Université de Lille

CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille

SAFARI

Doctoral School: MADIS-631

Integrated design of resilient system-of-systems

PhD THESIS
to obtain the title of

Doctor from University of Lille
Speciality : Automatic, Production

Defended by

Jun Jiang

prepared at
CRIStAL CNRS-UMR 9189
defended on November 14, 2024
at CRIStAL, Lille

Jury

Reviewers :	Pr. Taha Boukhobza	-	University of Lorraine, France
	Pr. Kenn Steger-Jensen	-	University of South-Eastern Norway, Norway
President of jury :	Pr. Ahmed Mebarki	-	Gustave Eiffel University, France
Examiners :	Dr. Ourania Tzoraki	-	University of the Aegean, Greece
	Dr. Lydie Nouveliere	-	Uiversity of Evry, France
Supervisor :	Pr. Rochdi Merzouki	-	University of Lille, France
Co-Supervisor :	Dr. Othman Lakhal	-	University of Lille, France
Guest :	Mr. Francisco Blanquer	-	CMA CGM, France

École Doctorale: MADIS-631

Conception intégrée de système de systèmes résilients

THÈSE

soumise à l'Université de Lille en vue de l'obtention de

Doctorat

Spécialité : Automatique, Productique

par

Jun Jiang

préparée au laboratoire

CRIStAL CNRS-UMR 9189

soutenue le 14 novembre 2024

à CRIStAL, Lille

Jury

Rapporteurs :	Pr. Taha Boukhobza	-	Université de Lorraine, France
	Pr. Kenn Steger-Jensen	-	Université du Sud-Est de la Norvège, Norvège
Président du jury :	Pr. Ahmed Mebarki	-	Université Gustave Eiffel , France
Examinateurs :	Dr. Ourania Tzoraki	-	Université de l'Égée, Grèce
	Dr. Lydie Nouveliere	-	Université d'Évry, France
Directeur de thèse :	Pr. Rochdi Merzouki	-	Université de Lille, France
Co-Encadrant:	Dr. Othman Lakhal	-	Université de Lille, France
Invité :	Mr. Francisco Blanquer	-	CMA CGM, France

Integrated design of resilient system-of-systems

Abstract

This thesis aims to develop a comprehensive, integrated framework to enhance the resilience of a system-of-systems (SoS). As SoS grow more complex and interdependent, ensuring their resilience is critical for maintaining effectiveness and reliability in the face of internal failures, external disruptions, and communication breakdowns. The proposed framework is structured into three key components: structural analysis, diagnosis, and decision-making.

The first component focuses on analyzing the controllability of multi-level SoS, which consist of heterogeneous component systems (CSs) with diverse dynamics, using the Sylvester equation. Some physical component systems (PCSs) may become uncontrollable due to malfunctions. Controllability analysis of the overall SoS contributes to resilient structural design, allowing the system to remain functional despite localized disruptions and laying the foundation for diagnosis and reconfiguration.

The second part addresses SoS diagnosis, proposing a performance evaluation framework to assess SoS status comprehensively. AI-based performance classifiers are employed to detect performance degradation, allowing reactive monitoring of the system's status under varying levels of disruption. This diagnosis process informs decisions on necessary reconfigurations to restore system performance.

The final part focuses on decision-making for resilience reconfiguration. Building on structural analysis and diagnosis, the decision-making strategy includes three scenarios: resilience implementation through bottom-up monitoring and top-down management, advance resilience planning based on disaster prediction, and recovery time estimation for unpredictable disruptions. The first scenario implements a resilience framework where faulty behaviours are detected using AI-based fault detection and isolation (FDI) algorithms, enabling top-down reconfiguration to optimize overall system performance. The second scenario focuses on proactive resilience planning for predictable disruptions (e.g., natural disasters) based on data-driven disaster prediction. In cases of unpredictable disasters, the framework estimates recovery time using optimized scheduling and routing of repair crews with heterogeneous capacities. In conclusion, this resilient integrated design framework provides: structural analysis for system controllability, performance evaluation for system diagnosis, and resilience reconfiguration strategies. These approaches make the SoS more structured, reactive, and efficient, ultimately enhancing its ability to recover from disruptive events and maintain operational stability.

Keywords: system-of-systems (SoS), controllability, multi-level, hypergraph modeling, system performance analysis, artificial intelligence (AI), classification, fault detection and isolation (FDI), resilience plan, decision making.

Conception intégrée de système de systèmes résilients

Résumé

Cette thèse vise à développer un cadre intégré et complet pour renforcer la résilience d'un système de systèmes (SoS). à mesure que les SoS deviennent plus complexes et interdépendants, garantir leur résilience est crucial pour maintenir leur efficacité et leur fiabilité face aux défaillances internes, aux perturbations externes et aux ruptures de communication. Le cadre proposé est structuré en trois composants clés : l'analyse structurelle, le diagnostic et la prise de décision.

Le premier composant se concentre sur l'analyse de la contrôlabilité des SoS multi-niveaux, qui se composent de systèmes composants hétérogènes (CS) aux dynamiques diverses, en utilisant l'équation de Sylvester. Certains systèmes composants physiques (PCS) peuvent devenir incontrôlables en raison de dysfonctionnements. L'analyse de la contrôlabilité de l'ensemble du SoS contribue à une conception structurelle résiliente, permettant au système de rester fonctionnel malgré des perturbations localisées et posant les bases pour le diagnostic et la reconfiguration.

La deuxième partie traite du diagnostic des SoS, en proposant un cadre d'évaluation des performances pour évaluer de manière exhaustive l'état des SoS. Des classificateurs de performance basés sur l'IA sont utilisés pour détecter la dégradation des performances, permettant une surveillance réactive de l'état du système face à différents niveaux de perturbation. Ce processus de diagnostic informe les décisions sur les reconfigurations nécessaires pour restaurer les performances du système.

La dernière partie se concentre sur la prise de décision pour la reconfiguration résiliente. En se basant sur l'analyse structurelle et le diagnostic, la stratégie de prise de décision inclut trois scénarios : la mise en œuvre de la résilience par une surveillance ascendante et une gestion descendante, la planification avancée de la résilience basée sur la prédiction des catastrophes, et l'estimation du temps de récupération pour des perturbations imprévisibles. Le premier scénario met en œuvre un cadre de résilience où les comportements défaillants sont détectés à l'aide d'algorithmes de détection et d'isolation des pannes (FDI) basés sur l'IA, permettant une reconfiguration descendante pour optimiser la performance globale du système. Le deuxième scénario se concentre sur la planification proactive de la résilience pour des perturbations prévisibles (par exemple, les catastrophes naturelles) basée sur la prédiction des catastrophes à partir de données. En cas de catastrophes imprévisibles, le cadre estime le temps de récupération en optimisant la planification et l'acheminement des équipes de réparation aux capacités hétérogènes.

En conclusion, ce cadre de conception résilient et intégré propose : une analyse structurelle pour la contrôlabilité du système, une évaluation des performances pour le diagnostic du système, et des stratégies de reconfiguration résiliente. Ces ap-

proches rendent le SoS plus structuré, réactif et efficace, renforçant ainsi sa capacité à se remettre des événements perturbateurs et à maintenir sa stabilité opérationnelle.

Keywords: Système de systèmes (SoS), contrôlabilité, multi-niveaux, modélisation par hypergraphe, analyse de performance du système, intelligence artificielle (IA), classification, détection et isolation des défauts (FDI), plan de résilience, prise de décision.

Acknowledgments

This research work has been accomplished in Université de Lille, at Laboratory Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL) (UMR CNRS 9189) within the SoftE (System of Systems Engineering) team of the ToPSyS (Tolerance Prognosis System of Systems) group.

Above all, I would like to express my sincere thanks to my supervisor Rochdi Merzouki, for his kind guidance and contiuous support over the past four years since my Master's studies. His insightful feedback and encouragement have been instrumental in shaping both this thesis and my development as a researcher.

I would also like to thank my co-supervisor, Dr. Othman Lakhal, for his thoughtful input and support during my PhD. Special thanks to my colleague Mr. Abdelkader Belarouci, for his vaulable help in my research work.

Meanwhile, I would like to thank all the jury members for their time and effort to review my thesis reviewers. My gratitude extends to my thesis reviewers: Prof. Taha Boukhobza and Prof. Kenn Steger-Jensen, and other jury members: Prof. Ahmed Mebarki, Dr. Ourania Tzoraki and Dr. Lydie Nouveliere, as well as the guest Mr. Francisco Blanquer. Their precious comments help me to improve my research greatly. These invaluable suggestions not only affect my thesis and defence, but are also of great importance in my future research.

Many thanks to my lab mates at CRIStAL: Mario Sanz Lopez, Gérald Dherbomez, Maxime Duquesne. Additionally, I would like to express my thanks to my colleagues: Xinrui Yang, Abdeslem Smahi, Abbass Chreim, Mahdi Boukerdja, Xiaoqin Zhai.

Finally, and most importantly, I would like to thank my family: my wife Yiwen Chen, my parents, my elder sister and the rest of my family for their love and support. It's because of their love that I am able to pursue my PhD degree. Especially my wife, great thanks to her infinite love, support and understanding throughout this entire process. She has made countless sacrifices to help me. I would also like to express my gratitude to myself for enduring so many challenges, staying the course through a long and difficult journey, and never giving up.

Contents

Acronyms **1**

1 Introduction **3**
1.1 General Introduction . . . 3
1.2 Thesis Scientific Context . . . 4
1.3 Research gap . . . 4
1.4 Thesis objectives . . . 5
1.5 Thesis contributions . . . 6
1.6 Thesis organization . . . 6
1.7 Disseminated results . . . 11
1.7.1 Journal Papers: . . . 11
1.7.2 International Conferences: . . . 11
1.7.3 Other Contributions: . . . 11

2 State of art for designing resilient SoS **13**
2.1 System-of-systems . . . 13
2.2 Graph-based and hypergraph-based modelling . . . 17
2.3 Resilience analysis . . . 19
2.4 Structural analysis . . . 21
2.4.1 Controllability . . . 21
2.5 Diagnosis . . . 23
2.5.1 AI-based classification and prediction . . . 23
2.5.2 Data-driven based prediction . . . 30
2.6 Decision making . . . 31
2.6.1 Crew dispatch . . . 32
2.7 Conclusion . . . 33

3 Modelling of multi-level collaborative heterogeneous systems **35**
3.1 Management decomposition of multi-level SoS structure . . . 35
3.2 Graph-based communication and organization topology . . . 37
3.3 Hypergraph-based communication and organization topology . . . 40
3.4 Hybrid graph/hypergraph-based communication and organization topology . . . 42
3.5 Nonlinear and linear dynamic component system modelling . . . 45
3.5.1 Nonlinear component system modelling . . . 45
3.5.2 Linear component system modelling . . . 45
3.6 SoS resilience metric . . . 46
3.7 Maier's Properties . . . 48
3.8 Conclusion . . . 49

4 SoS structure analysis of multi-level SoS controllability 51
4.1 Introduction . . . 51
4.2 State-controllability of heterogeneous SoS . . . 54
4.2.1 A two-layer SoS . . . 54
4.2.2 Controllability analysis based on Sylvester equation . . . 59
4.2.3 A three-layer SoS . . . 66
4.2.4 Two-layer decomposition: a divide-and-conquer viewpoint . . . 67
4.3 Qualitative and quantitative tests . . . 73
4.4 Conclusion . . . 76

5 Towards classification of operational status of SoS 78
5.1 Introduction . . . 78
5.2 Communication topology . . . 81
5.2.1 Graph-based communication among PCSs . . . 82
5.2.2 Hypergraph-based organization/communication between PCSs and MCSs of managerial level 1 . . . 82
5.2.3 Hypergraph-based organization between MCSs in managerial levels . . . 83
5.3 Problem formulation . . . 84
5.3.1 Problem formulation of Type-I communication topology . . . 84
5.3.2 Problem formulation of Type-II communication topology . . . 84
5.4 Feature construction of SoS performance classification . . . 85
5.4.1 Feature of PCSs . . . 86
5.4.2 Feature of communication quality . . . 87
5.4.3 Feature of MCSs of managerial level 1 . . . 90
5.4.4 Feature of MCSs of managerial level n . . . 93
5.4.5 Feature of SoS Management efficiency . . . 93
5.5 Qualitative and quantitative test . . . 96
5.5.1 Simulation . . . 96
5.5.2 Classification results . . . 97
5.6 Conclusion . . . 100

6 Resilient decision making of SoS 104
6.1 Introduction . . . 104
6.2 A resilience implementation framework of SoS . . . 105
6.2.1 Hypergraph-based functional and informational SoS modelling 105
6.2.2 Resilient design of SoS . . . 108
6.2.3 Physical twin of the smart port . . . 114
6.2.4 Conclusion . . . 118
6.3 SoS Resilient Planning Facing Predictable Disaster . . . 119
6.3.1 Modelling and problem formulation . . . 119
6.3.2 Data-driven based scenario prediction framework via multi-scale time series analysis . . . 121
6.3.3 Optimization for resilience planning based on disaster prediction 124

6.3.4 Qualitative and quantitative tests . . . 125
6.3.5 Validation of resilience planning of digital twin via Flexsim . 129
6.3.6 Comparison and analysis . . . 142
6.3.7 An extended disaster scenario . . . 144
6.3.8 Comparison and analysis . . . 149
6.3.9 Conclusion . . . 151
6.4 SoS Resilient Recovery Time Estimation . . . 151
6.4.1 Recovery time estimation of SoS . . . 152
6.4.2 Qualitative and quantitative tests . . . 158
6.4.3 Validation of RTE via Flexsim . . . 159
6.4.4 Conclusion . . . 162
6.5 Conclusion . . . 162

7 Conclusion and prospective **164**
7.1 Summary of main results . . . 164
7.2 Future works . . . 167
7.2.1 Controllability analysis . . . 167
7.2.2 Performance analysis . . . 168
7.2.3 Reconfiguration strategy . . . 168

Appendices **170**

A Preliminaries **172**
A.1 Graph and Hypergraph . . . 172
A.1.1 Graph . . . 172
A.1.2 Hypergraph . . . 174
A.2 Basic concepts in control theory . . . 175
A.2.1 Controllability . . . 175
A.2.2 Observability . . . 177
A.2.3 Cooperative output regulation . . . 178
A.3 AI-based classification and prediction . . . 180
A.3.1 AI-based prediction . . . 192
A.4 Multiple Traveling Salesman Problem (mTSP) . . . 197

B Coordinate transformation **200**

Bibliography **201**

List of Figures

1.1 Contents and contributions of thesis 10

2.1 An example of 3-layer SoS. $CS_{0,i}$ denotes the i-th PCS in physical level 0. $CS_{1,j}$ denotes the j-th supervisor (MCS) in management level 1. $CS_{2,j'}$ denotes the j'-th supervisor (MCS) in management level 2. 15

3.1 Communication/Network Topology Types. 38
3.2 An example for graph-based communication and organization topology of SoS modelling. 39
3.3 Organization and communication between PCSs and MCSs, and among MCSs. 42
3.4 Hybrid graph/hypergraph-based communication and organization topology. 44

4.1 In [Wu 2020, Wu 2023], the considered CSs shown in Fig.(a) belong to the same layer are of homogeneous dynamics. Given in Fig.(b), the considered CSs are of heterogeneous dynamics in one layer. . . . 54
4.2 An example of two-layer SoS over a directed chain graph $\mathcal{G}$ 58
4.3 An example of two-layer SoS over a directed star graph $\mathcal{G}$ 58
4.4 Two-layer decomposition in 3-layer SoS 69
4.5 Two-layer SoS subject to multiple graphs 73
4.6 3-layer SoS subjected to directed and undirected tree 77

5.1 Structure decomposition of SoS: 3-layer SoS with 8 PCSs and 3 MCSs. 83
5.2 Degree of operability of four SoS performances 85
5.3 Examples of management efficiency 95
5.4 Closed-loop control of the i-th PCS ($\sum_i$) [Jiang 2023a] 96
5.5 3D t-SNE visualization 98

6.1 An example of 3-layer SoS with bottom-up monitoring and top-down reconfiguration. $CS_{0,i}$ denotes the i-th component system in physical level 0. $CS_{1,j}$ denotes the j-th supervisor in management level 1. $CS_{2,j'}$ denotes the j'-th supervisor in management level 2. 105
6.2 $\mathcal{H}_{SoS}$ related to SoS in Fig.6.1. 106
6.3 An example of function-based hypergraph $\mathcal{H}_{fun}$ with $e_{fun,1}$, $e_{fun,2}$ and $e_{fun,3}$: hyperedges $e_{fun,1}$, $e_{fun,2}$ and $e_{fun,3}$ represents video, positioning and wifi functions. $e_{fun,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,8}\}$, $e_{fun,2} = \{CS_{0,1}, CS_{0,2}, CS_{0,4}, CS_{0,6}\}$, $e_{fun,3} = \{CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$. 107

6.4 NN-based FDI: Step 1 to step 4 show how to train the NN-based FDI model. The annotation is done manually to identify the location of the fault. After the training, skip step 2 is the process of our NN-based FDI. 108
6.5 Informational Observability and Controllability. Based on above hypergraph, $CS_{0,1}$ is directly observable but not controllable to $CS_{1,1}$. $CS_{0,2}$ is informationally observable with the help of hyperpath $HP^{dist}_{v_1,v_2}$ in $\mathcal{E}_{dist}$ but not informationally controllable to $CS_{1,1}$. $CS_{0,3}$ and backup $CS^{a}_{0,1}$ are directly observable and controllable to $CS_{1,1}$. $CS_{0,4}$ is directly controllable but not observable to $CS_{1,1}$. $CS_{0,5}$ is informationally controllable with the help of hyperpath $HP^{dist}_{v_4,v_5}$ but not informationally observable to $CS_{1,1}$. With the aid of backup $CS^{a}_{0,1}$, $CS_{0,6}$ are both informationally observable and controllable to $CS_{1,1}$. $CS_{0,7}$ is neither informationally observable nor controllable. 111
6.6 Resilience implementation of SoS 112
6.7 Organization structure of physical twin. 114
6.8 KNN based movement predictor 116
6.9 Training accuracy of NN-based FDI model 116
6.10 UAV (backup component systems) based reconfiguration. 117
6.11 Example of Flood scenario estimation in resilience planning. 122
6.12 Simulated rainfall and its corresponding capacity of drainage system. 126
6.13 Digital twin by Flexsim. 132
6.14 Workloads of vehicle, train and ship in Control group 0 133
6.15 Workloads of vehicle, train and ship in Control group 0 in one day . 134
6.16 Workloads of vehicle, train and ship in Experimental group I 135
6.17 Workloads of vehicle, train and ship in Experimental group II with $Per^{lb} = 60\%$ 135
6.18 State Gantt of trains and ships in Experimental group II with $Per^{lb} = 60\%$ 136
6.19 Workloads of vehicle, train and ship in Experimental group II with $Per^{lb} = 70\%$ 137
6.20 State Gantt of trains and ships in Experimental group II with $Per^{lb} = 70\%$ 137
6.21 Workloads of vehicle, train and ship in Experimental group II with $Per^{lb} = 80\%$ 138
6.22 State Gantt of trains and ships in Experimental group II with $Per^{lb} = 80\%$ 138
6.23 Workloads of vehicle, train and ship in Experimental group II with $Per^{lb} = 90\%$ 138
6.24 State Gantt of trains and ships in Experimental group II with $Per^{lb} = 90\%$ 139
6.25 Workloads of ship in Experimental group III with $Per^{lb} = 60\%$ . . . 139
6.26 State Gantt of trains and ships in Experimental group I with $Per^{lb} = 60\%$ 140

6.27 Workloads of ship in Experimental group III with $Per^{lb} = 70\%$ . . . 140
6.28 State Gantt of trains and ships in Experimental group I with $Per^{lb} = 70\%$ . . . 141
6.29 Workloads of ship in Experimental group III with $Per^{lb} = 80\%$ . . . 141
6.30 State Gantt of trains and ships in Experimental group III with $Per^{lb} = 80\%$ . . . 142
6.31 Operating performance between control group 0 and experimental group I during the whole operating time horizon. . . . 142
6.32 Operating performance between control group 0 and experimental group II (concluding 4 cases) during the whole operating time horizon. 143
6.33 Operating performance between control group 0 and experimental group III (concluding 3 cases) during the whole operating time horizon. 144
6.34 Generated wind disaster scenario. . . . 146
6.35 Workloads of vehicle, train and ship in Experimental group I under wind disaster. . . . 146
6.36 State Gantt of trains and ships in Experimental group I under wind disaster. . . . 147
6.37 Workloads of vehicle, train and ship in Experimental group II with $Per^{lb} = 90\%$ under wind disaster. . . . 147
6.38 State Gantt of trains and ships in Experimental group II with $Per^{lb} = 90\%$ under wind disaster. . . . 148
6.39 Workloads of vehicle, train and ship in Experimental group III with $Per^{lb} = 60\%$ under wind disaster. . . . 149
6.40 State Gantt of trains and ships in Experimental group III with $Per^{lb} = 60\%$ under wind disaster. . . . 149
6.41 Operating performance between control group 0 and experimental group I under wind disaster during the whole operating time horizon. 150
6.42 Operating performance between control group 0, experimental group II and experimental group III under wind disaster during the whole operating time horizon. . . . 150
6.43 Recovery with 6 crews . . . 159
6.44 Workloads of inoperable PCSs with the optimized 5 crews recovery strategy. . . . 161
6.45 State Gantt of inoperable PCSs with the optimized 5 crews recovery strategy. . . . 161

A.1 The relationship among AI, ML and DL. . . . 180
A.2 Examples of Logistic function (Sigmoid function) . . . 182
A.3 An example of two leaves DT . . . 184
A.4 An example of RF . . . 185
A.5 An example of SVM: the criterion that SVM seeks to optimize. . . . 187
A.6 An example of $k = 4$ to visualize the process of KNN. . . . 189
A.7 Overview of a NN's learning process. . . . 191
A.8 Linear Regression with one independent variable. . . . 192

A.9 The difference between Linear Regression and Polynomial Regression. 194

List of Tables

5.1 Disturbances among 5 classes of SoS 101
5.2 Results ofSoS Performance classification for Type-I 102
5.3 Results of SoS Performance classification for Type-II 103

6.1 Historical data related to multi-scale time series used in classification and regression 124
6.2 Results of Rainfall Prediction 126
6.3 Parameters used in the resilience planning optimization 130
6.4 Results of resilience planning optimization 131
6.5 Results of the optimization 160

Acronyms

AI Artificial Intelligence. vi, ix, 8, 9, 20, 23, 24, 26–28, 30, 31, 34, 80, 81, 100, 108, 121, 151, 166, 180, 192

CM Corrective Maintenance. 152

CPS Cyber-Physical System. 8, 166

CS Component System. vii, 3, 5, 7, 8, 13–18, 21, 23, 33, 34, 37–40, 44, 48–57, 68, 71, 72, 76–80, 85, 93, 96, 97, 100, 105–111, 113, 115, 132, 152, 162, 165–168

DL Deep Learning. ix, 24, 26, 180, 190, 197

DNN Deep Neural Network. 180

DT Decision Trees. ix, 24, 26–28, 182–185, 195

FDI Fault Detection and Isolation. viii, 108, 109, 111, 114, 116, 118, 119, 166

FNN Feedforward Neural Network. 190

GPS Global Positioning System. 111

KNN K-Nearest Neighbors. viii, ix, 24–26, 29, 80, 97, 99, 108, 116, 187–190, 196, 197

LR Linear Regression. 26, 27, 192

LSS Large-Scale Systems. 3, 14, 32, 73, 129

LTI Linear Time-Invariant. 22, 23, 52, 176, 178

MAE Mean Absolute Error. 193, 197

MAS Multi-Agent Systems. 21–23, 34, 51–53, 60, 76

MCS Managerial Component System. vii, 8, 14, 15, 35, 36, 39–44, 53–55, 59, 60, 66–68, 70, 72–77, 80–94, 96, 97, 99, 100, 164–167

MIMO Multi-Input/Multi-Output. 22, 23, 52

ML Machine Learning. ix, 20, 31, 180, 182, 183, 185, 192

MLP Multi-Layer Perceptron. 97, 190

MSE Mean Squared Error. 193, 195, 197

mTSP Multiple Traveling Salesman Problem. 197, 198

MTZ Miller-Tucker-Zemlin. 199

NN Neural Network. viii, ix, 24, 26, 29, 30, 80, 97, 108, 116, 180, 190, 191, 197

NS Networked System. 21–23, 34, 52, 53, 168

OLS Ordinary Least Squares. 193

PBH Popov-Belevitch-Hautus. 53, 56, 57, 71, 72, 74, 76, 176–178

PCS Physical Component System. vii, ix, 5, 7–9, 14, 15, 35, 36, 38–44, 46, 47, 49, 51, 53–56, 60, 66–68, 73–77, 80–94, 96, 97, 99, 100, 108, 114, 116, 117, 119–121, 125, 129, 151–153, 155–159, 161, 162, 164–168, 179

PM Preventive Maintenance. 152

PR Polynomial Regression. 26, 27, 194, 195

ReLU Rectified Linear Unit. 191, 197

RF Random Forest. ix, 24–28, 80, 97, 99, 183, 185, 195, 196

RMSE Root Mean Squared Error. 193, 197

ROC-AUC Receiver Operating Characteristics - Area Under the Curve. 184, 197

RTE Recovery Time Estimation. 152

SoS System-of-Systems. iv, vii, viii, xi, 3–9, 13–19, 21, 23, 33–38, 40–44, 46–49, 51–59, 61–63, 66–74, 76–81, 83, 85–87, 90, 93, 96, 97, 100–109, 111–121, 125, 151, 152, 155, 158, 159, 161–168, 175

SVM Support Vector Machine. ix, 24, 25, 28, 31, 97, 185–187, 196

SVR Support Vector Regression. 26, 28, 29, 196

t-SNE t-distributed Stochastic Neighbor Embedding. 99

UAV Unmanned Aerial Vehicle. viii, 117, 166

Chapter 1

Introduction

Contents

1.1 General Introduction . . . 3
1.2 Thesis Scientific Context . . . 4
1.3 Research gap . . . 4
1.4 Thesis objectives . . . 5
1.5 Thesis contributions . . . 6
1.6 Thesis organization . . . 6
1.7 Disseminated results . . . 11
1.7.1 Journal Papers: . . . 11
1.7.2 International Conferences: . . . 11
1.7.3 Other Contributions: . . . 11

1.1 General Introduction

In the contemporary landscape of technological advancement and organizational complexity, the concept of System-of-Systems (SoS) has become significant for describing, understanding and managing the sophisticate interconnected networks of systems [Jamshidi 2008]. Unlike traditional single systems, a SoS is characterised by the integration and coordination of multiple, independent Component Systems (CSs), each capable of functioning autonomously, yet collectively contributing to overarching goals and capabilities. To this end, the focus of SoS engineering extends beyond individual system performance to encompass the interactions and emergent behaviours of the entire network [Sousa-Poza 2008].

The evolution of SoS reflects the growing complexity and inter-connectivity of Large Scale System Large-Scale Systems (LSS). The development of information technology, communication networks, and systems engineering enables the integration of diverse CSs across various domains, allowing more sophisticated and resilient capabilities. This integration of multiple independent and interoperable CSs, enhances the capability of SoSs over the sum of individual systems [Madni 2014]. SoS integrates diverse functionalities of heterogeneous CSs, which lays the ground for its capacities to accomplish more complicated tasks that one single system cannot, leveraging the strengths and meanwhile compensating the weaknesses of the individual CS [Mostafavi 2011]. Besides, the cooperation among CSs leads to the better

performance as well as efficiency, generating value that exceeds the combined total of the individual components [DiMario 2009].

Given the increasing complexity and interdependence of SoS, resilience in SoS design and management is essential for maintaining their effectiveness and reliability [Han 2012a]. In an SoS, multiple independent subsystems interact and depend on each other, meaning that a failure in one area can potentially trigger a chain reaction across the entire system. Resilience helps mitigate these risks by ensuring that the system can absorb shocks, adapt to changing conditions, and continue to function even when individual components fail or face stress [Mebarki 2017a,Mebarki 2017b]. Resilience also ensures that the SoS can maintain an acceptable level of performance during disruptions, allowing for continuity of operations in the face of natural disasters, cyber-attacks, or other unforeseen events. By reducing downtime and improving recovery times, resilient systems minimize the economic and operational impacts of disruptions, leading to significant cost savings and increased efficiency. Ultimately, resilience in SoS is essential for maintaining performance, security, and economic viability, ensuring that these complex systems can thrive in dynamic, uncertain environments while continuing to deliver critical services and functions.

In complex and dynamic environments, SoS are often exposed to uncertainties such as external disturbances, component failures, natural disasters, and communication issues. To this end, the integrated design of resilient SoS is of great importance in maximizing the performance, safety, and reliability of SoS, enabling it to operate robustly in the face of disruptions and to continue delivering critical functions.

1.2 Thesis Scientific Context

This thesis is conducted at Laboratory Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL) (UMR CNRS 9189) within the SoftE (System of Systems Engineering) team of the ToPSyS (Tolerance Prognosis System of Systems) group. The scientific context concerns the integrated design of resilient SoS including structural analysis (controllability analysis), diagnosis (performance evaluation and classification) and decision making (resilience implementation, prediction-based planning in advance, recovery time estimation with crew dispatch for repairing). The objective is to enhance the overall SoS performance to ensure SoS remains operational and effective even in the face of disruptions, failures, or unexpected changes. This research has been developed under the supervision of Prof. Rochdi Merzouki, Professor at the University of Lille and Dr. Othman Lakhal, Associate Professor at the University of Lille.

1.3 Research gap

Although we have already witnessed various works on the system resilience analysis, most of them were emerged on single system, rather than the complexity system-

of-systems. Designing resilient SoS presents several significant challenges due to the inherent complexity, which includes:

- In the structural analysis, SoS are often dynamic, with CSs frequently heterogeneous, interdependent and suffering from malfunctions. Designing a resilient structure adds a layer of difficulty in ensuring sustained robustness and adaptability.

- In the diagnosis, SoS comprises multiple CSs with diverse functions and interactions. Evaluating and classifying the SoS performance with complex interactions among CSs, including how changes or failures in one system can impact others, is highly intricate.

- In the decision making, effective resilience in SoS depends on the interdependence and coordination among diverse CSs. Misalignment in objectives, priorities, or communication protocols between these systems can hinder overall resilience.

Addressing these difficulties requires an integrated approach of resilient SoS, incorporating advanced SoS modelling, controllability analysis for a robust structure, comprehensive performance evaluation and classification for diagnosis, resilience implementation, prediction-based planning in advance, and crew dispatch optimization-based recovery time estimation for decision making.

1.4 Thesis objectives

The objective of this thesis focus on the integrated design of resilient SoSs. As SoS become increasingly complex and interdependent, designing and managing them with resilience is crucial to ensure their effectiveness and reliability. The integrated design of resilient SoSs can be decomposed into the following steps: the controllability analysis for structural analysis, performance evaluation and classification for diagnosis, and resilience reconfiguration for decision making. Accordingly, the main results of this thesis are structured with the following objective:

- Analyze the controllability of SoS under networked interactions while a part of Physical Component Systems (PCSs) are out of control. This helps SoS to remain operational facing with malfunctions in a part of PCSs caused by various disturbances and risks.

- Develop an evaluation framework to assess the performance of the System of Systems SoS and classify its operational status. This classification will provide users with critical insights into the SoS's condition, facilitating its diagnosiss. It also supports the identification of necessary and/or sufficient reconfigurations required to recover from any performance degradation within the SoS.

- Develop a resilience plan for decision-making within a System of Systems SoS, incorporating the following elements: a resilience implementation framework that combines bottom-up monitoring with top-down reconfiguration; advance resilience planning based on disaster predictions for predictable events; and recovery time estimation based on repair crew dispatch for unpredictable disasters. These measures help the SoS prepare for both predictable and unpredictable events, optimize crew deployment, restore system performance, and accurately estimate recovery time.

1.5 Thesis contributions

The contributions of this thesis can be summarised as follows:

- Chapter 3 introduces three types of network topologies, graph-based, hypergraph-based, and hybrid graph/hypergraph-based topologies, to represent the communication and organization in SoS modeling. It also outlines the unique characteristics and suitable research areas for each topology, highlighting their specific applications across various chapters of the study

- Chapter 4 investigates the controllability of multi-level heterogeneous SoSs within directed graph-based networks. The considered CSs are of heterogeneous dimensions, dynamics and interactions. With the aid of the modified Sylvester equation, some necessary and sufficient conditions have been obtained to ensure the controllability of two-level SoS. The results are thereafter extended to multi-level SoS from a divide-and-conquer perspective [Jiang 2024a].

- Chapter 5 presents an assessment framework designed to evaluate the performance of an SoS from a holistic perspective, enabling its classification for system diagnosis [Jiang 2023a].

- Chapter 6 outlines the resilience reconfiguration strategy for SoS. First, a resilience implementation framework is introduced to address faults and malfunctions, featuring bottom-up monitoring and top-down reconfiguration [Jiang 2022]. Second, a resilience planning strategy is developed for predictable disasters, utilizing disaster forecasts to enhance preparedness. Finally, an optimized approach for repair crew routing and scheduling is proposed to respond to unpredictable disasters, providing the basis for estimating SoS recovery time [Jiang 2024c].

1.6 Thesis organization

The detailed organization and its corresponding contributions of each chapter can be summarised in Fig.1.1 with the following descriptions:

Chapter 1 establishes the framework and scope of the thesis. It begins with a general introduction to SoS, laying the groundwork for the research, followed by a discussion of the scientific context that situates the study within its academic domain. The research gap section identifies key unresolved issues the thesis seeks to address, while the thesis objectives define the specific aims of the study. The thesis organization outlines the structure of the work, and the contributions section highlights the novel insights and advancements made. Finally, the disseminated results section details the research outputs, including journal articles, conference presentations, and other contributions, showcasing the study's impact and dissemination..

Chapter 2 offers a comprehensive review of essential concepts and methodologies relevant to the research. It begins by exploring SoS, focusing on their complexity and interconnectivity. A detailed literature review follows, covering graph-based and hypergraph-based modeling approaches used to represent SoS structures. The chapter then shifts to resilience analysis, examining strategies to enhance system resilience. Structural analysis, including controllability, assesses the ability of systems to maintain functionality amid disruptions. The diagnosis section discusses methods for identifying system issues, incorporating AI-based classification and data-driven prediction techniques. Lastly, the decision-making section highlights strategies for system reconfiguration, emphasizing crew dispatch optimization, demonstrating the practical applications of these concepts in sustaining system performance.

Chapter 3 subsequently introduces the formulation of multi-level collaborative heterogeneous systems, utilizing graph-based and/or hypergraph-based topologies to describe the communication and organization of SoS modeling. It also presents the state-space representation of PCSs with either nonlinear or linear dynamics. For multi-level SoS, a management decomposition strategy is proposed to limit interactions to adjacent levels, thereby simplifying overall management and mitigating the risk of information overload. The graph-based topology is then introduced to depict the physical and logical structure of a SoS, along with the flow of data within it. This point-to-point representation of interactions in physical dynamics facilitates a detailed analysis of the SoS's dynamic response, laying the groundwork for controllability analysis discussed in Chapter 4. In devising the reconfiguration strategy, the focus shifts from physical dynamics to the multi-way relationships among CSs across different levels. To address this, a hypergraph-based topology is introduced into SoS modeling, which encapsulates multi-way relationships within a single hyperedge, thereby reducing redundancy and providing more concise and understandable models for resilience reconfiguration, as discussed in Chapter 6. Building on these methodologies, a hybrid graph/hypergraph-based topology is proposed. This approach uses traditional graphs for pairwise interactions to describe the heterogeneous interactions within the PCSs dynamics, while reserving hypergraphs for multi-way interactions that capture interconnections across different levels. The hybrid approach forms the foundation for the comprehensive SoS evaluation framework, presented in Chapter 5, which incorporates both dynamic response and management efficiency.

To establish a resilient structure, Chapter 4 delves into the controllability of

multi-level heterogeneous SoSs within directed graph-based networks. Given that some CSs may experience malfunctions, only a subset of them can be influenced by external inputs. Therefore, ensuring the controllability of the SoS is crucial to maintaining resilience.

The CSs considered in this work exhibit a higher degree of heterogeneity compared to previous studies, encompassing various dimensions, heterogeneous dynamics, and non-identical inner-coupling matrices. First, for a two-layer heterogeneous SoS, the chapter presents a framework that mitigates the computational complexity of controllability analysis using a modified Sylvester equation, commonly applied in cooperative output regulation. Subsequently, a two-layer decomposition approach simplifies the controllability analysis of more complex, higher-level SoSs by transforming them into lower-layer components.

The chapter adopts a "divide-and-conquer" strategy to reduce the complexity of controllability analysis in hierarchical heterogeneous SoSs, proposing necessary and/or sufficient conditions that require less computational effort. Several examples, including star, chain, circle, and tree graph structures, are provided to validate the effectiveness of the approach.

In Chapter 5, an assessment framework is proposed to evaluate the performance of SoS, so as to classify them for system diagnosis. The objectives of the framework are to inform users the SoS status and determine the sufficient and or necessary reconfigurations to recover from SoS degradation. To assess the performance degradation of SoS facing with faults, perturbations and communication issues, various performance features are extracted from different aspects, such as the response of PCSs, the task-oriented attributes of Managerial Component Systems (MCSs), the inter/intra-layer communication among CSs of multiple levels and the top-down management efficiency. These features provide a comprehensive view of the system's performance. Using the proposed feature construction and Artificial Intelligence (AI)-based classifiers, the SoS performance is classified into five status types: Normal, Robust, Fault-tolerance, Resilience and Breakdown. These classifications indicate the degree of malfunction and the urgency or necessity of a resilient response. This helps engineers identify the level of degradation in the SoS and decide on appropriate actions to restore its performance. Finally, a simulation of a three-level SoS with three MCSs and eight PCSs is conducted to validate the effectiveness of the feature construction for SoS performance. Various algorithms are utilized in the simulation, demonstrating high accuracy in both the training and test sets.

Chapter 6 explores the resilience reconfiguration strategy for SoSs in decision-making. This contribution is structured into three main components. First, we propose a resilience implementation approach designed to actively detect physical faults that could impact the SoS and adjust overall operating modes accordingly. This approach integrates both bottom-up monitoring and top-down reconfiguration strategies. Utilizing a Fault Detection and Isolation (FDI) algorithm applied to Cyber-Physical Systems (CPSs) at the foundational level, the bottom-up monitoring system can identify faulty behaviors and relay this information to higher-level supervisors. In response, the supervisor can issue instructions to lower-level CSs

to optimize the overall performance of the SoS, a process we refer to as top-down reconfiguration.

Unlike previous works that primarily focused on reconfiguration from a single perspective, this chapter presents a comprehensive framework that encompasses communication, operational independence, and mission objectives, all represented through a hypergraph derived from these various viewpoints. To demonstrate the effectiveness of the theoretical findings, we conclude with a case study of the physical twin of a smart port, considering multiple scenarios to validate our approach.

The second part focuses on the resilience planning in advance for SoS facing disruptive disaster scenarios on the basis of disaster prediction. Taking flash flood caused by excessive rainfall as an example of disaster, a hybrid rainfall prediction model is proposed in terms of the data-driven AI and multi-scale time series analysis. In simulation, the prediction model is test with Indonesia daily climate data from 2010 to 2022 including 173 sampling locations among 34 provinces. In view of the disaster prediction, the resilience planning optimization aims at allocating the operating hours of PCS at different operating sites in advance, to guarantee the performance degradation to a certain extent and meanwhile ensure the minimal operation cost. A case study on the digital twin of smart port via Flexsim facing flash flood or wind is provided finally to allocate the operating hours of human operators, trucks, freight trains and ships in port. In both the flood and wind scenarios, the groups with disaster prediction-based resilient planning have better performance than those without it, which reveals the effectiveness of the proposed strategy.

Facing with unpredictable disasters, the third part of this chapter studies the optimization-based resilient recovery time estimation for SoS with diverse resources and capacities. Regarding the crew dispatch in a large-scale SoS, it's natural to see distinct crews possess different capacities and performance in the repair of alternative PCSs. Facing with inevitable malfunctions of PCSs, the proposed optimization strategy incorporates not only the crew dispatch and PCS restoration, but also the heterogeneous capacity distribution that reflects the cooperation inside the SoS. The designed multi-objective optimization function minimizes the crew travel time, malfunction lost and potential risk. The estimation of recovery time is calculated based on the solution to the optimization. The case study is performed on the digital twin of smart port using Flexsim, with nine inoperable PCSs, eight crews and eight types of faults with alternative repair times, which shows the effectiveness of the proposed strategy.

In the context of both predictable and unpredictable disasters, the strategies outlined above facilitate proactive preparation, optimize crew dispatch, aid in restoring SoS performance and estimates the recovery time. These approaches enable a more structured and efficient response, ensuring that the SoS is better equipped to maintain operational continuity and quickly recover from disruptions.

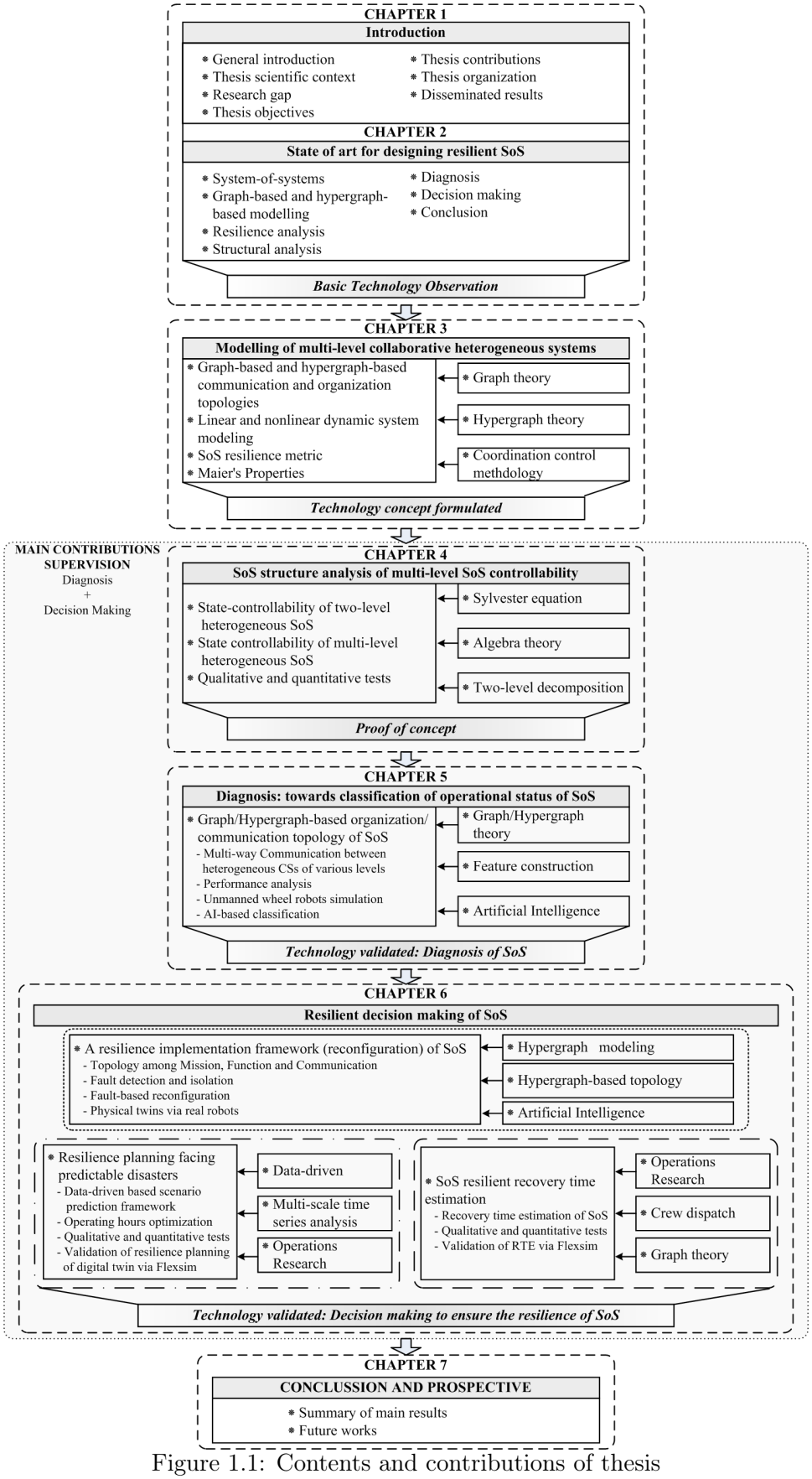


Figure 1.1: Contents and contributions of thesis

1.7 Disseminated results

The research contributions obtained during the development of this thesis have been published in:

1.7.1 Journal Papers:

- **Jiang J**, Chen Y, Lakhal O, Merzouki R. Controllability of heterogeneous multi-agent systems via cooperative output regulation. Journal of the Franklin Institute, Volume 361, Issue 15,2024: 107133, https://doi.org/10.1016/j.jfranklin.2024.107133

- **Jiang J**, Smahi A, Chen Y, Lakhal O, Merzouki R. AI-based Performance Classification of Multi-level System-of-Systems via Hypergraph Modelling. IEEE Access, (Under third term revision)

1.7.2 International Conferences:

- **Jiang J**, Lakhal O, Merzouki R. A resilience implementation framework of system-of-systems based on hypergraph model[C], 2022 17th Annual System of Systems Engineering Conference (SOSE). IEEE, 2022: 487-492.

- **Jiang J**, Chen Y, Lakhal O, Merzouki R. AI-based SoS performance classification for resilience reaction[C], 2023 18th Annual System of Systems Engineering Conference (SoSe). IEEE, 2023: 1-6.

- **Jiang J**, Chen Y, Lakhal O, Merzouki R. Event-triggered Reconfiguration Scheduling of Hypergraph-based System-of-systems: the Resilience Reaction[C], 2024 IEEE International Systems Conference (SysCon). IEEE, 2024: 1-7.

- **Jiang J**, Chen Y, Lakhal O, Merzouki R. System-of-Systems Resilient Recovery Time Estimation via Heterogeneous Capacity Distribution[C], 2024 19th Annual System of Systems Engineering Conference (SoSE). IEEE, 2024: 288-293.

1.7.3 Other Contributions:

- Chen Y, Wen G, Rahmani A, **Jiang J** and Huang T. Memory fusion controller of fractional-order systems with sliding memory window under intermittent sampled-data transmission[J]. IEEE Transactions on Cybernetics, 2023.

- Chen Y, Wen G, Rahmani A, **Jiang J** and Huang T. Resilient Stepped Transmission and Control for Nonlinear Systems against DoS Attacks[J]. Automatica, 2024. (Accepted)

- Chreim A, Chen Y, Smahi A, **Jiang J**, Merzouki R. Towards Supervision of Stochastic System of Systems Engineering: A Multi-Level Hypergraph Approach[J]. IEEE Access, 2024. (Early Access)
- Chen Y, Merzouki R, **Jiang J**, Defoort M and Djemai M. Memory Fusion Sampled-data Control of Fractional-order Heterogeneous Multi-agent Systems subject to DoS Attacks and Time Delays: A Resilient Binary Sampled-data Scheme[J]. IEEE Transactions on Cybernetics, 2024. (Under 1st review after major revisions)

Chapter 2

State of art for designing resilient SoS

Contents

2.1 System-of-systems 13
2.2 Graph-based and hypergraph-based modelling 17
2.3 Resilience analysis 19
2.4 Structural analysis 21
2.4.1 Controllability 21
2.5 Diagnosis 23
2.5.1 AI-based classification and prediction 23
2.5.2 Data-driven based prediction 30
2.6 Decision making 31
2.6.1 Crew dispatch 32
2.7 Conclusion 33

In terms of the thesis objectives and context introduced in Chapter 1, some necessary state of art of the considered concepts, together with the evolution of research topics and the utilized approaches in the integrated design of resilient SoS will be introduced in this Chapter.

2.1 System-of-systems

SoS refers to a complex network of independent, self-contained systems that are combined to achieve a higher-level functionality that no single system can accomplish on its own. Unlike traditional systems, which are designed as standalone entities, SoS comprises multiple CSs that maintain their own operational goals, management, and functionality but work together to fulfill overarching objectives towards broader objectives and enhanced capabilities. Consequently, SoS engineering shifts the focus from the performance of individual systems to the interactions, interdependencies, and emergent behaviours that arise within the entire network [Jamshidi 2008, Sousa-Poza 2008].

By integrating, replacing, repairing or removing CSs, SoS allows quick response to the changing requirements, resources and environments [Wudka 2020]. Meanwhile, SoS supports increasing or decreasing number of CS, the scalability of SoS

enable it to maintain the operational efficiency facing with growing workload and increasing demands over time [Fortino 2021]. What's more, the distributed nature of SoS provides redundancy, which implies if there exists faults, malfunctions or even damages in one CS, others can compensate for the performance degradation or loss of functionality [Uday 2014]. Moreover, for the same cooperative task, one SoS may provide distinct strategies, arrangements and approaches by allocating multiple CSs, which greatly improves the robustness and reliability of SoS [Gianetto 2013]. On the basis of this, the cost of SoS can be optimized by the resources sharing among CSs, leading to lower resources consumption and more efficient utilization [Davendralingam 2013]. Furthermore, SoS can aggregate data in virtue of the observations from multiple CSs, providing a more comprehensive and accurate information base for decision-making [Lautenbacher 2006, Eusgeld 2011]. At the same time, the integration of observations improve the situational awareness of SoS, allowing for more informed and timely decisions. Overall, compared with individual systems, SoS provides enhanced integrated functionality and coordination, better flexibility and scalability, improved fault tolerance and robustness, resource optimization, more comprehensive observation and decision-making.

Characterizing a SoS involves identifying and understanding its unique properties, components, interactions, and overarching goals. The key characteristics that distinguish an SoS from various complex, LSSs include operational and managerial independence of constituent systems, geographical distribution, evolutionary development, emergent behaviour [Maier 1998]. The operational independence reveals each CS belongs to the SoS can operate independently and realise it own functionality without necessarily depending on the other systems, fulfilling customer-operator purposes on their own. On the basis of it, the managerial independence takes a step further, which indicates all CSs are obtained and integrated independently, maintaining operational existence autonomously. Besides, the geographical distribution suggests CSs are typically distributed across different locations, allowing robust communication and coordination mechanisms. Moreover, the evolutionary development shows the evolution of SoS with the passage of time facing changing requirements and environments. Furthermore, the cooperation and interactions among CSs leads the the emergent behaviour of SoS, which is not predictable from the behaviour of one individual CS but relates to the involved interconnectivity and cooperation among CSs inside the SoS.

The CSs in SoS can be divided into PCSs and MCSs of hierarchical managerial levels, that functions together as an integration, respecting the principle proposed in [Maier 1998]. SoS can achieve complex coordination tasks compared with the sum of single systems, the interoperability of SoS ensures efficient cooperation with promising high performance, and at the same time, provides a higher level perspective [Bourne 2018]. An example is given in Fig.2.1 to show a simple 3-layer SoS.

Leveraging the power of integration and interoperability to address intricate and large-scale challenges, the concept of SoS has already garnered significant attention across multiple domains, such as smart cities [Axelsson 2018, Elshenawy 2018,

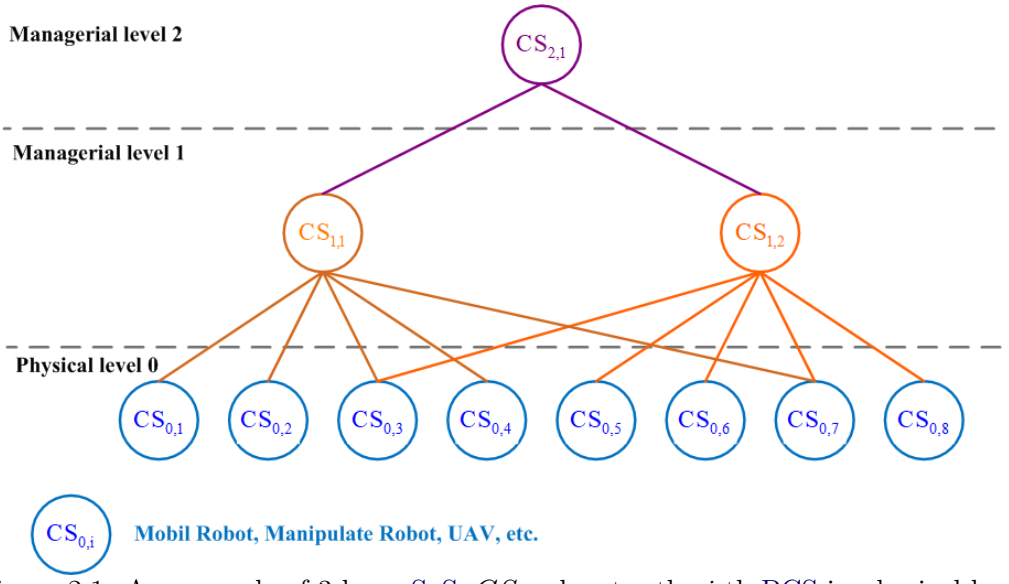


Figure 2.1: An example of 3-layer SoS. $CS_{0,i}$ denotes the i-th PCS in physical level 0. $CS_{1,j}$ denotes the j-th supervisor (MCS) in management level 1. $CS_{2,j'}$ denotes the j'-th supervisor (MCS) in management level 2.

Lee 2019], energy system [Zhao 2018, Uslar 2019], disaster response and management [Rehman 2021, Andrade 2022], healthcare system [Hata 2009, Grigoroudis 2013], logistics system [Nowicki 2010, Choi 2018], and so on. In the field of smart cities, [Elshenawy 2018] introduced a three-pillar framework for intelligent transportation SoS designed to support daily intelligent transportation operations in smart cities within the context of the Internet of Things. The framework was built on three pillars that unify and integrate dispersed cyber-physical components, facilitating coordinated planning among city stakeholders. This work further studied on the operation of these pillars and illustrated how they enable the dynamic provisioning of integrated intelligent SoS for transportation operations. a transportation governance framework has been proposed in virtue of SoS concept in [Lee 2019]. By viewing the land transportation system as a SoS in smart cities compose of autonomous, belonging, connected, diverse and emergent CSs, the authors emphasised the integration of various kinds of transportation modes, including train system, bus system, taxi system and road system. Using Singapore as a case study, this work demonstrated how effective the SoS-based governance framework enables smart cities to operate efficiently and effectively.

Regarding the area of energy system, [Zhao 2018] proposed a hierarchical decentralised SoS architecture for the energy management of a multi-microgrid system, formulating it as a bi-level optimization problem. In this regard, the level of individual microgrids focus on the multiple-stage robust optimization to handle the issue of renewable energy sources uncertainty. Meanwhile, the level of multi-microgrid system further addressed the problem of spatially unbalanced demand and generation in the area. Besides, [Uslar 2019] provided a comprehensive overview of the

application of Smart Grid Architecture Model in the design and validation of SoS in the power and energy domain, facing with increasing complexity of modern and sustainable power and energy systems.

Considering the applications in the field of disaster response and management, [Rehman 2021] proposed a flood warning, monitoring, and rescue SoS with the help of hierarchical coloured Petri-nets based model. Through rigorous model-based proof obligations and exhaustive verification techniques, the research ensured the correctness and reliability of the system in managing flood-related emergencies. [Andrade 2022] presented an SoS modelling and analysis of resilience in scalable traffic management for emergency response operations, with parameterized simulation of fire propagation and response, covering fire dynamics, coordination and planning, aerial firefighting, and fireline construction. Using the proposed model, the analysis evaluated the impact of reduced communication lag and enhanced surveillance on firefighting performance and resilience across various nominal and fault scenarios encountered during firefighting operations.

Refer to the SoS applications in healthcare system, [Hata 2009] introduced an SoS into human health management, as the first level of daily monitoring. In addition, the SoS in medical diagnostic imaging is discussed, with clinical usage of a magnetic resonance imaging scanner for the diagnosis of certain diseases. Moreover, the authors proposed an SoS in medical ultrasonic surgery support device, showing a novel ultrasonic support system for supporting crash bone orthopedic surgery. [Grigoroudis 2013] proposed a SoS-based approach to model healthcare system and design strategies that improve the health level of a population, enabling the hierarchical structure of national healthcare systems and providing analytical results for the CSs through simulations at each SoS level.

For logistics systems, the application of SoS concept has been reported in [Nowicki 2010], which established a SoS-based framework for performance-based logistics to describe its essence with the help of the grounded theory analysis. [Choi 2018] treated the global supply chain as an SoS. By incorporating big data technologies into SoS approach, the authors enhanced the global supply chain management with a risk management example.

SoS analysis is inherently challenging due to the complex nature of integrating multiple independent systems that interact dynamically to achieve a higher-level goal. One of the primary challenges is the sheer complexity and emergent behaviour of SoS, where the combined interactions of the CSs can produce unexpected outcomes that are not easily predictable from the properties of individual systems. Interoperability is another significant challenge, as it requires different systems—often built with diverse technologies to work together seamlessly through communications. This task becomes even more difficult in the presence of internal disruptions and communication issues, leading to potential incompatibilities that can cause integration failures, increased costs, and delays.

The presence of external disruptions adds another layer of difficulty, as each system within the SoS may have its own functions and operational constraints, making coordination and unified decision-making complex in the overall SoS restoration.

This decentralization often results in conflicting priorities and complicates the enforcement of standards, policies, and resource allocation. The requirements for SoS are typically uncertain, evolving, and sometimes poorly defined, necessitating adaptive and flexible approaches in design, analysis, and validation. These challenges make SoS analysis a dynamic, complex, and often unpredictable field, requiring innovative approaches to effectively design and manage interconnected systems. The integrated design of resilient SoS is a challenge task, particularly in the comprehensive evaluation of SoS performance and resilience reconfiguration in the presence of disruptions.

2.2 Graph-based and hypergraph-based modelling

Various kinds of modelling approaches have been employed in previous literature to capture the intricate relationships, organizations, behaviours and interdependencies among CSs. For example, [Hela 2021] utilized hierarchical coloured Petri Nets to model the cyber-physical level and the managerial levels of the SoS. [Mahulkar 2009] proposed an agent-based modelling to develop workflow simulations involving a ship's crew conducting routine maintenance, watch duty, and reporting functions. [Soyez 2017] proposed a multi-level agent-based modelling for SoS in both organizational and functional aspects. Besides, game theory has been employed in [Axelsson 2019] to model and analyse SoS, providing incentives to the independently operated and managed constituents. [Kumar 2017] developed a bond graph modelling method for a class of mechatronic SoS, combining the behavioral and the organizational modelling approaches. [Ge 2014] adopted the graph model for conflict resolution methodology to address the problem of multi-stakeholder system portfolio decision analysis encountered in architecting a SoS with desired capabilities.

A suitable model can be developed from various perspectives, such as organization, function, and communication, all of which are crucial for the design, analysis, evaluation, management, and reconfiguration of SoS. organizational viewpoint focuses on the hierarchical structure, roles, responsibilities, and relationships within the organization that interacts with or manages the system. Most previous modelling method focus on the organizational viewpoint [Hela 2021, Mahulkar 2009, Axelsson 2019, Ge 2014], in which the cooperation among SoSs are vividly exhibited but lacks other necessary perspectives. Functional modelling describes the capacity of each SoS and what kinds of functions it undertake in multiple missions. Additionally, communication-based modelling shows the information flow, including data exchange between components and external entities. Notice that preceding research works mainly consider only one point of view, that is to say, the effect of other viewpoints are ignored. Consequently, it's desirable to develop a unified modelling to combine all above mentioned perspectives to facilitate the following discussion.

Among all modelling approaches, graph-based modelling [Ge 2014, Marvin 2014] offers a clearer visual representation of complex relationships and interactions between components, making it easier to understand and communicate the structure

and dynamics of the SoS. They effectively capture and illustrate the intricate dependencies and interactions among various elements within the SoS, which can be challenging to represent using other modelling techniques. Graph models are flexible and can be adapted to represent different types of relationships (e.g., functional, communicational, or organizational) and dynamic changes in the system. In addition, graph models provides scalability, scaling effectively as the number of components and relationships increases. Additionally, graph-based models can integrate with other analytical tools and methodologies, providing more comprehensive viewpoint for SoS modelling.

Nevertheless, for a complicated SoS, the classical graph theory, which represents systems as nodes connected by edges, is often insufficient for modelling the multi-way interactions among diverse CSs. Traditional graphs limit the representation to pairwise relationships, which can lead to oversimplifications and misrepresentations. In contrast, hypergraphs, where hyperedges can connect more than two nodes, offer a more flexible and comprehensive framework for depicting multi-way interactions [Berge 1984, Zhang 2018]. This enhanced capability is crucial for accurately capturing more complex interactions. In this regard, hypergraph is particularly suitable for the modelling of SoS, where the interdependencies among CSs are often multifaceted. Unlike classical graphs, hypergraphs can represent higher-order relationships among multiple components, enabling more comprehensive modelling of SoS. Apart from this, a SoS may exhibit hierarchical structures, where CSs are of different managerial levels, which in turn may be supervised by other CSs. Hypergraphs can naturally accommodate these hierarchical relationships, allowing for multi-level modelling and analysis. This capability is essential for SoS, where different levels of abstraction and granularity are often required.

Various studies have demonstrated the efficacy of hypergraphs in capturing the intricate interactions within SoS across different domains. [Khalil 2012a] utilized the hypergraph to model a class of multi-level SoS with applications in the supervision of intelligent transportation systems. The developed hypergraph model showed effectiveness in detecting and isolating faults in the supervision. [Khalil 2015] developed a hierarchical model of SoS based on hypergraphs and constraint satisfaction problems. The developed model enabled operators to perform a bottom-up analysis of the hypergraphs, identifying the failure of a local system and assessing its impact on other interconnected systems. Meanwhile, a top-down analysis of the hypergraphs allowed for the examination of potential reconfiguration of the entire SoS to ensure that the maximum number of constraints is satisfied. [Jiang 2022] proposed various hypergraphs to present the organizational, communicational and functional structure of SoS, in which the related reconfiguration processes can be expressed by the corresponding hypergraph. Besides, [Jiang 2024b] developed a SoS hypergraph model is developed to depict the time-varying organizational structure of SoS, where the dual of hypergraph was adopted to describe the time-varying bidirectional information transmission among CSs with communication disruption and time-delay. According to the hypergraph modelling, a SoS performance assessment methodology was devised from various viewpoints, including task, energy cost

and communication quality. In addition, an event-triggered scheduling scheme was designed for the resilience reconfiguration of SoS.

In conclusion, hypergraph modelling represents a significant advancement in the field of complex system modelling, offering a robust framework for representing and analysing the multifaceted interactions within SoS. By advancing hypergraph theory and its applications, researchers can unlock new possibilities for enhancing the performance of SoS across various domains.

2.3 Resilience analysis

The term “resilience” derives from the Latin word "resiliere”, which means “to bounce back”. Resilience commonly refers to the capacity of a system to restore its normal status after a disruptive event. Until now, there is no unified definition for system resilience yet. For instance, the resilience in [Haimes 2009] was defined as the capability of a system to endure significant disruptions while maintaining acceptable levels of performance, and to recover within a reasonable time frame and at manageable costs and risks. [Vugrin 2010] described the system resilience as its ability to effectively minimize both the extent and duration of deviations from its intended performance level in the context of a specific disruptive event or series of events. Moreover, a part of definitions are similar, though many overlap with several existing concepts such as robustness, fault-tolerance, flexibility, survivability, and agility, among others.

Based on the observation of existing studies, we can witness two essential elements in most existing system resilience definitions, that is, the negative effect on the system caused by disruptions and recovery efforts to restore the system performance. Although there is no unique insight to define the system resilience, some similarities can be observed from the existing literature. Most previous works [Sheffi 2005, Bhamra 2011, Park 2013, Hosseini 2016] emphasised the capacity of a system to withstand, absorb, adapt to, and recover from disruptions or adverse conditions. This includes the ability to maintain essential functions and structure despite facing internal and external challenges, to adapt and transform in response to changing conditions, and to recover quickly and efficiently after disturbances. Additionally, resilience encompasses several key attributes such as robustness, redundancy, resourcefulness, and rapidity.

In modern society, system resilience is becoming increasingly important and complicated. With the rise of interconnected systems, network resilience has become a critical area of research. Increased interdependence modern systems are more complex and interconnected than ever before, which implies that a failure in one part of the system can have cascading effects, impacting other parts. [Albert 2000] explored the resilience of complex networks, highlighting the role of network topology in determining the robustness and vulnerability of networks to attacks and failures. Additionally, due to the development of digital technology, cyber security threats have turned to be more prevalent and sophisticated. Cyber-

attacks can disrupt critical services and cause widespread damage. The resilience of cyber systems are essential to protect against these threats and ensure continuity of operations [Linkov 2013, Nazir 2015, Zou 2021] . Moreover, the increasing complexity of system results in more involved effect caused by failures, malfunctions and disruptions [Pumpuni-Lenss 2017, Fraccascia 2018].

Advances in technology, data analytics, and interdisciplinary collaboration have led to innovative strategies and tools for enhancing system resilience. Firstly, the design of resilient structure is essential for building and maintaining system resilience. Collaborative strategy that involves interdependence are particularly effective in addressing the complex nature of modern systems. In [Linkov 2018], a risk governance approach has been addressed, integrating quantitative experimental data with qualitative expert insights for characterizing and balancing the risks, benefits, costs, and societal implications of emerging technologies. [Grigorian 2019] developed a methodology for earthquake-resilient structure, following principles of full cycle performance control and embracing a holistic approach to design-led analysis. Secondly, developing effective metrics and assessment frameworks is crucial for measuring and improving resilience. These tools help identify vulnerabilities and prioritize resilience-building measures. [Mebarki 2017a, Mebarki 2017b] developed a theoretical model and established metrics for assessing structural resilience, focusing on the general behaviour of physical systems after experiencing damage from natural or industrial hazards. [Cimellaro 2010] presented a set of quantitative measures for assessing the resilience of health care facilities subjected to earthquakes, providing a standardized way to evaluate and compare resilience. [Francis 2014] proposed an alternative metric for measuring resilience that incorporates knowledge uncertainty as an integral input into evaluating system resilience.

Taking one step further, resilience reconfiguration involves a multi-faceted approach that integrates organizational strategies, disaster response and planning, risk managements to create systems that are not only capable of bouncing back from shocks but are also adept at foreseeing potential threats. Recent research in resilience reconfiguration has focused on developing advanced optimization models and algorithms. [Kavousi-Fard 2018] developed an effective model for microgrid optimal scheduling with dynamic network reconfiguration. The proposed network reconfiguration can effectively modify local power flow, offering opportunities to reduce distribution network losses during grid-connected operation and minimize potential load curtailments during islanded operation. In [Lei 2020], the reconfiguration-related optimization problem subject to radiality constraints was proposed for distribution system with applications in microgrid formation. Another key area of research is the application of AI and Machine Learning (ML) techniques to predict and respond to system disruptions. In this regard, [Zheng 2020] proposed an uncertainty set construction network based on deep neural networks to adaptively construct the uncertainty set for distributed generator and loads using historical data. By connecting deep learning and robust optimization, a general robust method for distribution network reconfiguration while hedging against the risk brought by uncertainties in active distribution networks was developed. [Ji 2021] introduced a novel real-time

autonomous dynamic reconfiguration method to reduce the cost of power loss and switch action of distribution network based on the deep learning algorithm, where the method can be decision-making from the historical control dataset and the real-time system state.

2.4 Structural analysis

Structural analysis of a Networked System (NS) involves examining the arrangement, connections, and relationships between components or nodes within a network. This analysis focuses on understanding how the system's topology, that is, the way its elements are connected, affects its overall behaviour, performance, and resilience. The controllability analysis of NSs is a important part of structural analysis, which allows for the precise management and manipulation of its behaviour.

2.4.1 Controllability

Controllability demonstrates the ability to guide a system from any initial state to any desired state within a finite time frame. This fundamental concept in control theory was initially introduced for single systems [Hespanha 2018]. With the development of communication technology, the concept of controllability of NSs, which consist of interconnected and interdependent components. This topic has gained increasing attention and meanwhile introduced new layers of complexity to the analysis of controllability [Tanner 2004, Rahmani 2009, Guan 2016, Guan 2017, Wang 2020].

NSs, including Multi-Agent Systemss (MASs) and SoS, are characterised by intricate interactions among numerous CSs, each potentially differing in dynamics and behaviour. These systems are prevalent in diverse fields such as robotics [Ismail 2018], sensor networks [Raghavendra 2006], and ground vehicle [Xu 2022], where coordinated behaviour and collective dynamics are essential. The heterogeneity of agents and the complexity of their interactions pose significant challenges to the traditional controllability criteria. Understanding and analysing the controllability of NSs are crucial for the effective design and operation of these systems. It involves not only assessing the capability of the entire network to reach a desired state but also devising strategies to manage and utilize the interconnections among agents efficiently. Generally speaking, the controllability analysis of the NSs can be divided into two aspects: the controllability of each sub system and coordination interaction based on network communication. The controllability analysis of NSs focuses on how to maintain the controllability of the whole NS while a part of sub systems are out of external control signals, which helps the users to devise a robust networked control structure.

The controllability analysis on NSs was firstly emerged on MASs with first-order fully actuated agents via algebraic and graph-theoretic methods, such as [Rahmani 2009, Guan 2017]. In [Rahmani 2009], all agents were divided into leader and follower groups, in which the leader was viewed as the exogenous input. All agents updated their state based on the information from its neighbours following

a consensus algorithm. [Guan 2017] supposed both leaders and followers satisfied first-order dynamics, but only the leaders were subjected to external input signal. The authors demonstrated the controllability of the whole MASs while only a part of agents are under input in virtue of the almost equitable partition in graph theory. Notice that the above two paper only considered homogeneous first-order agents. Moreover, in [Guan 2016], the authors worked on a hybrid MAS with not only first-order but also second-order fully actuated agents. In addition, both continuous and discrete MASs were studied. The authors verified that controllability of communication topology is the necessary condition but not the sufficient condition of the controllability of MAS, which is different from that of homogeneous MASs. The presented theoretical analysis showed the technical difficulties and complexity in the controllability of heterogeneous MASs, even they only studied the mixing of first and second order fully-actuated agents.

The results were extended to MASs [Zhang 2013,Guo 2022] and networked Multi-Input/Multi-Output (MIMO) systems [Zhao 2015, Wang 2016, Hao 2018, Wu 2020] with Linear Time-Invariant (LTI) dynamics. [Zhang 2013] supposed each agents in the MAS is subject to homogeneous LTI dynamics, in which only the leaders are under external input signal. The considered MAS is diffusively Coupled following the consensus protocol. The authors provided provide lower and upper bounds for the controllable subspaces in terms of the distance partitions and the maximal almost equitable partitions, respectively. [Wang 2016] considered a NSs with homogeneous MIMO LTI dynamics and identical interactions, where only a part of nodes are subject to external control input. The authors illustrated that the controllability of such NSs cannot be decoupled into the controllability of individual node-systems and network topology. In addition, it was demonstrated that the controllability of network topology, the controllability and observability of the nodes are necessary condition for the controllability of the NSs under some mild conditions, but none of them are sufficient. The results were further extended to nondiagonalizable topology matrix in [Hao 2018], in which the authors proposed a necessary and sufficient condition for controllability to further observe the effects of the network topology, node-system dynamics, external control inputs, and inner interactions on the controllability of the whole NSs.

Additionally, [Wu 2020] explored the controllability of the deep-coupling dynamical NSs of multiple layers. The nodes in the same layer have the identical LTI dynamics while the nodes in different layers subject to nonidentical system matrices. Both inter-layer and intra-layer couplings have been considered in the interactions among agents. The controllability of NSs was illustrated based on Algebraic methods. While above studies focus on static communication topology graphs, the results were extended to agents with homogeneous LTI dynamics in [Tian 2021] and [Lu 2020] under switching topologies. In [Tian 2021], the authors investigated the controllability of MASs with identical continuous linear dynamics and switching interconnection topologies by virtue of the switching sequence and the constructed subspace sequence. [Lu 2020] further extended the results to sampled-data controller. The authors proved that the controllability property can be preserved if

an unsampled subsystem is controllable and the sampling time is not pathological. Moreover, the group controllability of two-time-scale linear continuous and discrete MASs were investigated in [Long 2018] and [Long 2020] respectively, where the considered MASs are fully actuated or of homogeneous LTI dynamics. [Liu 2023] and [Wang 2020] studied the controllability of MASs with Cartesian product networks and Corona product network, but the agents they considered remain fully actuated.

It is important to note that the reviewed literature assumed homogeneity for each agent or CS within the MASs or NSs, encompassing identical dimensions, dynamics, and coupling matrices. For NSs of heterogeneous LTI dynamics, the controllability of networked MIMO systems with heterogeneous system matrices was studied in [Xiang 2019], though each node still had identical dimensions and inner-coupling matrices. The results were further extended to NSs with nonidentical inner-coupling matrices in [Kong 2022], but the considered NSs still had nodes with invariant dimensions.

In an SoS structure, the CSs have heterogeneous dynamics and diverse dimensions. Additionally, the interactions among agents vary with CSs and their neighbours. These properties bring extra challenges into the controllability analysis of SoS. In the presence of local disruptions, some CSs may be out of control. The controllability helps in the maintaining operations of the overall SoS, which plays an essential role in the SoS resilience. Thus, the controllability of SoSs with heterogeneous CS still remains a topic worth exploring.

2.5 Diagnosis

Diagnosis is the process of monitoring, identifying, evaluating and analyzing issues within a system to ensure its optimal performance and functionality. This process involves detecting faults and evaluating their impacts, which lays the ground for implementing corrective measures to restore or maintain system performance. Effective diagnosis is crucial for maintaining reliability, minimizing downtime, and ensuring that systems operate efficiently. In this regard, AI-based classification and prediction, and data-driven prediction methods play important roles in the diagnosis process, which enhance the ability to detect, classify, and forecast system issues effectively.

2.5.1 AI-based classification and prediction

2.5.1.1 Classification

AI classification, a subfield of machine learning, represents a pivotal advancement in the realm of data analysis and pattern recognition. It involves the development of algorithms and models that can autonomously categorize data into predefined classes based on learned patterns and features. The evolution of AI classification has been driven by the exponential growth of data, increased compu-

tational power, and sophisticated algorithmic innovations, leading to transformative applications across diverse domains, such as disease diagnosis [Mei 2020] and risk stratification [Jiang 2023b], customer segmentation [Kasem 2024], quality control [Lughofer 2009], object detection [Jena 2022] and performance classification [Yılmaz 2019]. AI classification is foundational in enabling systems to make informed decisions and predictions without explicit human intervention. This capability is harnessed through the meticulous process of training models on extensive datasets, where the algorithm discerns the intricate relationships and characteristics within the data. The resultant models can then be deployed to classify new, unseen data with remarkable accuracy and efficiency. Key methodologies employed in AI classification include Decision Trees (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (KNN) and Deep Learning (DL) through Neural Network (NN), each offering unique strengths suited to specific types of data and classification tasks.

A DT is a hierarchical model resembling a flowchart, where each internal node denotes a test on an attribute [Kotsiantis 2013]. Each branch represents the possible outcomes of that test, and each leaf node signifies a class label (the decision reached after evaluating all attributes). The sequences of nodes from the root to the leaf nodes illustrate the classification rules. DTs can effectively handle structured data, including both categorical and numerical features. Besides, they are capable of capturing nonlinear relationships within the data. Their primary advantages include ease of interpretation and visualisation, no requirement for feature scaling, and robustness to outliers, making them an attractive choice for various applications, such as power systems and marketing. For instance, [Vanfretti 2020] employed DTs to classify various operating conditions concerning voltage stability in power systems. The proposed method leveraged a novel and flexible classification criterion that identifies operating conditions near or within regions of voltage instability, incorporating operational requirements. [Han 2012b] utilized DTs to extract significant parameters related to long-term value, credit, and loyalty, creating a practical customer value evaluation system. Applied to telecom operators in China, the model demonstrated high accuracy in identifying high-value customers. Nevertheless, DTs are prone to overfitting, particularly when they become overly complex, and can exhibit instability due to their sensitivity to small changes in the data [Sharma 2016]. Additionally, they may demonstrate bias towards dominant classes in imbalanced datasets and struggle with computational efficiency when applied to large datasets. The greedy nature of DTs, which optimizes local decisions without guaranteeing a globally optimal structure, further limits their capacity to capture intricate interactions between features.

To mitigate these drawbacks, ensemble methods such as RFs, are commonly employed [Fratello 2018]. RFs are an ensemble learning algorithm used primarily for classification. The method involves constructing multiple decision trees during training and outputting the mode of the classes [Fawagreh 2014]. By training each tree on a different subset of the data and features, random forests reduce overfitting and variance, improving the model's accuracy and robustness. The algorithm

can handle large datasets with many features and provides insights into feature importance. RFs classification has numerous applications across various domains due to its robustness and accuracy, such as ecology and land-cover. For example, [Cutler 2007] explored the application of RFs in ecological studies. The authors highlighted the method's accuracy in handling complex ecological data, including species distribution and habitat classification. They demonstrated the advantages of RFs over traditional statistical methods, such as its ability to manage large datasets, handle various data types, and provide measures of variable importance. [Rodriguez-Galiano 2012] evaluated the performance of RFs algorithm in categorising land-cover types using remote sensing data. It demonstrated that RFs excelled in accuracy and robustness compared to other classifiers. The study supported the adoption of RF as a robust and effective method for land-cover classification, highlighting its practical advantages in remote sensing applications. However, RFs can be computationally intensive, requiring significant resources for training and prediction, and the resulting model can be complex and less interpretable compared to individual decision trees.

Additionally, SVM is a powerful classification algorithm that aims to find the optimal hyperplane that separates data points of different classes with the maximum margin. It is particularly effective for high-dimensional spaces and cases where the number of dimensions exceeds the number of samples [Cervantes 2020]. SVM is well-suited for binary classification tasks and can handle non-linear relationships by using kernel functions to map data into higher-dimensional spaces. The primary advantages of SVM include its effectiveness in high-dimensional settings and robustness to overfitting, especially in scenarios with clear margins of separation. SVM is one kind of versatile and powerful classifiers with applications across various domains due to their effectiveness in handling high-dimensional data and complex decision boundaries. For example, [Huang 2007] studied the application of SVM in credit scoring, aiming to improve the accuracy and reliability of predicting creditworthiness. The study demonstrated that SVM achieved an identical classificatory accuracy with relatively few input features. Nevertheless, SVM can be computationally intensive and may perform poorly with very large datasets or when the classes are not well-separated. Additionally, choosing the right kernel and tuning hyperparameters can be challenging, impacting the model's performance.

Moreover, KNN is a straightforward, instance-based classification algorithm that classifies data points based on the majority class of their k nearest neighbours in the feature space. During classification, the algorithm calculates the distance between the new data point and all training instances, identifies the k closest points, and assigns the class label based on the majority vote among these neighbours [Peterson 2009]. It is particularly effective for problems with clear, local patterns and works well with small to medium-sized datasets where class boundaries are not explicitly defined by a model. KNN is intuitive and easy to implement, with the main advantages being its simplicity, ease of understanding, and ability to adapt to changes in data without needing a retraining phase [Bhatia 2010]. KNN classification is a simple, intuitive, and versatile algorithm used in a wide range of

applications across various domains. For instance, [Li 2014] proposed an intrusion detection system for wireless sensor networks using the KNN classification algorithm. The proposed system could distinguish abnormal nodes from normal ones by analysing their unusual behaviours, enabling efficient and rapid intrusion detection by enhancing the wireless ad hoc on-demand distance vector routing protocol. However, KNN classification still has several disadvantages, including high computational costs during prediction as it requires calculating distances to all training samples, sensitivity to irrelevant or redundant features, and reduced performance with large datasets due to its reliance on distance metrics, which can be affected by the curse of dimensionality.

DL is a subset of machine learning based on NNs with representation learning. The term "deep" highlights the presence of multiple layers within the network [LeCun 2015]. On the one hand, it is particularly effective for tasks involving image and speech recognition, natural language processing, and time-series forecasting. The advantages of DL include high performance, automatic feature extraction, scalability, and versatility across various domains. On the other hand, DL requires large amounts of labeled data, significant computational resources, and extensive training time, and often results in models that are complex and difficult to interpret. The NN-based classification has wide applications over various disciplines. For instance, [Shajin 2023] presented a hierarchical DL NN model for classifying brain tumours. The proposed framework utilized a multi-layered neural network to extract and analyse features at different levels of abstraction, enhancing classification accuracy. [Urban 2018] provided a comprehensive overview of how four different research groups approach the use of NN for acoustic modelling in speech recognition systems. The paper synthesised their shared insights and methodologies, emphasising how NN improves the accuracy and robustness of speech recognition by modelling complex acoustic features more effectively than traditional methods.

2.5.1.2 Prediction

AI-based prediction involves using advanced artificial intelligence techniques to forecast future outcomes based on historical data. AI-based prediction enables accurate and insightful forecasts across various fields like stock market prediction in finance [Ray 2018], the drug response prediction [Adam 2020], export sales forecasting [Sohrabpour 2021] and weather prediction [Haupt 2018]. By applying algorithms such as Linear Regression (LR), which models linear relationships; Polynomial Regression (PR) for capturing non-linear patterns; DT regression and RF regression for structured decision-making and ensemble accuracy; Support Vector Regression (SVR) for handling complex data; KNN for intuitive predictions; and DL based on NN for modelling intricate patterns in large datasets. These methods leverage historical data to uncover patterns and make informed predictions about future events or trends.

LR is a fundamental algorithm in AI prediction used to model the relationship between a dependent variable and one or more independent variables through

a linear equation. It is most effective with numerical data where the relationship is approximately linear and the data are free from multicollinearity and extreme outliers [Seber 2012]. Its main advantages include simplicity, computational efficiency, and ease of interpretation, making it suitable for applications such as economic forecasting, real estate valuation, and sales prediction [Maulud 2020]. For instance, [Gopalakrishnan 2018] explored the application of linear regression to forecast sales values in the context of e-commerce, helping them increase their profits, enhance their brand, and stay competitive with market trends by also generating customer satisfaction. However, LR has limitations, including the assumption of linearity, sensitivity to outliers, and potential issues with multicollinearity.

To overcome the deficiency, PR serves as an extension to LR used to model the relationship between a dependent variable and one or more independent variables by fitting a polynomial equation to the data [Ostertagová 2012]. This technique is useful for capturing non-linear relationships in data, making it a powerful tool for AI-based predictions when linear models are insufficient. PR is suitable for datasets where the relationship between the independent and dependent variables is non-linear but can be approximated by a polynomial function. The data should have a sufficient range and variety of values to adequately fit the polynomial curve. It works best when there is a single predominant relationship that can be expressed as a polynomial, avoiding high multicollinearity among variables. The technique's main advantages include its flexibility in modelling complex relationships and its relative interpretability compared to other non-linear models. However, it also poses risks such as overfitting, increased model complexity, and unreliable extrapolations. Applications of RF range from weather prediction to soil permeability and compaction prediction, as demonstrated in various studies and academic literature. For example, [Zaw 2009] explored the application of PR to predict rainfall in Myanmar. The authors utilized historical rainfall data and relevant climatic factors to develop a PR model that captures the non-linear relationships inherent in weather data, where the PR prediction model presented better accuracy compared with the LR prediction model. [Ahangar-Asr 2011] presented a study on PR using to predict soil permeability and compaction characteristics, which are critical parameters in geotechnical engineering. The models trained from experimental data were able to capture many physical relationships between soil parameters. Meanwhile, the models are also capable of represent ing the degree to which individual contributing parameters affect the maximum dry density.

Additionally, DT regression is a machine learning technique used to predict continuous outcomes by splitting data into subsets based on feature values. Each internal node in the tree represents a decision based on a feature, while each leaf node represents a continuous value prediction [De Ville 2013]. DT regression is well-suited for structured data and can handle both numerical and categorical variables. The applications of DT regression has been reported from drug reaction prediction to heart disease prediction. In this regard, [Hammann 2010] investigated the use of decision tree algorithms to predict adverse drug reactions, aiming to enhance drug safety and patient outcomes. By employing DT induction, the authors identified

the chemical, physical, and structural properties of compounds that make them likely to cause adverse drug reactions. The models demonstrated high predictive accuracies for allergic, renal, central nervous system, and hepatic adverse drug reactions. [Maji 2019] presented the application of DT algorithm in the heart disease prediction, aiming to improve early diagnosis and patient care. The authors showed that models trained can achieve high predictive accuracy, making them reliable tools for identifying individuals at risk of heart disease. The advantages of DT regression include interpretability, no need for data scaling, and the ability to handle both numerical and categorical data while modelling non-linear relationships [Mienye 2019]. However, DT regression has several disadvantages, such as a tendency to overfit, sensitivity to small data variations, potential bias towards dominant features, and limited ability to predict beyond the training data range. Additionally, large trees can become complex and difficult to interpret.

Besides, RF regression is an ensemble learning method used in AI prediction to estimate continuous outcomes by constructing multiple decision trees and averaging their predictions [Cutler 2012]. It is well-suited for structured data with both numerical and categorical features and can handle large datasets with complex, non-linear relationships. RF regression offers robust performance and versatility by combining multiple decision trees to reduce overfitting and handle non-linear relationships with minimal data preprocessing. It provides insights into feature importance and generally performs well across diverse datasets. However, it can be computationally intensive, challenging to interpret due to its complexity, and may require significant memory. Additionally, its effectiveness might decrease with sparse data and it can exhibit bias towards variables with many levels, requiring careful consideration of hyperparameters for optimal results. Applications of RF regression range from battery capacity estimation to crop yield prediction. For instance, RF regression was utilized in [Li 2018] for online capacity estimation of lithium-ion batteries. The proposed models effectively learned the dependency of battery capacity on features extracted from charging voltage and capacity measurements, showing high accuracy in the evaluation of health states of different batteries under varied cycling conditions. Besides, [Jeong 2016] presented the applications of to predict crop yield responses to climate and biophysical variables at global and regional scales for wheat, maize, and potato, using multiple linear regression as a benchmark for comparison.

In addition, SVR extends the principles of SVM to regression problems. It works by mapping input features into a high-dimensional space and finding a hyperplane that best fits the data while maintaining a margin of tolerance for errors [Zhang 2020a]. SVR aims to minimize the prediction errors within a specified margin while ensuring that the model is as flat as possible, which helps in managing overfitting and achieving robust generalization. It excels with datasets featuring high dimensionality, complex non-linear relationships, and a moderate size, where capturing intricate patterns is crucial. It is effective in handling noisy data and finding a balance between model complexity and training error, making it well-suited for scenarios where a clear margin of tolerance for errors can be defined. Despite its effectiveness in capturing complex patterns and handling high-dimensional data,

SVR can be computationally intensive and sensitive to the choice of hyperparameters and kernel functions, making it essential to carefully tune these parameters for optimal performance. SVR is widely applied in the prediction across various domains, such as stock market prediction and project control forecasting. For example, SVR was adopted in [Zhang 2020a] for the prediction of Volatile Stock Market. The authors proposed an extension of the standard SVR incorporating margins adaptation. The experimental results presented the high accuracy of the model trained in the prediction for the Hang Seng Index. [Wauters 2014] presented the application of SVR in the project control forecasting to predict key project performance metrics, such as cost and schedule. By applying SVR to historical project data, including project duration, budget, and resource allocation, the study demonstrated that SVR effectively forecasts these metrics with high accuracy.

Moreover, KNN Regression is a simple yet effective machine learning technique used for predictive modelling. In KNN regression, the prediction for a given data point is made by averaging the target values of its k nearest neighbours in the feature space [Song 2017]. The proximity of these neighbours is determined using distance metrics, such as Euclidean distance. KNN regression is non-parametric and does not assume any underlying data distribution, making it flexible and easy to understand. KNN regression is ideal for small to medium-sized datasets with low to moderate dimensionality, where capturing local patterns is essential and a meaningful distance metric can be defined. It effectively handles cases where relationships between features and target values are localized rather than global. However, KNN regression can become computationally expensive as the dataset grows, and its performance heavily depends on the choice of k and the distance metric. Additionally, it may struggle with high-dimensional data due to the curse of dimensionality, where distance calculations become less meaningful. The applications of KNN regression in prediction have been reported in multiple domains, such as gene function prediction and energy aware consolidation. For instance, [Yao 2006] introduced a general framework for gene function prediction in terms of the KNN algorithm, combining regression techniques to infer a similarity metric as a weighted combination of a set of base similarity measures. The proposed algorithm outperformed traditional prediction methods in terms of accuracy and reliability, offering valuable insights for functional genomics research. [Farahnakian 2013] proposed a dynamic virtual machine consolidation algorithm based on KNN regression to minimize the number of active physical servers in a data center, thereby reducing energy costs. The authors presented that the consolidation method can identify when a host becomes over-utilized and when a host becomes under-utilized.

Furthermore, NN is also utilized to forecast outcomes by learning from data through multiple layers of interconnected nodes [Adya 1998]. NN models, such as feedforward NNs [Bebis 1994], convolutional NNs [Li 2021], and recurrent NNs [Medsker 1999], are trained to automatically extract and learn hierarchical features and complex patterns from large datasets. This capability allows for highly accurate predictions in various domains, including image recognition, speech processing, and time-series forecasting. NN-based prediction leverages the depth and complexity of

these models to handle non-linear relationships and high-dimensional data. It is suitable for high-dimensional and complex data types, such as images, sequences, and structured data, thanks to its ability to automatically extract and learn features. Though it demands significant computational power and extensive data, and can be challenging to interpret due to its intricate structure. NN regression has a wide range of applications across various fields due to its ability to model complex, nonlinear relationships between inputs and outputs. For example, [Cigizoglu 2006] explored the application of NN in the prediction of river sediment yield. The NN were trained using daily river flow and suspended sediment data belonging to Juniata Catchment in USA and the NN-based estimations were found significantly superior to conventional method results. [Schlechtingen 2011] evaluated and compared the effectiveness of NN and traditional regression methods for detecting faults in wind turbines. The study found that neural network models outperformed regression-based methods, offering higher accuracy and better handling of complex, nonlinear relationships in the data.

2.5.2 Data-driven based prediction

In the age of information, the ability to harness and analyze vast amounts of data has transformed various fields, from business to healthcare, finance to environmental science. Data-driven prediction is a crucial aspect of this transformation, leveraging sophisticated algorithms and computational techniques to forecast future events and trends based on historical data. Unlike traditional methods that rely on theoretical models and assumptions, data-driven prediction utilizes empirical evidence to generate insights and make informed decisions [Clauset 2017]. By employing techniques such as AI-based prediction [Woldaregay 2019] and time series analysis [Box 2015], data-driven prediction can uncover patterns, identify correlations, and generate accurate forecasts that drive strategic planning and operational efficiency. As data continues to grow in volume and complexity, the advancements in prediction methodologies become increasingly essential for addressing challenges, optimizing processes, and gaining a competitive edge in various domains. This approach not only enhances decision-making but also fosters innovation by providing actionable insights derived from real-world data. Due to the AI-based prediction models have already been introduced above, we mainly presented time series analysis in this part.

Time series analysis is a pivotal technique in data-driven prediction, particularly for forecasting and understanding data points collected sequentially over time [Weigend 2018], such as, financial metrics, sales figures, weather observations, and operational data. This approach is essential for analyzing trends, seasonal patterns, and cyclical behaviours within temporal datasets. By examining how a variable evolves over time, time series analysis provides insights into underlying patterns and future movements. Additionally, there is a risk of overfitting to historical data, and advanced models may be computationally demanding. Despite these challenges, time series analysis remains a robust tool for making informed predictions

and understanding temporal dynamics. The applications of time series analysis in prediction has been reported in various domains. For example, [Kraft 1977] examined the factors influencing the prices of common stocks using time series analysis techniques. The study identified key determinants such as earnings, dividends, interest rates, and macroeconomic indicators, analyzing their impact on stock price movements over time. Besides, [Martín 2010] presented a methodology for forecasting global solar irradiance using time series analysis to enhance the planning of energy production in solar thermal power plants. Moreover, combining time series analysis with AI-based prediction, [Cao 2003] introduced an enhanced forecasting method for financial time series using a SVM with adaptive parameters. The adaptive SVM model was tested on various financial time series datasets, demonstrating superior performance in forecasting accuracy compared to standard SVM.

Moreover, in order to improve prediction accuracy and capture complex patterns, the multi-scale time series prediction was proposed to analyze and model time series data across different time scales. This technique involves examining data at various temporal resolutions, such as daily, weekly, and monthly, to understand and predict trends, seasonal variations, and other temporal dynamics more comprehensively [Nawroth 2006]. Due to the features are extracted from various scales, multi-scale time series is capable of learning from different aspects of the time series data. This helps in identifying patterns that may not be apparent when considering only a single time scale. In addition, combining insights from multiple scales can provide the model with robustness and accuracy. It helps in addressing issues like seasonality and long-term trends more effectively. [Jiang 2017] introduced an innovative method that enhances the analysis of financial multi-scale time series by combining permutation entropy (PE) with the Gini-Simpson index, providing valuable information for understanding the inner mechanism of financial markets. Besides, [Zheng 2022] presented a multi-scale time-series dataset to support the development of data-driven ML approaches for the reliable operation of future electric grids. The authors showed that this dataset will drive practical ML research in safety-critical systems, while also allowing ML researchers to advance the decarbonization of energy sectors.

2.6 Decision making

Decision making is the process of selecting a course of action from multiple alternatives to achieve a specific goal or solve a problem. It involves evaluating options, considering potential outcomes, and making informed choices to optimize results. Crew dispatch is a specific application of decision-making within operational contexts where personnel management is crucial. It involves allocating crew members to tasks, shifts, or assignments to ensure efficient and effective operations.

2.6.1 Crew dispatch

The crew dispatch problem is a critical challenge in the maintenance and restoration of LSS, involving the optimal assignment, scheduling and routing of crew members to ensure efficient operations and achieve restorations facing disruptions. This problem is ubiquitous across various industries, including power distribution network [Arif 2018, Chen 2018, Lei 2019, Zhang 2020b, Wang 2024, Pang 2024], logistics [Desaulniers 1998, Li 2020], road transport [Drexl 2013], railways [Heil 2020], aviation [Díaz-Ramírez 2014, Ahmed 2018] and public transit [Yang 2009, Malucelli 2019], where crews are essential for the operation of services and repair. Facing malfunctions, damage, emergencies, and disasters encountered in reality, crew dispatch plays a critical role in the restoration and maintenance of LSS. The complexity of crew dispatch arises from the need to consider multiple constraints and objectives, such as minimizing operational costs and lost caused by disruptions, adhering to labour regulations, maximising crew utilization, and ensuring the satisfaction of service requirements.

In the past decades, we have witnessed a number of research dedicated to developing advanced methods and optimization algorithms to solve the crew dispatch problem. Classical approaches often relied on mathematical programming techniques, such as dynamic programming and mixed-integer linear programming. For example, [Carvalho 2007] formulated the crew dispatch as a dynamic programming problem to minimize energy not supplied in the restoration of large-scale distribution outages. [Duque 2016] investigated the crew scheduling and routing in the emergency repair of a rural road network damaged by the occurrence of a natural disaster with dynamic programming, incorporating travelling salesman problem. [Kim 2018] studied the repair crew scheduling problem for short-term disasters, in which aspects of damage vary at certain times. To minimize the weighted sum of total damages caused in isolated areas and completion time of a repair crew, an optimization model was proposed in terms of the mixed integer programming, which can be reduced to a variation of the travelling salesman problem with simple modifications. [Van Hentenryck 2015] discussed the repair crew routing in transmission systems, where a deterministic two-stage approach was proposed to separate the routing and restoration models. In the first stage, a restoration ordering problem was addressed using mixed-integer linear programming. The objective was to determine an optimal repair sequence to maximise the restored loads. The second stage involved formulating the routing problem as a constraint programming model, which was solved using neighbourhood search algorithms and randomised adaptive decomposition.

Moreover, taking into account the uncertainties, [Arif 2018] addressed the repair crew routing and restoration problem using the two-stage stochastic mixed-integer linear programming, incorporating flow network and spanning tree methods. The process began with dispatching repair crews to address the damaged components and the next stage involved restoring the distribution system through the use of distributed generators and system reconfiguration. Notice that in [Arif 2018], there was only one type of crews. The result was accordingly expanded by [Chen 2018],

who included various types of sites and agents, such as operating, repair, and energization sites, each with distinct routing models and depot capacity constraints. The authors solved the distribution service restoration in the framework of a synthetic mixed-integer linear programming including switch operation, crew dispatch, and component repair. The integrated model also took into account the skill sets required for different types of crews, based on the universal routing model. [Lei 2019] added mobile power source assignment into the optimization, co-optimizing crew and power source dispatch with priority-weight and integration. The proposed model integrated various timescales for distribution system restoration and repair crews/mobile power source dispatch, as well as the interconnection between transportation and power networks. [Zhang 2020b] combined maintenance and restoration crew dispatch in a co-optimization framework, using an event-based model with variable time steps. To address the challenges related to timescales in the model formulation, technical constraints for system operation are assessed at each energization step instead of at each time step. Furthermore, [Wang 2024] formulated the resilience-driven dispatch problem of mobile power sources and repair crews in a decentralised framework, utilizing a hierarchical multi-agent reinforcement learning method.

Note that the routing and restoration models in the aforementioned works do not adequately capture the resource and capability distribution in a SoS with diverse CSs and heterogeneous crews possessing varying capacities. In power distribution network restoration, the roles of crews are limited, as is the total number of malfunction scenarios at all damaged sites. That's the reason why studies such as [Arif 2018, Lei 2019, Wang 2024, Pang 2024] , and others considered only homogeneous crews. Although [Chen 2018, Zhang 2020b] further classified crews into different clusters based on their roles in operations, repair, energization, and maintenance, they had to employ different routing models for each group. In a scalable SoS with multiple crews, each crew may possess various types of capacities, and each CS might face diverse malfunction scenarios. Therefore, it is impractical to simply divide crews into functional groups and develop numerous models for all clusters. Addressing the capacity distribution problem while integrating it with crew dispatch and restoration remains an area that requires further exploration.

2.7 Conclusion

This chapter provides a comprehensive exploration of the current state-of-art methodologies and concepts related to SoS analysis, especially the approaches that will be utilized in the resilience enhancement.

It begins with an overview of SoS, emphasizing the complexity and interconnected nature of modern systems that consist of multiple, independent CSs working together towards a common goal. This section introduces the characteristics, properties, development and applications of SoS, highlighting the challenges inherent in managing such complex structures and interactions, particularly when they face internal or external disruptions.

Following this, graph-based and hypergraph-based modelling techniques are introduced as essential tools for representing the intricate relationships within SoS. These modelling approaches facilitate a deeper understanding of the interdependencies between CSs, enabling more effective analysis and management strategies that account for multiple levels and interactions.

The discussion on resilience analysis underscored the importance of maintaining system functionality in the face of failures. The resilience analysis can be decomposed into the resilient structure, evaluation and resilience reconfiguration strategy. AI techniques are increasingly used to predict and respond to disruptions, offering dynamic reconfiguration solutions that help systems quickly adapt to changes, reduce operational costs, and enhance overall resilience.

Structural analysis, particularly focusing on controllability, was explored as a crucial aspect of SoS devise. Controllability of NSs, including SoS and MASs, determines the extent to which a system can stay operational in the presence of disruptions in a part of CSs. This analysis aims to understand how the system's topology impacts its overall behaviour, performance, and resilience, which is crucial for enabling precise management and manipulation of the system's behaviour.

The diagnosis section delves into methods for monitoring and detecting performance degradation within SoS. Emphasis was placed on AI-based classification and prediction, which leverages advanced algorithms to classify system states. Data-driven prediction approaches were also discussed, showcasing how real-time data can be used to anticipate system disruptions and inform proactive maintenance strategies.

The chapter concluded with a focus on Decision Making, particularly strategies for resilience reconfiguration. This section addressed the decision-making frameworks that guide system adjustments, such as dispatching repair crews effectively under varying disruption scenarios. The Crew Dispatch subsection specifically highlighted the optimization of resource allocation in response to unpredictable system failures, demonstrating the practical application of theoretical models in real-world settings.

In summary, this chapter synthesized key concepts and methodologies essential for understanding and enhancing the resilience of complex SoS. By integrating structural analysis, diagnosis, and decision-making, the chapter established a foundational framework for developing resilient SoS capable of maintaining operational stability amidst disruptions. The insights gained from this review provide a critical basis for the subsequent chapters, which will further explore the implementation of these strategies in enhancing system resilience and performance.

Chapter 3

Modelling of multi-level collaborative heterogeneous systems

Contents

3.1 Management decomposition of multi-level SoS structure . . 35
3.2 Graph-based communication and organization topology . . 37
3.3 Hypergraph-based communication and organization topology 40
3.4 Hybrid graph/hypergraph-based communication and organization topology 42
3.5 Nonlinear and linear dynamic component system modelling 45
3.5.1 Nonlinear component system modelling 45
3.5.2 Linear component system modelling 45
3.6 SoS resilience metric 46
3.7 Maier's Properties 48
3.8 Conclusion 49

Based on the state of art presented in Chapter 2, especially the graph-based and hypergraph-based modelling, we further explore the modelling of multi-level SoS in this Chapter. Three kinds of modelling methods are developed based on the graph and hypergraph theory, which are suitable for different research topics in terms of characteristics of graph and hypergraph. Above all, to depict the hierarchical organization structure of the multi-level SoS, the management decomposition will be introduced firstly.

3.1 Management decomposition of multi-level SoS structure

The considered SoS model exhibits a multi-level structure, wherein level 0 is designated as the physical level, consisting solely of PCSs. All subsequent higher levels are designated as managerial levels, consisting solely of MCSs. The MCSs are supervisors embedded with computers. Throughout this paper, we propose the following assumptions based on the *management decomposition* in [Jiang 2022]

Assumption 3.1.1 *(SoS management decomposition) For a n-level SoS, PCSs exist only at the physical level* 0*, while MCSs are present at the managerial levels* k*, where* $k = 1, 2, ..., n$*. Each MCS of managerial level* n *can only directly transmit/receive information to MCSs in managerial levels* n *and* $n-1$*. Each MCS of managerial level* $k, k = 2, 3, ..., n-1$ *can only directly transmit/receive information to MCSs in managerial levels* k*,* $k-1$ *and* $k+1$*. Each MCS of managerial level* 1 *can only directly transmit/receive information to MCSs in managerial level* 1 *and level* 2*, as well as PCSs in physical level* 0*. Each PCS of physical level* 0 *can only transmit/receive information to MCSs in managerial level* 1 *and other PCSs in physical level* 0*.*

Assumption 3.1.1 outlines the hierarchical structure and information flow within a multi-level SoS. It specifies the interactions of PCSs and MCSs across different levels of the hierarchy. Assumption 3.1.1 ensures that each MCS handles specific scopes of supervision and coordination, which simplifies the overall management of the SoS. By limiting interactions to adjacent levels, the hierarchy helps in maintaining clear communication pathways and reduces the risk of information overload or miscommunication.

Assumption 3.1.2 *(Streamlined mission management) The* j*-th MCS in managerial level* 1 *supervise only one basic mission, noted as the* j*-th mission. If the* i*-th PCS is supervised by the* j*-th MCS, then it functions in the* j*-th mission.*

Assumption 3.1.2 guarantees that resources can be allocated more efficiently since each MCS in the managerial level 1 controls resources specifically for its mission. There is less risk of resource conflicts or inefficiencies arising from shared or competing demands. In addition, the MCSs in higher levels further monitor and control the MCSs in lower levels to ensure cooperation and avoid conflicts among MCSs.

Assumption 3.1.3 *(Intra-layer information sharing among MCSs) MCSs in the same level can exchange information with each other.*

The intra-layer information sharing in Assumption 3.1.3 supports a decentralized management approach. Instead of relying on a single central authority, MCSs at the same level can make decisions collaboratively based on shared information. Additionally, when MCSs exchange information, they can align their strategies and operations, ensuring that their actions do not conflict with each other. This consistency is vital for maintaining the overall coherence of the SoS.

Assumption 3.1.4 *(PCSs monitoring) Each PCS in physical level* 0 *must be supervised by at least one MCS in managerial level* 1*. Each MCS in managerial level* 1 *must supervise at least one PCS.*

Assumption 3.1.4 ensures that all physical operations are conducted under managerial oversight, reducing the risk of errors, inefficiencies, or unauthorized activities.

Based on the management decomposition multi-level SoS model, we want to visualize the relationships among CSs. Therefore, we firstly introduce the graph-based topology to describe the communication and organization of the SoS. Graph-based topology offers several advantages due to its inherent ability to represent complex relationships, interactions, and dependencies between different components within an SoS.

3.2 Graph-based communication and organization topology

As one type of network topology, the graph-based topology is introduced to describe the physical and logical structure of a SoS as well as how data flows insides the SoS (among the CSs). There are several different logical and physical topologies to build a secure, robust, and easily maintainable topology. The most popular configurations (see in Fig. 3.1) include:

- **Point-to-Point Topology:** As the simplest topology, it involves connecting two nodes or devices using a common link. This type of topology works with the advantages, such as high bandwidth and speed, low latency, easy maintenance, etc.
- **Bus Topology:** Also known as backbone topology, it connects all of the devices or nodes to one primary line. It is easy to connect or remove devices in this topology without affecting any other device. Bus topology offers the benefits, such as easy implementation, cost efficiency, etc.
- **Ring Topology:** The devices or nodes in ring topology are connected in a circular manner, forming a closed loop. In this topology, each device is connected to exactly two other devices, creating a continuous pathway for data transmission. The advantages of using this topology include reduced data collision, high data transfer speed, etc.
- **Star Topology:** As a common used topology, all nodes are individually connected to a central connection point. Star topology improves network security by preventing data from passing through every device. Some of the other advantages include centralized control, easy scalability and reconfiguration, cost-efficiency, etc.
- **Tree Topology:** Tree topology consists of a parent-child hierarchy in which each node is either directly or indirectly connected via bus networks. Nodes branch linearly from a root node, and two connected nodes share only one interconnection. The advantages of using tree topology include easy maintenance, extended distance network coverage, limited data loss, etc.
- **Mesh Topology:** Dedicated point-to-point links connect each device to the other, which allows data to be transferred directly between the two devices

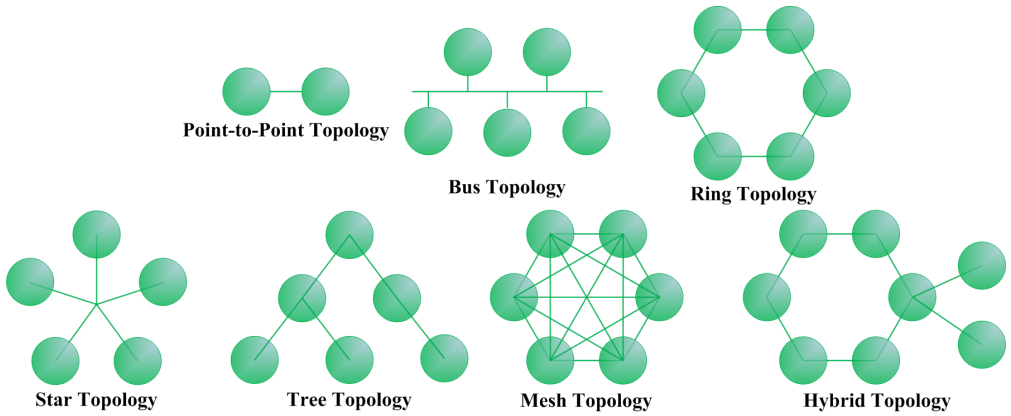


Figure 3.1: Communication/Network Topology Types.

without flowing through other devices in the same topology. Its advantages include fast communication speed, stronger privacy, better security, reduced channel congestion, etc.

- **Hybrid Topology:** Hybrid topology combines at least two other types of topologies. The advantages of using hybrid topology include improved flexibility, increased reliability, etc.

Graph-based topologies provide an intuitive and visual representation of the various CSs and their interconnections within an SoS. Generally, vertices (nodes) represent individual CSs, while edges represent the interactions and information exchange between these CSs, making it easier to understand the structure and dynamics of the SoS.

The communication and organization topology in this thesis is a hybrid graph topology which contains ring topology, star topology and tree topology. As a hierarchical system, the relationship different levels in SoS can be better described by tree topology. For the managerial levels in SoS, CS in higher levels connects at least one CS in the next lower level which forms the star topology. CSs in the same managerial level also have connections which form ring topology in the two or more adjacent levels. For physical level, topology among PCSs contains ring topology and/or star topology.

A general graph-based communication and organization topology of the m-level SoS model is given below:

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \Gamma)$ denote a directed weighted graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. The vertex set $\mathcal{V}$ satisfies $\mathcal{V} = \mathcal{V}_0 \cup \mathcal{V}_1 \cup, ..., \cup \mathcal{V}_m$, in which $\mathcal{V}_0 = \{CS_{0,1}, CS_{0,2}, ..., CS_{0,m_0}\}$ and $\mathcal{V}_n = \{CS_{n,1}, CS_{n,2}, ..., CS_{n,m_n}\}, n = 1, 2, ..., m$. Vertex $CS_{0,k_0}, k = 1, 2, ..., n$ represents the k_0-th PCS in physical level 0 and vertex $CS_{n,k_n}, n = 1, 2, ..., m$ represents the k_n-th MCS in managerial level n. The edge set $\mathcal{E}$ satisfies $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, where $(CS_{i,k_i}, CS_{j,k'_j}) \in \mathcal{E}$ implies there exists interactions between the k_i-th CS in level i and the k'_j-th CS in level j. In particular, if $i = j$, then $(CS_{i,k_i}, CS_{j,k'_j}) \in \mathcal{E}$ implies the k_i-th CS in level i can transmit information

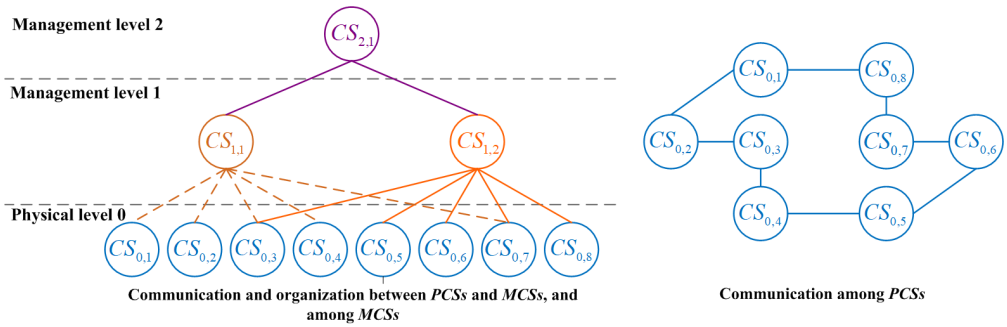


Figure 3.2: An example for graph-based communication and organization topology of SoS modelling.

to the k'_j-th CS in the same level. That is, for CSs in the same level, the interaction among CSs only refers to the communication relationship. If $i < j$, then the interactions $(CS_{i,k_i}, CS_{j,k'_j}) \in \mathcal{E}$ implies the k_i-th CS in level i is supervised by the k'_j-th CS in level j, and the k_i-th CS in level i can transmit information to the k'_j-th CS in level j. If $i > j$, then $(CS_{i,k_i}, CS_{j,k'_j}) \in \mathcal{E}$ implies the k_i-th CS in level i can monitor the k'_j-th CS in level j, and the k_i-th CS in level i can transmit information to the k'_j-th CS in level j. This indicates for CSs in different levels, the interaction among CSs includes both communication and organization relationship.

Let $\Gamma = [\gamma_{ij}] \in \mathbb{R}^{(\sum_{i=0}^{m} m_i)\times(\sum_{i=0}^{m} m_i)}$ signify the weighted adjacency matrix of $\mathcal{G}$. To address the weighted adjacency matrix Γ for the whole SoS, it's necessary to attribute a unified index for each CS. The index of vertex CS_{n,k_n} (the k_n-th CS in level) is defined as $k_n + \sum_{i=n+1}^{m} m_i$. Rewrite vertex CS_{n,k_n} as $v_{k_n+\sum_{i=n+1}^{m} m_i}$ with unified index $k_n + \sum_{i=n+1}^{m} m_i$. The vertex set $\mathcal{V}$ can be rewritten as $\mathcal{V} = \{v_1, v_2, ..., v_{\sum_{i=0}^{m} m_i}\}$ and sub vertex set $\mathcal{V}_n$ can be rewritten as $\mathcal{V}_n = \{v_{\sum_{i=\sum_{i=n+1}^{m} m_i}^{m} +1}, v_{\sum_{i=\sum_{i=n+1}^{m} m_i}^{m} +2}, ..., v_{\sum_{i=\sum_{i=n+1}^{m} m_i}^{m} +m_n}\}, n = 0, 1, 2, ...,$. In terms of the unified index of each vertex, the weighted adjacency matrix Γ can be defined as $\gamma_{ij} \neq 0$ if $(v_i, v_j) \in \mathcal{E}$, otherwise $\gamma_{ij} = 0$. Besides, $\gamma_{ii} = 0$ for all $i = 1, 2, ..., N$. $N_i = \{j | (v_i, v_j) \in \mathcal{E}, j = 1, 2, ..., N\}$ represents the neighbor index set of vertex v_i.

An example is given in Fig.3.2, to better interpret the graph-based communication and organization for SoS modelling. The tree topology, provided on the left side of Fig.3.2, illustrates the communication and organization between PCSs and MCSs, and among MCSs for the 3-level SoS model. It can be seen from the left side of Fig.3.2 that the 1-st and 2-nd MCSs in managerial level 1 are supervised by the 1-st MCS in managerial level 2, represented by the edges between vertices $CS_{2,1}$ and $CS_{1,i}, i = 1, 2$. The 1-st, 2-nd, 3-rd, 4-th and 7-th PCSs in physical level 0 are supervised and monitored by the 1-st in managerial level 1, represented as the edges between vertices $CS_{1,1}$ and $CS_{0,i}, i = 1, 2, 3, 4, 7$. Besides, the 3-th, 5-th, 6-th, 7-th and 8-th PCSs in physical level 0 are supervised and monitored by the 2-nd in managerial level 1, represented as the edges between vertices $CS_{1,2}$ and $CS_{0,i}, i = 3, 5, 6, 7, 8$. To prevent edges crossings in the graph, we address the ring

topology to describe the communication among PCSs on the right side of Fig.3.2, it is seen from the ring topology that for $i = 1, 2, ..., 7$ the i-th PCS can communicate with the $i + 1$-th PCS , represented by the edges between $CS_{0,i}$ and $CS_{0,i+1}$. Additionally, the edge between $CS_{0,1}$ and $CS_{0,8}$ demonstrates the communication between the 1-st and the 8-th PCSs .

Building on the above discussion, the graph-based topology effectively provides a clear, point-to-point representation of communication and organization in SoS modelling. It enables detailed analysis in the dynamic response of SoS subject to diverse interactions in physical dynamics. In the controllability analysis of the following Chapter 4, we will utilize the defined graph-based topology with adjustments in the SoS modelling. The graph-based topology clearly describes the heterogeneous interactions between agents' dynamics, which lays the ground for the controllability of networked SoS in Chapter 4.

3.3 Hypergraph-based communication and organization topology

Notice that in the above graph-based topology, each edge connects only two vertices, which limits the ability to represent complex, multi-way relationships. If a relationship involves more than two entities, it must be broken down into multiple pairwise connections. This drawback can be reflected by the repeated edges between MCSs and PCSs, which can lead to loss of clarity and increased complexity. Moreover, as the number of levels and component systems increases, the number of edges can become overwhelming, complicating the topology. For instance, in Fig.3.2, we had to use two separate diagrams to describe the communication and organization of the SoS to avoid edge crossings. Additionally, When modelling SoS, redundancy in edges is more likely, especially when trying to represent multi-way relationships or interactions that involve multiple CSs. This redundancy can clutter the model and make it harder to extract meaningful insights.

In this regard, hypergraphs allow hyperedges to connect multiple vertices simultaneously, providing a more natural representation of the multi-way relationships. By allowing hyperedges to connect multiple nodes in a single relationship, hypergraphs can reduce the number of edges required, leading to more scalable and maintainable models. Hypergraphs reduce redundancy by encapsulating multi-way relationships within a single hyperedge, leading to more concise and understandable models. That is the reason why we introduce the hypergraph-based topolgy to depict the communication and organization in SoS modelling.

A general hypergraph-based communication and organization topology of the m-level SoS model is given below:

Denote $\mathcal{H}_0 = (\mathcal{V}_0, \mathcal{E}_1)$ as the hypergraph to describe the communication and organization among MCSs in managerial level 1 and PCSs in physical level 0. The constitution of vertex set $\mathcal{V}_0 = \{CS_{0,1}, CS_{0,2}, \ldots, CS_{0,m_0}\}$ is the same to the $\mathcal{V}_0$ of graph $\mathcal{G}$, where vertex $CS_{0,k_0}, k_0 = 1, 2, ..., m_0$ represents the i-th PCS in physcial

level 0. The hyperedge set $\mathcal{E}_1$ is defined as $\mathcal{E}_1 = \{CS_{1,1}, CS_{1,2}, \ldots, CS_{1,m_1}\}$, in which hyperedge CS_{1,k_1} is the k_1-th MCS in managerial level 1. For $k_1 \in \{1, 2, ..., m_1\}$, hyperedge CS_{1,k_1} is a subset of vertex set V_0, where $CS_{0,k_0} \in CS_{1,k_1}$ implies the k_0-th PCS is supervised by the k_1-th MCS of managerial level 1. In addition, it also implies the i-th PCS can communicate with the k_1-th MCS. The cardinality of hyperedge CS_{1,k_1} is denoted as $|CS_{1,k_1}|$, which represents the number of nodes composed in the hyperedge CS_{1,k_1}, that is, the quantity of PCSs monitored by k_1-th MCS of managerial level 1. Denote $I(\mathcal{H}_0) \in \mathbb{R}^{m_0 \times m_1}$ as the incidence matrix of hypergraph $\mathcal{H}_0$, whose element $[I_{ij}(\mathcal{H}_0)]$ is defined by the indicator function $I(\cdot, \cdot)$ below

$$I_{ij}(\mathcal{H}_0) = I(CS_{0,i}, CS_{1,j}) = \begin{cases} 1, & \text{if } CS_{0,i} \in CS_{1,j}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

Notice that, in hypergraph $\mathcal{H}_1$, the Assumption 3.1.4 ensures every hyperedge $CS_{1,k_1}, k_1 \in \{1, 2, ..., m_1\}$ in hypergraph H_0 is not empty. That is, there must exist at least one vertex $CS_{0,k_0}, k \in \{1, 2, ..., m_0\}$ such that $CS_{0,k_0} \in CS_{1,k_1}$, which ensures $CS_{1,k_1} \neq \emptyset$.

The communication and organization between MCSs in managerial levels n and $n+1$ $(n = 1, 2, ..., m-1)$ can be described by the hypergraph $\mathcal{H}_n = (\mathcal{V}_n, \mathcal{E}_{n+1})$. The vertex set $\mathcal{V}_n = \{CS_{n,1}, CS_{n,2}, \ldots, CS_{n,m_n}\}$ is the same to the $\mathcal{V}_n$ of graph $\mathcal{G}$, where vertex CS_{n,k_n} represents the k_n-th MCS in managerial level n. The hyperedge set $\mathcal{E}_{n+1}$ satisfies $\mathcal{E}_{n+1} = \{CS_{n+1,1}, CS_{n+1,2}, \ldots, CS_{n+1,m_{n+1}}\}$, in which $CS_{n+1,k_{n+1}}$ indicates the k_{n+1}-th MCS in managerial level $n+1$. Note that for $n = 1, 2, ..., m-1$, While $CS_{n,k_n}, k_n \in \{1, 2, ..., m_n\}$ signifies the hyperedge in hypergraph $\mathcal{H}_{n-1}$, it denotes the vertex in hypergraph $\mathcal{H}_n$. Nevertheless, both of them represent the k_n-th MCS in managerial level n. In addition, $CS_{n,k_n} \in CS_{n+1,k_{n+1}}$ implies the k_n-th MCS is supervised and monitored by the k_n-th MCS of managerial level n. In addition, it also implies k_n-th MCS in level n can communicate with k_{n+1}-th MCS in level $n+1$. Similar to that that of $\mathcal{H}_0$, the incidence matrix of hypergraph $\mathcal{H}_n$ is denoted as $I(\mathcal{H}_n) \in \mathbb{R}^{m_n \times m_{n+1}}$ with element $[I_{ij}(\mathcal{H}_n)]$,

$$I_{ij}(\mathcal{H}_n) = I(CS_{n,i}, CS_{n+1,j}) = \begin{cases} 1, & \text{if } CS_{n,i} \in CS_{n+1,j}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

In addition, due to we have presumed the intra-layer information sharing among MCSs in the same managerial level. There is no need to draw hyperedges among $CS_{n,k_n}, k_n = 1, 2, ..., m_n$.

To better illustrate the hypergraph-based communication and organization topology, an example is given in Fig.3.3 for a 3-level SoS. There are two hypergraphs $\mathcal{H}_0$ and H_1 in Fig.3.3. In hypergraph H_0 that describe the organization and communication between PCSs and MCSs, there are two hyperedges $CS_{1,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,7}\}$ and $CS_{1,2} = \{CS_{0,3}, CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$, which implies the i-th $(i = 1, 2, 3, 4, 7)$ PCSs are supervised and monitored by and can communicate with the 1-th MCS in managerial 1 $(CS_{1,1})$. Besides, the i-th $(i = 3, 5, 6, 7.8)$ PCSs are supervised by 2-th MCS in managerial level 1 $(CS_{1,2})$.

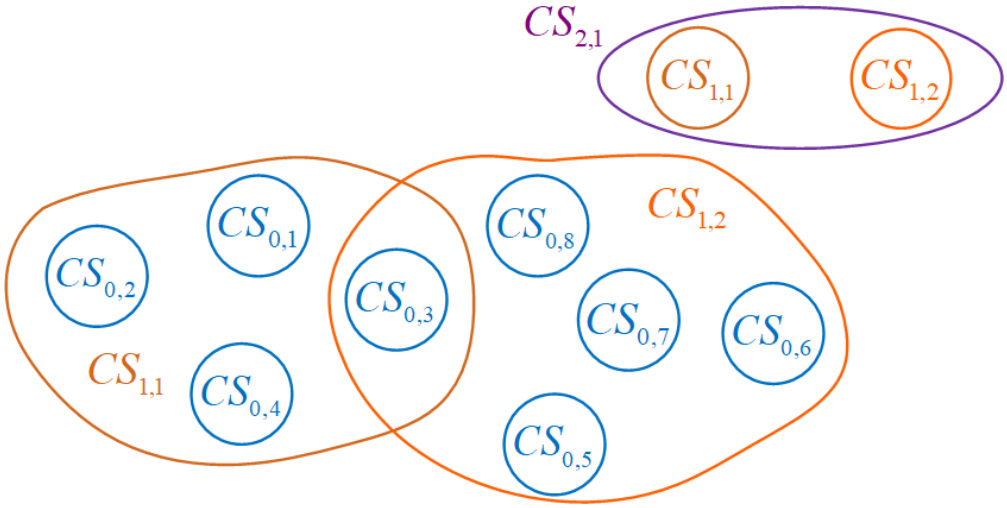


Figure 3.3: Organization and communication between PCSs and MCSs, and among MCSs.

The communication and organization between MCSs in managerial levels 1 and 2 are addressed in hypergraph H_1, where hyperedge $CS_{2,1} = \{CS_{1,1}, CS_{1,2}\}$. This reveals $1, 2$-th MCSs in managerial 1 $(CS_{1,1}, CS_{1,2})$ are supervised and monitored by and can communicate with the 1-th MCS in managerial level 2.

It's seen from Fig.3.3 that, hypergraphs can express compound relationships more naturally by connecting all relevant nodes within a single hyperedge, simplifying the representation. It offers a more flexible and scalable approach for capturing complex, multi-way, and overlapping relationships, making it a better choice in describing the communication and organization across levels for SoS modelling. By capturing complex relationships and interactions, hypergraphs provide decision-makers with deeper insights into the implementation, reconfiguration and planning. Meanwhile, the rich representation of interactions in hypergraphs allows for more precise optimization techniques, helping to estimate the SoS recovery time. The hypergraph topology will be adopted in Chapter 6 with adjustments.

3.4 Hybrid graph/hypergraph-based communication and organization topology

It's seen that the communication among PCSs has not been shown in Fig.3.3. This is because the interconnections among PCSs is influenced by the heterogeneous interactions between their dynamics. Therefore, even when using a hypergraph to represent their relationships, we still need to express these connections as point-to-point interactions as we did in the graph-based topology. Therefore, we propose the hybrid graph/hypergraph-based topology for the SoS modelling. By using traditional graphs for pairwise interactions to describe the heterogeneous interactions in the PCS dynamics and reserving hypergraphs for multi-way interactions to cap-

ture interconnections across different levels, the overall model remains scalable and manageable without losing essential details. Apart from this, visualizing a complex SoS can be challenging, but the hybrid approach allows for different visualization techniques tailored to the interaction types. Graph-based models are easier to visualize and interpret the point-to-point interactions among PCSs, while hypergraphs provide simpler and clearer presentation for the communication and organization between MCSs and PCSs, among MCSs in distinct managerial levels.

A hybrid graph/hypergraph-based communication and organization topology of the m-level SoS model is introduced based on the above mentioned graph-based topology and hypergraph-based topology.

Observe that, there is no organizational relationship among PCSs in the same physical level 0. Firstly, for all PCSs in physical level 0, we propose a propose a graph $\mathcal{G} = (\mathcal{V}_0, \mathcal{E}_0)$ as the directed communication graph among all PCSs with vertex set $\mathcal{V}_0$ and edge set $\mathcal{E}_0$. The definition of vertex set $\mathcal{V}_0 = \{CS_{0,1}, CS_{0.2}, \ldots, CS_{0,m_0}\}$ is the same to that of graph-based topology for SoS modelling in Chapter 3.2. Each node CS_{0,m_0} represents the i-th PCS in physical level 0. $\mathcal{E}_0 = \{e_1, e_2, \ldots, e_p\} \subseteq \mathcal{V}_0 \times \mathcal{V}_0$ is the edge set. There exists an edge $e_q = (CS_{0,i}, CS_{0,j}) \in \mathcal{E}_0$ if the i-th PCS can transmit information to the j-th PCS. The directed path in graph $\mathcal{G}$ is a sequence of nodes CS_{0,i_1}, CS_{0,i_2},...,CS_{0,i_s} such that $(CS_{0,i_k}, CS_{0,i_{k+1}}) \in \mathcal{E}_0$ for all $k = 1, 2, ..., s-1$. $A = [a_{ij}] \in \mathbb{R}^{m_0 \times m_0}$ is the adjacency matrix related to communication graph $\mathcal{G}$, where

$$a_{ij} = \begin{cases} 1, & \text{if } (CS_{0,i}, CS_{0,j}) \in \mathcal{E}_0, \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

Besides, $a_{ii} = 0$ for all $i = 1, 2, ..., n$. The neighbor set of $CS_{0,i}$ implies the index set of PCSs that can transmit information to the i-th PCS, which is defined as

$$N_i = \{j | CS_{0,j} \in \mathcal{V}_0, (CS_{0,j}, CS_{0,i}) \in \mathcal{E}_0\}. \tag{3.4}$$

The corresponding outputs of PCSs in the neighbor set N_i are available for the i-th PCS in its dynamics. The definitions and meanings of hypergraphs $\mathcal{H}_n, n = 0, 1, 2, ..., m-1$, related vertex sets $\mathcal{V}_n, n = 0, 1, 2, ..., m-1$ and hyperedge sets $\mathcal{E}_n, n = 1, 2, ..., m$ are the same to that of Chapter 3.3, which shows the communication and organization between PCSs and MCSs, among MCSs.

To better illustrate the hypergraph-based communication and organization topology, an example is given on the right side of Fig.3.4 for a 3-level SoS. In graph $\mathcal{G}$, the edges among $CS_{0,i}$ and $CS_{0,j}$ represents the communication exchange among the i, j-th PCSs ($i, j = 1, 2, ..., 8$). In hypergraph $\mathcal{H}_0$, the two hyperedges $CS_{1,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,7}\}$ and $CS_{1,2} = \{CS_{0,3}, CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$ imply the i-th ($i = 1, 2, 3, 4, 7$) PCSs are supervised and monitored by the first MCS in managerial level 1 ($CS_{1,1}$). And i-th ($i = 3, 5, 6, 7.8$) PCSs are supervised by the second MCS in managerial level 1 ($CS_{1,2}$). Additionally, in hypergraph $\mathcal{H}_1$, hyperedge $CS_{2,1} = \{CS_{1,1}, CS_{1,2}\}$ reveals the first and the second MCSs in managerial

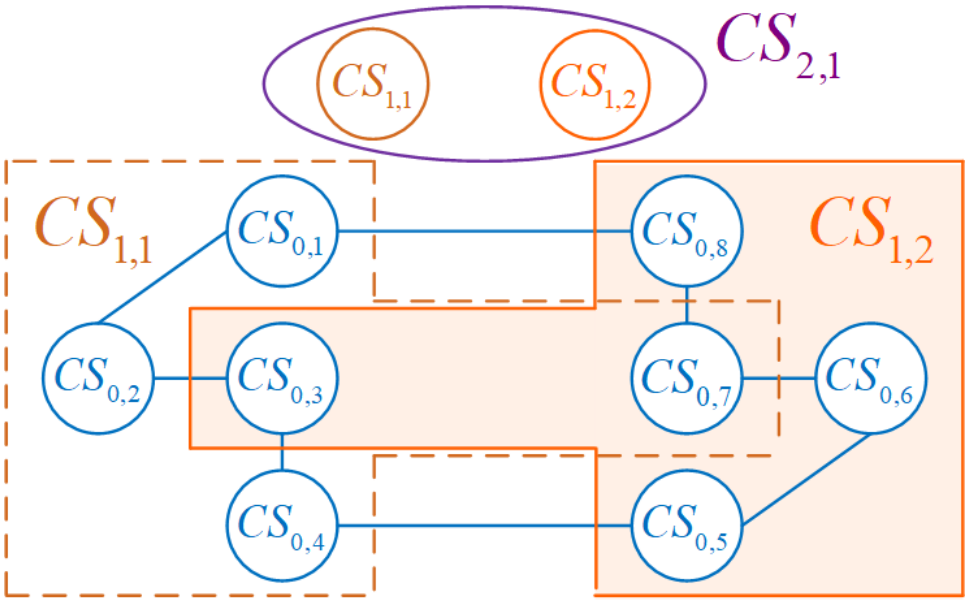


Figure 3.4: Hybrid graph/hypergraph-based communication and organization topology.

level 1 $(CS_{1,1}, CS_{1,2})$ are supervised and monitored by the first MCS in managerial level 2. The hyperedges also indicates the communications between them.

It can be seen from Fig.3.4 that the hybrid graph/hypergraph-based topology illustrates the sophisticated communication and organization of SoS modelling in a simply and clear way, leveraging the strengths of both graph-based and hypergraph-based topologies. It creates a more effective and accurate representation. Traditional graphs excel at representing pairwise interactions, which are common in PCS dynamics. Hypergraphs allow for the representation of multi-way interactions, which are crucial for capturing interconnections across different levels of the SoS. To propose a comprehensive evaluation framework for SoS performance, both dynamic response of various PCSs, MCSs performance and communication issue among CSs should be taken into account. That's the reason why we utilized the hybrid graph/hypergraph-based topology with modifications in Chapter 5 for the assessment and classification of SoS performance.

3.5 Nonlinear and linear dynamic component system modelling

3.5.1 Nonlinear component system modelling

For each PCS subject to nonlinear dynamics, its state space representation can be given by:

$$\begin{aligned} \dot{x_i}(t) &= f_i(t, x_i(t), u_i^F(t), d_i(t)), \\ u_i^F(t) &= \lambda_i^F u_i(t) + u_i^d(t), \lambda_i^F \in (0, 1], \\ u_i(t) &= h_i(t, y_i(t), Pdr_{ij}(t) \cdot y_j(t - \tau_{ij})), j \in N_i \\ y_i(t) &= g_i(t, u_i(t), x_i(t), n_i(t)). \end{aligned} \tag{3.5}$$

where $x_i(t) \in \mathbb{R}^{n_i}$ indicates the state vector of i-th PCS. $u_i(t) \in \mathbb{R}^{q_i}$ denotes the input vector, $u_i^F(t)$ represents the control input with/without fault which depends on its two input fault values λ_i^F (multiplicative fault) and $u_i^d(t)$ (additive fault). $y_i(t) \in \mathbb{R}^{p_i}$ signifies the output vector. $d_i(t)$ is external disturbance. $n_i(t)$ denotes output fault (measurement noise). $f_i(\cdot)$, $h_i(\cdot)$ and $g_i(\cdot)$ are continuous nonlinear functions. N_i expresses its neighbors that share their outputs with i-th PCS.

3.5.2 Linear component system modelling

For each PCS subject to linear dynamics, its state space representation can be given by:

$$\begin{aligned} \dot{x_i}(t) &= A_i x_i(t) + B_i u_i(t) + D_i d_i(t), \\ y_i(t) &= C_i x_i(t) + E_i u_i(t) + F_i n_i(t), \\ u_i^F(t) &= \lambda_i^F u_i(t) + u_i^d(t), \\ u_i(t) &= \sum_{j \in N_i} Pdr_{ij}(t) \bar{H}_{ij} y_j(t - \tau_{ij}) + \bar{H}_{ii} y_i(t). \end{aligned} \tag{3.6}$$

Additionally, the nonlinear dynamics in equation (3.5) can be approximated by the linear dynamics in equation (3.6) with the following approximations, in which the corresponding constant matrices of suitable dimensions are calculated in terms of partial derivative:

$$A_i \approx \frac{\partial f_i\left(t, x_i(t), u_i^F(t), d_i(t)\right)}{\partial x_i(t)} \tag{3.7}$$

$$B_i \approx \frac{\partial f_i\left(t, x_i(t), u_i^F(t), d_i(t)\right)}{\partial u_i^F(t)} \tag{3.8}$$

$$D_i \approx \frac{\partial f_i\left(t, x_i(t), u_i^F(t), d_i(t)\right)}{\partial d_i(t)} \tag{3.9}$$

$$C_i \approx \frac{\partial g_i\left(t, x_i(t), u_i(t), n_i(t)\right)}{\partial x_i(t)} \tag{3.10}$$

$$E_i \approx \frac{\partial g_i\left(t, x_i(t), u_i(t), n_i(t)\right)}{\partial u_i(t)} \tag{3.11}$$

$$F_i \approx \frac{\partial g_i\left(t, x_i(t), u_i(t), n_i(t)\right)}{\partial n_i(t)} \tag{3.12}$$

$$\bar{H}_{ii} \approx \frac{\partial h_i\left(t, y_i(t), Pdr_{ij}(t), y_j(t-\tau_{ij})\right)}{\partial y_i(t)} \tag{3.13}$$

$$\bar{H}_{ij} \approx \frac{\partial h_i\left(t, y_i(t), Pdr_{ij}(t), y_j(t-\tau_{ij})\right)}{\partial\left(Pdr_{ij}(t) \cdot y_j(t-\tau_{ij})\right)} \tag{3.14}$$

3.6 SoS resilience metric

The resilience reflects the capacity of a system to withstand, adapt, and recover from disruptions by some feasible reactions, whether these arise from natural disasters, technical failures, cyber-attacks, or other unforeseen challenges. In an SoS, resilience is not just about the robustness of individual systems but about the overall capability of the entire network to maintain its critical functions and avoid excessive recovery cost despite disturbances.

To formulate the resilience of the overall SoS, we first define the following function $\varphi_{i,j}(\cdot)$ for the time resilience metric of the i-th PCS in the j-th mission:

$$\varphi_{i,j}(T_{i,j}^{ro}|\bar{T}_{i,j}^{ro}) = \begin{cases} 1, & \text{if } T_{i,j}^{ro} \leq \bar{T}_{i,j}^{ro} \\ 0, & \text{otherwise,} \end{cases} \tag{3.15}$$

where the independent variable $T_{i,j}^{ro}$ represents the recovery and task operation duration of the i-th PCS in the j-th mission. Note that $T_{i,j}^{ro}$ satisfies

$$T_{i,j}^{ro} = T_i^r + T_{i,j}^o, \tag{3.16}$$

in which T_i^r is the recovery duration of the i-th PCS facing with disruptions. And $T_{i,j}^o$ is the operation duration of the i-th PCS to contribution its functions in the j-th mission. In addition, $\bar{T}_{i,j}$ is the maximal recovery and task operation duration of the i-th PCS in the j-th mission. The value of $\bar{T}_{i,j}$ relates to the importance of i-th PCS in the j-th mission and the significance of the j-th mission. The selection of $\bar{T}_{i,j}$ is flexible and is up to the manager. It's seen that $\varphi_{i,j}(T_{i,j}^{ro}|\bar{T}_{i,j}^{ro}) = 1$ if the recovery and task operation duration of the i-th PCS in the j-th mission is less than the upper bound $\bar{T}_{i,j}$, which implies i-th PCS is time resilient in the j-th mission. Otherwise, $\varphi_{i,j}(T_{i,j}^{ro}|\bar{T}_{i,j}^{ro}) = 0$, which indicates the related recovery and task operation duration exceeds the time limit and thus the i-th PCS is not time resilient in the j-th mission.

On the basis of (3.15), the definition of the function $\varphi_j(\cdot)$ for the time resilience metric of the j-th mission is given as

$$\varphi_j(\varphi_{i,j}, CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}) = \prod_{CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}} \varphi_{i,j}. \tag{3.17}$$

In terms of (3.15), $\varphi_j(\cdot)$ equals to 1 if and only if all $\varphi_{i,j} = 1$ for $CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}$ in hypergraph $\mathcal{H}_0$. That is, all the PCSs in the j-th mission is time resilient to the j-th mission. If there exists at least one PCS cannot achieve the recovery and

accomplish its functions in the j-th mission in its own upper bound duration, then the j-th mission is not time resilient facing with the disruptions.

Moreover, the definition of the function $\varphi(\cdot)$ for the time resilience metric of the SoS is shown as

$$\varphi(\varphi_j, CS_{1,j} \in \mathcal{V}_1) = \prod_{CS_{1,j} \in \mathcal{V}_1} \varphi_j, \tag{3.18}$$

which implies the overall SoS is time resilient if all basic missions are time resilient. If there exists at least one mission that is not time resilient, then the overall SoS is not time resilient.

Similarly, the definition of cost resilience function can be given as follows. Firstly, the following function $\sigma_{i,j}(\cdot)$ for the cost resilience metric of the i-th PCS in the j-th mission:

$$\sigma_{i,j}(C_{i,j}^{ro} | \bar{C}_{i,j}^{ro}) = \begin{cases} 1, & \text{if } C_{i,j}^{ro} \leq \bar{C}_{i,j}^{ro} \\ 0, & \text{otherwise,} \end{cases} \tag{3.19}$$

where the independent variable $C_{i,j}^{ro}$ represents the recovery and task operation cost of the i-th PCS in the j-th mission. $C_{i,j}^{ro}$ satisfies

$$C_{i,j}^{ro} = C_i^r + C_{i,j}^o, \tag{3.20}$$

in which C_i^r is the recovery cost of the i-th PCS facing with disruptions. And $C_{i,j}^o$ is the operation cost of the i-th PCS in the j-th mission. In addition, $\bar{C}_{i,j}$ is the upper bound for the sum of the recovery and task operation cost of the i-th PCS in the j-th mission. The value of $\bar{C}_{i,j}$ is the maximal cost of the i-th PCS in the j-th mission in practice. The selection of $\bar{C}_{i,j}$ is flexible and is up to the manager, which may refer to the priority of the i-th PCS in the j-th mission and the importance of the j-th mission. It's seen that $\sigma_{i,j}(C_{i,j}^{ro} | \bar{C}_{i,j}^{ro}) = 1$ if the recovery and task operation cost of the i-th PCS in the j-th mission is less than the upper bound $\bar{C}_{i,j}$, which implies i-th PCS is cost resilient in the j-th mission. Otherwise, $\sigma_{i,j}(C_{i,j}^{ro} | \bar{C}_{i,j}^{ro}) = 0$, which indicates the related recovery and task operation cost exceeds the maximal cost and thus the i-th PCS is not cost resilient in the j-th mission.

On the basis of (3.19), the definition of the function $\sigma_j(\cdot)$ for the cost resilience metric of the j-th mission is given as

$$\sigma_j(\sigma_{i,j}, CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}) = \prod_{CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}} \sigma_{i,j}. \tag{3.21}$$

In terms of (3.19), $\sigma_j(\cdot)$ equals to 1 if and only if all $\sigma_{i,j} = 1$ for $CS_{0,i} \in \mathcal{V}_0, CS_{0,i} \in CS_{1,j}$ in hypergraph $\mathcal{H}_0$. That is, all the PCSs in the j-th mission is cost resilient to the j-th mission. If there exists at least one PCS cannot achieve the recovery and accomplish its functions in the j-th mission with the limited cost, then the j-th mission is not cost resilient facing with the disruptions.

Moreover, the definition of the function $\sigma(\cdot)$ for the cost resilience metric of the SoS is shown as

$$\sigma(\sigma_j, CS_{1,j} \in \mathcal{V}_1) = \prod_{CS_{1,j} \in \mathcal{V}_1} \sigma_j, \tag{3.22}$$

which implies the overall SoS is cost resilient if all basic missions are cost resilient. If there exists at least one mission that is not cost resilient, then the overall SoS is not cost resilient.

In virtue of the time and cost resilience of SoS, the overall resilience metric Res of SoS can be defined as

$$Res = \varphi(\varphi_j, CS_{1,j} \in \mathcal{V}_1) \cdot \sigma(\sigma_j, CS_{1,j} \in \mathcal{V}_1). \tag{3.23}$$

Note that $Res = 1$ if and only if both $\varphi(\varphi_j, CS_{1,j} \in \mathcal{V}_1)$ and $\sigma(\sigma_j, CS_{1,j} \in \mathcal{V}_1)$ are equal to 1. That is, facing with disruption SoS can not only restore all missions in required duration, but also avoid excessive cost. If at least one of $\varphi(\varphi_j, CS_{1,j} \in \mathcal{V}_1)$ and $\sigma(\sigma_j, CS_{1,j} \in \mathcal{V}_1)$ is equal to 0, then $Res = 0$, which indicates the overall SoS is not resilient if it is not time resilient and or not cost resilient.

3.7 Maier's Properties

A SoS is characterized by several distinct features that differentiate it from a traditional system. According to [Maier 1998], the key properties include operational and managerial independence of constituent systems, geographical distribution, evolutionary development, emergent behaviour.

- **Operational independence**: Each CS within an SoS operates independently, achieving its own objectives and functioning autonomously. In the graph-based, hypergraph-based, and hybrid graph/hypergraph-based topology, each CS is represented by individual vertex $CS_{i,j}$ (the j-th CS in level i) satisfying

  $$\forall CS_{i,j}, CS_{i',j'} \in \text{SoS}, Op_{i,j} \neq Op_{i',j'}, \text{if } CS_{i,j} \neq CS_{i',j'}, \tag{3.24}$$

  which indicates the operational process Op of distinct CS is independent with other CSs.

- **Managerial independence**: The CSs are managed independently, often by different organizations or stakeholders. They may be developed, operated, and maintained independently, yet are brought together to fulfill a higher-level purpose.

  In the graph-based, hypergraph-based, and hybrid graph/hypergraph-based topology, each CS satisfies

  $$\forall CS_{i,j}, CS_{i',j'} \in \text{SoS}, M_{i,j} \cap M_{i',j'} = \emptyset, \text{if } CS_{i,j} \neq CS_{i',j'}, \tag{3.25}$$

which indicates the managerial process M of distinct CS is independent with other CSs.

- **Geographical Distribution**: CSs are dispersed over diverse geographical areas. Each CS satisfies

$$\forall CS_{i,j}, CS_{i',j'} \in \text{SoS}, \, l_{i,j} \cap l_{i',j'} = \emptyset, \text{if } CS_{i,j} \neq CS_{i',j'}, \tag{3.26}$$

  which indicates the physical connection l of distinct CS is independent with other CSs.

- **Cooperative Behaviour**: The interactions inside the SoS lead to cooperative behaviours that are not predictable solely from understanding the individual systems. These behaviours arise from the complex interactions and are a hallmark of an SoS . In the dynamic system modelling of, there exists interactions in the control input. That is, the control input of each PCS subject to the output of its neighbours, which reflects the sophisticated interactions inside the SoS. For nonlinear systems,

$$u_i(t) = h_i(t, y_i(t), Pdr_{ij}(t) \cdot y_j(t - \tau_{ij})), j \in N_i. \tag{3.27}$$

  For linear systems,

$$u_i(t) = \sum_{j \in N_i} Pdr_{ij}(t) \bar{H}_{ij} y_j(t - \tau_{ij}) + \bar{H}_{ii} y_i(t). \tag{3.28}$$

- **Evolutionary Development**: SoS evolves over time, adapting to changing requirements and environments. For both nonlinear dynamics in (3.5) and linear dynamics in (3.6), the states of systems evolves over time. That is,

$$\forall CS_{i,j} \in \text{SoS}, \, (SoS_{t+1} = SoS_t \setminus CS_{i,j}) \vee (SoS_{t+1} = SoS_t \cup CS_{i,j}), \tag{3.29}$$

  which reflects the evolution of systems with the passage of time.

3.8 Conclusion

In this Chapter, the management decomposition is developed firstly to depict the multi-level structure of SoS. Next, three kinds of modelling methods: graph-based modelling, hypergraph-based modelling, hybrid graph/hypergraph-based modelling are proposed to describe the communication and organization of SoS structure. Besides, both nonlinear and linear dynamics of PCSs are presented. The SoS resilience metric is proposed to measure the resilience performance of SoS under disruptions. Five essential properties characterizing SoS have been tested throughout the three proposed modelling. Point out that the pairwise edges in graph-based modelling enable detailed analysis on the interactions in the dynamics of SoS. Thus,

it will be adopted in the controllability analysis in Chapter 4. In developing the reconfiguration strategy, our focus shifts towards the multi-way relationships among CSs across various levels rather than emphasizing physical dynamics. To achieve this, we incorporate a hypergraph-based topology into the SoS modelling, which encapsulates multi-way relationships within a single hyperedge, thereby reducing redundancy. This approach offers more concise and comprehensible models for resilience reconfiguration, which will be utilized in Chapter 6. We propose a hybrid graph/hypergraph-based topology that employs traditional graphs for pairwise interactions to represent heterogeneous interactions within PCS dynamics, while utilizing hypergraphs for multi-way interactions to capture interconnections across different levels. This hybrid approach forms the foundation for the comprehensive evaluation framework for the SoS in Chapter 5

Chapter 4

SoS structure analysis of multi-level SoS controllability

Contents

4.1 Introduction 51
4.2 State-controllability of heterogeneous SoS 54
4.2.1 A two-layer SoS 54
4.2.2 Controllability analysis based on Sylvester equation 59
4.2.3 A three-layer SoS 66
4.2.4 Two-layer decomposition: a divide-and-conquer viewpoint . . 67
4.3 Qualitative and quantitative tests 73
4.4 Conclusion 76

4.1 Introduction

Based on the graph-based modelling and linear dynamics of PCSs state representation in Chapter 3, the controllability analysis of multi-level SoS will be presented in this Chapter, considering the heterogeneous dynamics, diverse dimension and varying interactions among CSs. The interactions among CSs varies with CS and their neighbours.

Multi-level modeling of an SoS has been reported in several works, based on Hypergraphs [Khalil 2012b] in order to supervise the failing modes of dynamic CS, or based on MAS [Soyez 2017] for the simulation of the evolutionary behavior of managerial CS, or based on the extended Bond Graph [Kumar 2018] for modeling collaborative behavior between dynamic CSs and managerial CSs. All of these models have been developed to analyze the organizational properties of an SoS, namely: operational independence, managerial independence, collaborative behavior, geographical dispersion and evolutionary development. However, these models did not deal with the analysis of structural properties such as controllability, which is the subject of this work, extended to multi-levels SoS. It proposes a high-level perspective to cope with the growing complexity of large-scale system.

SoS and MAS share a number of common features, such as emergent, collaborative and distributed [Gorod 2008]. Nevertheless, a SoS demands higher autonomy

for each component system compared with the agents in MAS, including operational and managerial independence [Maier 1998, Gorod 2008]. On the contrary, MAS only requires partial autonomy and allows passive agents such as obstacles, apply or key [Kubera 2010]. Overall, SoS can be regarded as a distinctive subclass of MAS, characterized by more rigorous demands and constraints. The advances of SoS have been reported in various applications, such as supply chain management [Jaradat 2017], telecommunications networks [Tsilipanos 2012] and air transportation [DeLaurentis 2011], etc.

Controllability, one of the basic concept in control theory, was firstly studied in single dynamic system [Hespanha 2018, Van Der Woude 2019]. It reveals the quantification of capacity to steer a system to any prescribed state from arbitrary initial state within a finite duration. For a large-scale networked system, including SoS, MAS or complex network, the intricate networked interconnections among CSs render the controllability analysis more involved, which implies the computational inefficiency of the traditional controllability criterion related to single systems [Zhou 2021]. Hereby, the controllability of the whole systems can be divided into two aspects: the controllability of each CS and inner-coupling based on network communication [Chapman 2014]. The controllability analysis of SoS focuses on how to maintain the controllablility of the whole system while a part of CSs are not controllable under failures, which helps us to devise a resilient networked control structure.

The works tangentially related are the controllability analysis of NSs [Lou 2018] and MASs [Guan 2016]. For instance, the controllability of leaderless and leader-follower MASs has been investigated in [Tanner 2004, Zhu 2022, Hou 2016] and [Guan 2016, Zhang 2021a] via algebraic and/or graph-theoretic methods respectively. Note that, in the above considered systems, each CS is regarded as an one-dimensional single/double-integrator node with neighbor interconnection-based dynamics. The results were extended to MASs [Zhang 2013, Guo 2022] and networked MIMO systems [Zhao 2015, Wang 2016, Hao 2018, Wu 2020] with LTI dynamics. Nevertheless, note that the above reviewed literature presupposed homogeneity for each agent or CS in the MASs or NSs, including the same dimensions, dynamics and coupling matrices. In [Xiang 2019], the controllability of networked MIMO systems with heterogeneous system matrices was investigated, but the dimension and inner-coupling matrix of each node are still identical. The results were further extended to NSs with nonidentical inner-coupling matrices [Kong 2022], but the considered NSs still subject to the nodes of invariant dimension.

For a SoS, it's expected to exhibit a variety of functions to cope with complex tasks [Gorod 2008], which requires a high-degree of heterogeneity, such as CSs of different dimensions. For example, in [Jiang 2022], a 3-dimensional T-quad robot and a 4-dimensional drone are included in one SoS to achieve the navigation collaboratively. Furthermore, observe that the above mentioned works mainly consider single-layer networked structure, where all CSs are treated on an equal footing. Nevertheless, the effectiveness of multi-layer NSs has already been discussed in [Boccaletti 2014]. For a SoS, a multi-level structure can achieve the functional

hierarchy [Thacker 2017, Jiang 2022] with inter/intra-layer connections. The controllability of multi-layer NSs is investigated in [Wu 2020, Wu 2023], but the considered CSs still subject to homogeneous dynamics in each layer.

Inspired by the above discussion, the controllability of heterogeneous SoS is studied in this chapter. Hereby, the SoS organization follows a multi-layer structure, which is divided into PCSs in physical level 0 and MCSs in various managerial levels $n, n = 1, 2, ...$, based on the management decomposition in [Jiang 2022]. The contribution of this work can be summarized as follows:

1. Inspired by the widely applied output regulation approach for coordination control of heterogeneous MASs [Ma 2015, Yaghmaie 2016], the controllability of two-layer heterogeneous SoS is investigated based on the modified Sylvester equations. This deeply alleviates the computational complexity of controllability analysis compared with classical Popov-Belevitch-Hautus (PBH) test. Moreover, the considered SoS is of high-degree heterogeneity, where each CS is of multiple dimension, different coupling matrix and nonidentical dynamics. In comparison to preceding literature emerged on homogeneous NSs [Zhao 2015, Wang 2016, Hao 2018, Wu 2020], heterogeneous NSs of identical coupling matrices [Xiang 2019] and/or the same dimension [Kong 2022], the considered SoS is more applicable and the controllability analysis is more challenging.

2. Thereafter, a two-layer decomposition approach is proposed for multi-layer SoS. It deeply simplifies the controllability analysis by dividing the higher-level SoS into multiple lower-layer components, which further form a newly generated lower-level SoS. Above properties enable us to cope with the controllability of multi-layer SoS in a two-layer framework, where we can firstly find controllable lower-level components and then observe the controllability of generated lower-level SoS. While previous controllability analysis on multi-layer NSs subject to homogeneous dynamics in the same layer [Wu 2020, Wu 2023], the proposed method can deal with heterogeneous dynamics in the same layer. Fig.4.1 is addressed to illustrate the difference between the systems considered in [Wu 2020, Wu 2023] and this chapter.

Some necessary and/or sufficient controllability conditions are derived for heterogeneous multi-layer SoS. A number of illustrative instances of various typical graphs, including star, chain, circle and tree are addressed to substantiate the validity of theoretical results.

Notations. Throughout this chapter, $\sigma(P)$ stands for the eigenvalue of matrix P. $Im(P)$ and $Ker(P)$ denote the column space and kernel space of matrix P. $diag\{a_1, a_2, ..., a_n\}$ represents (block) diagonal matrix with main diagonal entities $a_1, a_2, ..., a_n$. $\mathbf{0}$ ($\mathbf{0}_{n\times m}$) signifies the zero matrix of suitable dimension ($n \times m$ dimension).

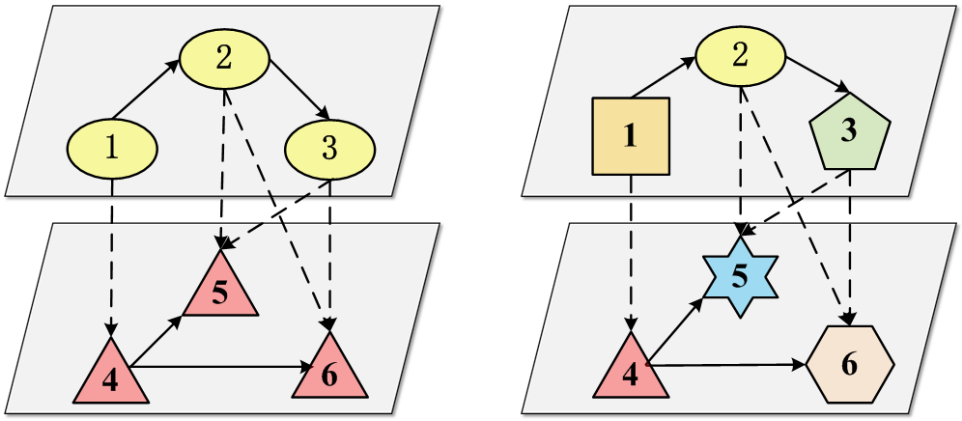


Figure 4.1: In [Wu 2020, Wu 2023], the considered CSs shown in Fig.(a) belong to the same layer are of homogeneous dynamics. Given in Fig.(b), the considered CSs are of heterogeneous dynamics in one layer.

4.2 State-controllability of heterogeneous SoS

The SoS model considered in this work obeys a hierarchical organization including, MCSs in multiple managerial levels and PCSs in physical level 0. The MCSs in SoS are usually regarded as supervisors embedded with computers. Throughout this chapter, we propose the following assumption based on the *management decomposition* in [Jiang 2022].

Assumption 4.2.1 *(SoS management decomposition) Each MCS in level n can only receive/transmit information to CSs in n, $n-1$ and $n+1$ levels in terms of the directed graph. For PCSs in level 0, it can only communicate with MCSs in level 1 and other PCSs according to the directed graph.*

4.2.1 A two-layer SoS

In this part, we firstly consider the simplest case, i.e., a two-layer SoS with only one MCS. The MCS with an embedded computer that can supervise related PCSs over digraph-based network. Meanwhile, a virtual dynamic model is established in the embedded computer of 1-th autonomous MCS, whose parameters can be designed by the engineer to ensure the controllability of SoS. The dynamics of 1-th MCS virtual model Σ_1 and i-th ($i = 2, 3, ..., N$) PCS Σ_i in two-layer SoS Σ related to graph $\mathcal{G}$ can be depicted by

$$\begin{cases} \dot{x}_1(t) = A_1 x_1(t) + \sum_{j=2}^{n} \gamma_{1j} H_{1j} y_j(t) + \theta_1 B_1 u_1(t) \\ \dot{x}_i(t) = A_i x_i(t) + \sum_{j=1}^{n} \gamma_{ij} H_{ij} y_j(t) + \theta_i B_i u_i(t) \\ \qquad\qquad\qquad\qquad i = 2, 3, ..., N \\ y_i(t) = C_i x_i(t), x_i(0) = x_{i0}, \end{cases} \tag{4.1}$$

in which $x_1(t) \in \mathbb{R}^{n_1}$ and $y_1(t) \in \mathbb{R}^{m_1}$ represent the pseudo state and output of MCS. For $i = 2, 3, ..., N$, $x_i(t) \in \mathbb{R}^{n_i}$ and $y_i(t) \in \mathbb{R}^{m_i}$ denote the state and output of all PCSs.

Let $\bar{u}_i(t) = \sum_{j=1}^{n} \gamma_{ij} H_{ij} y_j(t) + \theta_i B_i u_i(t)$ stands for the controller of i-th CS, which consists of outputs received from its neighbor and external control signal $u_i(t)$. Matrices $A_i \in \mathbb{R}^{n_i \times n_i}$, $B_i \in \mathbb{R}^{n_i \times q_i}$ and $C_i \in \mathbb{R}^{m_i \times n_i}$ are system, input and output matrices of i-th plant. $H_{ij} \in \mathbb{R}^{n_i \times m_i}$ is the inner-coupling matrix that describes the interaction among CSs. The weight of interaction between i-th and j-th systems is denoted by $\gamma_{ij} \in \mathbb{R}$, which is the entries of adjacency matrix $\Gamma = [\gamma_{ij}] \in \mathbb{R}^{N \times N}$ related to graph $\mathcal{G}$. $\theta_i \in \{0, 1\}$ is a binary signal, in which $\theta_i = 1$ if i-th plant is subject to external input signal $u_i(t)$; else, $\theta_i = 0$. Let matrix $\Theta = diag\{\theta_1, \theta_2, ..., \theta_N\} \in \mathbb{R}^{N \times N}$. It's seen that pair (Γ, Θ) reflects the network topology of SoS.

Remark 4.2.1 *The term "heterogeneous" in this work implies each CS can be described by*

- *different dynamics, i.e., different system matrices A_i, B_i, C_i;*
- *multiple dimensions, i.e., the dimension of state, output input, n_i, m_i, q_i may vary for each CS;*
- *nonidentical coupling matrices H_{ij} which can be selected by designer due to the autonomy of each CS in SoS.*

Compared with previous research mainly focus on homogeneous CSs [Wang 2016, Hao 2018, Chen 2017, Zhang 2021b, Hao 2022] or heterogeneous CSs of the same dimension and coupling matrices [Xiang 2019, Guan 2016], the controllability analysis of considered SoS (4.1) is more involved and challenging. In a large-scale SoS, it's common to see heterogeneous CSs of different dimension as well as system matrices in one coordination task, such as T-quad robot and drone [Jiang 2022].

Naturally suppose $n_1 \geq n_i$, $q_1 \geq q_i$, $n_i \geq m_i$ and $n_i \geq q_i$ for $i = 1, 2, ..., N$. In addition, due to the virtual model of 1-th MCS can be designed by the users, we make the following assumption.

Assumption 4.2.2 *The virtual pair (A_1, B_1) is controllable.*

Note that, although pair (A_1, B_1) is controllable, the binary signal θ_1 may equal to 0 or 1 due to possible faults or failures, which implies **the root of SoS network is not assumed to be controllable**.

Denote $x(t) = [x_1^T(t), x_2^T(t), ..., x_N^T(t)]^T \in \mathbb{R}^{\sum_{i=1}^{N} n_i}$, $\bar{u}(t) = [\bar{u}_1^T(t), \bar{u}_2^T(t), ..., \bar{u}_N^T(t)]^T \in \mathbb{R}^{\sum_{i=1}^{N} q_i}$, $u(t) = [u_1^T(t), u_2^T(t), ..., u_N^T(t)]^T \in \mathbb{R}^{\sum_{i=1}^{N} q_i}$ and $y(t) = [y_1^T(t), y_2^T(t), ..., y_N^T(t)]^T \in \mathbb{R}^{mN}$. Equation (4.1) can be rewritten in a compact form

$$\begin{cases} \dot{x}(t) = (\tilde{A} + \tilde{H})x(t) + \tilde{\Theta}\tilde{B}u(t) \\ y(t) = \tilde{C}x(t), \end{cases} \tag{4.2}$$

where $\tilde{A} = diag\{A_1, A_2, ..., A_N\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\tilde{H} = [\gamma_{ij} H_{ij} C_j] \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\tilde{\Theta} = diag\{\theta_1 I_{n_1}, \theta_2 I_{n_2}, ..., \theta_N I_{n_N}\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\tilde{B} = diag\{B_1, B_2, ..., B_N\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} q_i}$ and $\tilde{C} = diag\{C_1, C_2, ..., C_N\} \in \mathbb{R}^{\sum_{i=1}^{N} m_i \times \sum_{i=1}^{N} n_i}$.

Let $\Xi = \tilde{A} + \tilde{H}$ and $\Omega = \tilde{\Theta}\tilde{B}$. SoS (4.1) can be rewritten as

$$\begin{aligned} \dot{x}(t) &= \Xi x(t) + \Omega u(t) \\ y(t) &= \tilde{C} x(t) \end{aligned} \tag{4.3}$$

Definition 1 *(State-controllability) SoS (4.1) is said to be state-controllable if the state of all CSs can reach any desired state from arbitrary initial value with suitable external input $u(t)$.*

It's intuitive to obtain the controllable subspace of SoS (4.1) as $\mathcal{K} = im\Omega + \Xi im\Omega +, ..., +\Xi^{N-1} im\Omega$ and SoS (4.1) is state-controllable if and only if $dim(\mathcal{K}) = \sum_{i=1}^{N} n_i$. The following task is proposed throughout this chapter.

Task 1 *Given triple (A_i, B_i, C_i) for all i-th PCS and Θ, devise suitable virtual triple (C_1, A_1, B_1), graph $\mathcal{G}$ and inner-coupling matrix H_{ij} such that SoS (4.1) is controllable.*

Theorem 4.2.1 *SoS (4.1) is state-controllable if and only if there only exists zero solution $\xi = [\xi_1^T, \xi_2^T, ..., \xi_N^T]^T \in \mathbb{C}^{\sum_{i=1}^{N} n_i}, \xi_i \in \mathbb{C}^{n_i}, i = 1, 2, ..., N$ to the following equations,*

$$\begin{cases} \xi_i^T (sI_{n_i} - A_i) - \sum_{j=1}^{N} \gamma_{ji} \xi_j^T H_{ji} C_i = \boldsymbol{0} \\ \theta_i \xi_i^T B_i = \boldsymbol{0}, i = 1, 2, ..., N \end{cases} \tag{4.4}$$

for arbitrary $s \in \mathbb{C}$.

Proof: Based on PBH test [Hespanha 2018], it's seen that SoS (4.1) is state-controllable if and only if there only exists zero solution $\xi \in \mathbb{C}^{\sum_{i=1}^{N} n_i}$ such that $\xi^T \Xi = s\xi^T$ and $\xi^T \Omega = 0$ for arbitrary $s \in \mathbb{C}$, which is equivalent to (4.4). Thus, we finalize the proof. ■

Corollary 4.2.1 *SoS (4.1) is state-controllable only if for the k-th CS Σ_k such that $N_k = \emptyset$, it satisfies pair (A_k, B_k) is state-controllable and $\theta_k = 1$.*

Proof: For k-th CS without neighbor ($N_k = \emptyset$), (4.4) is transformed into

$$\begin{cases} \xi_i^T (sI_{n_i} - A_i) - \sum_{j=1, j\neq k}^{N} \gamma_{ji} \xi_j^T H_{ji} C_i = \mathbf{0} \\ \theta_i \xi_i^T B_i = \mathbf{0}, i = 1, 2, ..., N, i \neq k \\ \xi_k^T (sI_{n_k} - A_k) - \sum_{j=1}^{N} \gamma_{jk} \xi_j^T H_{jk} C_k = \mathbf{0} \\ \theta_k \xi_k^T B_k = \mathbf{0}, \end{cases} \tag{4.5}$$

If $\theta_k = 0$ or (A_k, B_k) is not controllable, adopt (s^*, ξ_k^*) as a left eigenvalue and eigenvector pair of matrix A_k such that $\begin{cases}(\xi_k^*)^T(s^* I_{n_k} - A_k) = \mathbf{0}, \\ (\xi_k^*)^T B_k = \mathbf{0}.\end{cases}$ Then ξ satisfying $\xi_k = \xi_k^*$ and $\xi_j = \mathbf{0}_{n_j}, j \neq k$ is a nonzero solution to equation (4.4) with $s = s^*$. According to Theorem 4.2.1, SoS (4.1) is not state-controllable. ■

Corollary 4.2.2 *If there exists i-th CS such that for arbitrary* $j = 1, 2, ..., n$, $rank([H_{ij}, B_i]) = rank(B_i)$. *SoS (4.1) is state-controllable only if* (A_i, B_i) *is state-controllable.*

Proof: If $rank([H_{ij}, B_i]) = rank(B_i)$, there must exist matrices $X_{ij} \in \mathbb{R}^{q_i \times m_j}$ such that $H_{ij} = B_i X_{ij}$. Denote $W_i(s)$ as the i-th row block of $[sI - \Xi, \Omega]$. Due to $H_i = B_i X_i$, it's easy to see that $rank(W_i(s)) \leq rank([sI_{n_i} - A_i, B_i]) = n_i$, which implies $[sI - \Xi, \Omega]$ is not full row rank. Based on PBH test, SoS (4.1) is state-controllable. ■

Corollary 4.2.3 *If there exists k-th CS* Σ_k *which is not under external control i.e.,* $\theta_k = 0$ *or* (A_k, B_k) *is not state-controllable, SoS (4.1) is state-controllable only if the k-th CS's neighbor set* $N_k \neq \emptyset$.

Proof: If $N_k = \emptyset$, $rank(W_k(s)) = rank([sI_{n_k} - A_k, B_k]) \leq n_k$ due to $\theta_k = 0$ or (A_k, B_k) is not state-controllable. Thus, $[sI - \Xi, \Omega]$ is not full row rank and SoS (4.1) is not state-controllable. ■

Remark 4.2.2 *Corollaries 4.2.1, 4.2.2 and 4.2.3 help us to gain an insight into the state-controllability of SoS* Σ *(4.1). For arbitrary i-th CS, it's state-controllability mainly relates to the following two aspects:*

- *controllability of pair* $(A_i, \theta_i B_i)$;
- *measured outputs from neighbor* N_i *via network.*

For a CS Σ_i *of neighbor set* $N_i = \emptyset$, *pair* $(A_i, \theta_i B_i)$ *has to be state-controllable due to there is no external outputs (Corollary 4.2.1). Meanwhile, for an uncontrollable pair* $(A_i, \theta_i B_i)$, *the neighbor set of CS* Σ_i *cannot be empty (Corollary 4.2.3). Moreover, in Corollary 4.2.2,* $rank([H_{ij}, B_i]) = rank(B_i)$ *implies* $Im(H_{ij}) \subset Im(B_i)$. *Thus,* $(A_i, \theta_i B_i)$ *must be controllable owing to neighbor's output cannot provide extra controllable space for i-th CS.*

Corollary 4.2.4 *Suppose Assumption 4.2.2 holds. SoS (4.1) subjects to the directed chain topology graph* $\mathcal{G}$ *in Fig.4.2. The following results hold. SoS (4.1) is state-controllable*

p1. ***only if*** $\theta_1 = 1$.

p2. ***if*** *for* $i = 2, ..., N-1$, $Ker(C_i^T H_{i+1,i}^T) \cap Ker(\theta_{i+1} B_{i+1}^T) = \boldsymbol{0}_{n_{i+1}}$. *And for arbitrary* $s \in \mathbb{C}$,

$$rank(\begin{bmatrix} sI - A_1 & B_1 \\ \gamma_{21} H_{21} C_1 & \boldsymbol{0} \end{bmatrix}) = n_1 + n_2. \tag{4.6}$$

Proof: (**necessity of p1**) Due to neighbor set $N_1 = \emptyset$, the results can be obtained directly from Corollary 4.2.1.

(**necessity and sufficiency of p2**) For the SoS under digraph in Fig.4.2, equations (4.4) in Theorem 4.2.1 are transformed into

$$\begin{cases} \xi_i^T(sI_{n_i} - A_i) - \gamma_{i+1,i}\xi_{i+1}^T H_{i+1,i}C_i = \mathbf{0}, i = 1, 2, ..., N-1, \\ \xi_N^T(sI_{n_N} - A_N) = \mathbf{0}, \\ \theta_i\xi_i^T B_i = \mathbf{0}, i = 1, 2, ..., N. \end{cases} \tag{4.7}$$

Based on equation (4.6), there only exists zero solution $\xi_1 = \mathbf{0}_{n_1}$ and $\xi_2 = \mathbf{0}_{n_2}$ such that $\xi_1^T(sI_{n_1} - A_1) - \gamma_{2,1}\xi_2^T H_{2,1}C_2 = \mathbf{0}$. Thus, $\xi_3^T H_{32}C_2 = 0$. Due to $\theta_3\xi_3^T B_3 = 0$ and $Ker(C_2^T H_{32}^T) \cap Ker(\theta_3 B_3^T) = \mathbf{0}_{n_3}$, $\xi_3 = \mathbf{0}_{n_3}$. It follows in the same procedure that $\xi_i = \mathbf{0}_{n_i}, i = 3, ..., N$. Thus, equations (4.7) only have zero solutions. ■

Figure 4.2: An example of two-layer SoS over a directed chain graph $\mathcal{G}$

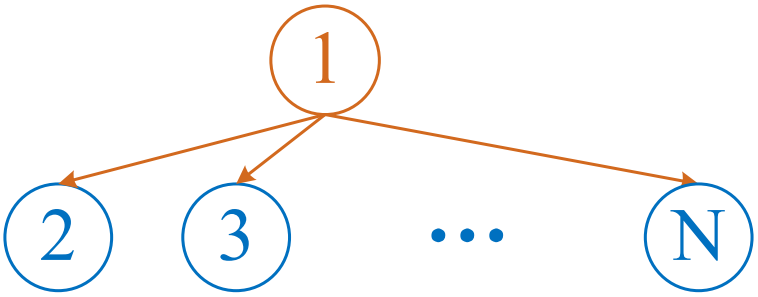


Figure 4.3: An example of two-layer SoS over a directed star graph $\mathcal{G}$

Corollary 4.2.5 *Suppose Assumption 4.2.2 holds. SoS (4.1) subjects to the directed star topology graph $\mathcal{G}$ in Fig.4.3. The following results hold. SoS (4.1) is state-controllable*

p1. ***only if*** $\theta_1 = 1$.

p2. ***if and only if*** *for arbitrary $s \in \cup_{i=2,i\in Unc}^{n}\sigma_{Unc}(A_i, \theta_i B_i)$, there only exists zero solution $\xi = [\xi_1^T, \xi_2^T, ..., \xi_N^T]^T \in \mathbb{C}^{\sum_{i=1}^N n_i}$ to equations*

$$\begin{cases} \xi_1^T(sI_{n_1} - A_1) - \sum_{j=2,j\in Unc}^{N} \gamma_{j1}\xi_j^T H_{j1}C_1 = \boldsymbol{0} \\ \theta_1\xi_1^T B_1 = \boldsymbol{0}, \end{cases} \tag{4.8}$$

and

$$\begin{cases} \xi_i^T(sI_{n_i} - A_i) = \boldsymbol{0} \\ \theta_i\xi_i^T B_i = \boldsymbol{0}, i \in Unc. \end{cases} \tag{4.9}$$

where $\sigma_{Unc}(A_i, \theta_i B_i)$ denotes the $(A_i, \theta_i B_i)$ uncontrollable eigenvalue set of A_i. That is, for all $s \in \sigma_{Unc}(A_i, \theta_i B_i)$, there exists nonzero $\xi_i \in \mathbb{C}^{n_i}$ such that equation (4.9) exists. In addition, Unc is the index set of uncontrollable systems, where $Unc = \{i | (A_i, B_i)$ is uncontrollable or $\theta_i = 0, i = 2, 3, ..., N\}$.

p3. ***if*** *$\forall i, j \in Unc, i \neq j, \sigma_{Unc}(A_i, \theta_i B_i) \cap \sigma_{Unc}(A_j, \theta_j B_j) = \emptyset$. In addition, $\theta_1 = 1$ and for arbitrary $s \in \cup_{i=2, i \in Unc}^{n} \sigma_{Unc}(A_i, \theta_i B_i)$, if $s \in \sigma_{Unc}(A_i, \theta_i B_i)$ and $i \in Unc$, one has*

$$rank(\begin{bmatrix} sI - A_1 & B_1 \\ \gamma_{i1} H_{i1} C_1 & \boldsymbol{0} \end{bmatrix}) = n_1 + n_i. \tag{4.10}$$

Proof: For the SoS under digraph in Fig.4.3, equations (4.4) in Theorem 4.2.1 are transformed into

$$\begin{cases} \xi_1^T (sI_{n_1} - A_1) - \sum_{j=1}^{N} \gamma_{j1} \xi_j^T H_{j1} C_1 = \mathbf{0}, \\ \theta_1 \xi_1^T B_1 = \mathbf{0}, \end{cases} \tag{4.11}$$

and

$$\begin{cases} \xi_i^T (sI_{n_i} - A_i) = \mathbf{0}, \\ \theta_i \xi_i^T B_i = \mathbf{0}, i = 2, ..., N. \end{cases} \tag{4.12}$$

(**necessity of p1**) Due to neighbor set $N_1 = \emptyset$, the results can be obtained directly from Corollary 4.2.1.

(**necessity and sufficiency of p2**) The necessity is obvious. For sufficiency proof, the value of $s \in \mathbb{C}$ can be divided into the following three cases. Firstly, if $s \notin \cup_{i=2}^{n} \sigma(A_i)$, then equation (4.12) exists only for $\xi_i = 0_{n_i}, i = 2, 3, ..., N$. Secondly, if $s \notin \cup_{i=2, i \in Unc} \sigma_{Unc}(A_i, \theta_i B_i)$ but $s \in \cup_{i=2}^{n} \sigma(A_i)$, equation (4.12) also, exists only for $\xi_i = 0_{n_i}, i = 2, 3, ..., N$. Meanwhile, equation (4.11) turns into $\begin{cases} \xi_1^T (sI_{n_1} - A_1) = \mathbf{0}, \\ \theta_1 \xi_1^T B_1 = \mathbf{0}. \end{cases}$ Due to $(A_1, \theta_1 B_1)$ is state-controllable, $\xi_1 = 0_{n_1}$. Thus, there only exists zero solutions for equations (4.11) and (4.12). Thirdly, for arbitrary $s \in \cup_{i=2, i \in Unc}^{n} \sigma_{Unc}(A_i, \theta_i B_i)$, for $i \notin Unc$, equation (4.12) exists only for $\xi_i = 0_{n_i}, i \notin Unc$. Therefore, there only exists zero solutions to equations (4.11) and (4.12), if and only if, there only exists zero solutions to equations (4.8) and (4.9).

(**sufficiency of p3**) For $s \in \cup_{i=2, i \in Unc}^{n} \sigma_{Unc}(A_i, \theta_i B_i)$, owing to $\forall i, j \in Unc, i \neq j, \sigma_{Unc}(A_i, \theta_i B_i) \cap \sigma_{Unc}(A_j, \theta_j B_j) = \emptyset$, equations (4.8) and (4.9) exist only for $s = s^* \in \sigma_{Unc}(A_{i^*}, \theta_{i^*} B_{i^*}), i^* \in Unc$ and ξ such that $\xi_i = 0_{n_i}, i \neq 1$ or i^*. However, in terms of (4.10), one obtains $rank(\begin{bmatrix} s^* I - A_1 & B_1 \\ \gamma_{i^*1} H_{i^*1} C_1 & \mathbf{0} \end{bmatrix}) = n_1 + n_{i^*}$, which implies there only exists zero solutions $\xi_1 = \mathbf{0}_{n_1}$ and $\xi_{i^*}^T = \mathbf{0}_{n_{i^*}}$. Hence, SoS (4.1) is state-controllable based on **p2**. ■

4.2.2 Controllability analysis based on Sylvester equation

Above theorems inspire us that the controllability of SoS (4.1) relies on the devise of 1-th virtual model of MCS (i.e., virtual triple (C_1, A_1, B_1)). It may lead to

reduced analysis complexity if a part of "heterogeneous" terms of PCSs can be transformed into "homogeneous" as the 1-th virtual system, such as the widely applied output regulation approach in heterogeneous large-scale systems [Su 2011]. Motivated by the widely applied output regulation method in coordination control of heterogeneous MASs [Ma 2015, Yaghmaie 2016], the following studies based on the Sylvester equations are addressed.

Assumption 4.2.3 *For $i = 2, ..., N$, there exist suitable nonzero solutions $\Pi_i \in \mathbb{R}^{n_i \times n_1}$ and solutions $F_i \in \mathbb{R}^{q_i \times n_1}$, $Q_i \in \mathbb{R}^{m_1 \times m_i}$, $H_i \in \mathbb{R}^{n_1 \times m_1}$, $d_i \in \mathbb{R}$ such that*

$$\Pi_i(A_1 + d_i H_i C_1) = A_i \Pi_i + \theta_i B_i F_i, \tag{4.13}$$

$$Q_i C_i \Pi_i = C_1. \tag{4.14}$$

Set $\Pi_1 = I_{n_1}$, $Q_1 = I_{m_1}$, $\Pi = diag\{\Pi_1, \Pi_2, ..., \Pi_N\}$ and $Q = diag\{Q_1, Q_2, ..., Q_N\}$.

Remark 4.2.3 *When $\theta_i \neq 0$, in terms of [Barlow 1992, Wu 2007], there exists unique solution Π_i to equation (4.13) for arbitrary selected F_i if and only if for $\sigma(A_1 - d_i H_i C_1) \cap \sigma(A_i) = \emptyset$. For $\theta_i = 0$, there exists nonzero solution to equation (4.13) only if $\sigma(A_1 - d_i H_i C_1) \cap \sigma(A_i) \neq \emptyset$. Actually, if (C_1, A_1) is observable, the arbitrary pole placement of matrix $A_1 - d_i H_i C_1$ can be ensured. Meanwhile, the existence of Q_i to equation (4.14) is related to the row space matrix C_1. Due to triple (C_1, A_1, B_1) is a virtual model and can be devised by the users regarding the autonomy of the 1-th MCS, above mentioned requirements are easy to be satisfied. Besides, until now, we have witnessed various algorithms to compute the full rank solution Π_i for Sylvester equation (4.13), such as [Carvalho 2003, Datta 2011]. Moreover, when $d_i = 0$, Assumption 4.2.3 is widely adopted in the coordination control of heterogeneous MASs [Ma 2015, Yaghmaie 2016], which is known as output regulation [Su 2011] or internal model control [Francis 1976].*

Theorem 4.2.2 *Suppose Assumption 4.2.3 holds. Set inner-coupling matrix as $H_{ij} = \Pi_i H_j Q_j$, where $\Pi_i \in \mathbb{R}^{n_i \times n_1}$ is the solution to equations (4.13) and (4.14) with $\Pi_1 = I_{n_1}$ and $H_i \in \mathbb{R}^{n_1 \times m_1}$. SoS (4.1) is state-controllable if*

$$rank([\Pi(I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1} \hat{H} I_N \otimes C_1), \Omega]) = \sum_{i=1}^{N} n_i, \tag{4.15}$$

where $\hat{H} = diag\{H_1, H_2, ..., H_N\}$ and $D = diag\{d_1, d_2, ..., d_N\}$ with $d_1 = 0$. Above sufficient condition is equivalent to there only exists zero solution $\xi = [\xi_1^T, \xi_2^T, ..., \xi_N^T]^T \in \mathbb{C}^{\sum_{i=1}^{N} n_i}, \xi_i \in \mathbb{C}^{n_i}, i = 1, 2, ..., N$ to the following equations,

$$\begin{cases} \xi_i^T \Pi_i (sI_{n_1} - A_1 - d_i H_i C_1) - \sum_{j=1}^{N} \gamma_{ji} \xi_j^T \Pi_j H_i C_1 = \boldsymbol{0}, \\ \theta_i \xi_i^T B_i = \boldsymbol{0}, i = 1, 2, ..., N, \end{cases} \tag{4.16}$$

for arbitrary $s \in \mathbb{C}$.

Proof: Suppose SoS (4.1) is not state-controllable, upon Theorem 4.2.1 there exists one nonzero vector $\xi^* = [(\xi_1^*)^T, (\xi_2^*)^T, ..., (\xi_N^*)^T]^T \in \mathbb{C}^{\sum_{i=1}^N n_i}$ such that equations (4.4) exist for $s = s^* \in \mathbb{C}$. Right multiplying Π_i on the first equation of (4.4), for $i = 2, 3, ..., N$ it yields

$$(\xi_i^*)^T(s^*\Pi_i - A_i\Pi_i - \theta_i B_i F_i) - \sum_{j=1}^{N} \gamma_{ji}(\xi_j^*)^T H_{ji} C_i \Pi_i = \mathbf{0},$$

where $H_{ji} = \Pi_j H_i Q_i$. Besides, $\theta_i(\xi_i^*)^T B_i F_i = \mathbf{0}$ due to $\theta_i(\xi_i^*)^T B_i = \mathbf{0}$. In terms of Assumption 4.2.3, one has

$$(\xi_i^*)^T\Pi_i(s^* I_{n_1} - (A_1 + d_i H_i C_1)) - \sum_{j=1}^{N} \gamma_{ji}(\xi_j^*)^T \Pi_j H_i C_1 = \mathbf{0}.$$

For $i = 1$, due to $\Pi_1 = I_{n_1}$ and $d_1 = 0$, $(\xi_1^*)^T\Pi_1(sI_{n_1} - (A_1 + d_1 H_1 C_1)) - \sum_{j=1}^N \gamma_{j1}(\xi_j^*)^T\Pi_j H_1 C_1 = (\xi_1^*)^T(sI_{n_1} - A_1) - \sum_{j=1}^N \gamma_{j1}(\xi_j^*)^T\Pi_j H_1 C_1 = \mathbf{0}$. It therefore follows that there exists a nonzero solution ξ^* to equations (4.16) with $s = s^* \in \mathbb{C}$. Rewrite (4.16) in a compact form

$$\begin{cases} \xi^T\Pi(I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1}\hat{H} I_N \otimes C_1) = \mathbf{0}, \\ \xi^T\Omega = \mathbf{0}, i = 1, 2, ..., N, \end{cases} \tag{4.17}$$

which is equal to condition (4.15) if there only exists zero solution ξ to (4.17). ■

Theorem 4.2.3 *Suppose Assumption 4.2.3 holds. Set inner-coupling matrix $H_{ij} = \Pi_i H_j Q_j, j \neq i$, where $\Pi_i \in \mathbb{R}^{n_i \times n_1}$ is the solution to equations (4.13) and (4.14) with $\Pi_1 = I_{n_1}$ and $H_i \in \mathbb{R}^{n_1 \times m_1}$. If for $i = 2, ..., N$, Π_i is full row rank, then SoS (4.1) is state-controllable if*

$$\begin{aligned} &rank([I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1}\hat{H} I_N \otimes C_1, \Theta \otimes I_{n_1}\Pi^{-1}\tilde{B}]) \\ &= nN_1, \end{aligned} \tag{4.18}$$

where $\Pi^{-1} = diag\{\Pi_1^{-1}, \Pi_2^{-1}, ..., \Pi_N^{-1}\}$. Π_i^{-1} is the right inverse of Π_i such that $\Pi_i\Pi_i^{-1} = I_{n_i}$. Above necessary and sufficient condition is equivalent to there only exists zero solution $\eta = [\eta_1^T, \eta_2^T, ..., \eta_N^T]^T \in \mathbb{C}^{Nn_1}, \eta_i \in \mathbb{C}^{n_1}, i = 1, 2, ..., N$ to the following equations,

$$\begin{cases} \eta_i^T(sI_{n_1} - A_1 - d_i H_i C_1) - \sum_{j=1}^N \gamma_{ji}\eta_j^T H_j C_1 = \boldsymbol{0}, \\ \theta_i \eta_i^T \Pi_i^{-1} B_i = 0, i = 1, 2, ..., N, \end{cases} \tag{4.19}$$

for arbitrary $s \in \mathbb{C}$.

Proof: If Π_i is full row rank for $i = 2, ..., N$, there must exist a right inverse Π_i^{-1} such that $\Pi_i\Pi_i^{-1} = I_{n_i}$, where $\Pi_i^{-1} \in \mathbb{R}^{n_1 \times n_i}$ is full column rank.

Suppose SoS (4.1) is not state-controllable, upon Theorem 4.2.2 there exists one nonzero vector $\xi^* = [(\xi_1^*)^T, (\xi_2^*)^T, ..., (\xi_N^*)^T]^T \in \mathbb{C}^{\sum_{i=1}^N n_i}$ to equations (4.16) with $s = s^* \in \mathbb{C}$. Set $\eta_i^* = \Pi_i^T \xi_i^*, i = 2, 3, ..., N$ and $\eta_1 = \xi_1$. Owing to $\Pi_1 = I_{n_1}$,

$$(\xi_i^*)^T \Pi_i = (\eta_i^*)^T, i = 1, 2, ..., N. \tag{4.20}$$

Substituting (4.20) into (4.16), it yields $\eta^* = [(\eta_1^*)^T, (\eta_2^*)^T, ..., (\eta_N^*)^T]^T \in \mathbb{C}^{Nn_1}, \eta_i^* \in \mathbb{C}^{n_1}, i = 1, 2, ..., N$ is a nonzero solution to equation (4.19) with $s = s^* \in \mathbb{C}$. Our proof is thus finalized. ■

Lemma 4.2.1 *Suppose Assumption 4.2.3 holds. Set inner-coupling matrix $H_{ij} = \Pi_i H_j Q_j$, where $\Pi_i \in \mathbb{R}^{n_i \times n_1}$ is the solution to equations (4.13) and (4.14) with $\Pi_1 = I_{n_1}$ and $H_i \in \mathbb{R}^{n_1 \times m_1}$. If for $i = 2, ..., N$, Π_i is full row rank, then SoS (4.1) is state-controllable if*

$$\begin{aligned} &rank(\begin{bmatrix} I_N \otimes (sI_{n_1} - A_1) & \Theta \otimes I_{n_1} \Pi^{-1} \tilde{B} & (\Gamma + D) \otimes I_{n_1} \hat{H} \\ I_N \otimes C_1 & \boldsymbol{0} & I_{Nm_1} \end{bmatrix}) \\ &= N(n_1 + m_1). \end{aligned} \tag{4.21}$$

Proof: It's easy to see that equation (4.18) exists if and only if $rank(\Psi) = N(n_1 + m_1)$, where

$$\Psi = \begin{bmatrix} \hat{\Psi} & \Theta \otimes I_{n_1} \Pi^{-1} \tilde{B} & \mathbf{0} \\ I_N \otimes C_1 & \mathbf{0} & I_{Nm_1} \end{bmatrix} \tag{4.22}$$

and $\hat{\Psi} = I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1} \hat{H} I_N \otimes C_1$. Notice that $\begin{bmatrix} I_{Nn_1} & (\Gamma + D) \otimes I_{n_1} \hat{H} \\ \mathbf{0} & I_{Nm_1} \end{bmatrix} \Psi = \begin{bmatrix} I_N \otimes (sI_{n_1} - A_1) & \Theta \otimes I_{n_1} \Pi^{-1} \tilde{B} & (\Gamma + D) \otimes I_{n_1} \hat{H} \\ I_N \otimes C_1 & 0 & I_{Nm_1} \end{bmatrix}$. Thus, equation (4.18) is equivalent to equation (4.21). It thus proceeds with Theorem 4.2.3 that SoS (4.1) is controllable if equation (4.21) exists. ■

Lemma 4.2.2 *Assume for $i = 2, ..., N$, Π_i is full row rank and the coupling matrix H_i in Assumption 4.2.3 satisfies $H_i = H \in \mathbb{R}^{n_1 \times m_1}$ for $d_i \neq 0$. Equation (4.18) holds only if pair $(\Gamma + D, \Theta)$ is state-controllable.*

Proof: Suppose pair $(\Gamma + D, \Theta)$ is uncontrollable. Then there must exist a nonzero vector $z_1 = [z_{11}, z_{12}, ..., z_{1N}]^T \in \mathbb{C}^N$ and a scalar $s_1 \in \mathbb{C}$ such that

$$\begin{cases} z_1^T (s_1 I_N - (\Gamma + D)) = \mathbf{0}, \\ z_1^T \Theta = \mathbf{0}. \end{cases} \tag{4.23}$$

Let (s_2, z_2) be one left eigenvalue and eigenvector pair of matrix $A_1 + s_1 H C_1$, where $z_2 \in \mathbb{C}^{n_1}$ and a scalar $s_2 \in \mathbb{C}$. It thus follows that there exists nonzero vector $z_1 \otimes z_2$

and complex number s_2 such that

$$\begin{aligned}&(z_1^T \otimes z_2^T)[I_N \otimes (s_2 I_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1} \hat{H} I_N \otimes C_1] \\ &= (z_1^T \otimes z_2^T)[I_N \otimes (s_2 I_{n_1} - A_1) - s_1 I_N \otimes HC_1] \\ &= (z_1^T \otimes z_2^T)[I_N \otimes (s_2 I_{n_1} - (A_1 + s_1 HC_1))] = 0.\end{aligned} \tag{4.24}$$

Meanwhile, $(z_1^T \otimes z_2^T)(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = (z_1^T \Theta \otimes z_2^T)\Pi^{-1}\tilde{B} = 0$. It follows that $rank([I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes HC_1, \Theta \otimes I_{n_1}\Pi^{-1}\tilde{B}]) < nN_1$. ■

Lemma 4.2.3 *Assume for $i = 2, ..., N$, Π_i is full row rank. Equation (4.18) holds only if pair $(I_N \otimes A_1 + (\Gamma + D) \otimes I_{n_1} \hat{H} I_N \otimes C_1, \Theta \times I_{n_1})$ is controllable, that is homogeneous networked SoS,*

$$\begin{cases} \dot{x}_i(t) = A_1 x_i(t) + H_i(\sum_{j=1}^N \gamma_{ij} y_j(t) + d_i y_i(t)) + \theta_i u_i(t) \\ \qquad\qquad\qquad\qquad i = 1, 2, 3, ..., N, \\ y_i(t) = C_1 x_i(t), x_i(0) = x_{i0}, \end{cases} \tag{4.25}$$

is controllable.

Proof: Suppose SoS (4.25) is uncontrollable. Then, there must exist a nonzero vector $\eta^* = [(\eta_1^*)^T, (\eta_2^*)^T, ..., (\eta_N^*)^T]^T \in \mathbb{C}^{Nn_1}$ and a scalar $s^* \in \mathbb{C}$ such that

$$\begin{cases} \eta_i^T(sI_{n_1} - A_1 - d_i H_i C_1) - \sum_{j=1}^N \gamma_{ji} \eta_j^T H_j C_1 = \mathbf{0}, \\ \theta_i \eta_i^T = 0, i = 1, 2, ..., N. \end{cases} \tag{4.26}$$

Thus, vector η^* and scalar s^* are also nonzero solution to equation (4.19), which implies equation (4.18) does not hold. ■

Next, we consider the situation that matrix $\Gamma + D$ is diagonalizable, that is, there exists nonsingular matrix $P = [p_{ij}]$ such that $P(\Gamma + D)P^{-1} = \Lambda = \text{Diag}\{\lambda_1, \lambda_2, ..., \lambda_N\}, \lambda_i \in \sigma(\Gamma + D)$. Next, if we set $H_i = H$, then equation (4.18) holds if and only if the following pair is controllable

$$(I_N \otimes A_1 - (\Gamma + D) \otimes HC_1, (\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B}) \tag{4.27}$$

It's obvious that

$$\begin{aligned}&(P \otimes I_{n_1})(I_N \otimes A_1 - (\Gamma + D) \otimes HC_1)(P^{-1} \otimes I_{n_1}) \\ =&(I_N \otimes A_1 - \Lambda \otimes HC_1) = \begin{bmatrix} S + \lambda_1 HC_1 & & & \\ & S + \lambda_2 HC_1 & & \\ & & \ddots & \\ & & & S + \lambda_N HC_1 \end{bmatrix},\end{aligned} \tag{4.28}$$

which implies $\sigma(I_N \otimes A_1 - (\Gamma + D) \otimes HC_1) = \cup_{i=1}^N \sigma(A_1 + \lambda_i HC_1)$. Denote $\sigma(A_1 + \lambda_i HC_1) \triangleq \{\gamma_1^i, \gamma_2^i, ..., \gamma_{q_i}^i\}$. Let $E_j^i \triangleq \{\eta_{ij}^1, \eta_{ij}^2, ..., \eta_{ij}^{\varphi_{ij}}\}$ be a basis of characteristic

subspace for matrix $(A_1+\lambda_i HC_1)^T$ related to eigenvalue $\gamma_j^i, j=1,2,...,q_i$ (i.e., the basis of all left eigenvectors of $A_1+\lambda_i HC_1$ related to eigenvalue γ_j^i), where φ_{ij} is the geometric multiplicity of eigenvalue γ_j^i for $A_1+\lambda_i HC_1$.

Let $p_i=[p_{i1},p_{i2},...,p_{iN}]^T, i=1,2,...,N$, then $P=[p_1,p_2,...,p_N]^T$. Due to $(P\otimes I_{n_1})(I_N\otimes A_1-(\Gamma+D)\otimes HC_1)(P^{-1}\otimes I_{n_1})=(I_N\otimes A_1-\Lambda\otimes HC_1)$, one has $(P\otimes I_{n_1})(I_N\otimes A_1-(\Gamma+D)\otimes HC_1)=(I_N\otimes A_1-\Lambda\otimes HC_1)(P\otimes I_{n_1})$, which reveals

$$(p_i^T\otimes I_{n_1})(I_N\otimes A_1-(\Gamma+D)\otimes HC_1)(P^{-1}\otimes I_{n_1})=(A_1+\lambda_i HC_1)(p_i^T\otimes I_{n_1}). \tag{4.29}$$

It readily follows that for $k=1,2,...,\varphi_{ij}$,

$$\begin{aligned}&(\eta_{ij}^k)^T(p_i^T\otimes I_{n_1})(I_N\otimes A_1-(\Gamma+D)\otimes HC_1)(P^{-1}\otimes I_{n_1})\\ =&(\eta_{ij}^k)^T(A_1+\lambda_i HC_1)(p_i^T\otimes I_{n_1})=\gamma_j^i(\eta_{ij}^k)^T(p_i^T\otimes I_{n_1}),\end{aligned} \tag{4.30}$$

which reflects $(\eta_{ij}^k)^T(p_i^T\otimes I_{n_1}), k=1,2,...,\varphi_{ij}$ are the left eigenvectors of matrix $I_N\otimes A_1-(\Gamma+D)\otimes HC_1$ related to eigenvalue γ_j^i. In addition,

$$(\eta_{ij}^k)^T(p_i^T\otimes I_{n_1})=[p_{i1}\gamma_j^i(\eta_{ij}^k)^T,p_{i2}\gamma_j^i(\eta_{ij}^k)^T,...,p_{iN}\gamma_j^i(\eta_{ij}^k)^T]=p_i^T\otimes(\eta_{ij}^k)^T. \tag{4.31}$$

Based on the above derivation we can obtain the following results.

Theorem 4.2.4 *Suppose Assumption 4.2.3 holds. Set inner-coupling matrix $H_{ij}=\Pi_i H_j Q_j, j\neq i$, where $\Pi_i\in\mathbb{R}^{n_i\times n_1}$ is the solution to equation (4.13) and (4.14) with $\Pi_1=I_{n_1}$ and $H_i=H\in\mathbb{R}^{n_1\times m_1}$. If for $i=2,...,N$, Π_i is full row rank and matrix $\Gamma+D\in\mathbb{R}^{N\times N}$ is diagonalizable, then*

*(**i**). SoS (4.1) is state-controllable **if** conditions **c.1**-**c.3** hold.*

- ***c.1*** *For arbitrary $i\neq j, i,j=1,2,...,N$, $\sigma(A_1+\lambda_i HC_1)\cap\sigma(A_1+\lambda_j HC_1)=\emptyset$;*
- ***c.2*** *Pair $(\Gamma+D,\Theta)$ is controllable;*
- ***c.3*** *For all $\lambda\in\sigma(\Gamma+D)$,*

$$(A_1-\lambda HC_1,\Pi_i^{-1}B_i),\theta_i\neq 0 \tag{4.32}$$

is controllable.

*(**ii**). SoS (4.1) is state-controllable **if** conditions **c.2**-**c.4** hold.*

- ***c.4*** *If there exists a common eigenvalue λ for matrices $A_1+\lambda_{i_1}HC_1,...,A_1+\lambda_{i_l}HC_1, l\geq 2$, where $\gamma_{k_m}^{i_m}=\lambda\in\cap_{m=1}^l\sigma(A_1+\lambda_{i_m}HC_1), m=1,...,l$, then vectors $[\theta_1 p_{i_1 1}(\eta_{i_1k_1}^1)^T\Pi_1^{-1}B_1,\theta_2 p_{i_1 2}(\eta_{i_1k_1}^1)^T\Pi_2^{-1}B_2,...,\theta_N p_{i_1 N}(\eta_{i_1k_1}^1)^T\Pi_N^{-1}B_N]$,...,*
  $[\theta_1 p_{i_1 1}(\eta_{i_1k_1}^{\varphi_{i_1k_1}})^T\Pi_1^{-1}B_1,\theta_2 p_{i_1 2}(\eta_{i_1k_1}^{\varphi_{i_1k_1}})^T\Pi_2^{-1}B_2,...,\theta_N p_{i_1 N}(\eta_{i_1k_1}^{\varphi_{i_1k_1}})^T\Pi_N^{-1}B_N]$,...,
  $[\theta_1 p_{i_l 1}(\eta_{i_lk_l}^1)^T\Pi_1^{-1}B_1,\theta_2 p_{i_l 2}(\eta_{i_lk_l}^1)^T\Pi_2^{-1}B_2,...,\theta_N p_{i_l N}(\eta_{i_lk_l}^1)^T\Pi_N^{-1}B_N]$,...,
  $[\theta_1 p_{i_l 1}(\eta_{i_lk_l}^{\varphi_{i_lk_l}})^T\Pi_1^{-1}B_1,\theta_2 p_{i_l 2}(\eta_{i_lk_l}^{\varphi_{i_lk_l}})^T\Pi_2^{-1}B_2,...,\theta_N p_{i_l N}(\eta_{i_lk_l}^{\varphi_{i_lk_l}})^T\Pi_N^{-1}B_N]$

are linearly independent. Point out that $p_{i_m}^T, m = 1, 2, ..., l$ *is the* i_m*-th column of matrix* P^T *(also the left eigenvalue of matrix* $\Gamma + D$ *related to eigenvalue* λ_{i_m}*).* $\{\eta_{i_m k_m}^1, ..., \eta_{i_m k_m}^{\varphi_{i_m k_m}}\}, m = 1, 2, ..., l$ *are a basis of all left eigenvectors of matrix* $A_1 + \lambda_{i_m} HC_1$ *related to eigenvalue* $\gamma_{k_m}^{i_m}$*.*

Proof: Suppose SoS (4.1) is uncontrollable, then there exists a nonzero vector $\eta \in \mathbb{R}^{Nn_1}$ with $s \in \mathbb{C}$ such that $\eta^T(sI_{Nn_1} - (I_N \otimes A_1 - (\Gamma + D) \otimes HC_1)) = \mathbf{0}_{Nn_1}$ and $\eta^T(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = \mathbf{0}_{Nn_1}$. Thus, one has $s \in \sigma(I_N \otimes A_1 - (\Gamma + D) \otimes HC_1) = \cup_{i=1}^N \sigma(A_1 + \lambda_i HC_1)$ and η^T is the left eigenvector of matrix $I_N \otimes A_1 - (\Gamma + D) \otimes HC_1$ corresponding to eigenvalue s.

(**i**) Based on **c.1**, for arbitrary $s = \gamma_j^i \in \sigma(A_1 + \lambda_i HC_1), i = 1, 2, ..., N$, $s \notin \cap_{k=1, k\neq i}^N \sigma(A_1 + \lambda_k HC_1)$. As per (4.30), every left eigenvector η of $I_N \otimes A_1 - (\Gamma + D) \otimes HC_1$ can be decomposed into a linear combination $\eta = \sum_{m=1}^{\varphi_{ij}} \beta_m(p_i) \otimes (\eta_{ij}^m)$. It follows from $\eta^T(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = \mathbf{0}_{Nn_1}$ that

$$
\begin{aligned}
&\eta^T(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = (\sum_{m=1}^{\varphi_{ij}} \beta_m(p_i^T) \otimes (\eta_{ij}^m)^T)((\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B}) \\
=&((p_i^T\Theta) \otimes (\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T))(\Pi^{-1}\tilde{B}) = (\sum_{m=1}^{\varphi_{ij}} \beta_m(p_i^T\Theta) \otimes (\eta_{ij}^m)^T)(\Pi^{-1}\tilde{B}) \\
=&[\theta_1 p_{i1}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T)), \theta_2 p_{i2}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T)), ..., \theta_N p_{iN}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T))] \\
&\times \begin{bmatrix} \Pi_1^{-1}B_1 & & & \\ & \Pi_2^{-1}B_2 & & \\ & & \ddots & \\ & & & \Pi_N^{-1}B_N \end{bmatrix} \\
=&[\theta_1 p_{i1}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T))\Pi_1^{-1}B_1, \theta_2 p_{i2}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T))\Pi_2^{-1}B_2, ..., \\
&\theta_N p_{iN}(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T))\Pi_N^{-1}B_N] = \mathbf{0}
\end{aligned} \tag{4.33}
$$

Note that $\sum_{m=1}^{\varphi_{ij}} \beta_m \eta_{ij}^m$ is the left eigenvector of matrix $A_1 + \lambda_i HC_1$ related to eigenvalue γ_j^i. In view of **c.3**,

$$
(\sum_{m=1}^{\varphi_{ij}} \beta_m(\eta_{ij}^m)^T)\Pi_j^{-1}B_j \neq \mathbf{0}, \text{ if } \theta_j \neq 0. \tag{4.34}
$$

Additionally, owing to $(\Gamma + D, \Theta)$ is controllable (**c.2**)

$$
[\theta_1 p_{i1}, \theta_2 p_{i2}, ..., \theta_N p_{iN}] \neq \mathbf{0}_N^T \tag{4.35}
$$

Since $\theta_j p_{ij} = 0$ if $\theta_j = 0$, there must exist at least one $j^* = 1, 2, ..., N$ such that

$$\theta_{j^*} p_{ij^*} (\sum_{m=1}^{\varphi_{ij}} \beta_m (\eta_{ij}^m)^T)) \Pi_{j^*}^{-1} B_{j^*} \neq \mathbf{0}, \tag{4.36}$$

which violates (4.33). Consequently, SoS (4.1) is controllable.

(**ii**) If $s \in \sigma(A_1 + \lambda_i HC_1)$ but $s \notin \cup_{k=1,k\neq j}^{N} \sigma(A_1 + \lambda_k HC_1)$, then the derivation to verify $\eta^T(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = \mathbf{0}_{Nn_1}$ is violated is the same to the proof of (**i**). If s is a common eigenvalue of matrices $A_1 + \lambda_{i_1} HC_1, A_1 + \lambda_{i_2} HC_1, ..., A_1 + \lambda_{i_l} HC_1, l \geq 2$, we denote the related eigenvalue as $s = \gamma_{k_m}^{i_m} \in \sigma(A_1 + \lambda_{i_m} HC_1) = \{\gamma_1^{i_m}, \gamma_2^{i_m}, ..., \gamma_{q_{i_m}}^{i_m}\}, m = 1, ..., l, k_m \in \{1, 2, ..., q_{i_m}\}$. In terms of (4.30), η can be decomposed into a linear combination $\eta = \sum_{m=1}^{l} \sum_{j=1}^{\varphi_{i_m k_m}} \omega_k^{i_m}(p_{i_m}) \otimes (\eta_{i_m k_m}^j)$. It thus follows that

$$\begin{aligned}
&\eta^T(\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B} = \sum_{m=1}^{l} \sum_{j=1}^{\varphi_{i_m k_m}} \omega_k^{i_m}(p_{i_m})^T \otimes (\eta_{i_m k_m}^j)^T ((\Theta \otimes I_{n_1})\Pi^{-1}\tilde{B}) \\
&= \sum_{m=1}^{l} \sum_{j=1}^{\varphi_{i_m k_m}} \omega_k^{i_m}(p_{i_m}^T \Theta) \otimes (\eta_{i_m k_m}^j)^T (\Pi^{-1}\tilde{B}) \\
&= \sum_{m=1}^{l} \sum_{j=1}^{\varphi_{i_m k_m}} \omega_k^{i_m}(p_{i_m}^T \Theta) \otimes (\eta_{i_m k_m}^j)^T \begin{bmatrix} \Pi_1^{-1} B_1 & & & \\ & \Pi_2^{-1} B_2 & & \\ & & \ddots & \\ & & & \Pi_N^{-1} B_N \end{bmatrix} \\
&= \sum_{m=1}^{l} \sum_{j=1}^{\varphi_{i_m k_m}} [\theta_1 p_{i_m 1} (\eta_{i_m k_m}^j)^T \Pi_1^{-1} B_1, \theta_2 p_{i_m 2} (\eta_{i_m k_m}^j)^T \Pi_2^{-1} B_2, ..., \\
&\theta_N p_{i_m N} (\eta_{i_m k_m}^j)^T \Pi_N^{-1} B_N] = \mathbf{0},
\end{aligned} \tag{4.37}$$

which reveals vectors $[\theta_1 p_{i_1 1}(\eta_{i_1 k_1}^1)^T \Pi_1^{-1} B_1, \theta_2 p_{i_1 2}(\eta_{i_1 k_1}^1)^T \Pi_2^{-1} B_2, ..., \theta_N p_{i_1 N}(\eta_{i_1 k_1}^1)^T \Pi_N^{-1} B_N]$,...,
$[\theta_1 p_{i_1 1}(\eta_{i_1 k_1}^{\varphi_{i_1 k_1}})^T \Pi_1^{-1} B_1, \theta_2 p_{i_1 2}(\eta_{i_1 k_1}^{\varphi_{i_1 k_1}})^T \Pi_2^{-1} B_2, ..., \theta_N p_{i_1 N}(\eta_{i_1 k_1}^{\varphi_{i_1 k_1}})^T \Pi_N^{-1} B_N]$,...,
$[\theta_1 p_{i_l 1}(\eta_{i_l k_l}^1)^T \Pi_1^{-1} B_1, \theta_2 p_{i_l 2}(\eta_{i_l k_l}^1)^T \Pi_2^{-1} B_2, ..., \theta_N p_{i_l N}(\eta_{i_l k_l}^1)^T \Pi_N^{-1} B_N]$,...,
$[\theta_1 p_{i_l 1}(\eta_{i_l k_l}^{\varphi_{i_l k_l}})^T \Pi_1^{-1} B_1, \theta_2 p_{i_l 2}(\eta_{i_l k_l}^{\varphi_{i_l k_l}})^T \Pi_2^{-1} B_2, ..., \theta_N p_{i_l N}(\eta_{i_l k_l}^{\varphi_{i_l k_l}})^T \Pi_N^{-1} B_N]$ are linear dependent and **c.4** is violated. Thus, SoS (4.1) is controllable. ■

4.2.3 A three-layer SoS

Consider a three-layer SoS with one MCS in managerial level 2 and a number of MCSs in managerial level 1. The related structure of a typical three-layer SoS is given in Fig.4.4. In addition to the two-layer heterogeneous SoS model mentioned in Section 4.2.1, the MCS in managerial level 2 also contains an embedded computer that can supervise related MCSs in managerial level 1. The dynamics of 1-th virtual model Σ_1 of MCS in managerial level 2, i-th ($i = 2, 3, ..., m+1$) virtual model Σ_i of MCSs in managerial level 1, and i-th ($i = m+2, m+3, ..., N$) PCSs Σ_i in a

three-layer SoS Σ related to graph $\mathcal{G}$ can be depicted by

$$\begin{cases} \dot{x}_1(t) = A_1x_1(t) + \sum_{j=2}^{m+1} \gamma_{1j}H_{1j}y_j(t) + \theta_1 B_1u_1(t), \\ \dot{x}_i(t) = A_ix_i(t) + \sum_{j=1}^{n} \gamma_{ij}H_{ij}y_j(t) + \theta_i B_iu_i(t), \\ \qquad\qquad\qquad\qquad i = 2, 3, ..., m+1, \\ \dot{x}_i(t) = A_ix_i(t) + \sum_{j=2}^{n} \gamma_{ij}H_{ij}y_j(t) + \theta_i B_iu_i(t), \\ \qquad\qquad\qquad\qquad i = m+2, m+3, ..., N, \\ y_i(t) = C_ix_i(t), x_i(0) = x_{i0}. \end{cases} \tag{4.38}$$

in which $x_1(t) \in \mathbb{R}^{n_1}$ and $y_1(t) \in \mathbb{R}^{m_i}$ represent the pseudo state and output of virtual model of MCS in managerial level 2. For $i = 2, 3, ..., m+1$, $x_i(t) \in \mathbb{R}^{n_i}$ and $y_i(t) \in \mathbb{R}^{m_i}$ denote the pseudo state and output of virtual model of MCSs in managerial level 1. For $i = m+2, m+3, ..., N$, $x_i(t) \in \mathbb{R}^{n_i}$ and $y_i(t) \in \mathbb{R}^{m}$ signify the state and output of all PCSs. Definitions of controller $\bar{u}_i(t)$, external input signal $u_i(t)$, matrices $A_i \in \mathbb{R}^{n_i \times n_i}$, $B_i \in \mathbb{R}^{n_i \times q_i}$, $C_i \in \mathbb{R}^{m_i \times n_i}$, $H_{ij} \in \mathbb{R}^{n_i \times m_i}$, $\Gamma \in \mathbb{R}^{N \times N}$ and scalars $\gamma_{ij} \in \mathbb{R}$ and $\theta_i \in \{0, 1\}$ are the same as that of Section 4.2.1. Moreover, due to there is no information change between 1-th MCS Σ_1 in managerial level 2 and i-th PCSs Σ_i in physical level 0, $\gamma_{1i} = \gamma_{i1} = 0, i = m+2, m+3, ..., N$.

The definition of augmented vectors $x(t)$, $u(t)$, $\bar{u}(t)$ $y(t)$ and augmented matrices $\tilde{A}$, $\tilde{H}$, $\tilde{\Theta}$, $\tilde{B}$, $\tilde{C}$, Ξ, Ω are the same as that of Section 4.2.1. Similar to Section 4.2.1, the dynamics of SoS (4.38) can be rewritten in a compact version the same as equation (4.3). Different from Section 4.2.1, the considered three-layer SoS consists of MCSs in two distinct levels. Besides, all virtual model of i-th MCSs, $i = 1, 2, ..., m+1$ are naturally supposed to be controllable.

Assumption 4.2.4 *For MCSs, the virtual pairs* $(A_i, B_i), i = 1, 2, ..., m+1$ *are controllable.*

Note that, the binary signal $\theta_i, , i = 1, 2, ..., m+1$ may equal to 0 or 1 due to possible faults or failures. Thus, each MCS is not assumed to be under control, which implies **the roots of SoS network are not assumed to be controllable**.

Firstly, it's easy to obtain that **Theorem 4.2.1**, **Corollary 4.2.1**, **Corollary 4.2.2** and **Corollary 4.2.3** also exist for three-layer heterogeneous SoS (4.38).

4.2.4 Two-layer decomposition: a divide-and-conquer viewpoint

To simplify the analysis on controllability of three-layer SoS on the basis of the results in Section 4.2.1, the following two-layer decomposition approach is firstly proposed.

Denote $\tau_j, j \in \{2, 3, ..., m+1\}$ as the index set of two-layer component. Each $\tau_j = \{j, i | i \in \{m+2, ..., n\}\}$ is composed by j-th MCS and a group of PCSs. Set $\tau_1 = \{1\}$. The following definition is given.

Definition 2 $\{\tau_j, j = 1, 2, ..., m+1\}$ *is said to be a two-layer decomposition of three-layer SoS if* $\cup_{j=1}^{m+1} \tau_j = \mathcal{F}$ *and* $\tau_j \cap \tau_k = \emptyset, j \neq k$.

Let $|\tau_j|$ denote the number of CSs in component τ_j. Based on the two-layer component $\tau_j, j \in \{2, 3, ..., m+1\}$, we can establish the j-th sub-graph $\mathcal{G}_j$ and j-th two-layer augmented system $\tilde{\Sigma}_j$, which is composed of j-th MCS and i-th PCS satisfying $i \in \tau_j$. For the j-th sub-graph $\mathcal{G}_j = (\mathcal{V}_j, \mathcal{E}_j, \Gamma_j)$, the corresponding j-th sub-vertex set is given as $\mathcal{V}_j = \{v_j\} \cup \{v_i, i \in \tau_j\}$. Besides, the j-th sub-edge set is shown as

$$\mathcal{E}_j = \{(v_p, v_q) | p, q \in \tau_i \& (v_p, v_q) \in \mathcal{E}\} \subset \mathcal{V}_j \times \mathcal{V}_j. \tag{4.39}$$

In addition, $\Gamma_j \in \mathbb{R}^{|\tau_j| \times |\tau_j|}$ is the corresponding adjacency matrix of sub-graph $\mathcal{G}_i$.

Besides, the dynamics of the j-th two-layer augmented system $\tilde{\Sigma}_j$ can be described by

$$\begin{cases} \dot{\tilde{x}}_j(t) = \tilde{A}_j \tilde{x}_j(t) + \sum_{k=1}^{m+1} \tilde{\gamma}_{jk} \tilde{H}_{jk} \tilde{y}_k(t) + \tilde{\theta}_j \tilde{B}_j \tilde{u}_j(t), \\ \qquad\qquad\qquad\qquad i = 2, 3, ..., m+1, \\ \tilde{y}_j(t) = \tilde{C}_j \tilde{x}_j(t), \tilde{x}_j(0) = \tilde{x}_{j0}, \end{cases} \tag{4.40}$$

where $\tilde{x}_j(t) = [x_j^T(t), x_i^T(t), i \in \tau_j]^T$ and $\tilde{y}_j(t) = [y_j^T(t), y_i^T(t), i \in \tau_j]^T$ represent the augmented state and output of $\tilde{\Sigma}_j$ related to two-layer component τ_j. $\tilde{\gamma}_{jk}$ describes the information exchange between augmented system $\tilde{\Sigma}_j$ and $\tilde{\Sigma}_k$ ($\tilde{\Sigma}_1 = \Sigma_1$), which is defined as

$$\tilde{\gamma}_{jk} = \begin{cases} 1, \text{if } \exists\ p \in \tau_j, q \in \tau_k, \gamma_{pq} \neq 0, \\ 0, \text{else.} \end{cases} \tag{4.41}$$

And $\tilde{\theta}_j$ is defined as

$$\tilde{\theta_j} = \begin{cases} 1, \text{if } \exists\ p \in \tau_j, \theta_p \neq 0, \\ 0, \text{else.} \end{cases} \tag{4.42}$$

Augmented matrices $\tilde{A}_j \in \mathbb{R}^{\sum_{i\in\tau_j} n_i \times \sum_{i\in\tau_j} n_i}$, $\tilde{H}_{jk} \in \mathbb{R}^{\sum_{i\in\tau_j} n_i \times \sum_{i\in\tau_k} m_i}$, $\tilde{B}_j \in \mathbb{R}^{\sum_{i\in\tau_j} n_i \times \sum_{i\in\tau_j} q_i}$ and $\tilde{C}_j \in \mathbb{R}^{\sum_{i\in\tau_j} m_i \times \sum_{i\in\tau_j} n_i}$ are designed based on the dynamics of three-layer SoS (4.38).

Moreover, in terms of sub-graph $\mathcal{G}_i$ and two-layer component τ_j, the following two-layer SoS reflects the internal two-layer organization (including j-th MCS and i-th PCSs, $i \in \tau_j$) of augmented system $\tilde{\Sigma}_j$:

$$\begin{cases} \dot{x}_j(t) = A_j x_j(t) + \sum_{k\in\tau_j \cap N_j} \gamma_{jk} H_{jk} y_k(t) + \theta_j B_j u_j(t) \\ \dot{x}_i(t) = A_i x_i(t) + \sum_{k\in\tau_j \cap N_i} \gamma_{ik} H_{ik} y_k(t) + \theta_i B_i u_i(t), i \in \tau_j \\ y_i(t) = C_i x_i(t), x_i(0) = x_{i0}. \end{cases} \tag{4.43}$$

Furthermore, observe that the 1-th MCS, i.e., Σ_1 and all augmented component

systems $\tilde{\Sigma}_j$ (4.40) form a new two-layer SoS $\tilde{\Sigma}$ as follows

$$\begin{cases} \dot{x}_1(t) = A_1x_1(t) + \sum_{j=2}^{m+1} \tilde{\gamma}_{1j}\tilde{H}_{1j}\tilde{y}_j(t) + \theta_1 B_1 u_1(t), \\ \dot{\tilde{x}}_i(t) = \tilde{A}_i\tilde{x}_i(t) + \sum_{j=1}^{n} \tilde{\gamma}_{ij}\tilde{H}_{ij}\tilde{y}_j(t) + \tilde{\theta}_i\tilde{B}_i\tilde{u}_i(t), \\ \qquad\qquad i = 2, 3, ..., m+1, \\ \tilde{y}_1(t) = y_1(t) = C_1x_1(t), x_1(0) = x_{10}, \\ \tilde{y}_i(t) = \tilde{C}_i\tilde{x}_i(t), \tilde{x}_i(0) = \tilde{x}_{i0}, i = 2, 3, ..., m+1. \end{cases} \tag{4.44}$$

Let $\bar{A} = diag\{A_1, \tilde{A}_2, ..., \tilde{A}_{m+1}\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\bar{H}_C = [\tilde{\gamma}_{ij}\tilde{H}_{ij}\tilde{C}_j] \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\bar{\Theta} = diag\{\theta_1 I_{n_1}, \tilde{\theta}_2 I_{\sum_{k\in\tau_2} n_k}, ..., \tilde{\theta}_{m+1} I_{\sum_{k\in\tau_{m+1}} n_k}\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i}$, $\bar{B} = diag\{B_1, \tilde{B}_2, ..., \tilde{B}_{m+1}\} \in \mathbb{R}^{\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} q_i}$ and $\bar{C} = diag\{C_1, \tilde{C}_2, ..., \tilde{C}_{m+1}\} \in \mathbb{R}^{\sum_{i=1}^{N} m_i \times \sum_{i=1}^{N} n_i}$.

Let $\tilde{\Xi} = \bar{A} + \hat{H}$ and $\tilde{\Omega} = \bar{\Theta}\bar{B}$. The generated two-layer $\tilde{\Sigma}$ (4.44) can be rewritten as

$$\begin{aligned} \dot{\tilde{x}}(t) =& \tilde{\Xi}x(t) + \tilde{\Omega}u(t), \\ \tilde{y}(t) =& \bar{C}\tilde{x}(t), \end{aligned} \tag{4.45}$$

where $\tilde{x} = [x_1^T(t), \tilde{x}_2^T(t), ..., \tilde{x}_{m+1}^T(t)]^T$ and $\tilde{y}(t) == [y_1^T(t), \tilde{y}_2^T(t), ..., \tilde{y}_{m+1}^T(t)]^T$ are control input and output of generated SoS (4.44). The information exchange of the two-layer SoS $\tilde{\Sigma}$ can be described by a generated graph $\tilde{\mathcal{G}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}}, \tilde{\Gamma})$, where $\tilde{\mathcal{V}} = \{\tilde{v}_1, \tilde{v}_2, ..., \tilde{v}_{m+1}\}$ represents the vertex set. The edge set is defined as

$$\tilde{\mathcal{E}} = \{(\tilde{v}_i, \tilde{v}_j) | \exists\ p \in \tau_i, q \in \tau_j, (v_p, v_q) \in \mathcal{E}\} \subset \tilde{\mathcal{V}} \times \tilde{\mathcal{V}}. \tag{4.46}$$

Meanwhile, $\tilde{\Gamma} = [\tilde{\gamma}_{ij}] \in \mathbb{R}^{(m+1)\times(m+1)}$ signifies the adjacency matrix of graph $\tilde{\mathcal{G}}$.

An example of two-layer decomposition for three-layer SoS is given in the following example and Fig.4.4.

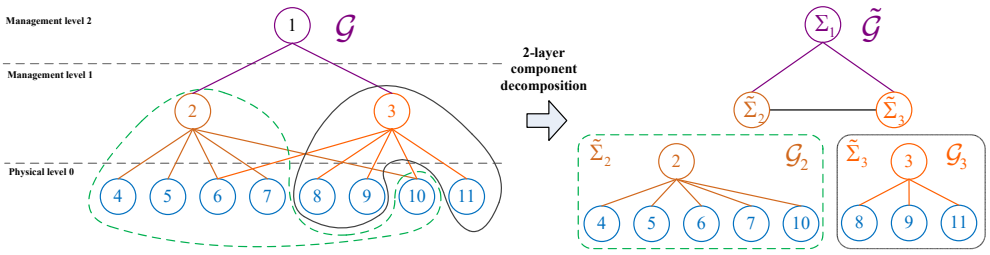


Figure 4.4: Two-layer decomposition in 3-layer SoS

Example 4.2.1 *An example of two-layer decomposition for the 3-layer SoS is shown in Fig.4.4, where SoS (4.4) is decomposed into* $\tau_1 = \{1\}$ *with two-layer components* $\tau_2 = \{2, 4, 5, 6, 7, 10\}$ *and* $\tau_3 = \{3, 8, 9, 11\}$. *Related to* τ_2 *and* τ_3, *we can establish two sub-graphs* $\mathcal{G}_2$, $\mathcal{G}_3$ *and corresponding augmented systems* $\tilde{\Sigma}_2$ *and* $\tilde{\Sigma}_3$. *Moreover, in view of Fig.4.4, the internal structures of augmented systems* $\tilde{\Sigma}_2$, $\tilde{\Sigma}_3$ *are two-layer SoSs related to graph* $\mathcal{G}_2$ *and* $\mathcal{G}_3$ *respectively. Furthermore, it's seen from Fig.4.4*

that, 1-th MCS Σ_1 and augmented systems $\tilde{\Sigma}_2$, $\tilde{\Sigma}_3$ form a new two-layer SoS, whose information exchange can be described by a newly generated graph $\tilde{\mathcal{G}}$. The dynamics of newly generated two-layer SoS $\tilde{\Sigma}$ related to graph $\tilde{\mathcal{G}}$ can be given by

$$\begin{cases} \dot{x}_1(t) = A_1x_1(t) + \sum_{j=2}^{3} \tilde{\gamma}_{1j}\tilde{H}_{1j}\tilde{y}_j(t) + \theta_1 B_1 u_1(t), \\ \dot{\tilde{x}}_i(t) = \tilde{A}_i\tilde{x}_i(t) + \sum_{j=1}^{3} \tilde{\gamma}_{ij}\tilde{H}_{ij}\tilde{y}_j(t) + \tilde{\theta}_i\tilde{B}_i\tilde{u}_i(t), i = 2, 3, \\ \tilde{y}_1(t) = y_1(t) = C_1x_1(t), x_1(0) = x_{10}, \\ \tilde{y}_i(t) = \tilde{C}_i\tilde{x}_i(t), \tilde{x}_i(0) = \tilde{x}_{i0}, i = 2, 3. \end{cases} \tag{4.47}$$

where for 1-th MCS Σ_1, $\tilde{H}_{12} = [H_{12}, 0, 0, 0, 0, 0]$ and $\tilde{H}_{13} = [H_{13}, 0, 0, 0]$, $\tilde{\gamma}_{12} = 1$ and $\tilde{\gamma}_{13} = 1$. For augmented system

$\tilde{\Sigma}_2$, $\tilde{x}_2 = [x_2^T(t), x_4^T(t), x_5^T(t), x_6^T(t), x_7^T(t), x_{10}^T(t)]^T$,

$\tilde{y}_2 = [y_2^T(t), y_4^T(t), y_5^T(t), y_6^T(t), y_7^T(t), y_{10}^T(t)]^T$,

$\tilde{B}_2 = diag\{\theta_2B_2, \theta_4B_4, \theta_5B_5, \theta_6B_6, \theta_7B_7, \theta_{10}B_{10}\}$,

$\tilde{C}_2 = diag\{C_2, C_4, C_5, C_6, C_7, C_{10}\}$, $\tilde{H}_{21} = [\gamma_{21}H_{21}^T, 0, 0, 0, 0, 0]^T$,

$\tilde{\gamma}_{21} = 1$ *and* $\tilde{\gamma}_{23} = 1$.

For augmented system

$\tilde{\Sigma}_3$, $\tilde{x}_3 = [x_3^T(t), x_8^T(t), x_9^T(t), x_{11}^T(t)]^T$, $\tilde{y}_3 = [y_3^T(t), y_8^T(t), y_9^T(t), y_{11}^T(t)]^T$,

$\tilde{B}_3 = diag\{\theta_3B_3, \theta_8B_8, \theta_9B_9, \theta_{11}B_{11}\}$,

$\tilde{C}_3 = diag\{C_3, C_8, C_9, C_{11}\}$, $\tilde{H}_{31} = [\gamma_{31}H_{31}^T, 0, 0, 0]^T$,

$\tilde{\gamma}_{31} = 1$ *and* $\tilde{\gamma}_{32} = 1$. *In addition, the rest system matrices are given in (4.48) and $\tilde{\theta}_j, j = 2, 3$ are given by (4.42).*

Lemma 4.2.4 *Pair $(\tilde{A}_j, \tilde{\theta}_j\tilde{B}_j)$ is state-controllable if and only if the internal two-layer SoS (4.43) of augmented system $\tilde{\Sigma}_j$, related to sub-graph $\mathcal{G}_j$ and component τ_j, is state-controllable.*

Proof: For instance, we verify one typical component example $\tau_2 = \{2, m+2, m+3, ..., q\}$ and the proof of other component can be similarly obtained based on suitable transformation. In terms of (4.38), system matrices of $\tilde{\Sigma}_2$ can be given as

$$\tilde{A}_2 = \begin{bmatrix} A_2 & \gamma_{2,m+2}H_{2,m+2}C_{m+2} & \cdots & \gamma_{2q}H_{2q}C_q \\ \gamma_{m+2,2}H_{m+2,2}C_2 & A_{m+2} & \cdots & \gamma_{m+2,q}C_q \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{q2}H_{q2}C_2 & \gamma_{q,m+2}H_{q,m+2}C_{m+2} & \cdots & A_q \end{bmatrix} \tag{4.49}$$

and $\tilde{B}_2 = diag\{\theta_2B_2, \theta_{m+2}B_{m+2}, \cdots, \theta_qB_q\}$. In addition, $\tilde{\theta}_j$ is given by (4.42).

It's easy to see that pair $(\tilde{A}_2, \tilde{\theta}_2\tilde{B}_2)$ is state-controllable if and only if the following two-layer SoS (4.50), related to sub-graph $\mathcal{G}_2$ and component τ_2, is state-

$$
\begin{aligned}
\tilde{A}_2 &= \begin{bmatrix} A_2 & \gamma_{24}H_{24}C_4 & \gamma_{25}H_{25}C_5 & \gamma_{26}H_{26}C_6 & \gamma_{27}H_{27}C_7 & \gamma_{2,10}H_{2,10}C_10 \\ \gamma_{42}H_{42}C_2 & A_4 & 0 & 0 & 0 & 0 \\ \gamma_{52}H_{52}C_2 & 0 & A_5 & 0 & 0 & 0 \\ \gamma_{62}H_{62}C_2 & 0 & 0 & A_6 & 0 & 0 \\ \gamma_{72}H_{72}C_2 & 0 & 0 & 0 & A_7 & 0 \\ \gamma_{10,2}H_{10,2}C_2 & 0 & 0 & 0 & 0 & A_{10} \end{bmatrix}, \\
\tilde{H}_{23} &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \gamma_{63}H_{63}C_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \gamma_{10,3}H_{10,3}C_2 & 0 & 0 & 0 \end{bmatrix}; \\
\tilde{A}_3 &= \begin{bmatrix} A_3 & \gamma_{38}H_{38}C_8 & \gamma_{39}H_{39}C_9 & \gamma_{3,11}H_{3,11}C_{11} \\ \gamma_{83}H_{83}C_3 & A_8 & 0 & 0 \\ \gamma_{93}H_{93}C_3 & 0 & A_9 & 0 \\ \gamma_{11,3}H_{11,3}C_3 & 0 & 0 & A_{11} \end{bmatrix}, \\
\tilde{H}_{32} &= \begin{bmatrix} 0 & 0 & 0 & \gamma_{36}H_{36} & 0 & \gamma_{3,10}H_{3,10} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};
\end{aligned}
\tag{4.48}
$$

controllable

$$
\begin{cases}
\dot{x}_2(t) = A_1x_1(t) + \sum_{i=m+2}^{q} \gamma_{2j}H_{1j}y_j(t) + \theta_2B_2u_2(t), \\
\dot{x}_i(t) = A_ix_i(t) + \gamma_{i2}H_{i2}y_2(t) + \sum_{j=m+2}^{q} \gamma_{ij}H_{ij}y_j(t), \\
\qquad\quad + \theta_iB_iu_i(t), i = m+2, m+3, ..., q, \\
y_i(t) = C_ix_i(t), x_i(0) = x_{i0}.
\end{cases}
\tag{4.50}
$$

Based on PBH criterion, (4.50) is state-controllable if and only if pair $(\tilde{A}_j, \tilde{\theta}_j\tilde{B}_j)$ is state-controllable. ■

Remark 4.2.4 *Notice that when observe the controllability of internal two-layer SoS (4.43) of augmented system $\tilde{\Sigma}_j$ related to component τ_j, only intra-component interactions are considered. Moreover, the inter-component interconnections only present in the interactions of generated two-layer SoS $\tilde{\Sigma}$ with respect to graph $\tilde{\mathcal{G}}$.*

Theorem 4.2.5 *Three-layer SoS Σ (4.38) is state-controllable if and only if there exists a two layer decomposition $\{\tau_j, j = 1, 2, ..., m+1\}$ such that the newly obtained two-layer SoS $\tilde{\Sigma}$ (4.44), composed of Σ_1 and all augmented component systems $\tilde{\Sigma}_j$ (4.40), is state-controllable.*

Proof: Observe that three-layer SoS (4.38) is said to be state-controllable if the state of all CS can reach any desired state from arbitrary initial value with suitable

control inputs, which is equivalent to the state of Σ_1 and the augmented state $\tilde{x}_j(t) = [x_j^T(t), x_i^T(t), i \in \tau_j]^T$ of all augmented CSs $\tilde{\Sigma}_j, j = 2, 3, ..., m+1$ can reach any desired state from arbitrary initial value with suitable control inputs. ■

Theorem 4.2.6 *Suppose Assumptions 4.2.4 holds. Three-layer SoS (4.38) is state-controllable if there exists a two-layer decomposition such that*

c1. For 1-th MCS, $\theta_1 = 1$.

c2. For all $j = 2, 3, ..., m+1$, the internal two-layer SoS (4.43) of augmented systems $\tilde{\Sigma}_j$ are state-controllable.

c3. For all $i, j = 1, 2, ..., m+1$, $rank([\tilde{H}_{ij}, \tilde{\theta}_i \tilde{B}_i]) = rank(\tilde{\theta}_i \tilde{B}_i)$.

Proof: Similar to Corollary 4.2.2, $rank([\tilde{H}_{ij}, \tilde{\theta}_i \tilde{B}_i]) = rank(\tilde{\theta}_i \tilde{B}_i)$ ensures there must exist $\tilde{X}_{ij} \in \mathbb{R}^{\sum_{k\in\tau_i} q_k \times \sum_{k\in\tau_j} m_k}$ such that $\tilde{\theta}_i \tilde{B}_i \tilde{X}_{ij} = \tilde{H}_{ij}$. Let $\tilde{X} = [\tilde{X}_{ij}] \in \mathbb{R}^{\sum_{i=1}^N q_i \times \sum_{i=1}^N n_i}$ it therefore follows that,

$$\bar{\Theta} \bar{B} \tilde{X} \bar{C} = \bar{H}_C, \tag{4.51}$$

which yields, for arbitrary $s \in \mathbb{C}$,

$$rank([sI - \tilde{\Xi}, \tilde{\Omega}]) = rank([\bar{A} + \bar{H}_C, \bar{\Theta}\bar{B}]) = rank([\bar{A}, \bar{\Theta}\bar{B}]). \tag{4.52}$$

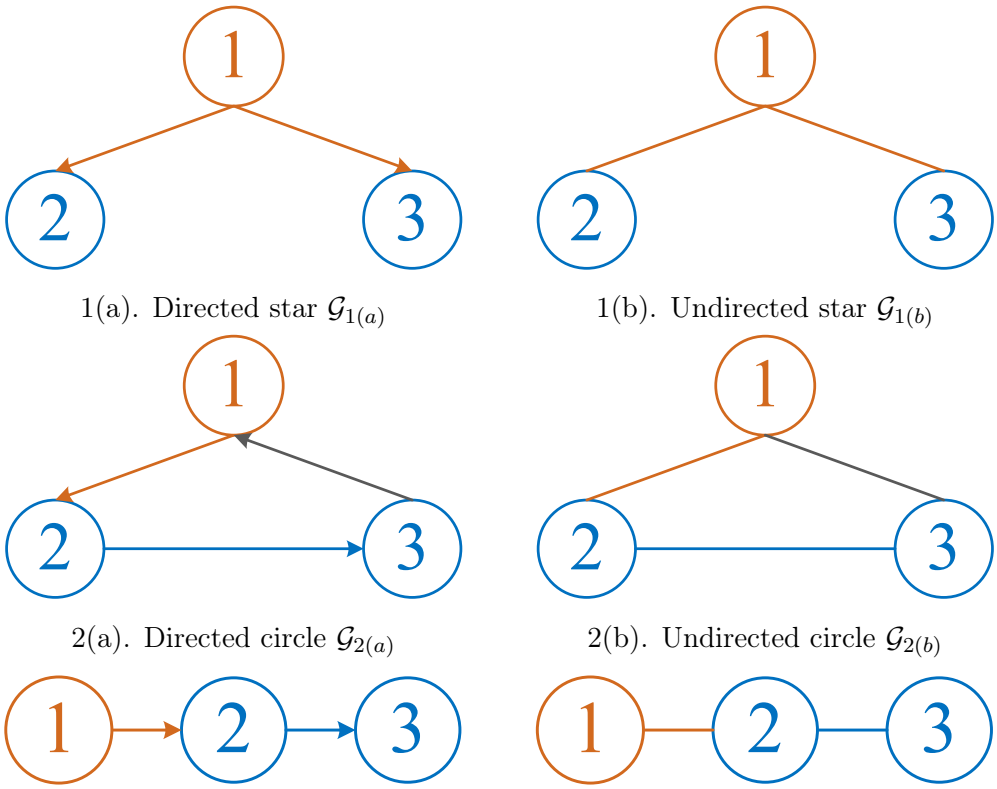
In addition, $rank([\bar{A}, \bar{\Theta}\bar{B}]) = \sum_{i=1}^N n_i$ if and only if pair (A_1, B_1) and pairs $(\tilde{A}_j, \tilde{\theta}_j \tilde{B}_j), j = 2, 3, ..., m+1$ are state-controllable. According to Lemma 4.2.4, pairs $(\tilde{A}_j, \tilde{\theta}_j \tilde{B}_j), j = 2, 3, ..., m+1$ are state-controllable due to two-layer SoS (4.43) is state-controllable. Hence, based on PBH criterion, the two-layer generated SoS (4.44) is state-controllable. In view of Theorem 4.2.4, three-layer SoS (4.38) is state-controllable. ■

Remark 4.2.5 *Above discussion enables us to transform the controllability analysis of three-layer SoS Σ (4.38) into the controllability analysis of two-layer SoS $\tilde{\Sigma}$, two-layer augmented systems $\tilde{\Sigma}_i, i = 2, 3, ..., m+1$ with the following steps:*

s1. Find a suitable two-layer decomposition $\{\tau_j, j = 1, 2, ..., m+1\}$ such that for each component τ_j, the two-layer augmented systems $\tilde{\Sigma}_i, i = 2, 3, ..., m+1$ (4.43) related to graph $\mathcal{G}_j$ is controllable based on the theoretical results in Section 4.2.1.

s2. Verify the controllability of two-layer generated systems $\tilde{\Sigma}$ related to graph $\tilde{\mathcal{G}}$.

*Furthermore, for a n-layer SoS, we can firstly apply $(n-1)$-layer decomposition, $(n-2)$-layer decomposition,..., 2-layer decomposition step by step. Then, observe the controllability of 2-layer augmented systems following **Step 2** in the 2-layer decomposition and the 2-layer generated systems $\tilde{\Sigma}$ (the 3-layer components in 3-layer decomposition) following **Step 2**. It proceeds with the same procedure that we observe the controllability of $(n-1)$-layer component in $n-1$-layer decomposition.*



1(a). Directed star $\mathcal{G}_{1(a)}$ 1(b). Undirected star $\mathcal{G}_{1(b)}$

2(a). Directed circle $\mathcal{G}_{2(a)}$ 2(b). Undirected circle $\mathcal{G}_{2(b)}$

3(a). Directed chain $\mathcal{G}_{3(a)}$ 3(b). Undirected chain $\mathcal{G}_{3(b)}$

Figure 4.5: Two-layer SoS subject to multiple graphs

Finally, the controllability of n-layer SoS will be transformed into a two-layer generated SoS composed of 1*-th MCS in managerial level* $n-1$ *and various* $(n-1)$*-layer components.*

Remark 4.2.6 *The two-layer decomposition can also be applied into the controllability of LSSs). In this regard, it's recommended to firstly divide the LSS into a number of controllable clusters and then observe the controllability of the generated LSS composed of augmented clusters. Note that when observe the controllability of each cluster, only inter-cluster couplings are involved. The two-layer decomposition approach actually provides a "divide-and-conquer" viewpoint.*

4.3 Qualitative and quantitative tests

Example 4.3.1 *To verify the effectiveness of theoretical results in Section 4.2.1, a two-layer SoS composed of* 1*-th MCS and two PCSs subject to typical star, circle and chain topologies are considered, which are given in Fig.4.5. The system matrices are*

shown as follows:

$$\begin{aligned} &A_1 = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}, B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C_1 = \begin{bmatrix} 1 & 1 \end{bmatrix}; \\ &A_2 = -1, \qquad B_2 = 1, \qquad C_2 = 1; \\ &A_3 = \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C_3 = \begin{bmatrix} -0.5 & -0.5 \end{bmatrix}, \end{aligned} \tag{4.53}$$

with $\theta_1 = 1$, $\theta_2 = 0$ *and* $\theta_3 = 1$. *It's easy to see for PCS* Σ_2, *pair* $(A_2, \theta_2 B_2)$ *is uncontrollable. Let* $H_i = H, i = 1, 2, 3$, $H_{12} = \Pi_1 H_2 = H$, $H_{13} = \Pi_1 H_3 = H$.*Note that SoS system matrices (4.53) satisfy Assumption 4.2.3 with* $d_1 = d_2 = 0$, $d_3 = 1$, $D = diag\{0, 0, 1\}$ *and*

$$\begin{aligned} &\Pi_2 = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad H_{21} = H_{23} = \Pi_2 H_3 = 2, \\ &\Pi_3 = \begin{bmatrix} -1 & -\frac{1}{3} \\ -1 & -\frac{5}{3} \end{bmatrix}, H_{31} = H_{32} = \Pi_3 H = \begin{bmatrix} -\frac{4}{3} \\ -\frac{8}{3} \end{bmatrix}. \end{aligned}$$

Notice that $\Pi_i, i = 1, 2, 3$ *is full row rank. Besides,* $Q_i = 1, i = 1, 2, 3$. *The adjacency matrix related to topology graphs presented in Fig.4.5 are shown as follows*

$$\begin{aligned} &\Gamma(\mathcal{G}_{1(a)}) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \Gamma(\mathcal{G}_{1(b)}) = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}; \\ &\Gamma(\mathcal{G}_{2(a)}) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \Gamma(\mathcal{G}_{2(b)}) = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}; \\ &\Gamma(\mathcal{G}_{3(a)}) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \Gamma(\mathcal{G}_{3(b)}) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \end{aligned}$$

Due to $H_i = H, i = 1, 2, 3$, $I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes I_{n_1} \hat{H} I_N \otimes C_1 = I_N \otimes (sI_{n_1} - A_1) - (\Gamma + D) \otimes HC_1$. It's calculated that for $i = 1, 2, 3$ and $j = a, b$, $rank([I_N \otimes (sI_{n_1} - A_1) - (\Gamma(\mathcal{G}_{i(j)}) + D) \otimes HC_1, \Theta \otimes I_{n_1} \Pi^{-1} \tilde{B}]) = nN_1$. Based on Theorem 4.2.3, SoS (4.1) composed of 1-th MCS and two PCSs (4.53) with respect to every graph $\mathcal{G}_{i(j)}$ in Fig.4.5 is state-controllable, which is consistent with the results obtained from classical PBH test.

Example 4.3.2 *To illustrate the validity of theoretical results in Section 4.2.3, a three-layer SoS composed of* 1*-th MCS in managerial level* 2*, two MCSs in managerial level* 1 *and four PCSs is considered. Two kinds of typical topology graphs, i.e., directed tree* $\mathcal{G}_{(a)}$ *and undirected tree* $\mathcal{G}_{(b)}$ *are shown in Fig.4.6. The system matrices*

of virtual model of 1*-th MCS are*

$$A_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -1 & -2 \end{bmatrix}, B_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, C_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}. \tag{4.54}$$

Besides, the coupling matrices of 1*-th MCS related to graphs in Fig.4.6 are* $H_{12} = H_{13} = H_{31} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ *and* $H_{21} = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$.

As shown in Fig.4.6, a two-layer decomposition $\{\tau_j, j = 1, 2, 3\}$ *is employed, where* $\tau_1 = \{1\}$, $\tau_2 = \{2, 4, 5\}$ *and* $\tau_3 = \{3, 6, 7\}$. *On the basis of* τ_2 *and* τ_3, *the corresponding generated graphs* $\tilde{\mathcal{G}}_{(a)}$ *and* $\tilde{\mathcal{G}}_{(b)}$, *sub-graphs* $\mathcal{G}_{2(a)}$, $\mathcal{G}_{3(a)}$, $\mathcal{G}_{2(b)}$ *and* $\mathcal{G}_{3(b)}$, *two-layer augmented systems* $\tilde{\Sigma}_2$, $\tilde{\Sigma}_3$ *and two-layer generated system* $\tilde{\Sigma}$ *are presented in Fig.4.6.*

In component $\tau_2 = \{2, 4, 5\}$ *and two-layer augmented system* $\tilde{\Sigma}_2$, *the corresponding system matirces are given by*

$$\begin{aligned} &A_2 = \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C_2 = \begin{bmatrix} 1 & -1 \end{bmatrix}; \\ &A_4 = 1, \qquad\qquad B_4 = 1, \qquad C_4 = 1; \\ &A_5 = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix}, B_5 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \qquad C_5 = \begin{bmatrix} 1 & -2 \end{bmatrix}, \end{aligned} \tag{4.55}$$

with $\theta_2 = \theta_5 = 1$ *and* $\theta_4 = 0$. *Hence, for PCS* Σ_4, *pair* $(A_4, \theta_4 B_4)$ *is uncontrollable. Notice that the system matrices in component* τ_2 *and two-layer augmented system* $\tilde{\Sigma}_2$ *satisfy Assumption 4.2.3 with* $d_2 = d_4^2 = d_5^2 = 0$, $D_2 = diag\{0, 0, 0\}$. *Let* $H_2 = H_4^2 = H_5^2 = \begin{bmatrix} 1 & -1 \end{bmatrix}^T$, $F_5^2 = \begin{bmatrix} 1 & 1 \end{bmatrix}$,

$$\begin{aligned} &\Pi_4^2 = \begin{bmatrix} 1 & -1 \end{bmatrix}, \quad H_{42} = \Pi_4^2 H_2 = 2, \quad Q_4^2 = 1; \\ &\Pi_5^2 = \begin{bmatrix} \frac{10}{3} & -\frac{2}{3} \\ -\frac{1}{3} & 1 \end{bmatrix}, H_{52} = \Pi_5^2 H_2 = \begin{bmatrix} 4 \\ -\frac{4}{3} \end{bmatrix}, Q_5^2 = \frac{3}{14}. \end{aligned}$$

In addition, $H_{24} = H_{25} = H_2$. *Next, considering component* $\tau_3 = \{3, 6, 7\}$ *and two-layer augmented system* $\tilde{\Sigma}_3$, *the corresponding system matrices are given by*

$$\begin{aligned} &A_3 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 1 \end{bmatrix}, B_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, C_3 = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}; \\ &A_6 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \qquad B_6 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad C_6 = \begin{bmatrix} 2 & -3 \end{bmatrix}; \\ &A_7 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 2 & 3 \end{bmatrix}, \quad B_7 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad C_7 = \begin{bmatrix} -1 & 4 & 1 \end{bmatrix}, \end{aligned} \tag{4.56}$$

with $\theta_3 = \theta_7 = 1$ *and* $\theta_6 = 0$. *Hence, for PCS* Σ_6, *pair* $(A_6, \theta_6 B_6)$ *is uncontrollable.*

Observe that the system matrices in component τ_3 *and two-layer augmented system* $\tilde{\Sigma}_3$ *also satisfy Assumption 4.2.3 with* $d_3 = d_6^3 = d_7^3 = 0$, $D_3 = diag\{0,0,0\}$. *Set* $H_3 = H_6^3 = H_7^3 = \begin{bmatrix}0 & 1 & 1\end{bmatrix}^T$, $F_7^3 = \begin{bmatrix}1 & 0 & 1\\ 1 & 1 & 0\end{bmatrix}$, $H_{36} = H_{37} = H_3$,

$$\Pi_6^3 = \begin{bmatrix}1 & -1 & -2\\ 1 & 0 & -1\end{bmatrix},\ H_{63} = \Pi_6^3 H_3 = \begin{bmatrix}-3\\ 1\end{bmatrix},\ Q_6^3 = \frac{1}{3};$$
$$\Pi_7^3 = \begin{bmatrix}\frac{19}{8} & \frac{5}{2} & -\frac{15}{8}\\ \frac{7}{8} & \frac{1}{2} & -\frac{3}{8}\end{bmatrix}, H_{73} = \Pi_7^3 H_3 = \begin{bmatrix}\frac{5}{8}\\ \frac{1}{8}\\ -\frac{3}{8}\end{bmatrix}, Q_7^3 = 2.$$

In addition, $H_{24} = H_{25} = H_2$.

For directed trees in sub-graphs $\mathcal{G}_{2(a)}$, $\mathcal{G}_{3(a)}$, *generated graph* $\tilde{\mathcal{G}}_{(a)}$ *related to original graph* $\mathcal{G}_a$ *and undirected trees in sub-graphs* $\mathcal{G}_{2(b)}$, $\mathcal{G}_{3(b)}$, *generated graph* $\tilde{\mathcal{G}}_{(b)}$ *related to original graph* $\mathcal{G}_b$, *their adjacency matrices are presented as*

$$\Gamma(\tilde{\mathcal{G}}_{(a)}) = \Gamma(\mathcal{G}_{2(a)}) = \Gamma(\mathcal{G}_{3(a)}) = \begin{bmatrix}0 & 0 & 0\\ 1 & 0 & 0\\ 1 & 0 & 0\end{bmatrix};$$
$$\Gamma(\tilde{\mathcal{G}}_{(b)}) = \Gamma(\mathcal{G}_{2(b)}) = \Gamma(\mathcal{G}_{3(b)}) = \begin{bmatrix}0 & 1 & 1\\ 1 & 0 & 0\\ 1 & 0 & 0\end{bmatrix}.$$

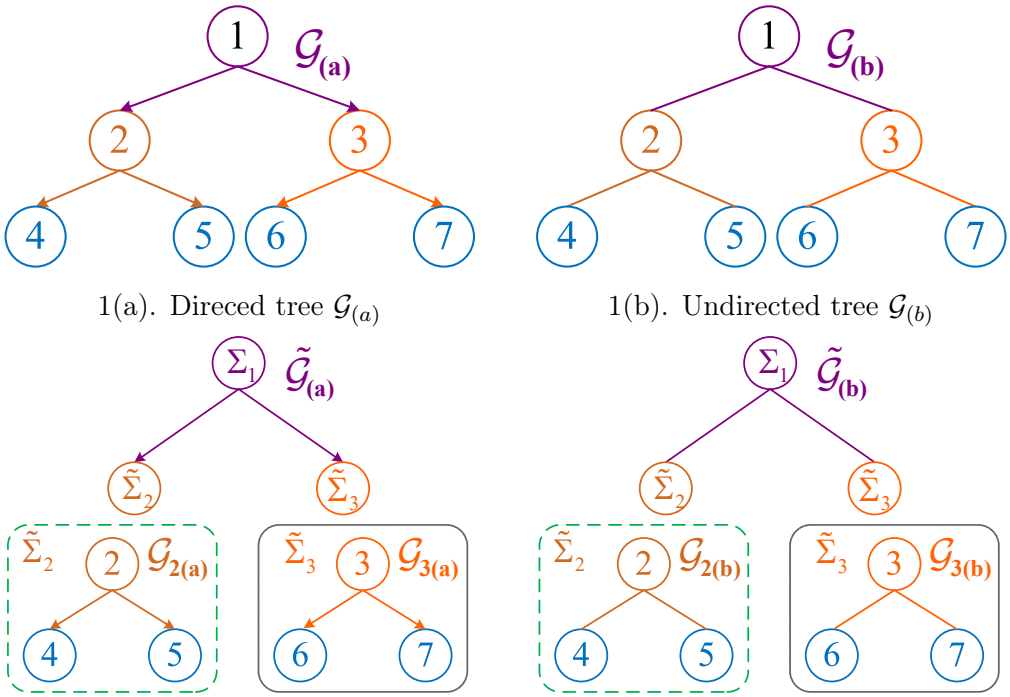
Based on Theorem 4.2.3, two-layer augmented systems $\tilde{\Sigma}_2$ *and* $\tilde{\Sigma}_3$ *related to sub-graphs* $\mathcal{G}_{2j}$ *and* $\mathcal{G}_{3j}, j = a, b$ *are controllable. In addition, for two-layer generated SoS* $\tilde{\Sigma}$,

$$\tilde{H}_{12} = \begin{bmatrix}H_{12} & \boldsymbol{0}_{3\times1} & \boldsymbol{0}_{3\times1}\end{bmatrix}, \tilde{H}_{13} = \begin{bmatrix}H_{13} & \boldsymbol{0}_{3\times1} & \boldsymbol{0}_{3\times1}\end{bmatrix};$$
$$\tilde{H}_{21} = \begin{bmatrix}H_{21} & 0 & \boldsymbol{0}_{1\times2}\end{bmatrix}^T,\quad \tilde{H}_{31} = \begin{bmatrix}H_{31} & \boldsymbol{0}_{1\times2} & \boldsymbol{0}_{1\times3}\end{bmatrix}^T,$$

from which we can obtain that $rank(\tilde{H}_{ij}, B_i) = rank(B_i), i.j = 1,2,3$. *Due to* 1*-th MCS is naturally supposed to be controllable, three-layer SoS is state-controllable in terms of Theorem 4.2.6, which is consistent with the result of well-known PBH test.*

4.4 Conclusion

The controllability of heterogeneous multi-layer SoS is discussed, where the SoS model follows a multi-level organization with MCSs of multiple managerial levels and PCSs in physical level 0. The considered heterogeneous CSs subject to non-identical dimensions, dynamics and coupling matrices. Motivated by the wildly employed output regulation approach in the coordination control of heterogeneous MASs, this work firstly introduces the modified Sylvester equation into the controllability analysis of SoS, which significantly reduce the computational complexity. The



1(a). Direced tree $\mathcal{G}_{(a)}$ 1(b). Undirected tree $\mathcal{G}_{(b)}$

2(a). Decomposition of $\mathcal{G}_{(a)}$ 2(b). Decomposition of $\mathcal{G}_{(b)}$

Figure 4.6: 3-layer SoS subjected to directed and undirected tree

analysis also reflects the impact of the MCS virtual model and coupling matrices on the controllability of the whole SoS. Furthermore, for multi-level large scale SoS, a two-layer decomposition method is proposed from the “divide-and-conquer" perspective, which enables us to transform a high-level SoS into several simple low-level components. The controllability problem of intricate high-level SoS is thus divided into the controllability of low-level components and a low-level generated SoS. A number of instances, including some typical graphs: star, chain, circle and tree are addressed to substantiate the effectiveness of the proposed theoretical results. The controllability analysis contributes to the integrated design of resilient SoS in the concept of structural analysis. This helps SoS to remain operational while a part of CSs are out of control in the presence of various disturbances and risks. It also lays the ground for the following performance evaluation and resilience reconfiguration. Without a controllable SoS, we can not observe dynamic features of PCSs in the performance assessment and perform resilient implementation to restore the overall SoS.

Chapter 5

Towards classification of operational status of SoS

Contents

5.1 Introduction . . . 78
5.2 Communication topology . . . 81
5.2.1 Graph-based communication among PCSs . . . 82
5.2.2 Hypergraph-based organization/communication between PCSs and MCSs of managerial level 1 . . . 82
5.2.3 Hypergraph-based organization between MCSs in managerial levels . . . 83
5.3 Problem formulation . . . 84
5.3.1 Problem formulation of Type-I communication topology . . . 84
5.3.2 Problem formulation of Type-II communication topology . . . 84
5.4 Feature construction of SoS performance classification . . . 85
5.4.1 Feature of PCSs . . . 86
5.4.2 Feature of communication quality . . . 87
5.4.3 Feature of MCSs of managerial level 1 . . . 90
5.4.4 Feature of MCSs of managerial level n . . . 93
5.4.5 Feature of SoS Management efficiency . . . 93
5.5 Qualitative and quantitative test . . . 96
5.5.1 Simulation . . . 96
5.5.2 Classification results . . . 97
5.6 Conclusion . . . 100

5.1 Introduction

Based on the controllability analysis in Chapter 4, we are able to ensure the overall SoS to remain operational under local disruptions, which enables us to measure CSs that will be utilized in this Chapter. Controllability plays a critical role in determining the extent to which a system can be guided toward desired states through external inputs, directly impacting the SoS's ability to achieve its objectives. By

linking controllability with performance evaluation and classification, we can better understand how control mechanisms and system interactions influence overall performance and efficiency.

For a practical multi-level SoS, it's inevitably to witness multiple kind of uncertainties that may degrade the performance to various degrees, which includes but not limited to disturbances, measurement noises, input faults, data loss and etc. Regarding the degree of degradation, the SoS performance may be classified into various status. For example, system's robustness [Zhou 1998] reflects the capacity to cope with the various uncertainties, which has been discussed intensively during the past decades [Bhattacharyya 2017]. To take a step further, the concept fault-tolerance indicates the system's ability to sustain tolerable performance subject to malfunctions. Compared with the robustness, fault-tolerance implies the failures in a part of system's components [Patton 1993], including hardware and/or software. Furthermore, if the performance degrades to an unacceptable level, then a reconfiguration is required to recover the SoS from severe deterioration with the help of backup resources, which exhibits the resilience reaction of SoS [Jiang 2022, Andrade 2022]. Observe that the resilient reconfiguration may fail due to limited resources, which leads to the breakdown of SoS.

The above discussion inspires the following questions regarding the SoS performance degradation and resilience reconfiguration:

- **Firstly**, how to develop a framework of evaluation from a more complex and comprehensive perspective for the performance of multi-level SoS, on the basis of which the users has the knowledge of the degradation degree in performance.

- **Additionally**, the second query focus on how to classify the performance of SoS based on the proposed evaluation framework, which help the engineer to locate the SoS performance status. That is to say, the users are able to know whether a resilience reaction should be performed and it's sufficient/insufficient to recover the SoS from severe performance deterioration.

Therefore, the resilience of SoS deserves imperative concerns, which implies the capability to response to and recover from the severe performance deterioration. As for resilience reaction, engineers should perform a reconfiguration in the organization and communication of SoS when it's performance degrades to an unacceptable value.

Observe that the row data collected from diverse heterogeneous CSs in multiple levels is of various magnitudes and dimensions, which may result in the low generalization as well as overfitting in the classification thereafter [Zhao 2009]. For performance evaluation, feature engineering is a crucial process to extract attributes from the data of SoS [Aamir 2019]. It thereafter lays the ground for the SoS performance classification. Refer to the above first query, the feature construction is a crucial step in the performance assessment [Tran 2016] to extract diverse attributes from the multiple measurements of SoS. This lays the ground for the second stage, that is, the classification.

In the field of performance classification/evaluation, we have witnessed various kinds of AI-based algorithms haven been utilized, such as fuzzy machine learning [Ishibuchi 1999], Naïve Bayes [Saritas 2019, Prati 2011], KNN [Rajagopal 2015], RF [Suksomboon 2022] and NN [Saritas 2019], etc.

As for AI-based classifier, it's quite significant that feature processing contributes a lot in developing a predictive model. It's worthwhile to point out that various aspects of performance lead to various performance measures, in other words, different features are applied to describe the various aspects of performance. That's why one should select features carefully to match the specific objectives [Hand 2012]. Consequently, we define five classes to describe different operational level of SoS, and propose a general framework for constructing SoS performance classification features including performance of PCS, communication, and MCS.

The main idea of feature construction at the physical level is to avoid low generalization and overfitting in classification. Considering the fact that different kinds of physical systems possessing inputs, states and outputs of various dimensions and/or order of magnitudes, it may lead to a classification model (classifier) with low generalization ability if we directly use the row data that are gathered from the physical systems as the features of PCSs.

As a summary, this chapter proposes a multi-level graph/hypergraph-based modelling method for the SoS to describe its complex organization and communication relation among diverse CSs of multiple levels. On the basis of this, a performance evaluation framework is established for SoS extracting features from various viewpoints. Thereafter, the classification is performed with an illustration example based on a number of classification approaches. The contribution of this chapter can be summarized as follows:

- Compared with the SoS model in [Boyd 2022, Soyez 2017, Silva 2020, Jiang 2023a, Jiang 2022, Khalil 2012a], this chapter further provides a multi-level hypergraph modelling based on the SoS management decomposition and inter/intra-layer information sharing to better describe a hierarchical organization and communication. Besides, in comparison to the PCSs considered in [Jiang 2023a], the instructions from related MCSs are included in the control input of PCSs, which provides a more practical scenario, and at the same time, increases the complexity of system performance analysis.

- The SoS performance is evaluated based on attributes from comprehensive viewpoints. In comparison to [Fan 2019] focusing on task oriented performance, various kinds of features are extracted based on the multi-level modelling considering CSs, communication and management efficiency. Compared with [Jiang 2023a], the existence of MCSs information in PCSs' dynamic renders the performance feature analysis more involved. All related performance features should be further improved considering the availability of MCSs' data and extra features should be compensated. The multi-level MCSs performance, the communication between CSs of multiple levels and management

efficiency are further considered. Notice that the features are flexible to be chosen regarding the practical available measurements.

- The performance classification is conducted in an illustration simulation with 8 PCSs and 3 MCSs, where the performance of SoS are divided into 5 classes. As an outcome of the comprehensive feature construction, various kinds of AI-based classification algorithms show high accuracy in both training set ans test set, which reveals the effectiveness of feature construction for SoS performance evaluation.

5.2 Communication topology

The considered SoS model is of multi-level structure, which is composed of MCSs) in multiple managerial levels and PCSs in physical level 0. The MCSs are supervisors embedded with computers. Throughout this chapter, we propose the following assumption based on the *management decomposition* in [Jiang 2022].

Assumption 5.2.1 *(SoS management decomposition) Each MCS of level $n, n \geq 2$ can only directly transmit/receive information to MCSs in n, $n-1$ and $n+1$ levels. Each MCS of level 1 can only directly receive/transmit information to MCSs in levels 1&2 and PCSss in level 0. Each PCS of level 0 can only receive/transmit information to MCSs in level 1 and other PCSs.*

Assumption 5.2.1 outlines the hierarchical structure and information flow within a multi-level SoS. It specifies the interactions of PCSs and MCSs across different levels of the hierarchy. Assumption 5.2.1 ensures that each MCS handles specific scopes of supervision and coordination, which simplifies the overall management of the SoS. By limiting interactions to adjacent levels, the hierarchy helps in maintaining clear communication pathways and reduces the risk of information overload or miscommunication.

Assumption 5.2.2 *(Intra-layer information sharing among MCSs) The MCS in the same level can exchange information with each other.*

The intra-layer information sharing in Assumption 5.2.2 supports a decentralized management approach. Instead of relying on a single central authority, MCSs at the same level can make decisions collaboratively based on shared information. Additionally, when MCSs exchange information, they can align their strategies and operations, ensuring that their actions do not conflict with each other. This consistency is vital for maintaining the overall coherence of the SoS.

Based on the management decomposition structure, the following hierarchical SoS modelling is introduced.

5.2.1 Graph-based communication among PCSs

Denote $\mathcal{G} = (\mathcal{V}_0, \mathcal{E}_0)$ as the directed communication graph among all PCSs with vertex set $\mathcal{V}_0$ and edge set $\mathcal{E}_0$. $\mathcal{V}_0 = \{CS_{0,1}, CS_{0.2}, \ldots, CS_{0,n}\}$ represents the vertex set, where $CS_{0,n}$ represents the i-th PCS in physical level 0. $\mathcal{E}_0 = \{e_1, e_2, \ldots, e_p\} \subseteq \mathcal{V}_0 \times \mathcal{V}_0$ is the edge set. There exists an edge $e_q = (CS_{0,i}, CS_{0,j}) \in \mathcal{E}_0$ if the i-th PCS can transmit information to the j-th PCS. The directed path in graph $\mathcal{G}$ is a sequence of nodes CS_{0,i_1}, CS_{0,i_2},...,CS_{0,i_s} such that $(CS_{0,i_k}, CS_{0,i_{k+1}}) \in \mathcal{E}_0$ for all $k = 1, 2, ..., s-1$. $A = [a_{ij}]$ is the adjacency matrix related to communication graph $\mathcal{G}$, where

$$a_{ij} = \begin{cases} 1, & \text{if } (CS_{0,i}, CS_{0,j}) \in \mathcal{E}_0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

Besides, $a_{ii} = 0$ for all $i = 1, 2, ..., n$. The neighbor set of $CS_{0,i}$ implies the index set of PCSs that can transmit information to the i-th PCS, which is defined as

$$N_i = \{j | CS_{0,j} \in \mathcal{V}_0, (CS_{0,j}, CS_{0,i}) \in \mathcal{E}_0\}. \tag{5.2}$$

The corresponding outputs of PCSs in the neighbor set N_i are available for the i-th PCS to generate its input.

5.2.2 Hypergraph-based organization/communication between PCSs and MCSs of managerial level 1

Denote $\mathcal{H}_0 = (\mathcal{V}_0, \mathcal{E}_1)$ as the organization/communication hypergraph with vertex set $\mathcal{V}_0 = \{CS_{0,1}, CS_{0,2}, \ldots, CS_{0,n}\}$ and hyperedge set $\mathcal{E}_1 = \{CS_{1,1}, CS_{1,2}, \ldots, CS_{1,m_1}\}$. Each vertex $CS_{0,i}$ represents the i-th PCS in physical level 0. Besides, CS_{1,k_1} denotes the k_1-th MCS in managerial level 1. For $k_1 \in \{1, 2, ..., m_1\}$, hyperedge CS_{1,k_1} is a subset of vertex set V_0, where $CS_{0,i} \in CS_{1,k_1}$ implies the i-th PCS is in the k_1-th mission that supervised by the k_1-th MCS of managerial level 1. In addition, it also implies the i-th PCS can receive/transmit the information from /to k_1-th MCS. The cardinality of hyperedge CS_{1,k_1} is denoted as $|CS_{1,k_1}|$, which represents the number of nodes composed in the hyperedge CS_{1,k_1}, that is, the quantity of PCSs monitored by k_1-th MCS of managerial level 1. Denote $I(\mathcal{H}_0) \in \mathbb{R}^{n \times m_1}$ as the incidence matrix of hypergraph $\mathcal{H}_0$, whose element $[I_{ij}(\mathcal{H}_0)]$ is defined by the indicator function $I(\cdot, \cdot)$ below

$$I_{ij}(\mathcal{H}_0) = I(CS_{0,i}, CS_{1,j}) = \begin{cases} 1, & \text{if } CS_{0,i} \in CS_{1,j}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

Assumption 5.2.3 *(PCSs monitoring) For each PCS in physical level* 0*, it must be supervised by at least one MCS in managerial level* 1*.*

Assumption 5.2.3 ensures that all physical operations are conducted under managerial oversight, reducing the risk of errors, inefficiencies, or unauthorized activities. In the hypergraph modelling, it ensures every hyperedge $CS_{1,k_1}, k_1 \in \{1, 2, ..., m_1\}$

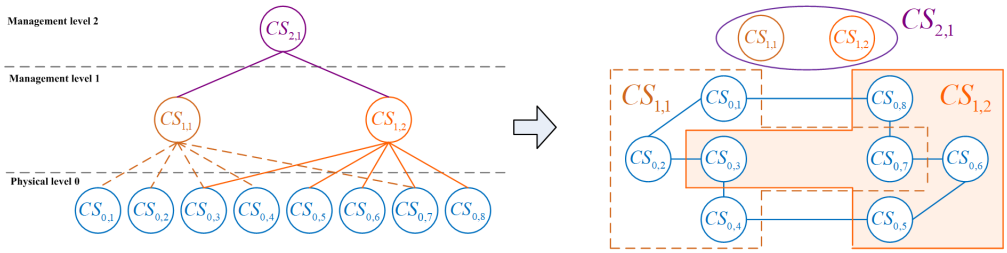


Figure 5.1: Structure decomposition of SoS: 3-layer SoS with 8 PCSs and 3 MCSs.

in hypergraph H_0 is not empty. That is, there must exist at least one vertex $CS_{0,k}, k \in \{1, 2, ..., n\}$ such that $CS_{0,k} \in CS_{1,k_1}$, which ensures $CS_{1,k_1} \neq \emptyset$.

5.2.3 Hypergraph-based organization between MCSs in managerial levels

The organization between MCSs in managerial levels n and $n+1$ ($n \geq 1$) can be described by a hypergraph $\mathcal{H}_n = (\mathcal{V}_n, \mathcal{E}_{n+1})$ with vertex set $\mathcal{V}_n = \{CS_{n,1}, CS_{n,2}, \ldots, CS_{n,m_n}\}$ and hyperedge set $\mathcal{E}_{n+1} = \{CS_{n+1,1}, CS_{n+1,2}, \ldots, CS_{n+1,m_{n+1}}\}$. While $CS_{n,k_n}, k_n \in \{1, 2, ..., m_n\}$ signifies the hyperedge in hypergraph $\mathcal{H}_{n-1}$, it denotes the vertex in hypergraph $\mathcal{H}_n$. Nevertheless, both of them represent the k_n-th MCS in managerial level n. Besides, $CS_{n+1,k_{n+1}}$ indicates the k_{n+1}-th MCS in managerial level $n+1$. In addition, $CS_{n,k_n} \in CS_{n+1,k_{n+1}}$ implies the k_n-th MCS is supervised by the k_n-th MCS of managerial level n. In addition, it also implies k_n-th MCS in level n can receive/transmit the information from/to k_{n+1}-th MCS in level $n+1$. Similar to that that of $\mathcal{H}_0$, the incidence matrix of hypergraph $\mathcal{H}_n$ is denoted as $I(\mathcal{H}_n) \in \mathbb{R}^{m_n \times m_{n+1}}$ with element $[I_{ij}(\mathcal{H}_n)]$,

$$I_{ij}(\mathcal{H}_n) = I(CS_{n,i}, CS_{n+1,j}) = \begin{cases} 1, & \text{if } CS_{n,i} \in CS_{n+1,j}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.4}$$

To better presents the above hypergraph-based multi-layer modelling of SoS, a three-layer SoS illustration example is given in Fig.5.1. The graph-based organization of the three-layer SoS is shown in the left side of Fig.5.1, where the communication among PCSs in physical level 0 hasn't been given due to complexity. With the help of hypergraph modelling, both organization and communication among PCSs and MCSs of different managerial levels can be drawn in a simple way. The graph-based communication among PCSs is shown in the right side of Fig.5.1. Additionally, in hypergraph H_0 that describe the organization and communication between PCSs and MCSs, there are two hyperedges $CS_{1,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,7}\}$ and $CS_{1,2} = \{CS_{0,3}, CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$, which implies the i-th ($i = 1, 2, 3, 4, 7$) PCSs are supervised by 1-th MCS in managerial 1 ($CS_{1,1}$). Besides, the i-th ($i = 3, 5, 6, 7.8$) PCSs are supervised by 2-th MCS in managerial level 1 ($CS_{1,2}$). The organization between MCSs in managerial levels

1 and 2 are addressed in hypergraph H_1, where hyperedge $CS_{2,1} = \{CS_{1,1}, CS_{1,2}\}$. This reveals 1, 2-th MCSs in managerial 1 ($CS_{1,1}, CS_{1,2}$) are monitored by 1-th MCS in managerial level 2.

5.3 Problem formulation

In this section, we introduce two kinds of communication topology, Type-I and Type-II, respectively. The first one (Type-I) relates to graph-based communication topology which means the communication is only between PCSs in level 0; the other one (Type-II) relates to the hypergraph-based organization and communication topology, that is, the communication is among PCSs and between MCSs&PCSs.

5.3.1 Problem formulation of Type-I communication topology

In Type-I, we depict the PCSs by nonlinear dynamics in the following form, for the i-th PCS,

$$\begin{aligned} &\dot{x_i}(t) = f_i(t, x_i(t), u_i^F(t), d_i(t)), \\ &u_i^F(t) = \lambda_i^F u_i(t) + u_i^d(t), \lambda_i^F \in (0, 1], \\ &u_i(t) = h_i(t, y_i(t), Pdr_{ij}(t) \cdot y_j(t - \tau_{ij})), j \in N_i \\ &y_i(t) = g_i(t, u_i(t), x_i(t), n_i(t)). \end{aligned} \tag{5.5}$$

where $x_i(t) \in \mathbb{R}^{n_i}$ indicates the state vector of i-th PCS. $u_i(t) \in \mathbb{R}^{q_i}$ denotes the input vector, $u_i^F(t)$ represents the control input with/without fault which depends on its two input fault values λ_i^F (multiplicative fault) and $u_i^d(t)$ (additive fault). $y_i(t) \in \mathbb{R}^{p_i}$ signifies the output vector. $d_i(t)$ is external disturbance. $n_i(t)$ denotes output fault (measurement noise). $f_i(\cdot)$, $h_i(\cdot)$ and $g_i(\cdot)$ are continuous nonlinear functions. N_i expresses its neighbors that share their outputs with i-th PCS.

5.3.2 Problem formulation of Type-II communication topology

In Type-II, for the i-th PCS, its nonlinear dynamics can be described by

$$\begin{aligned} \dot{x_i}(t) =& f_i(t, x_i(t), u_i^F(t), d_i(t)), \\ u_i^F(t) =& \lambda_i^F u_i(t) + u_i^d(t), \lambda_i^F \in (0, 1], \\ u_i(t) =& h_i(t, y_i(t), DL_{ij}(t) \cdot y_j(t - \tau_{ij}(t)), \\ & DL_{ik}^M(t) \cdot \xi_{ik}(t - \rho_{ik}(t))), j \in N_i, CS_{0.i} \in CS_{1,j} \\ y_i(t) =& g_i(t, u_i(t), x_i(t), n_i(t)). \end{aligned} \tag{5.6}$$

in which $x_i(t) \in \mathbb{R}^{n_i}$ is the state of the i-th PCS. $u_i(t) \in \mathbb{R}^{q_i}$ represents the original input vector. $u_i^F(t)$ denotes the control input that may subject to the input fault, where the existence of faults relate to the values of λ_i^F (multiplicative fault) and $u_i^d(t)$ (additive fault). $y_i(t) \in \mathbb{R}^{p_i}$ indicates the output vector. $n_i(t)$ is the output fault. $d_i(t)$ represents the external disturbance. $f_i(\cdot)$, $h_i(\cdot)$ and $g_i(\cdot)$ are continuous nonlinear functions. $\xi_{ik}(t)$ is the signal sent from the k-th MCS if the i-th PCS

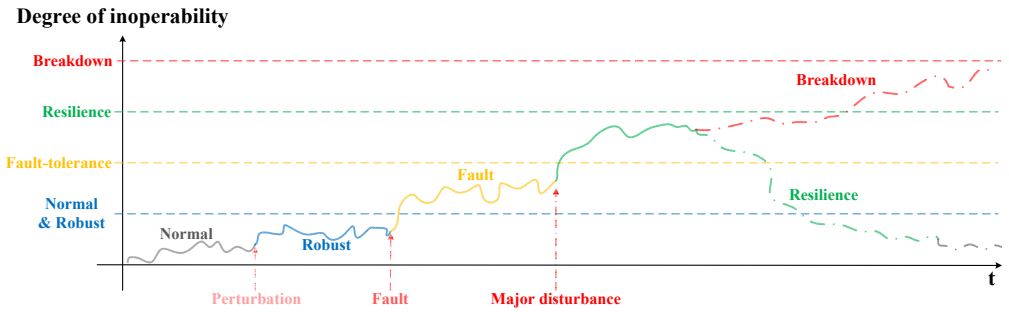


Figure 5.2: Degree of operability of four SoS performances

is supervised by the k-th MCS. Each PCS updates its control input based on the received output from its PCS neighbor and the signal from its related MCSs in managerial level 1. In Type-II, the information received from MCSs is considered in the controller updating of each PCS, where this kind of information exchange is ignored in [Jiang 2023a]. $\tau_{ij}(t)$ is the intra-layer time delay from j-th PCS to i-th PCS. $\rho_{ik}(t)$ is the inter-layer time delay from the k-th MCS in managerial level 1 to i-th PCS. Besides, $DL_{ij}(t)$ and $DL_{ik}^{M}(t)$ are corresponding Boolean that used to describe the data loss. Their definitions will be introduced later.

The MCSs of managerial level n control and monitor the behavior of CS in level $n-1$ (including PCS in level 0). At the same time, they also be supervised and observed by MCSs of managerial level $n+1$. The MCSs of each layer cannot be simply described by dynamic equations. Each MCS collect the information received from the CSs in lower level and calculate the related performance. Meanwhile, they receive instructions from the MCSs in higher level.

This chapter aims at establishing a SoS performance classification framework with comprehensive feature construction. Subsequently, detailed analysis will be provided.

5.4 Feature construction of SoS performance classification

We utilized the five SoS performance classes proposed in [Jiang 2023a], which are given as

- **Normal**: SoS is in the absence of any faults, external disturbance/noises and network service issue.
- **Robust**: SoS operates with insignificant errors in output in the presence of weak external disturbance and/or network service issue.
- **Fault-tolerance**: There exists tolerable degradation in the SoS performance due to the faults, disturbances or communication disruptions.
- **Resilience**: The SoS performance degrades to an unacceptable level and it's necessary to perform a resilience reaction (reconfiguration).

- **Breakdown**: Severe degradation emerges in SoS performance and it is beyond retrieve.

The Fig.5.2 in [Jiang 2023a] is shown to illustrate the evolution of inoperability degree related to the various classes, where the degree of inoperability is the inverse of performance. Given Fig.5.2, the higher the degree of inoperability, the lower the performance.

Comprehensive features of SoS performance are extracted from various viewpoints. Different from the feature construction in [Jiang 2023a], various hypergraph-based communication performance attributes between MCSs&PCSs are compensated, since the modified controller (5.6) includes the information from MCSs. In addition, the MCSs performance in multiple levels are further introduced based on the multi-level hypergraph-based modelling. Furthermore, the management efficiency performance is further introduced to reflect the cost of instruction flow in the top-down management. As a summary, the SoS performance can be divided into the following five parts:

- PCSs performance
- Communication performance including the information exchange among PCS and between MCSs&PCS
- MCSs performance in multiple levels
- Management efficiency

5.4.1 Feature of PCSs

In order to better characterize the performance of PCSs, we extract Rapid, Stable, Accuracy, Fault Detection and Fault Isolation as the features of PCSs from several perspectives and present them.

A.1 Rapid: Quantify the capacity of response to the target for the i-th PCS with the help of the *rise time*. The definition of Rapid is given by:

$$Rapid_i = min\left\{\frac{Tr_i^*}{Tr_i}, 1\right\}, \tag{5.7}$$

in which Tr_i^* represents the rise time of the i-th PCS in ideal status. Tr_i is the real one.

A.2 Stable: Describe the ability of the i-th PCS to stabilize within a given region based on the *settling time*. The definition of Stable is given as:

$$Stable_i = min\left\{\frac{Ts_i^*}{Ts_i}, 1\right\}, \tag{5.8}$$

where Ts_i^* is the settling time of the i-th PCS in ideal status. Ts_i is the real one.

A.3 Accuracy: Denote the accuracy of the i-th PCS to achieve its control objective on the basis of the *steady-state error*. The Accuracy attribute is defined as follows:

$$Accuracy_i = min\left\{\frac{ess_i^*}{ess_i}, 1\right\}, \tag{5.9}$$

where ess_i^* represents the steady-state error of the i-th PCS in normal operations. ess_i is the actual one.

A.4 Fault detection: Identify if there exist any faults the i-th PCS. The fault detection is defined by the Boolean value as

$$FD_i = \begin{cases} 0, & \text{No fault,} \\ 1, & \text{With fault.} \end{cases} \tag{5.10}$$

A.5 Fault isolation: Pinpoint the type of fault for the i-th PCS with the following feature coding, in which each case is marked with an unique code as

$$FI_i = \begin{cases} 0, & \text{No fault,} \\ 1, & \text{Fault type 1,} \\ ... & ... \\ N_{FI}, & \text{Fault type } N_{FI}. \end{cases} \tag{5.11}$$

5.4.2 Feature of communication quality

The communication performance of the whole SoS is affected by time delay, data loss and data congestion, which directly relates to the quality of network service.

B.1 Time delay: Represent the effect of the time delay between the message is sent and received for PCSs in two aspects. The Intra-layer Time delay effect feature between the i-th PCS (subscriber) and j-th PCS (publisher) at time t is given as:

$$TD_{ij}(t) = e^{-\tau_{ij}(t)} \in (0, 1], \tag{5.12}$$

where $\tau_{ij}(t)$ denotes the time delay of between the directed channel from the j-th PCS to the i-th PCS. There may exists $t_f > 0$ such that $\tau_{ij}(t_f) \neq \tau_{ji}(t_f)$ due to we consider directed communication.

Similarly, the definition of Inter-layer Time delay effect feature between the i-th PCS (subscriber) and the k-th MCS (publisher) at time t is given as:

$$TD_{ik}^M(t) = e^{-\rho_{ik}(t)} \in (0, 1], \tag{5.13}$$

where $\rho_{ik}(t)$ is the time delay from the k-th MCS to the i-th PCS. In addition, if there exists an instant t_f such that there is no time-delay at time t_f, that is $\tau_{ij}(t) = 0$ and/or $\rho_{ik}(t) = 0$. This results in $TD_{ij}(t) = 1$ and/or $TD_{ik}^M(t) = 1$. In addition, the larger the time-delay, the closer the two features to 0.

Moreover, note that $TD_{ij}(t) \to 0, \tau_{ij}(t) \to +\infty$ and $TD^M_{ik}(t) \to 0, \rho_{ik}(t) \to +\infty$.

In terms of the $\mathcal{G}$ that describes the communication among all PCSs, the communication performance matrix $A^{TD}(t) = [A^{TD}_{ij}(t)]$ with respect to Intra-layer time delay is given by

$$A^{TD}_{ij}(t) = \begin{cases} TD_{ij}(t), & \text{if } a_{ij} = 1, \\ 0, & \text{if } a_{ij} = 0, \end{cases} \tag{5.14}$$

in which $A = [a_{ij}]$ represents the adjacent matrix of graph $\mathcal{G}$. Point out that $A^{TD}(t_d) = A$ if there is no time delay at instant t_d,

Based on the hypergraph $\mathcal{H}_0$ between PCSs and MCSs in managerial level 0, the performance matrix $I^{TD_0}(t) = [I^{TD_0}_{ik}(t)]$ with respect to PCS and MCS communication time-delay is defined as

$$I^{TD_0}_{ik}(t) = \begin{cases} TD^M_{ik}(t), & \text{if } I_{ik}(\mathcal{H}_0) = 1, \\ 0, & \text{if } I_{ik}(\mathcal{H}_0) = 0, \end{cases} \tag{5.15}$$

where $I(\mathcal{H}_0) = [I_{ik}(\mathcal{H}_0)]$ is the incidence matrix with respect to hypergraph $\mathcal{H}_0$. Note that $I^{TD_0}(t_d) = I(\mathcal{H}_0)$ if there is no time delay at instant t_d,

The Intra-layer communication performance with respect to time-delay is given as

$$TD(t) = \frac{\|A^{TD}(t)\mathbf{1}_n\|}{\|A\mathbf{1}_n\|}, \tag{5.16}$$

where $\|\cdot\|$ denotes the Euclidean norm. $\mathbf{1}_n \in \mathbb{R}^n$ is a vector with all 1 entities. Observe that the numerator assess of communication performance under time delay among PCSs and the denominator is the desired one.

Additionally, the Inter-layer communication performance with respect to time-delay is defined by

$$TD^M(t) = \frac{\|I^{TD_0}(t)\mathbf{1}_{m_1}\|}{\|I(\mathcal{H}_0)\mathbf{1}_{m_1}\|}, \tag{5.17}$$

where the numerator is the assessment of communication performance under time delay between PCSs and MCSs. The denominator is the ideal one.

B.2 Data loss: The definition of Intra-layer Data loss between the i-th PCS and j-th PCS at time t is given by the following time-varying Boolean value:

$$DL_{ij}(t) = \begin{cases} 0, & \text{Data loss}, \\ 1, & \text{Otherwise}. \end{cases} \tag{5.18}$$

in which $DL_{ij}(t) = 1$ means the message from the j-th PCS to i-th PCS is complete at time t. And $DL_{ij}(t) = 0$ represents the data loss at instant t.

The definition of Inter-layer Data loss between the i-th PCS and the k-th MCS

of managerial level 1 at time t is defined by

$$DL_{ik}^{M}(t) = \begin{cases} 0, & \text{Data loss,} \\ 1, & \text{Otherwise.} \end{cases} \tag{5.19}$$

in which $DL_{ij}^{M} = 1$ means the message from the k-th MCS to i-th PCS is complete at time t. And $DL_{ij}^{M}(t) = 0$ represents the data loss at instant t.

The performance matrix $A^{DL}(t) = [A_{ij}^{DL}(t)]$ with respect to Intra-layer data loss is given by

$$A_{ij}^{DL}(t) = \begin{cases} DL_{ij}(t), & \text{if } a_{ij} = 1, \\ 0, & \text{if } a_{ij} = 0. \end{cases} \tag{5.20}$$

For a desired instant t_d with no data loss, $A^{LD}(t_d) = A$.

Similarly, the performance matrix $I^{DL_0}(t) = [I_{ik}^{DL_0}(t)]$ with respect to Inter-layer data loss is depicted by

$$I_{ik}^{DL_0}(t) = \begin{cases} DL_{ik}^{M}(t), & \text{if } I_{ik}(\mathcal{H}_0) = 1, \\ 0, & \text{if } I_{ik}(\mathcal{H}_0) = 0. \end{cases} \tag{5.21}$$

For a desired instant t_d with no data loss, $I^{DL_0}(t_d) = I(\mathcal{H}_0)$.

The Intra-layer communication performance related to data loss is defined by

$$DL(t) = \frac{\|A^{DL}(t)\mathbf{1}_n\|}{\|A\mathbf{1}_n\|}, \tag{5.22}$$

where the numerator evaluates the communication performance subject to data loss among PCSs. The denominator is the ideal one.

The Inter-layer communication performance related to data loss is shown as

$$DL^{M}(t) = \frac{\|I^{DL_0}(t)\mathbf{1}_{m_1}\|}{\|I(\mathcal{H}_0)\mathbf{1}_{m_1}\|}, \tag{5.23}$$

in which the numerator evaluate the data loss between PCSs and MCSs. And the denominator is the ideal one.

B.3 Data received frequency: During a given time period $[t-\delta(t), t]$, one measures the data received frequency that reflects the time-varying quality of the communication network. The Intra-layer Data received frequency from the j-th PCS to i-th PCS is defined as:

$$Pdr_{ij}(t) = \frac{Q_{ij}(t)}{Q_{ij}^{*}(t)}, t \geq \delta(t), \tag{5.24}$$

where $Q_{ij}(t)$ is the real quantity of messages that the i-th PCS receives from j-th PCS during $[t - \delta(t), t]$. $Q_{ij}^{*}(t)$ is the ideal one. $\delta(t)$ is a bounded time

period which could be devised flexibly according to the users and practical applications.

Similarly, the Inter-layer Data received frequency from the k-th MCS to i-th PCS is defined as:

$$Pdr_{ik}^{M}(t) = \frac{W_{ik}(t)}{W_{ik}^{*}(t)}, t \geq \delta(t), \tag{5.25}$$

where $W_{ik}(t)$ is the real quantity of messages that the the i-th PCS receives from k-th MCS during $[t - \delta(t), t]$. $W_{ik}^{*}(t)$ is the ideal one.

For $t \geq \delta(t)$, the performance matrix $A^{Pdr}(t) = [A_{ij}^{Pdr}(t)]$ with respect to intra-layer SoS data received frequency is given by

$$A_{ij}^{Pdr}(t) = \begin{cases} Pdr_{ij}(t), & \text{if } a_{ij} = 1, \\ 0, & \text{if } a_{ij} = 0. \end{cases} \tag{5.26}$$

Observe that if for a given instant t_d, all intra-layer data is successfully received during $[t_d - \delta(t_d), t_d]$, $A^{Pdr}(t_d) = A$. For $t \geq \delta(t)$, the performance matrix $I^{Pdr_0}(t) = [I_{ik}^{Pdr_0}(t)]$ with respect to inter-layer SoS data received frequency is given by

$$I_{ik}^{Pdr_0}(t) = \begin{cases} Pdr_{ik}^{M}(t), & \text{if } I_{ik}(\mathcal{H}_0) = 1, \\ 0, & \text{if } I_{ik}(\mathcal{H}_0) = 0. \end{cases} \tag{5.27}$$

If there exists an instant t_d such that all inter-layer messages are successfully received during time period $[t_d - \delta(t_d), t_d]$, $I^{Pdr_0}(t_d) = I(\mathcal{H}_0)$. The Inter-layer SoS Data received performance is defined as

$$Pdr(t) = \frac{\|A^{Pdr}(t)\mathbf{1}_n\|}{\|A\mathbf{1}_n\|}, t \geq \delta(t). \tag{5.28}$$

The Intra-layer SoS Data received performance is defined as

$$Pdr^{M}(t) = \frac{\|I^{Pdr_0}(t)\mathbf{1}_{m_1}\|}{\|I(\mathcal{H}_0)\mathbf{1}_{m_1}\|}, t \geq \delta(t). \tag{5.29}$$

5.4.3 Feature of MCSs of managerial level 1

In managerial level 1, the features of MCSs are constructed to show if each MCS can complete its supervised mission and quantify the quality of mission.

C.1 Time cost: Represent the PCSs' duration to reach the task in the k-th mission supervised by the k-th MCS of managerial level 1. The Time cost estimation of the i-th PCS at time t is defined as:

$$T_{ik}(t) = \frac{T_{ik}^{*}}{\hat{T}_{ik}(t)}, \tag{5.30}$$

in which T_{ik}^{*} is the pre-given time cost of the i-th PCS to achieve its objective in the k-th mission that supervised by the k-th MCS of managerial level 1

in a desirable state. Besides, the prediction $\hat{T}_{ik}(t)$ is calculated following the steps below.

The estimation is performed based on discrete-time approximation with sampling period T. Denote the sampling instant as $t_m, m = 0, 1, 2, ...$ and $t_{m+1} - t_m = T$. And t_0 is the initial time. Let $\hat{T}_{ik}(t)$ denote the time cost estimation of the i-th PCS in the k-th mission. In terms of (5.6), $\hat{T}_{ik}(t)$ can be defined as

$$\begin{aligned}\hat{T}_{ik}(t) = \hat{T}_{ik}(t_m) = t_m + T * \inf\{\delta(t_m) | x^*_{ik} - \tilde{x}_{ik}(t_m) \\ \leq \sum_{p=1}^{\delta(t_m)} T * f_i(t_{p+m}, \hat{x}_{ik}(t_{m+p}|t_m), \hat{u}_{ik}(t_{m+p}|t_m), 0)\},\end{aligned} \tag{5.31}$$

where x^*_{ik} is the objective state of i-th PCS related to its objective output y^*_{ik} in the k-th mission in the absence of any disturbances or network disruption. $\tilde{x}_{ik}(t_m) = g_i^{-1}(t_m, y_i(t_m - \rho_{ik}(t_m)), \hat{u}_{ik}(t_m), 0)$ is calculated by the real output measurement $y_i(t_m - \rho_{ik}(t_m))$ and the inverse $g_i^{-1}(\cdot)$ of output function $g_i(\cdot)$. It's worth pointing out that, the following discrete-time approximation is utilized in (5.31),

$$\begin{aligned}\int_{t_m}^{t_m + T*\delta(t_m)} f_i(s, x_i(s), u_i(s), d_i(s)) ds \\ \approx \sum_{p=1}^{\delta(t_m)} T * f_i(t_{p+m}, \hat{x}_{ik}(t_{m+p}|t_m), \hat{u}_{ik}(t_{m+p}|t_m), 0)\}.\end{aligned} \tag{5.32}$$

Notice that $\hat{x}_{ik}(t_{m+p}|t_m)$ is given by

$$\begin{aligned}&\hat{x}_{ik}(t_{m+p}|t_m) \\ &= \begin{cases} T * f_i(t_{m+p}, \tilde{x}_i(t_m), \tilde{u}_i(t_m), 0) + \tilde{x}_i(t_m), p = 1, \\ T * f_i(t_{m+p}, \hat{x}_{ik}(t_{m+p-1}|t_m), \hat{u}_{ik}(t_{m+p-1}|t_m), 0) \\ +\hat{x}_{ik}(t_{m+p-1}|t_m) \qquad\qquad\qquad\qquad , p \geq 2. \end{cases}\end{aligned} \tag{5.33}$$

For the k-th Mission (for the k-th MCS in managerial level 1), the estimation of input $u_i(t)$ is given by

$$\begin{aligned}\tilde{u}_i(t_m) = &h_i(t_m, y_i(t_m - \rho_{ik}(t_m)), y_j(t_m - \hat{\rho}_{jk}(t_m)), \\ &\xi_{is}(t)), j \in N_i, CS_{0,i} \in CS_{1,s},\end{aligned} \tag{5.34}$$

in which the k-th MCS accesses the signal $\xi_{is}(t)$ from other MCSs based on Assumption 5.2.2. The time-delay of information exchange among MCSs is not considered due to it is actually data transmission among computers. Besides,

the time delay $\hat{\rho}_{jk}(t_m)$ of neighbor PCS in (5.34) is defined as follows

$$\hat{\rho}_{jk}(t_m) = \begin{cases} \rho_{jk}(t_m), & \text{if } CS_{0,j} \in CS_{1,k} \text{ in } \mathcal{H}_0, \\ \min\{\rho_{js}(t_m) | CS_{0,j} \in CS_{1,s}\}, \text{Otherwise.} \end{cases} \tag{5.35}$$

This implies if the j-th PCS is supervised by the k-th MCS, then the k-th MCS can directly utilize the output of the j-th PCS. Otherwise, it access the output of the j-th PCS based on the information received from other MCS based on Assumption 5.2.2.

Besides, $\hat{u}_{ik}(t_{m+p}|t_m)$ is presented as

$$\begin{aligned} \hat{u}_{ik}(t_{m+p}|t_m) = h_i(t_{m+p}, \hat{y}_{ik}(t_{m+p}|t_m), \hat{y}_{jk}(t_{m+p}|t_m), \\ \xi_{ik}(t)), j \in N_i, \end{aligned} \tag{5.36}$$

where $\hat{y}_{ik}(t_{m+p}|t_m)$ is defined as

$$\hat{y}_{ik}(t_{m+p}|t_m) = g_i(t_{m+p}, \hat{u}_{ik}(t_{m+p}|t_m), \hat{x}_{ik}(t_{m+p}|t_m), 0). \tag{5.37}$$

C.2 Achievement: The definition of the Achievement $Achie_{ik}(t)$ of the i-th PCS at time t for the k-th mission is given based on the Boolean value. If the PCSs complete their missions within 2 times of desired time cost (the maximum tolerance time could be given by the users), then $Achie_{ik}(t) = 1$. 0 means unachieved. That is

$$Achie_{ik}(t) = \begin{cases} 1, T_{ik}(t) \leq 2, \\ 0, \text{Otherwise.} \end{cases} \tag{5.38}$$

The definition of Achievement of the k-th MCS at time t is given as below:

$$Achie_k^1(t) = \begin{cases} 1, & \text{if } \sum_{CS_{0,i} \in CS_{1,k}} Achie_{ik} = |CS_{1,k}|, \\ 0, & \text{Otherwise.} \end{cases} \tag{5.39}$$

C.3 Quality: Represent the weighted percentage of PCSs that can achieve their tasks, which is defined based on hypergraph $\mathcal{H}_0$

$$Q_k(t) = \sum_{CS_{0,i} \in CS_{1,k}} \delta_{ik} Achie_{ik}(t) \tag{5.40}$$

where $\delta_{ik} \in (0, 1)$ represents the weight of the i-th PCS in the k-th mission. Note that weight δ_{ik} satisfies

$$\sum_{CS_{0,i} \in CS_{1,k}} \delta_{ik} = 1, \text{in hypergraph } \mathcal{H}_0. \tag{5.41}$$

It actually reflects the importance of the i-th PCS in the k-th mission.

5.4.4 Feature of MCSs of managerial level n

In managerial level n, the features of MCSs concern about the following two aspect: the performance of supervised MCSs of managerial level $n-1$ and the hypergraph-based management efficiency.

D.1 Achievement: The definition of the achievement of the k_n-th MCSs in level n is defined based on the Boolean value, which shows if all supervised MCSs in managerial level $n-1$ achieve their objectives. It can be given based on the hypergraph $\mathcal{H}_{n-1}$ as

$$Achie^{n}_{k_n}(t) = \begin{cases} 1, & \text{if } \sum\limits_{\substack{CS_{n-1,k_{n-1}} \\ \in CS_{n,k_n}}} Achie^{n-1}_{k_{n-1}} = |CS_{1,k}|, \\ 0, & \text{Otherwise.} \end{cases} \tag{5.42}$$

D.2 Quality: Represent the weighted percentage of supervised MCSs that can complete their missions, which is defined based on hypergraph $\mathcal{H}_{n-1}$,

$$Q^{n}_{k_n}(t) = \sum_{CS_{n-1,k_{n-1}} \in CS_{n,k_n}} \delta^{n}_{k_{n-1},k_n} Achie_{ik}(t) \tag{5.43}$$

where $\delta^{n}_{k_{n-1},k_n} \in (0,1)$ represents the weight (importance) of k_{n-1}-th PCS for k_n-th mission. Similarly, weight $\delta^{n}_{k_{n-1},k_n}$ satisfies

$$\sum_{CS_{n-1,k_{n-1}} \in CS_{n,k_n}} \delta^{n}_{k_{n-1},k_n} = 1, \text{in hypergraph } \mathcal{H}_{n-1}. \tag{5.44}$$

5.4.5 Feature of SoS Management efficiency

In the whole SoS, if a number of transmission channels are interrupted, the information flow path will be changed with the help of other CSs based on the information sharing of MCSs in the same level and the communication of PCSs with their neighbors. Considering the top-down management, there is no doubt that the more CSs are included in the transmission path, which implies the decline of management efficiency.

E.1 Management Efficiency: Represent the inter-layer instruction flow efficiency from the MCSs of the top managerial level n to the i-th PCS of the bottom physical level 0. Considering the disruption among communication CSs, the following management rules are proposed for MCSs. If there is no disruption in communication channel, the instruction from the MCSs of managerial level n only go through $n-1$ node before arriving at the i-th PCS. However, in the presence of disruption in communication, the instruction has to go through other MCSs and/or PCSs to ensure the success of information flow according to Algorithm 1 and Algorithm 2. Denote $Non_i(t)$ as the number of nodes that

Algorithm 1 Management Rule for k_n-th MCS in level n, $n \geq 2$

Initialization: Draw the hypergraph $\mathcal{H}_{n-1}$ and hyperedge CS_{n,k_n}
for all k_{n-1} such that $CS_{n-1,k_{n-1}} \in CS_{n,k_n}$ **do**
  if transmission between CS_{n,k_n} and $CS_{n-1,k_{n-1}}$ is not disrupted **then**
    CS_{n,k_n} transmits the instruction to $CS_{n-1,k_{n-1}}$;
  else
    CS_{n,k_n} transmits the instruction to other $CS_{n-1,k'_{n-1}}$ satisfying $CS_{n-1,k'_{n-1}} \in CS_{n,k_n}$;
    CS_{n,k_n} receive the instruction from other MCSs $CS_{n-1,k'_{n-1}}$ via information sharing in the same level.

Algorithm 2 Management Rule for k_1-th MCS in level 1

Initialization: Draw the hypergraph $\mathcal{H}_0$ and graph $\mathcal{G}$
for all i such that $CS_{0,i} \in CS_{1,k_1}$ **do**
  if transmission between $CS_{0,i}$ and CS_{1,k_1} is not disrupted **then**
    CS_{1,k_1} transmits the instruction to $CS_{0,i}$;
  else
    CS_{1,k_1} transmits the instruction to other $CS_{0,i'}$ satisfying $CS_{0,i'} \in CS_{1,k_1}$;
    $CS_{0,i}$ receive the instruction via the direct path in graph $\mathcal{G}$ from $CS_{0,i'}$.

instruction traverse from the MCS in top managerial level n to the i-th PCS. We utilize $Non_i(t)$ to qualify the management efficiency $Mef_i(t)$, where

$$Mef_i(t) = \frac{Non_i^*}{Non_i(t)}, \tag{5.45}$$

where Non_i^* represents the number of nodes that instruction traverse in normal status.

To better illustrate the variation of $Non_i(t)$ in the presence of communication failures, an example with two cases and normal status are given in Fig.5.3. In normal status, the instruction flow from $CS_{2,1}$ to $CS_{0,1}$ only has to go through $CS_{1,1}$ in the absence of communication disruption. In CASE 1, due to the disruption in communication between $CS_{2,1}$ and $CS_{1,1}$, $CS_{2,1}$ has to transmit the information to $CS_{1,2}$ first and then share the information with $CS_{1,1}$ under Algorithm 1. The instruction has to traverse extra node $CS_{1,2}$. Thus, $Non_1(t) = 2$ while $Non_1^* = 1$, which suggests management efficiency $Mef_1(t) = \frac{1}{2}$. In CASE 2, the disruption in channel between $CS_{1,1}$ and $CS_{0,1}$ renders the information has to go through extra node $CS_{0,2}$ and then flow to $CS_{0,1}$ based on 2. It therefore follows that $Non_1(t) = 2$ and $Mef_1(t) = \frac{1}{2}$.

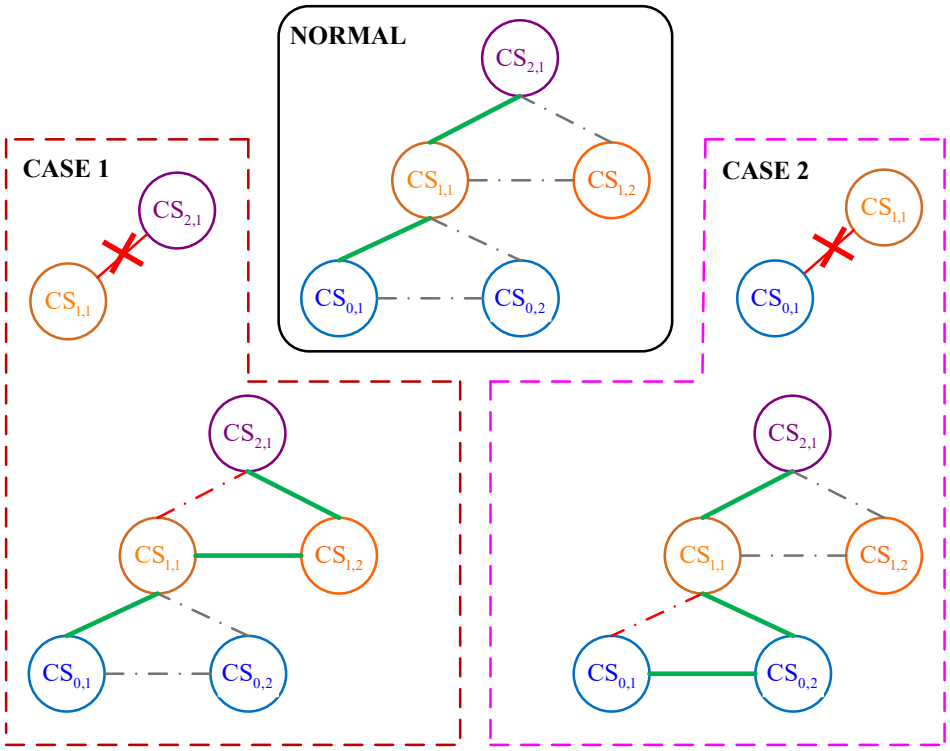


Figure 5.3: Examples of management efficiency

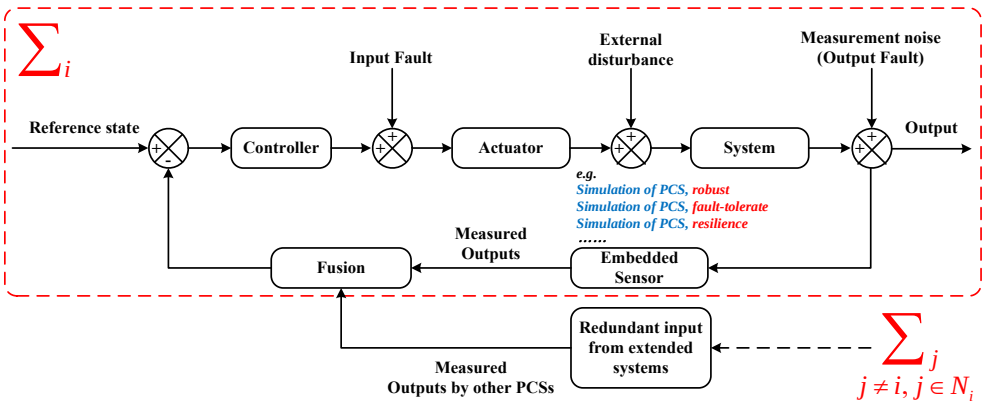


Figure 5.4: Closed-loop control of the i-th PCS ($\sum_i$) [Jiang 2023a]

5.5 Qualitative and quantitative test

In the simulation, we consider a multi-level SoS, whose organization is the same as [Jiang 2023a]. What's more, two types (Type-I & Type-II) communication topology with different constructed features are included in the simulation part.

In Type-I communication topology [Jiang 2023a], the communication is only between PCSs. For Type-II communication topology, in contrast to [Jiang 2023a] that the control input is generated based on the PCSs neighbor measurements, hereby, the dynamics of PCSs incorporates the MCSs' information in the controller. The whole SoS contains eleven CSs, in which eight PCSs are in physical level 0, two MCSs are in managerial level 1 and one MCS is in managerial level 2 (see in Fig.5.1). Each PCS is considered as under closed-loop feedback control, as shown in Fig.5.4.

As what we mentioned in Section 5.3, the system navigates under several types of disturbances, including input fault (external fault) λ_i^F and $u_i^d(t)$, output fault (measurement noise) $n_i(t)$, external disturbances $d_i(t)$, and communication failures. Three types of communications are considered in our SoS, the information exchange between the pair PCS-PCS, PCS-MCS and MCS-MCS. The communication graph is given in Fig.5.1. In PCS-PCS communication, we only consider the Neighbors' outputs of each PCS as a part of feedback to adjust its inputs to the controller, which represents the cooperative behavior in SoS. Compared with the simulation in [Jiang 2023a] only considers the PCS-PCS communication, the system analysis becomes more involved and complex in this work.

5.5.1 Simulation

The considered PCSs are wheeled mobile robots, whose dynamics are given in [Dong 2011]. To achieve the navigation tasks in certain areas, the objectives of PCSs are to track target position under the supervision of related MCSs. For Type-I communication topology, only the measurements from the neighbor PCSs are involved in the PCS controller. And for Type-II communication topology,both the measure-

ments from the neighbor PCSs and MCSs will present in the PCS controller, which shows the cooperation among CSs.

Various kinds of uncertainties are considered in the simulation, which includes the process disturbance $d_i(t)$, output fault $n_i(t)$, multiplicative and additive fault λ_i^F, $u_i^d(t)$ in the control input, communication delay, loss, received frequency $TD_{ij}(t), TD_{ij}^M(t), DL_{ij}(t), DL_{ik}^M, Pdr_{ij}(t), Pdr_{ik}^M(t)$.

To better simulate the possible situations in actual navigation, all disturbances and the corresponding values are generated randomly. The data loss in simulation is expressed by the bernoulli process for all the three types communications.

The 5 classes which are introduced in Section 5.4 are applied to describe the state of SoS. The disturbances among different classes in SoS's performance classification is given in table 5.1. ✓means navigation with this type of disturbance, ✗ represents without, △ denotes with or without. In table 5.1, $\boldsymbol{TD_{ij}^M(t)}$, $\boldsymbol{DL_{ij}^M(t)}$ and $\boldsymbol{Pdr_{ij}^M(t)}$ are only for Type-II.

Given Table 5.1, it seems that communication failures will not affect the final classification result. But in fact, although communication failures are not the main judgment criteria to distinguish Robust and Fault-tolerance classes, high level of communication failures will degrade the SoS performance. For example, it may lead the classification result from Robust/Fault-tolerance to Resilience or Breakdown, or from Resilience to Breakdown. To better understand it, readers may refer to Table 5.1 and what we address in the beginning of Section 5.4.

It is important to emphasize that the main difference between Resilience and Breakdown is whether there is a feasible reaction to restore the performance of the entire SoS to maintain its navigation in a acceptable finite time. To simplify, we consider a SoS to be Resilience if and only if one of the missions (in **1** level) is incompletable. That is, SoS is Breakdown when both missions in level **1** cannot be completed.

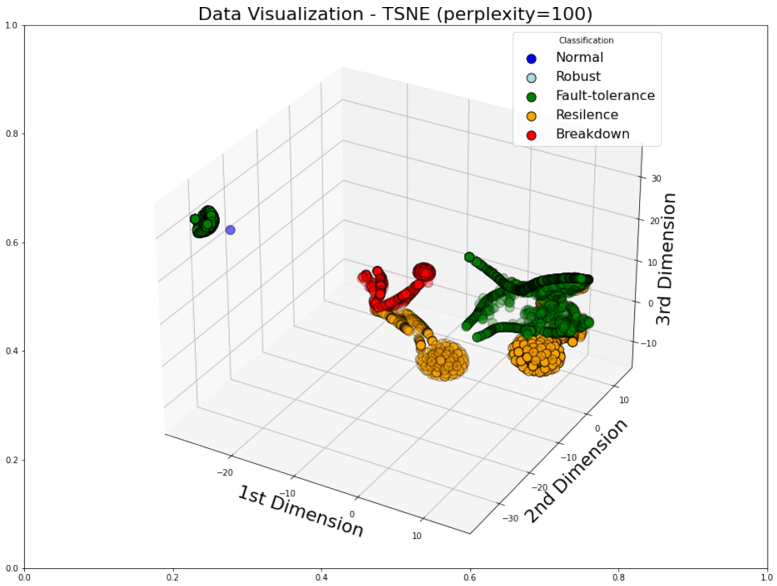
5.5.2 Classification results

Several commonly-used algorithms are chosen to train the classification model, including KNN, SVM, RF, Multi-Layer Perceptron (MLP) and NN.

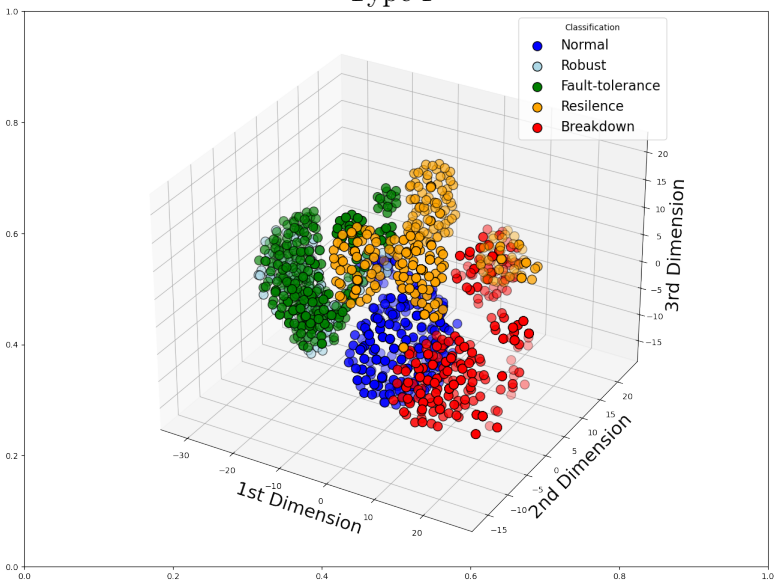
For all those methods, the data set is split randomly into two parts, **85%** for training set and **15%** for test set. To better show the advantages of our feature construction, all the above methods are applied without modification.

The simulation results of the two communication topology are given below:

- **Results of Type-I**:
  In total, $\mathbf{8,600}$ samples are simulated in Type-I, including **46** features to describe the performance of the whole SoS navigation, where **40** features (A.1-A.5) are related to **8** PCSs, **3** features (B.1 - B.3) are related to graph-based communication, **3** features (C.1 - C.3) are for MCSs in level **1** and **3** features are for MCS in level **2**.
  KNN algorithm is a distance-based algorithm and it classifies objects based



Type-I



Type-II

Figure 5.5: 3D t-SNE visualization

on their proximate neighbors' classes. In case Type-I, we have $\mathbf{5 \times 8}$ features to describe the quality of navigation in physical level which lets to lots of points stand closer in distance. That's the reason why KNN shows the worst results in classification (see in Table 5.2). Therefore, t-distributed Stochastic Neighbor Embedding (t-SNE) is used to visualize data in 3D (see in figure 5.5 Type-I) with six features (3 for communication and 3 for MCS), it's easy to find that the point is hard to separate via distance-based method due to the fact that there are lots of points gathered around the boundary of each case. But, by comparing the proportion of different classes, we can still get that the Normal class is the most concentrated case and the second one is Robust class. In other words, the rest three classes are more complicated than those two classes which fits what we address in table 5.1.
In Type-I, all the algorithms except KNN show good results, accuracy of training and test set are greater than **94.7%** and **92.0%** respectively. The best algorithm is RF, in which the accuracy of training and test set are **97.2%** and **96.9%**, respectively.

- **Results of Type-II**:
  In total, $\mathbf{23,000}$ samples are simulated in Type-II, including **71** features to describe the performance of the whole SoS navigation, where **56** features (A.1-A.5, C.1 & E.1) are related to **8** PCSs, **6** features (B.1 - B.3) are related to communication among PCS-PCS and PCS-MCS, **6** features (C.1 - C.3) are for MCS in level **1** and **3** features (C.1, D.1 & D.2) are for MCS in level **2**.
  In table 5.3, for each algorithm, we trained two classifiers based on different features. The first classifier is trained without feature of PCSs which is the control group of the one training with all the features addressed in Section 5.4. This indicates the **56** features of PCSs greatly improved the accuracy of classification, nearly **10%**.
  The 3D visualization of the data is given in Fig.5.5 Type-II. As the fundamental case, Normal class is at the centre of all those five classes. As for Robust and Fault-tolerance, all the missions are completed in those two classes, therefore, the samples are closer than the samples of Resilience and Breakdown.

The samples of Resilience and Breakdown are more dispersed than the two classes we just mentioned, because the resources of the disturbances are different. Not limited to the following reasons:

- Larger external disturbances $\boldsymbol{d_i(t)}$ will lead the class from Robust to Resilience / Breakdown.
- Larger input fault $\boldsymbol{\lambda_i^F}$ and $\boldsymbol{u_i^d(t)}$ and/or output fault $\boldsymbol{n_i(t)}$ will lead the class from Fault-tolerance to Resilience / Breakdown.
- Larger communication failures $(\boldsymbol{TD_{ij}(t), TD_{ij}^M(t)}$, $\boldsymbol{DL_{ij}(t), DL_{ik}^M, Pdr_{ij}(t), Pdr_{ik}^M(t))}$ will lead the class from Normal / Robust / Fault-tolerance to Resilience / Breakdown.

- .....

Different resources of class Resilience and Breakdown make the samples disperse in a certain side of the other three classes, the direction of the side means larger disturbances.

Compared with Type-I, in Type-II, all the algorithms show good results training with all the features which are highlighted with bold in Table 5.3. Because when all missions are completed, the status of PCSs will determine whether the classification result is Robust or Fault-tolerance. Hence, the classifiers without feature of PCS perform worse. Among those five algorithms, all the accuracy of training and test are greater than **93.4%** and **89.66%** respectively.

5.6 Conclusion

To quantify the performance degradation of SoS facing various kinds of uncertainties and malfunctions, a performance assessment framework is proposed for SoS. Based on the proposed multi-level graph/hypergraph-based model of SoS, multiple features are extracted regarding the performance of CSs of different level, communication and top-down management. After the feature construction, the AI-based classifier is employed to divide the SoS performance into **5** different status to inform the engineer the degree of performance deterioration. In this regard, various algorithms are used to train the classification model in the simulation of a SoS with **3** MCS and **8** PCS (wheeled mobile robots). The obtained high accuracy in both training and test sets show the effectiveness of the feature construction. The performance evaluation contributes to the integrated design of resilient SoS in the concept of system diagnosis. This helps SoS to actively detect the performance degradation in the presence of disturbances, faults and malfunctions. The classification of SoS performance informs managers the SoS status. It helps in determining the resilient reactions, that is, the resilient decision making to be introduced in the following chapter.

Table 5.1: Disturbances among **5** classes of SoS

Disturbances \ Classes	$d_i(t)$	$n_i(t)$	λ_i^F and $u_i^d(t)$	$TD_{ij}(t)$	$TD_{ij}^M(t)$	$DL_{ij}(t)$	$DL_{ij}^M(t)$	$Pdr_{ij}(t)$	$Pdr_{ij}^M(t)$
Normal	✗	✗	✗	✗	✗	✗	✗	✗	✗
Robust	✓	✗	✗	△	△	△	△	△	△
Fault-tolerance	△	✓	✗	△	△	△	△	△	△
	△	✗	✓	△	△	△	△	△	△
	△	✓	✓	△	△	△	△	△	△
Resilience	△	△	△	△	△	△	△	△	△
Breakdown	△	△	△	△	△	△	△	△	△

Table 5.2: Results ofSoS Performance classification for Type-I

Method		Accuracy of training set	Accuracy of test set
KNN	Round 1	**0.768940686**	0.688723072
	Round 2	0.767253382	0.701629366
	Round 3	0.765881042	**0.704490367**
	Round 4	0.739514116	0.667297352
	Round 5	0.750324106	0.698069009
SVM	Round 1	0.947332421	0.951162791
	Round 2	0.949110807	0.941085271
	Round 3	**0.949247606**	0.940310078
	Round 4	0.947606019	0.949612403
	Round 5	0.947058824	**0.952713178**
RF	Round 1	0.970314637	**0.968992248**
	Round 2	0.970998632	0.968217054
	Round 3	0.97127223	0.964341085
	Round 4	0.971682627	0.958139535
	Round 5	**0.971819425**	0.963565891
MLP	Round 1	0.955540356	0.948837209
	Round 2	**0.958139535**	**0.954263566**
	Round 3	0.954172367	0.939534884
	Round 4	0.954719562	0.941085271
	Round 5	0.955266758	0.946511628
NN 1	Round 1	0.9608755	0.944186032
	Round 2	**0.9616963**	0.939534903
	Round 3	0.95923394	**0.951162815**
	Round 4	0.9610123	0.945736408
	Round 5	0.9596443	0.93643409
NN 2	Round 1	0.9595075	0.945736408
	Round 2	**0.9607387**	0.935658932
	Round 3	0.95909715	**0.948062003**
	Round 4	0.95923394	0.944186032
	Round 5	0.95937073	0.93643409
NN 3	Round 1	0.9607387	**0.948062003**
	Round 2	**0.9612859**	0.939534903
	Round 3	0.9584131	0.920155048
	Round 4	0.95937073	0.941860437
	Round 5	0.9596443	0.940310061

Table 5.3: Results of SoS Performance classification for Type-II

Method		Accuracy of training set	Accuracy of test set
KNN	MCS features	0.8360	0.8188
	All features	**0.9340**	**0.8966**
SVM	MCS features	0.8550	0.8528
	All features	**0.9561**	**0.9510**
RF	MCS features	0.9102	0.8736
	All features	**1.0000**	**0.9571**
MLP	Round 1	0.8775	0.8814
	All features	**0.9534**	**0.9496**
NN	Round 1	0.8690	0.8757
	All features	**0.9576**	**0.9525**

Chapter 6

Resilient decision making of SoS

Contents

6.1 Introduction 104
6.2 A resilience implementation framework of SoS 105
6.2.1 Hypergraph-based functional and informational SoS modelling 105
6.2.2 Resilient design of SoS 108
6.2.3 Physical twin of the smart port 114
6.2.4 Conclusion 118
6.3 SoS Resilient Planning Facing Predictable Disaster 119
6.3.1 Modelling and problem formulation 119
6.3.2 Data-driven based scenario prediction framework via multi-scale time series analysis 121
6.3.3 Optimization for resilience planning based on disaster prediction 124
6.3.4 Qualitative and quantitative tests 125
6.3.5 Validation of resilience planning of digital twin via Flexsim . . 129
6.3.6 Comparison and analysis 142
6.3.7 An extended disaster scenario 144
6.3.8 Comparison and analysis 149
6.3.9 Conclusion 151
6.4 SoS Resilient Recovery Time Estimation 151
6.4.1 Recovery time estimation of SoS 152
6.4.2 Qualitative and quantitative tests 158
6.4.3 Validation of RTE via Flexsim 159
6.4.4 Conclusion 162
6.5 Conclusion 162

6.1 Introduction

In the previous chapter, we introduced performance evaluation and classification (diagnosis) of SoS, focusing on understanding the key performance indicators. This analysis provides critical insights into how systems perform individually and collectively. Through diagnosis, we can categorize SoS performance statuses, helping in determining necessary and sufficient reactions to restore the overall SoS

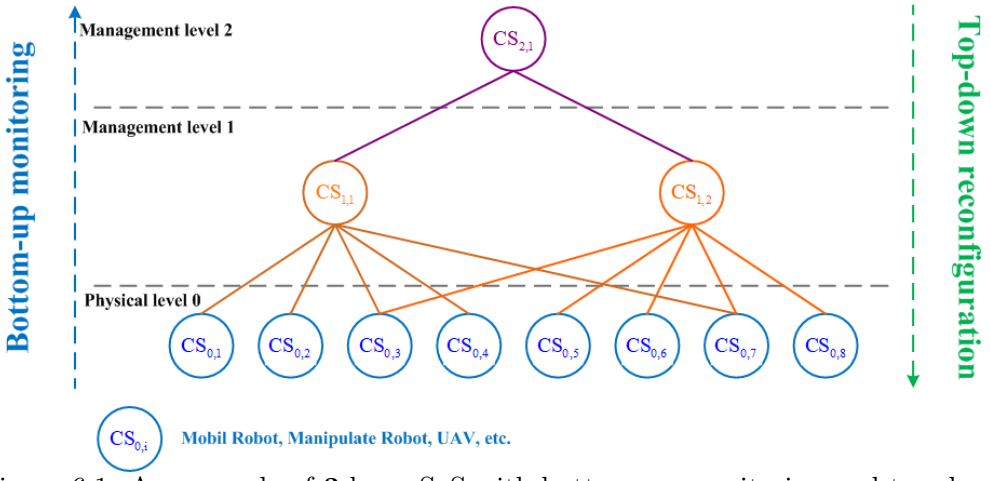


Figure 6.1: An example of **3**-layer SoS with bottom-up monitoring and top-down reconfiguration. $\boldsymbol{CS_{0,i}}$ denotes the $\boldsymbol{i}$-th component system in physical level **0**. $\boldsymbol{CS_{1,j}}$ denotes the $\boldsymbol{j}$-th supervisor in management level 1. $\boldsymbol{CS_{2,j'}}$ denotes the $\boldsymbol{j'}$-th supervisor in management level 2.

Building on this diagnostic foundation, this chapter shifts focus to decision-making within the SoS framework, particularly in the context of enhancing resilience. We explore various strategies, including resilience implementation, resilient planning based on disaster prediction and proactive measures that strengthen the SoS against predictable disasters. Additionally, facing within inevitable malfunctions of PCSs, we delve into the crew dispatch for repair, which are crucial for maintaining operational continuity and mitigating the impact of unforeseen events.

By linking performance evaluation with decision-making, we underscore the importance of data-driven strategies that not only diagnose current SoS performance but also inform adaptive, resilient responses to emerging challenges. This comprehensive approach ensures that the SoS is not only robust in its current state but also prepared to respond effectively to future uncertainties.

6.2 A resilience implementation framework of SoS

6.2.1 Hypergraph-based functional and informational SoS modelling

To illustrate the internal structure of SoS, we introduce the following three hypergraphs to describe the mission membership, function classification and communication among all component systems. In this work, we consider a two-layer SoS. Denote the mission-based SoS hypergraph as $\boldsymbol{\mathcal{H}_{SoS} = (\bar{V}, E, W)}$. $\boldsymbol{\bar{V} = V \cup V'}$ signifies the vertex set. Notice that $\boldsymbol{V = \{v_1, v_2, \ldots, v_n\}}$ represents the main vertex set, where vertex $\boldsymbol{v_i}$ is the CS $\boldsymbol{CS_{0,i}}$ in physical level 0. $\boldsymbol{CS_{0,i}}$ may undertake tasks in various missions. $\boldsymbol{V' = \{v^a_1, v^a_2, \ldots, v^a_{n'}\}}$ represents the auxiliary vertex set, where $\boldsymbol{v^a_{i'}}$ is the backup CSs $\boldsymbol{CS^a_{0,i'}}$. $\boldsymbol{E = \{e_1, e_2, \ldots, e_m, e_{m+1}\}}$ denotes

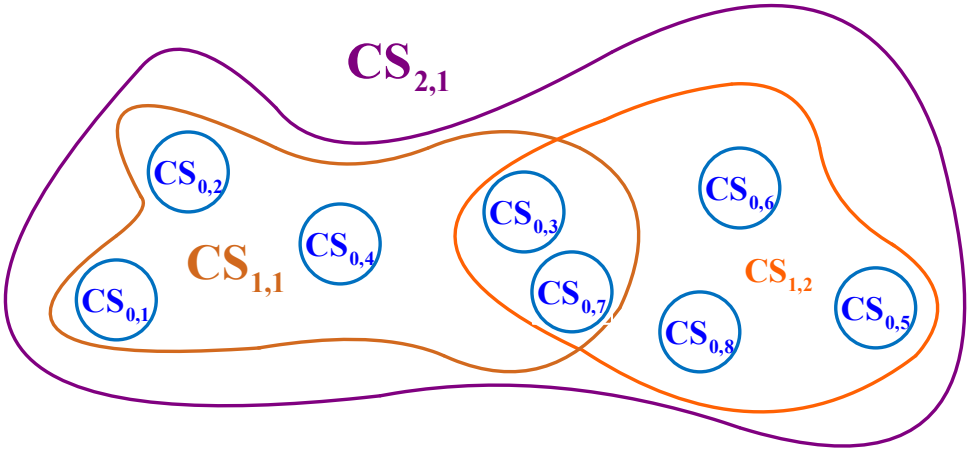


Figure 6.2: $\mathcal{H}_{SoS}$ related to SoS in Fig.6.1.

the hyperedge set of $\mathcal{H}_{SoS}$, where e_j signifies the j-th supervisor $CS_{1,j}$ in management level 1. Point out that $CS_{1,j}, j = 1, 2, ..., m$ supervises and manages the j-th mission in SoS. Moreover, $v_i \in e_j, j = 1, 2, ..., m$ implies CS $CS_{0,i}$ works for the j-th mission and can be supervised by $CS_{1,j}$. For all $i' = 1, 2, ..., n'$, $CS^a_{0,i'}$ doesn't undertake any mission at first and may be added into one of missions in the presence of faults. And e_{m+1} represents $CS_{1,m+1}$ who supervises all backup CSs. Thus, for all $i' = 1, 2, ..., n'$, $v^a_{i'} \notin E \setminus \{e_{m+1}\}$ initially and it might be added into one of hyperedges $e_j, j = 1, 2, ..., m$ thereafter. Fig.6.1 and Fig.6.2 show the constitution of a SoS and its corresponding mission-based SoS hypergraph as $\mathcal{H}_{SoS} = (V, E, W)$.

Let $\mathcal{H}_{fun} = (V, E_{fun})$ denote the function-based SoS hypergraph, in which the vertex set V and its meaning are the same as that of $\mathcal{H}_{SoS}$. $E_{fun} = \{e_{fun,1}, e_{fun,2}, \ldots, e_{fun,p}\}$ denotes the hyperedge set of $\mathcal{H}_{fun}$, where $e_{fun,k}$ represents the k-th function that will be used in SoS. Note that $v_i \in e_{fun,k}$ if CS $CS_{0,i}$ possesses the k-th function. An example of $\mathcal{H}_{fun}$ is shown in Fig.6.3, in which three functions: video, positioning and wifi are represented by hyperedges $e_{fun,1}$, $e_{fun,2}$ and $e_{fun,3}$ respectively. Observation of Fig.6.3 yields the following equations: $e_{fun,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,8}\}$, $e_{fun,2} = \{CS_{0,1}, CS_{0,2}, CS_{0,4}, CS_{0,6}\}$, $e_{fun,3} = \{CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$.

Set $\mathcal{H}^j_{fun} = (V_j, E^j_{fun}), j = 1, 2, ..., m$ as the function assignment SoS hypergraph for the j-th mission. V_j denotes the vertex set and $v_i \in V_j$ if $v_i \in e_j$ in $\mathcal{H}_{SoS}$. The hyperedge set of $\mathcal{H}^j_{fun}$ is denoted as $E^j_{fun} = \{e^j_{fun,1}, e^j_{fun,2}, ..., e^j_{fun,p_j}\}$. $v_i \in e^j_{fun,k}$ if $CS_{0,i}$ undertake the k-th function in j-th mission. Observe that $e^j_{fun,k} \neq e_j \cap e_{fun,k}$. The former represents the CSs who undertake the k^{th} function in j-th mission and the latter denotes the CSs who possess the k-th function in j-th mission. $CS_{0,i}$ may be capable of k-th function but doesn't undertake it in j^{th} mission.

Regarding the functions of $CS_{0,i}$ in the j-th mission, we define the weight matrix

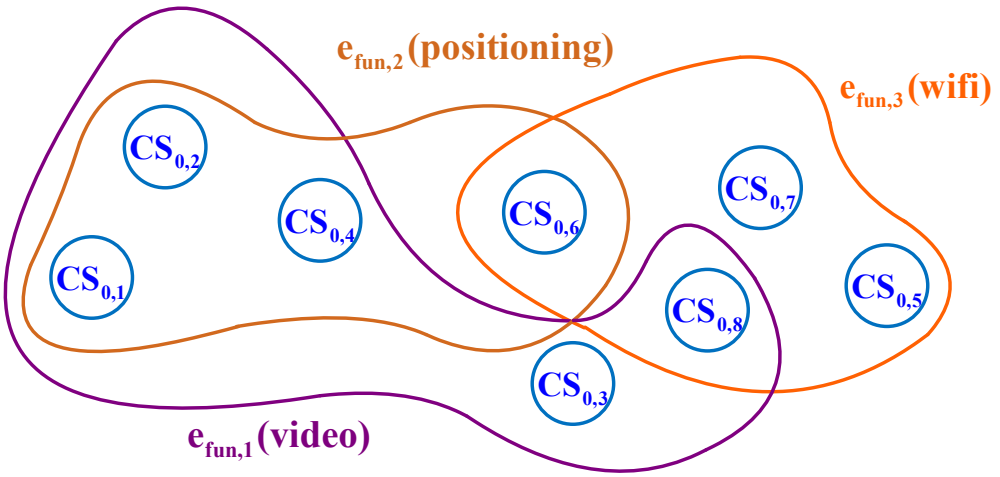


Figure 6.3: An example of function-based hypergraph $\mathcal{H}_{fun}$ with $e_{fun,1}$, $e_{fun,2}$ and $e_{fun,3}$: hyperedges $e_{fun,1}$, $e_{fun,2}$ and $e_{fun,3}$ represents video, positioning and wifi functions. $e_{fun,1} = \{CS_{0,1}, CS_{0,2}, CS_{0,3}, CS_{0,4}, CS_{0,8}\}$, $e_{fun,2} = \{CS_{0,1}, CS_{0,2}, CS_{0,4}, CS_{0,6}\}$, $e_{fun,3} = \{CS_{0,5}, CS_{0,6}, CS_{0,7}, CS_{0,8}\}$.

$\boldsymbol{W}$ of $\mathcal{H}_{SoS}$ as follows,

$$\omega_{i,j} = \sum_{k=1}^{p_j} I(v_i, e^j_{fun,k}) \tag{6.1}$$

$\omega_{i,j} = 0$ if $CS_{0,i}$ is not in the j-th mission. The larger the $\omega_{i,j}$, the more functions $CS_{0,i}$ undertakes in the j-th mission.

In addition, let δ_j denotes the weight of hyperedge e_j as follows:

$$\delta_j = \frac{1}{|e_j|} \sum_{v_i \in e_j} w_{i,j}, \tag{6.2}$$

which presents the weight of the j-th mission in the SoS. $|e_j|$ is the cadinality of the hyperedge e_j, that is, the number of vertices in hyperedge e_j. It is easy to see that $|e_j|$ indicates the number of CSs in the j-th mission.

Next, the communication-based SoS hypergraph is given by $\mathcal{H}_{trans} = (\bar{\mathcal{V}}, \mathcal{E}_{trans})$, which is designed to interpret the informational observability and controllability of CSs in SoS. Observe that the vertex set $\bar{\mathcal{V}} = \mathcal{V} \cup \mathcal{V}'$ of $\mathcal{H}_{trans}$ is the same as that of $\mathcal{H}_{SoS}$, where v_i denotes the corresponding $CS_{0,i}$ and $v^a_{i'}$ denotes backup $CS^a_{0,i'}$. The hyperedge set of $\mathcal{H}_{trans}$ is denoted as $\mathcal{E}_{trans}$, which can be decomposed into $\mathcal{E}_{trans} = \mathcal{E}_{obs} \cup \mathcal{E}_{cont} \cup \mathcal{E}_{dist}$. $\mathcal{E}_{obs}$ and $\mathcal{E}_{cont}$ reveals the information flow relationship between CS $CS_{0,i}$ and supervisor $CS_{1,j}$. $\mathcal{E}_{obs} = \{e^{obs}_1, e^{obs}_2, ..., e^{obs}_m\}$ and $v_i \in e^{obs}_j$ reveals $CS_{1,j}$ can receive information from $CS_{0,i}$ directly. Besides, $\mathcal{E}_{cont} = \{e^{cont}_1, e^{cont}_2, ..., e^{cont}_m\}$ and $v_i \in e^{cont}_j$ implies $CS_{1,j}$ can transmit instructions to $CS_{0,i}$ directly. $\mathcal{E}_{dist} = \{e^{dist}_1, e^{dist}_2, ..., e^{dist}_l\}$ reveals the information exchange among multiple $CS_{0,i}, i = 1, 2, ..., n$. Each hyperedge in $\mathcal{E}_{dist}$ contains only two vertices and

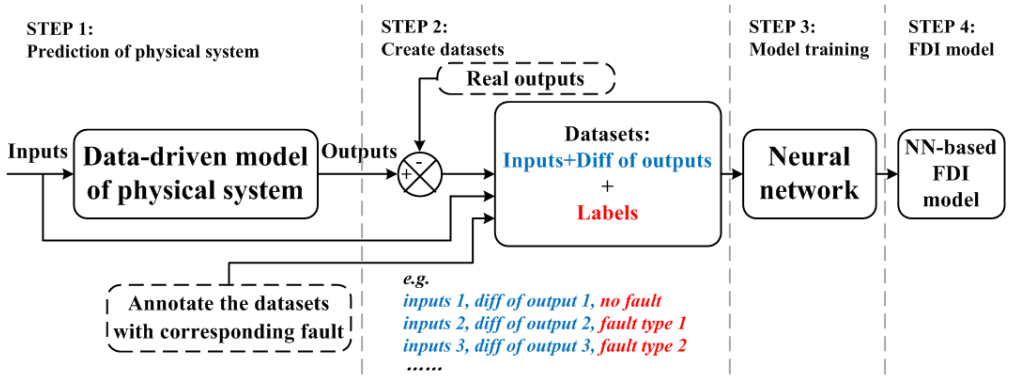


Figure 6.4: NN-based FDI: Step 1 to step 4 show how to train the NN-based FDI model. The annotation is done manually to identify the location of the fault. After the training, skip step 2 is the process of our NN-based FDI.

$\{v_i, v_j\} \in \mathcal{E}_{dist}$ if $d(CS_{0,i}, CS_{0,k}) \leq D$, where $d(\cdot,\cdot)$ represent the Euclid distance and D is the maximal communication distance between two CSs. In a SoS who operates normally without any faults, $v_i \in e_j^{obs} \& e_j^{cont}$ if $v_i \in e_j$. That is, for a $CS_{0,i}$ supervised by $CS_{1,j}$ in j-th mission, it can send/receive information to/from $CS_{1,j}$ directly.

6.2.2 Resilient design of SoS

6.2.2.1 Fault Detection and Isolation (FDI)

Fault Detection and Isolation (FDI) is one of the foundations of our SoS reconfiguration strategy. AI based FDI methods show many advantages such as high accuracy, efficiency, parallel processing, online learning abilities etc. KNN regression and NN are selected to take charge of FDI, and the result of the NN-based FDI is an important judgement condition related to the process of SoS reconfiguration.

As it is shown in Fig.6.4, considering the simple structure of the data, KNN regression is chosen to handle the prediction of the PCS's outputs. In step 1, a data-driven model of physical system is trained based on k-neighbors regression. Deep learning prediction model has the flexibility to provide prediction without knowing values of internal parameters. Step 2 shows how to create the training data set of NN-based FDI model. It means that for a specific inputs we remove some components manually to simulate the real faults and collect the data of real outputs. The training inputs include the inputs of physical system, the prediction of its outputs, its real outputs (highlight in blue in Fig.6.4). The target classification is annotated by its corresponding fault type (highlight in red in Fig.6.4). NN is used to detect and locate the fault automatically. It is worthwhile to point out that the quality and quantity are both important for the NN-based model training. Large data set with high quality will lead to high performance of the FDI model and vice versa. The training is finished after step 3 and step 4.

6.2.2.2 Reconfiguration

The FDI strategy proposed above enable to identify and localize the undesirable faults. Observation of FDI technique lays the ground for the reconfiguration of the SoS, which deserves imperative concerns to ensure the resilience of the SoS. The reconfigurability implies the capability of SoS to operate in normal when various faults occurs, e.g., part of functions of CSs $\boldsymbol{CS_{0,i}}$ are broken, the directed transmission between $\boldsymbol{CS_{0,i}}$ and $\boldsymbol{CS_{1,j}}$ is interrupted. Regarding this, while performing SoS reconfiguration, the following three aspects are fundamentally important:

A.1 Bidirectional communication between $\boldsymbol{CS_{0,i}}$ and $\boldsymbol{CS_{1,j}}$.

A.2 Operational independence of component systems $\boldsymbol{CS_{0,i}, i = \{1, 2, ..., n\}}$ in physical level $\mathbf{0}$.

A.3 Completion of $\boldsymbol{j^{th}}$ mission supervised by component systems $\boldsymbol{CS_{1,j}, j = \{1, 2, ..., m\}}$.

Regarding A.1, the first step of reconfiguration is to check the bidirectional communication between $\boldsymbol{CS_{0,i}}$ and $\boldsymbol{CS_{1,j}}$ work for the same $\boldsymbol{j}$-th mission. With the help of hypergraph $\boldsymbol{\mathcal{H}_{trans}}$, we define the informational observability (controllability) of $\boldsymbol{CS_{0,i}}$ to $\boldsymbol{CS_{1,j}}$ below, which describe $\boldsymbol{CS_{0,i}}$ can send (receive) information to (from) $\boldsymbol{CS_{1,j}}$ directly or indirectly.

Definition 3 ***Informational Observability:*** $\boldsymbol{CS_{0,i}}$ *is said to be informationally observable for* $\boldsymbol{CS_{1,j}}$ *if* $\boldsymbol{v_i \in e_j^{obs}}$ *(directly observable) or there exists a vertex* $\boldsymbol{v_k \in \bar{V} = V \cup V'}$ *and a hyperpath* $\boldsymbol{HP^{dist}_{v_i,\xi_k}}$ *in* $\boldsymbol{\mathcal{E}_{dist}}$ *such that* $\boldsymbol{v_k \in e_j^{obs}}$ *(indirectly observable, namely, there exist neighbors can help it to communicate with* $\boldsymbol{CS_{1,j}}$*).*

Definition 4 ***Informational Controllability:*** $\boldsymbol{CS_{0,i}}$ *is said to be informationally controllable for* $\boldsymbol{CS_{1,j}}$ *if* $\boldsymbol{v_i \in e_j^{cont}}$ *(directly controllable) or there exist a vertex* $\boldsymbol{v_k \in \bar{V}}$ *and a hyperpath* $\boldsymbol{HP^{dist}_{v_i,v_k}}$ *in* $\boldsymbol{\mathcal{E}_{dist}}$ *such that* $\boldsymbol{v_k \in e_j^{cont}}$ *in* $\boldsymbol{\mathcal{E}_{cont}}$ *(indirectly controllable, namely, there exist neighbors can help it to communicate with* $\boldsymbol{CS_{1,j}}$*).*

Thus, the feasibility of communication between $\boldsymbol{CS_{0,i}}$ and $\boldsymbol{CS_{1,j}}$ has been transformed to the informational observability and controllability in hypergraph $\boldsymbol{\mathcal{H}_{trans}}$. We address the communication-based reconfigurability of $\boldsymbol{CS_{0,i}}$ and the whole SoS below.

Definition 5 ***Communication-based reconfigurability:*** $\boldsymbol{CS_{0,i}}$ *is said to be reconfigurable for* $\boldsymbol{CS_{1,j}}$ *in communication if* $\boldsymbol{CS_{0,i}}$ *is informationally observable and controllable for* $\boldsymbol{CS_{1,j}}$ *in the presence of faults.* $\boldsymbol{CS_{0,i}}$ *is said to be reconfigurable in communication if* $\boldsymbol{CS_{0,i}}$ *is reconfigurable for all* $\boldsymbol{CS_{1,j}}$ *satisfying* $\boldsymbol{v_i \in e_j}$ *in communication. The whole SoS is said to be reconfigurable in communication if for all* $\boldsymbol{i = 1, 2, ..., n}$, $\boldsymbol{CS_{0,i}}$ *is reconfigurable in communication.*

In the presence of transmission failures, for example, if $\boldsymbol{CS_{0,i}}$ cannot send its information to $\boldsymbol{CS_{1,j}}$ directly, it implies $\boldsymbol{v_i}$ is deleted from $\boldsymbol{e_j^{obs}}$. Besides, if $\boldsymbol{CS_{0,i}}$ cannot receive instructions from $\boldsymbol{CS_{1,j}}$ directly, it reveals $\boldsymbol{v_i}$ is deleted from $\boldsymbol{e_j^{con}}$. To ensure the informational observability/controllability of $\boldsymbol{CS_{0,i}}$ to $\boldsymbol{CS_{1,j}}$, one feasible solution is to communicate with $\boldsymbol{CS_{0,j}}$ via its neighbors, that is, find a hyperpath in $\boldsymbol{\mathcal{E}_{dist}}$ from $\boldsymbol{v_i}$ to vertex $\boldsymbol{v_k}$ such that $\boldsymbol{v_k \in e_j^{obs}}$ or $\boldsymbol{v_k \in e_j^{cont}}$. Secondly, if no such vertex $\boldsymbol{v_k}$ is found in hypergraph, we choose an available backup CS $\boldsymbol{CS_{0,i}^a}$ and add it into the $\boldsymbol{j}$-th mission. That is, choose an isolated vertex $\boldsymbol{v_{i'}^a}$ such that $\boldsymbol{v_{i'}^a \notin \mathcal{E} \setminus \{e_{m+1}\}}$ from auxiliary vertex set $\boldsymbol{\mathcal{V}'}$ and add it into hyperedges $\boldsymbol{e_j^{obs}}$ and $\boldsymbol{e_j^{cont}}$. Meanwhile, add a new hyperedge $\boldsymbol{e_{i,i'}^a = \{v_i, v_{i'}^a\}}$ to $\boldsymbol{\mathcal{E}_{dist}}$ in $\boldsymbol{\mathcal{H}_{trans}}$. In $\boldsymbol{\mathcal{H}_{SoS}}$, add $\boldsymbol{v_{i'}^a}$ into $\boldsymbol{e_j}$. Thus, $\boldsymbol{v_{i'}^a \in \mathcal{E} \setminus \{e_{m+1}\}}$ and $\boldsymbol{v_{i'}^a}$ is no longer an isolated vertex in $\boldsymbol{\mathcal{H}_{trans}}$.

The communication-based hypergraph reconfiguration process of $\boldsymbol{CS_{0,i}}$ to $\boldsymbol{CS_{1,j}}$ can be summarized as follows.

* ***Step 1***: Verify the informational **observability**.

O.1. Check if $\boldsymbol{v_i \in e_j^{obs}}$ in $\boldsymbol{\mathcal{H}_{trans}}$. If true, $\boldsymbol{CS_{0,i}}$ is directly observable to $\boldsymbol{CS_{1,j}}$. Skip to Step.2. Else, go to Step.O.2.

O.2. Check if there exist a hyperpath $\boldsymbol{HP_{v_i,v_k}^{dist}}$ in $\boldsymbol{\mathcal{E}_{dist}}$ such that $\boldsymbol{v_k \in e_j^{obs}}$. If true, $\boldsymbol{CS_{0,i}}$ is indirectly observable to $\boldsymbol{CS_{1,j}}$. Skip to Step.2. Else, go to Step.O.3.

O.3. Check if there exists an vertex $\boldsymbol{v_{i'}^a}$ satisfying $\boldsymbol{v_{i'}^a \notin E \setminus \{e_{m+1}\}}$ in $\boldsymbol{V'}$. If true, add it into hyperedges $\boldsymbol{e_j^{obs}}$ and $\boldsymbol{e_j^{cont}}$. Meanwhile, add a new hyperedge $\boldsymbol{e_{i,i'}^a = \{v_i, v_{i'}^a\}}$ to $\boldsymbol{\mathcal{E}_{dist}}$ in $\boldsymbol{\mathcal{H}_{trans}}$. In $\boldsymbol{\mathcal{H}_{SoS}}$, add $\boldsymbol{v_{i'}^a}$ into $\boldsymbol{e_j}$. Thus, $\boldsymbol{CS_{0,i}}$ is reconfigurable to $\boldsymbol{CS_{1,j}}$. The communication-based reconfiguration is finished. Else, $\boldsymbol{CS_{0,i}}$ is unobservable and is not reconfigurable to $\boldsymbol{CS_{1,j}}$ in communication. Weakly delete $\boldsymbol{v_i}$ from $\boldsymbol{\mathcal{H}_{SoS}}$, $\boldsymbol{\mathcal{H}_{fun}}$, $\boldsymbol{\mathcal{H}_{fun}^j}$ and $\boldsymbol{\mathcal{H}_{trans}}$.

** ***Step 2***: Verify the informational **controllability**.

C.1. Check if $\boldsymbol{v_i \in e_j^{cont}}$ in $\boldsymbol{\mathcal{H}_{trans}}$. If true, $\boldsymbol{CS_{0,i}}$ is directly controllable to $\boldsymbol{CS_{1,j}}$. $\boldsymbol{CS_{0,i}}$ is reconfigurable for $\boldsymbol{CS_{1,j}}$. Else, go to Step.C.2.

C.2. Check if there exist a hyperpath $\boldsymbol{HP_{v_i,v_k}^{dist}}$ in $\boldsymbol{\mathcal{E}_{dist}}$ such that $\boldsymbol{v_k \in e_j^{cont}}$. If true, $\boldsymbol{CS_{0,i}}$ is indirectly controllable to $\boldsymbol{CS_{1,j}}$. $\boldsymbol{CS_{0,i}}$ is reconfigurable for $\boldsymbol{CS_{1,j}}$. Else, go to Step.C.3.

C.3. Check if there exists an vertex $\boldsymbol{v_{i'}^a}$ satisfying $\boldsymbol{v_{i'}^a \notin E \setminus \{e_{m+1}\}}$ in $\boldsymbol{V'}$. If true, add it into hyperedges $\boldsymbol{e_j^{obs}}$ and $\boldsymbol{e_j^{cont}}$. Meanwhile, add a new hyperedge $\boldsymbol{e_{i,i'}^a = \{v_i, v_{i'}^a\}}$ to $\boldsymbol{\mathcal{E}_{dist}}$ in $\boldsymbol{\mathcal{H}_{trans}}$. In $\boldsymbol{\mathcal{H}_{SoS}}$, add $\boldsymbol{v_{i'}^a}$ into $\boldsymbol{e_j}$. Thus, $\boldsymbol{CS_{0,i}}$ is controllable to $\boldsymbol{CS_{1,j}}$. Else, $\boldsymbol{CS_{0,i}}$ is informationally uncontrollable and is not reconfigurable to $\boldsymbol{CS_{1,j}}$ in communication. Weakly delete $\boldsymbol{v_i}$ from $\boldsymbol{\mathcal{H}_{SoS}}$, $\boldsymbol{\mathcal{H}_{fun}}$, $\boldsymbol{\mathcal{H}_{fun}^j}$ and $\boldsymbol{\mathcal{H}_{trans}}$.

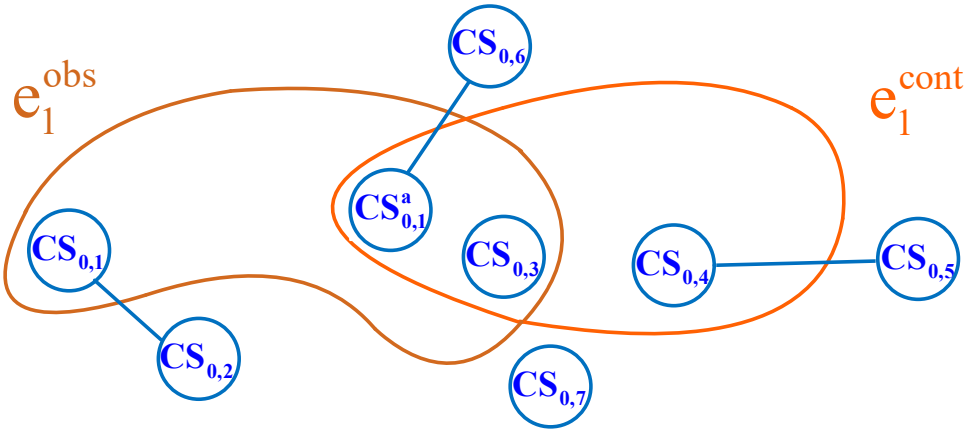


Figure 6.5: Informational Observability and Controllability. Based on above hypergraph, $\boldsymbol{CS_{0,1}}$ is directly observable but not controllable to $\boldsymbol{CS_{1,1}}$. $\boldsymbol{CS_{0,2}}$ is informationally observable with the help of hyperpath $\boldsymbol{HP^{dist}_{v_1,v_2}}$ in $\boldsymbol{\mathcal{E}_{dist}}$ but not informationally controllable to $\boldsymbol{CS_{1,1}}$. $\boldsymbol{CS_{0,3}}$ and backup $\boldsymbol{CS^a_{0,1}}$ are directly observable and controllable to $\boldsymbol{CS_{1,1}}$. $\boldsymbol{CS_{0,4}}$ is directly controllable but not observable to $\boldsymbol{CS_{1,1}}$. $\boldsymbol{CS_{0,5}}$ is informationally controllable with the help of hyperpath $\boldsymbol{HP^{dist}_{v_4,v_5}}$ but not informationally observable to $\boldsymbol{CS_{1,1}}$. With the aid of backup $\boldsymbol{CS^a_{0,1}}$, $\boldsymbol{CS_{0,6}}$ are both informationally observable and controllable to $\boldsymbol{CS_{1,1}}$. $\boldsymbol{CS_{0,7}}$ is neither informationally observable nor controllable.

*** ***Step 3***: Considering A.2, after communication-based reconfiguration, the operational independence of $\boldsymbol{CS_{0,i}}$ will be examined by **FDI** strategy. Note that FDI helps users to detect faults in functions of $\boldsymbol{CS_{0,i}}$ and determine whether $\boldsymbol{CS_{0,i}}$ can keep running and operate independently. It is worthwhile to point out that Global Positioning System Global Positioning System (GPS) signal lose is a special situation. Caused by obstacles in the operation environment, for example tall buildings and dead zone in port, we cannot solve it by replacing $\boldsymbol{CS_{0,i}}$. For the sake of efficiency, we treat GPS broken and signal lose as one case and handle it with backup systems. In this case, the reconfiguration follows O.3 or C.3, where backup system provides GPS signal instead of communication. If the examined $\boldsymbol{CS_{0,i}}$ cannot keep the operational independence but an backup CS $\boldsymbol{CS^a_{0,i'}}$ has been assigned into the $\boldsymbol{j}$-th mission and help $\boldsymbol{CS_{0,i}}$ to communicate with $\boldsymbol{CS_{1,j}}$, then we should call back the $\boldsymbol{CS^a_{0,i'}}$ from the $\boldsymbol{j}$-th mission. In addition, we should weakly delete $\boldsymbol{v_i}$ from $\boldsymbol{\mathcal{H}_{SoS}}$, $\boldsymbol{\mathcal{H}_{fun}}$, $\boldsymbol{\mathcal{H}^j_{fun}}$ and $\boldsymbol{\mathcal{H}_{trans}}$. In $\boldsymbol{\mathcal{H}_{SoS}}$, we delete auxiliary vertex $\boldsymbol{v^a_{i'}}$ from hyperedge $\boldsymbol{e_j}$. In $\boldsymbol{\mathcal{H}_{trans}}$, we delete $\boldsymbol{e^a_{i,i'} = \{v_i, v^a_{i'}\}}$ from $\boldsymbol{\mathcal{E}_{dist}}$ and remove $\boldsymbol{v^a_{i'}}$ from $\boldsymbol{e^{cont}_j, e^{obs}_j}$. It implies vertex $\boldsymbol{v^a_{i'}}$ satisfies $\boldsymbol{v^a_{i'} \notin \mathcal{E} \setminus \{e_{m+1}\}}$ again. Thus, backup $\boldsymbol{CS^a_{0,i'}}$ is available again for other CSs.

After communication-based reconfiguration and FDI of $\boldsymbol{CS_{0,i}}$, $\boldsymbol{CS_{0,i}}$ might be deleted from the SoS. That is to say, in hypergraph $\boldsymbol{\mathcal{H}^j_{fun}}$ satisfying $\boldsymbol{v_i \in e_j}$, $\boldsymbol{v_i}$ is removed from all hyperedges. In other cases, $\boldsymbol{CS_{0,i}}$ may keep the operational independence but some functions of $\boldsymbol{CS_{0,i}}$ may broke, for example the $\boldsymbol{k'}$th function is missing. In all $\boldsymbol{\mathcal{H}^j_{fun}}$ satisfying $\boldsymbol{v_i \in e_j}$, it implies $\boldsymbol{v_i}$ is removed from hyperedges

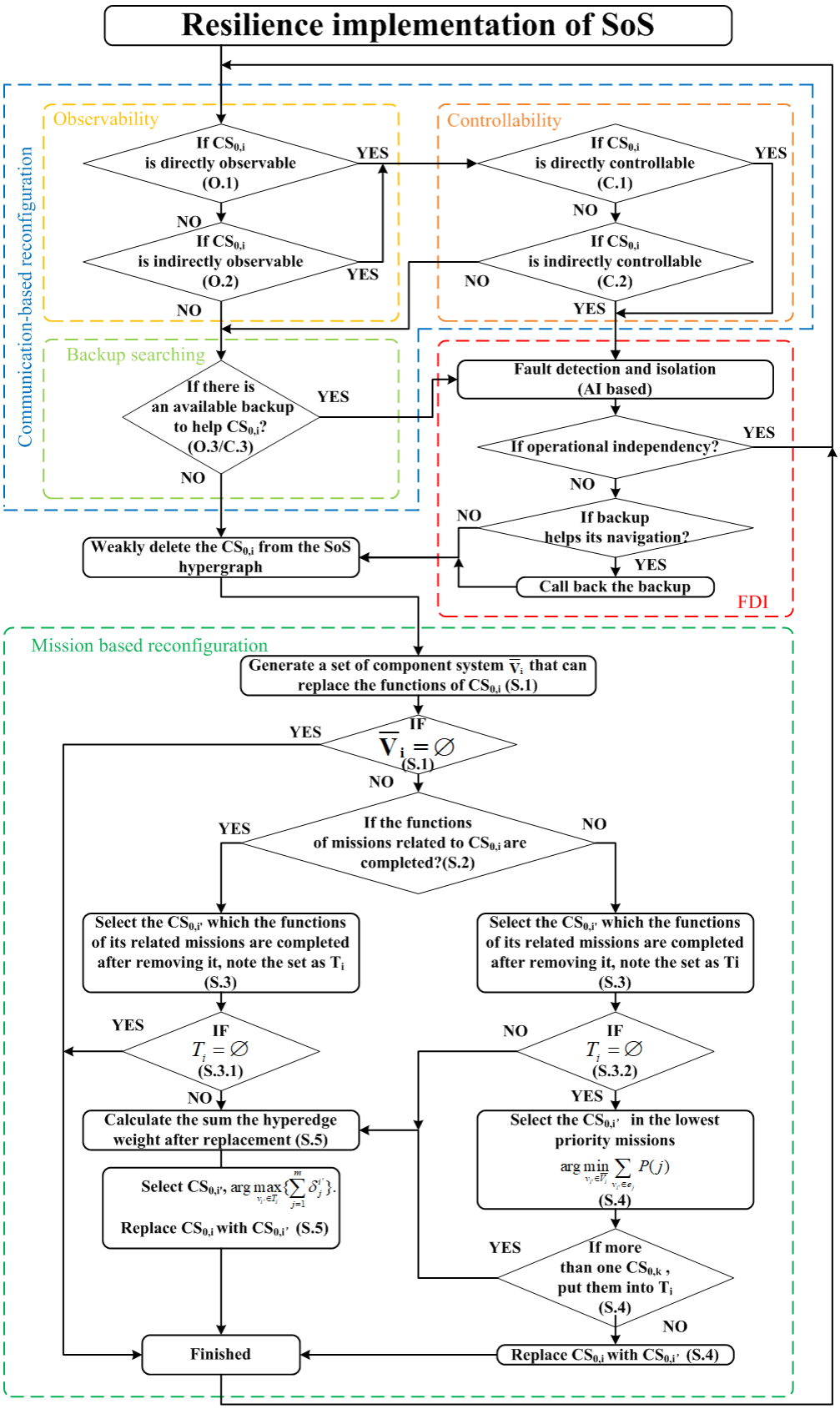


Figure 6.6: Resilience implementation of SoS

$e^{j}_{fun,k'}$. Nevertheless, the malfunction of $CS_{0,i}$ may cause serious consequences on the performance of the whole SoS. On account of A.3, to ensure the necessary the functions of related missions, we address the following mission-based reconfiguration process after isolation of $CS_{0,i}$

**** ***Step 4***: Mission-based **reconfiguration**.

S.1. Establish an optional vertex set $\bar{\mathcal{V}}_i$ for $CS_{0,i}$, where the vertices in $\bar{\mathcal{V}}_i$ represents the CSs having the same functions in j-th mission as $CS_{0,i}$. Namely, the vertices in $\bar{\mathcal{V}}_i$ represent the CSs that can undertake the same work of $CS_{0,i}$.

$$\bar{\mathcal{V}}_i = \{v_{i'} | i' \neq i, v_{i'} \in e_{fun,k} \text{ if } v_i \in e^{j}_{fun,k}, \text{ before deleting } v_i\} \tag{6.3}$$

If $\bar{\mathcal{V}}_i = \emptyset$, it means there is no available $CS_{0,i'}$ can undertake the necessary missing functions. The reconfiguration is finished. Else, $\bar{\mathcal{V}}_i \neq \emptyset$. Go to S.2.

S.2. Check for all k, j satisfying $v_i \in e^{j}_{fun,k}$ before deleting v_i,

$$e^{j}_{fun,k} \neq \emptyset, \text{after deleting } v_i \tag{6.4}$$

If above inequality exists, it implies although $CS_{0,i}$ is removed from SoS, the functions of its related missions are still completed. But the missions' efficiency may decrease. Go to S.3.1. After deleting v_i, if $e^{j}_{fun,k} = \emptyset$ for some k, j, it implies the function completeness of the corresponding missions are broken. Go to S.3.2.

S.3. Check whether there exists a $v_{i'}$ in $\bar{\mathcal{V}}_i$ that the function completeness of missions related to $CS_{0,i'}$ can be ensured if we removed $CS_{0,i'}$. In hypergraph, it means for each $v_{i'} \in \bar{\mathcal{V}}_i$, check the following inequality

$$e^{j'}_{fun,k'} \setminus \{v_{i'}\} \neq \emptyset \text{ if } v_{i'} \in e^{j'}_{fun,k'} \tag{6.5}$$

Denote T_i as the vertex set, where the vertices in T_i are $\sqsubseteq_{i'} \in \bar{\mathcal{V}}_i$ that satisfy inequality (6.5).

S.3.1. If $T_i = \emptyset$, we cannot find an available $CS_{0,i'}$ to keep efficiency. The reconfiguration is finished. Else, $T_i \neq \emptyset$. Go to step 5.

S.3.2. If $T_i = \emptyset$, we cannot find an available $CS_{0,i'}$ to ensure function completeness of all missions. Go to Step.4. Else, $T_i \neq \emptyset$. Go to step 5.

S.4. Define $P(j) : \{1, 2, ..., m\} \to \{\tau_1, \tau_2, ..., \tau_m\}$ as the priority function of the j-th mission. The larger the $P(j)$ is, the higher the priority of the j-th mission is. Select $v_{i'}$ such that

$$\arg \min_{v_{i'} \in \bar{\mathcal{V}}_i} \sum_{\sqsubseteq_{i'} \in e_j} P(j) \tag{6.6}$$

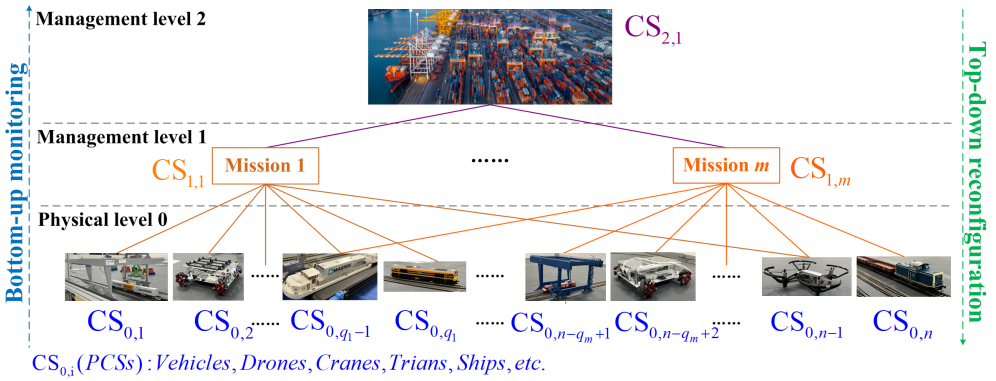


Figure 6.7: Organization structure of physical twin.

If $\arg\min_{v_{i'} \in \bar{\mathcal{V}}_i} \sum_{v_{i'} \in e_j} \boldsymbol{P(j)} \geq \sum_{v_i \in e_j} \boldsymbol{P(j)}$, it means the replacement will violate the priority principle, the reconfiguration is finalized. If there is only one $\boldsymbol{v_{i'}}$ can realize the minimal value of $\sum_{v_{i'} \in e_j} \boldsymbol{P(j)}$. Replace $\boldsymbol{CS_{0,i}}$ with $\boldsymbol{CS_{0,i'}}$. The reconfiguration is finished. If there exists not only one vertex in $\bar{\mathcal{V}}_i$ can realize the minimal value of $\sum_{v_{i'} \in e_j} \boldsymbol{P(j)}$, then put these vertices into $\boldsymbol{T_i}$.

S.5. For each $\boldsymbol{v_{i'}} \in \boldsymbol{T_i}$, suppose we replace $\boldsymbol{v_i}$ with $\boldsymbol{v_{i'}}$ in $\boldsymbol{\mathcal{H}_{SoS}}$ and all $\boldsymbol{\mathcal{H}^j_{fun}}$. Based on the newly generated hypergraph related to $\boldsymbol{v_{i'}}$, calculate weight matrix $\boldsymbol{W^{i'}}$ and hyperedge weight $\boldsymbol{\delta^{i'}_j}$ upon (6.1) and (6.2). Select $\boldsymbol{v_{i'}}$ such that

$$\arg\max_{v_{i'} \in \boldsymbol{T_i}} \sum_{j=1}^{m} \boldsymbol{\delta^{i'}_j} \tag{6.7}$$

The selected $\boldsymbol{v_{i'}}$ ensures the maximal efficiency reconfiguration. Then replace $\boldsymbol{CS_{0,i}}$ with $\boldsymbol{CS_{0,i'}}$. The reconfiguration is finished.

As a summary, we address the following flowchart Fig.6.6 to illustrate the resilience implementation of SoS, which contains both reconfiguration and FDI.

6.2.3 Physical twin of the smart port

6.2.3.1 System description

In this section, a SoS is applied to a set of Robots which simulated the real ports' facilities in the physical twin of the smart port (see in Fig.6.7) to validate our theoretical resilience implementation. The physical twin platform is designed to simulate the port environment under the framework of SPEED project [1].

The PCSs involve in this part including the following robots:

- **T-quad**: T-Quad (see $\boldsymbol{CS_{0,2}}$ and $\boldsymbol{CS_{0,n-2}}$ in Fig.6.7) is a multi-activity robot, based on an Arduino Mega card. In our case, it's a 4-wheel holonomic

[1] https://www.smartportsecosystem.com

robot (Mecanum wheels). It's applued to simulate the Reach stacker which is a vehicle used for handling intermodal cargo containers in small terminals or medium-sized ports.

- **Electric rubber-tired gantry cranes model**: There are two types of electric rubber-tired gantry cranes models (see $\boldsymbol{CS_{0,1}}$ and $\boldsymbol{CS_{0,n-3}}$ in Fig.6.7), which is applied to simulate the operations of rubber-tired gantry cranes, that is, moving containers on the platform to organize storage in piles, after they are unloaded from container vessels, and load them on trucks.

- **Electric train model**: There are two types of Electric train models (see $\boldsymbol{CS_{0,4}}$ and $\boldsymbol{CS_{0,n}}$ in Fig.6.7), which is applied to simulate the rail logistics in port.

- **Electric ship model**: The model (see $\boldsymbol{CS_{0,3}}$ in Fig.6.7) simulates the maritime transport in port concluding both the import and export logistics.

- **Tello drone**: Tello (see $\boldsymbol{CS_{0,n-1}}$ in Fig.6.7) is a mini drone equipped with an HD camera. It'a used to simulate the more powerful drones which are used in port to implement the monitoring and establishing indirect observation and control communication pathways.

What's more, the motion capture markers (OptiTrack[2]) is equipped around the top of the physical twin platform to track movement accurately in three-dimensional space.

In the SoS, there are $\boldsymbol{m+1}$ managerial CSs, including $\boldsymbol{CS_{2,1}}$ in management level $\mathbf{2}$, $\boldsymbol{CS_{1,1}}$ to $\boldsymbol{CS_{1,m}}$ in management level $\mathbf{1}$ and $\boldsymbol{n}$ physical CSs in physical level $\mathbf{0}$, the organization structure is given in Fig.6.7.

Each T-quad is equipped with a Raspberry Pi, a webcam and also an ultrasonic sensor. The ultrasonic sensor ensures that T-quad won't collide with the robot in front. Therefore, the hypergraphs of this SoS can be describe as below:

The mission-based SoS hypergraph $\boldsymbol{\mathcal{H}_{SoS}}$:

$$
\begin{aligned}
\boldsymbol{\mathcal{H}_{SoS}} =& (\bar{\mathcal{V}}, \mathcal{E}, \mathcal{W}), \\
\bar{\mathcal{V}} =& \mathcal{V} \cup \mathcal{V}' = \{CS_{0,1}, CS_{0,2}, ..., CS_{0,n}\} \cup \{CS^a_{0,1}, CS^a_{0,2}, ..., CS^a_{0,n'}\}, \\
\mathcal{E} =& \{CS_{1,1}, ..., CS_{1,m}\} = \{\{CS_{0,1}, CS_{0,2}, ..., CS_{0,q_1-1}, CS_{0,q_1}, CS_{0,n-1}\} \\
& , ..., \{CS_{0,3}, CS_{0,n-q_m+1}, CS_{0,n-q_m+2}, ..., CS_{0,n-1}, CS_{0,n}\}, \\
& \{CS^a_{0,1}, CS^a_{0,2}, ..., CS^a_{0,n'}\}\}
\end{aligned}
\tag{6.8}
$$

where $\boldsymbol{CS^a_{0,i'}, i' = 1, 2, ..., n'}$ represents the backup Tellos. Note that, each backup Tello with front camera is applied to achieve the communication-based reconfigura-

[2]OptiTrack is a leading motion capture technology company specializing in high-precision 3D tracking systems used in a variety of industries, including animation, virtual reality (VR), robotics, biomechanics, sports science, and drone tracking.

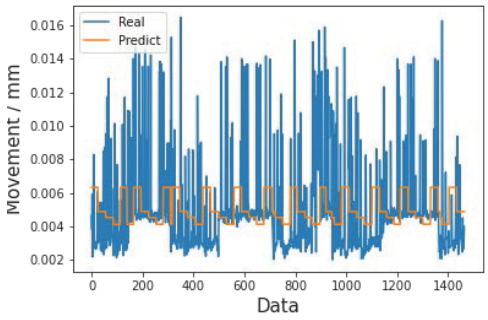


Figure 6.8: KNN based movement predictor

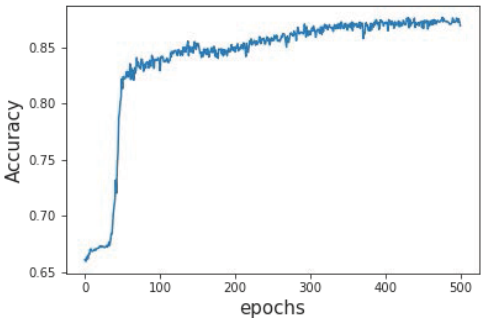


Figure 6.9: Training accuracy of NN-based FDI model

tion. The function-based SoS hypergraph $\mathcal{H}_{fun}$:

$$\begin{aligned} &\mathcal{H}_{fun} = \{\mathcal{V}, \mathcal{E}_{fun}\}, \\ &\mathcal{E}_{fun} = \{e_{Move}, e_{Ultrasonic}, e_{Video}, e_{Wifi}, e_{Positioning}\} \end{aligned} \tag{6.9}$$

6.2.3.2 Experimental Results

The resilient implementation mainly focus on the malfunctions of PCSs. Taking T-quads in mission 1 and mission m as examples, we propose following three scenarios:

1. Using $\boldsymbol{CS_{0,2}}$ in mission 1 as an example, discuss the situation where one of the wheels of T-quad is crashed.

2. Using $\boldsymbol{CS_{0,2}}$ in mission 1 as an example, discuss the situation where the positioning signal of T-quad is lost (remove the motion capture markers).

3. Using $\boldsymbol{CS_{0,2}}$ in mission 1 and $\boldsymbol{CS_{0,n-q_m+2}}$ in mission m as examples, discuss the situation where multiple T-quads are unobservable and/or uncontrollable.

For the first scenario, the SoS initializes as described in equations (6.8) and (6.9). $\boldsymbol{CS_{0,2}}$ pass ***Step 1*** and ***Step 2*** due to its ability to communicate with $\boldsymbol{CS_{1,1}}$. For ***Step 3***, a NN-based FDI model is trained based on the velocity of T-quads and the corresponding movements. There are four motions (go forward/backward/left/right) and three situations (normal/one wheel crashed/two wheels crashed). Training accuracy of NN-based FDI model is given in Fig.6.9. And the prediction of the model is shown in Fig.6.8.

The model is used to detect and locate the fault of driving system of the T-quad ($\boldsymbol{CS_{0,2}}$). When $\boldsymbol{CS_{1,1}}$ get the information that one of the wheels of $\boldsymbol{CS_{0,2}}$ is crashed, $\boldsymbol{CS_{1,1}}$ will switch the move mode of $\boldsymbol{CS_{0,2}}$ from four-wheel drive to two-wheel drive, and $\boldsymbol{CS_{0,2}}$ is still operational independent. This adaptation allows $\boldsymbol{CS_{0,2}}$ to remain operational independently, ensuring continued functionality despite the detected fault. Thus, the reconfiguration process is successfully completed after ***Step 3***. In more severe scenarios, such as when multiple critical components

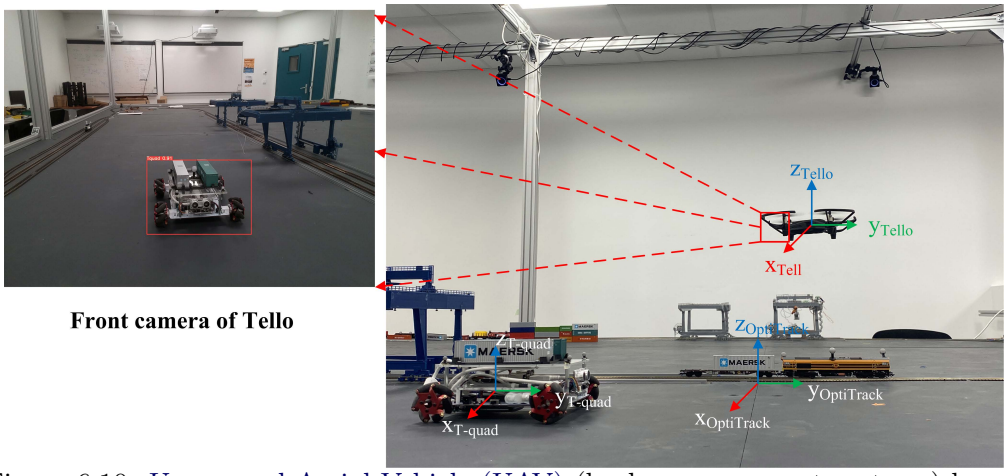


Figure 6.10: Unmanned Aerial Vehicle (UAV) (backup component systems) based reconfiguration.

are damaged, and simple reconfiguration is insufficient, the system requires the intervention of maintenance personnel. To this end, the crew dispatch optimization strategy for repairing inoperational PCSs will be further elaborated in Chapter 6.4.

Scenarios 2 and 3 begin with the same initialization process as described in Equations (6.8) and (6.9). In Scenario 2, $\boldsymbol{CS_{0,2}}$ successfully completes ***Step 1*** and ***Step 2*** without any faults detected in the driving system, indicating that the T-quad's mobility is fully operational. However, during ***Step 3***, $\boldsymbol{CS_{1,1}}$ receives all necessary information except for the precise position of $\boldsymbol{CS_{0,2}}$, which aligns with a special case defined in this step.

To address this situation, an additional backup, $\boldsymbol{CS^a_{0,1}}$, is integrated into the SoS by being assigned to $\boldsymbol{CS_{1,1}}$. The role of $\boldsymbol{CS^a_{0,1}}$ is to track the final position of $\boldsymbol{CS_{0,2}}$ through object detection techniques, enhancing the overall system's situational awareness and accuracy in positioning (see Fig. 6.10). $\boldsymbol{CS^a_{0,1}}$ utilizes its front camera to visually detect and follow $\boldsymbol{CS_{0,2}}$.

Once $\boldsymbol{CS^a_{0,1}}$ identifies $\boldsymbol{CS_{0,2}}$, $\boldsymbol{CS_{1,1}}$ processes the visual data to calculate the distance between the two systems. This distance measurement is crucial for accurately estimating the position of $\boldsymbol{CS_{0,2}}$. The calculation involves comparing the observed size of $\boldsymbol{CS_{0,2}}$ in the camera's pixel representation with its known real-world dimensions. By analyzing the proportional relationship between these sizes, the system transforms the known position of $\boldsymbol{CS^a_{0,1}}$ into an estimated position for $\boldsymbol{CS_{0,2}}$. This method effectively compensates for the lack of direct positional data, ensuring that $\boldsymbol{CS_{1,1}}$ can maintain accurate situational awareness and continue to monitor $\boldsymbol{CS_{0,2}}$ effectively.

This integration of object detection and relative positioning enhances the robustness of the system, allowing it to adapt to scenarios where standard positional data is unavailable. The overall approach highlights the system's ability to dynamically incorporate additional resources and methodologies to maintain operational integrity under diverse conditions.

Let the frame of OptiTrack be the base coordinate system which is denoted as $\mathfrak{R}_{base} = \{O_{Op}, X_{Op}, Y_{Op}, Z_{Op}\}$. The frame of Tello and T-quad are noted as $\mathfrak{R}_{Tel} = \{O_{Tel}, X_{Tel}, Y_{Tel}, Z_{Tel}\}$, $\mathfrak{R}_{Tq} = \{O_{Tq}, X_{Tq}, Y_{Tq}, Z_{Tq}\}$.

According to the coordinate transformation (see in Appendix B), T-quad ($CS_{0,2}$) coordinate in frame $\mathfrak{R}_{base}$ can be calculated by the following equation:

$$\begin{bmatrix} x_{Op} \\ y_{Op} \\ z_{Op} \end{bmatrix} = (R_2 R_1)^{-1} \begin{bmatrix} x_{Tq} \\ y_{Tq} \\ z_{Tq} \end{bmatrix} + \begin{bmatrix} x_{Tel}^{Op} \\ y_{Tel}^{Op} \\ z_{Tel}^{Op} \end{bmatrix} + R_1^{-1} \begin{bmatrix} x_{Tq}^{Tel} \\ y_{Tq}^{Tel} \\ z_{Tq}^{Tel} \end{bmatrix} \tag{6.10}$$

As for Scenario 3, it is observed that $CS_{0,2}$ in mission 1 and $CS_{0,n-q_m+2}$ in mission m are unobservable and/or uncontrollable during ***Step 1*** and ***Step 2***. Specifically, in the unobservable case, $CS_{1,1}$ fails to receive any information from $CS_{0,2}$, while $CS_{1,m}$ does not receive information from $CS_{0,n-q_m+2}$. In the uncontrollable case, $CS_{0,2}$ does not respond to commands issued by $CS_{1,1}$, and similarly, $CS_{0,n-q_m+2}$ does not respond to instructions from $CS_{1,m}$. These conditions indicate communication breakdowns and potential system faults that prevent effective coordination and control of the respective units.

To address these issues, the system initiates a communication-based reconfiguration strategy. $CS_{0,1}^{a}$ is dispatched to the last known position of $CS_{0,2}$ to re-establish a communication link and facilitate the reconfiguration process. Similarly, $CS_{0,i'm}^{a}$ is dispatched to the final position of $CS0, n - q_m + 2$ to achieve the same objective for mission m. The primary role of $CS_{0,1}^{a}$ is to act as an intermediary, enabling $CS_{0,2}$ to communicate effectively with $CS_{1,1}$, thus bridging the gap caused by the unobservability or uncontrollability issues. While performing this task, $CS_{0,1}^{a}$ also tracks the real-time position of $CS_{0,2}$, enhancing the accuracy of position data and ensuring the stability of the communication link.

Similarly, $CS_{0,i'm}^{a}$ not only assists in restoring communication between $CS0, n - q_m + 2$ and $CS_{1,m}$ but also continuously tracks the position of $CS_{0,n-q_m+2}$. This dual function of communication and tracking ensures that $CS_{0,n-q_m+2}$ remains correctly positioned and its status accurately reported, even under challenging conditions where direct control and observation are compromised. Since the positioning functions of $CS_{0,2}$ and $CS_{0,n-q_m+2}$ are operational, $CS_{0,1}^{a}$ and $CS_{0,i'm}^{a}$ can effectively receive position information from $CS1, 1$ and $CS_{1,m}$, respectively. This data plays a crucial role in facilitating communication-based reconfiguration.

6.2.4 Conclusion

Based on hypergraph theory, a resilience implementation framework of SoS is developed in this section, which is divided into bottom-up monitoring and top-down reconfiguration. Bottom-up monitoring implies the FDI on cyber-physical component systems at the bottom level and the fault information flow from lower level to higher level. Meanwhile, supervisors at higher levels rearrange the whole SoS to optimize the suited performance. Various hypergraph models are given from the perspectives of communication (i.e.information exchange), function and mission to

describe the complex relationship between component systems of SoS. Upon the multiple hypergraph models and FDI, the dynamic reconfiguration process includes communication-based, operational independence and mission-based reconfiguration. Above resilience implementation process is applied to the physical twin of the smart port considering several scenarios in case study. The experiment results show the effectiveness of our theoretical analysis.

6.3 SoS Resilient Planning Facing Predictable Disaster

This section focus on the resilience planning in advance for SoS facing disruptive disaster scenarios on the basis of disaster prediction. The structure of SoS is characterized by a proposed scenario-dependent hypergraph model, in which the defined edge-dependent weight function describe the multi-way interaction between the composed PCSs and operating sites.

6.3.1 Modelling and problem formulation

6.3.1.1 Basic assumptions

Based on the practical operations under disasters, the following assumptions are proposed to facilitate the resilience planning analysis:

A.1 All considered scenarios, which composes of three types: normal situation, multiple kinds of operating sites maintenance and various kinds of nature disasters, are predictable and foreseeable.

A.2 The managerial cost is presumed to be a constant value under multiple management activities.

A.3 The capacity for each operating site $\boldsymbol{os_m}$ is known and remains constant during the whole resilience planning time horizon.

A.4 Each PCS operates in normal status (without any malfunctions) during the whole resilience planning time horizon.

A.5 The considered PCSs (including human operators and machines) are heterogeneous. The unit workload and cost varies for each PCS. Namely, $\boldsymbol{L_i^u}/\boldsymbol{C_i^u}$ varies with different $\boldsymbol{i}$.

A.6 For each PCS, its unit cost and workload are invariant with scenario $\boldsymbol{s}$.

6.3.1.2 Hypergraph-based modelling in normal scenario

Denote $\boldsymbol{H = (V_P, E_{os}, W)}$ as the hypergraph describing the organization between PCSs and operating sites. Vertex set $\boldsymbol{V_P = \{v_1, v_2, ..., v_{|V_P|}\}}$ represents the set of PCSs). $\boldsymbol{v_i}$ represents the $\boldsymbol{i}$-th PCS ($\boldsymbol{PCS_i}$). The cardinality of set $\boldsymbol{V_P}$ is denoted as $\boldsymbol{|V_P|}$, which signifies the total number of PCSs. Hyperedge set

$\boldsymbol{E_{os}} = \{\boldsymbol{os_1}, \boldsymbol{os_2}, ..., \boldsymbol{os_{|E_{os}|}}\}$ stands for the set of operating site, where $|\boldsymbol{E_{os}}|$ is the cardinality of set $\boldsymbol{E_{os}}$, i.e., the total number of operating sites. Note that $\boldsymbol{v_i} \in \boldsymbol{os_m}$ if $\boldsymbol{PCS_i}$ is in the operating site $\boldsymbol{os_m}$. The indicator function of vertex $\boldsymbol{v_i}$ and edge $\boldsymbol{o_{sm}}$ is denoted as $\boldsymbol{I(v_i, os_m)}$ and satisfies

$$\boldsymbol{I(v_i, os_m)} = \begin{cases} \mathbf{1}, & \text{if } \boldsymbol{v_i} \in \boldsymbol{os_m} \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{6.11}$$

Considering the maximal capacity of each operating sites, the cardinality of hyperedge $\boldsymbol{os_m}$ satisfies

$$|\boldsymbol{os_m}| \leq \boldsymbol{\mathcal{CA}_m^{max}} \tag{6.12}$$

The edge-dependent weight function $\boldsymbol{W(v_i, os_m)}$ corresponds to hypergraph $\boldsymbol{H}$ is defined with various attributes

$$\boldsymbol{W(v_i, os_m)} =< \boldsymbol{C_{i,m}^u}, \boldsymbol{L_{i,m}^u}, \boldsymbol{oh_{i,m}}, \boldsymbol{T_{i,m}^d} >, \tag{6.13}$$

where attributes $\boldsymbol{C_{i,m}^u}$, $\boldsymbol{L_{i,m}^u}$, $\boldsymbol{oh_{i,m}}$ and $\boldsymbol{T_{i,m}^d}$ represent the unit cost, unit workload, operating hours and desired operating hours for $\boldsymbol{PCS_i}$ in the operating site $\boldsymbol{os_m}$ respectively.

Note that above attributes are independent with the emergence of disasters and only relate to the organization between PCSs and operating sites. Facing with various kinds of disasters (scenarios), the following scenario-dependent hypergraph is proposed.

6.3.1.3 Disaster scenario-dependent hypergraph-based modelling

Let $\boldsymbol{\mathcal{H}_s} = \boldsymbol{(V_P^s, E_{os}^s, W_s)}$ be a scenario-dependent hypergraph describing the organization of SoS in scenario $\boldsymbol{s}$. $\boldsymbol{V_p^s}$ denotes the vertex set, which represents the set of PCSs operates in scenario $\boldsymbol{s}$. The definition of $\boldsymbol{V_p^s}$ can be given by

$$\boldsymbol{V_p^s} = \{\boldsymbol{v_i} | \boldsymbol{I(v_i, s)} = \mathbf{1}, \boldsymbol{v_i} \in \boldsymbol{V_P}\} \tag{6.14}$$

Besides, $\boldsymbol{E_{os}^s}$ signifies the hyperedge set, which indicates the set of operating sites that contains PCSs in scenario $\boldsymbol{s}$. The constitution of $\boldsymbol{E_{os}^s}$ can be depicted by

$$\begin{aligned} \boldsymbol{os_m^s} = \{\boldsymbol{v_i} | \boldsymbol{I(v_i, s)} * \boldsymbol{I(v_i, os_m)} = \mathbf{1}, \boldsymbol{v_i} \in \boldsymbol{os_m}\}, \\ \boldsymbol{m} = \mathbf{1}, \mathbf{2}, ..., |\boldsymbol{E_{os}}|, \\ \boldsymbol{E_{os}^s} = \{\boldsymbol{os_m^s} | \boldsymbol{os_m^s} \neq \emptyset, \boldsymbol{m} = \mathbf{1}, \mathbf{2}, ..., |\boldsymbol{E_{os}}|\} \end{aligned} \tag{6.15}$$

The corresponding edge-dependent weight function $\boldsymbol{W_s(v_i, os_m)}$ are defined with the following attributes

$$\boldsymbol{W(v_i, os_m)} =< \boldsymbol{t_{i,m,s}^a}, \boldsymbol{t_{i,m,s}^b}, \boldsymbol{T_{i,m,s}^p} >, \tag{6.16}$$

in which $\boldsymbol{t^a_{i,m,s}}$ represents the predicted inoperable duration of $\boldsymbol{PCS_i}$ after $\boldsymbol{t_0}$ in scenario $\boldsymbol{s}$. Note that $\boldsymbol{t_0}$ is the predicted time for the occurrence of the scenario. And $\boldsymbol{t^b_{i,m,s}}$ implies the predicted duration of $\boldsymbol{PCS_i}$ planned to operates in $\boldsymbol{os_m}$ before $\boldsymbol{t_0}$ in scenario $\boldsymbol{s}$. $\boldsymbol{T^p_{i,m,s}}$ means the predicted operating time of $\boldsymbol{PCS_i}$ working in $\boldsymbol{os_m}$ in scenario $\boldsymbol{s}$.

6.3.1.4 Problem formulation

The planning-oriented optimization facing disasters for various kinds of complex systems have been research extensively during the past few decades. The existing results mainly focus on the repairing scheduling for the damaged sites. However, in this section we focus on ensuring the performance of SoS facing disasters based on resilience planning in advance based on the disaster prediction model.

The main task of SoS resilience planning can be divided into the following two parts:

- Establish a prediction model for the possible disasters.
- On the basis of the prediction of disasters, perform the planning optimization to minimize the operating cost and meanwhile respect the acceptable degradation in performance.

6.3.2 Data-driven based scenario prediction framework via multi-scale time series analysis

As we have mentioned in Section 6.3.1, the scenarios can be divided into three types. The first type refers to the normal situation that all the CSs operate well without any external disturbances according to plan. Note that, the total workload of SoS in normal scenario is regarded as the benchmark in the following resilience planning. The second type contains various kinds of foreseeable maintenance related to the operating sites, such as road maintenance, rail track maintenance, communication equipment maintenance, warehouse maintenance, etc. In maintenance scenario, the operating scheduling is known, on the basis of which we can update the work plan to reach an acceptable total workload corresponding to the benchmark of normal scenario. The final type includes multiple kinds of predictable nature disasters such as hurricane, flood or any other natural disasters which may cause significant degradation and inconvenience in the operating environment. Note that the malfunctions of PCSs are out of our discussion and we presuppose all PCSs are in normal operation in terms of **Assumption A.4** in subsection 6.3.1.1.

Observe that the influence of diverse disaster type scenarios varies with operating site $\boldsymbol{os_m}$ and time $\boldsymbol{t}$. In order to achieve an accurate disaster type scenario prediction, the key point is to collect a suitable data-set for the training of AI-based predictor. In this regard, an illustration example is shown in Section 6.3.2.1 below to present a feasible data collection and prediction framework of one kind of disasters: flash flood.

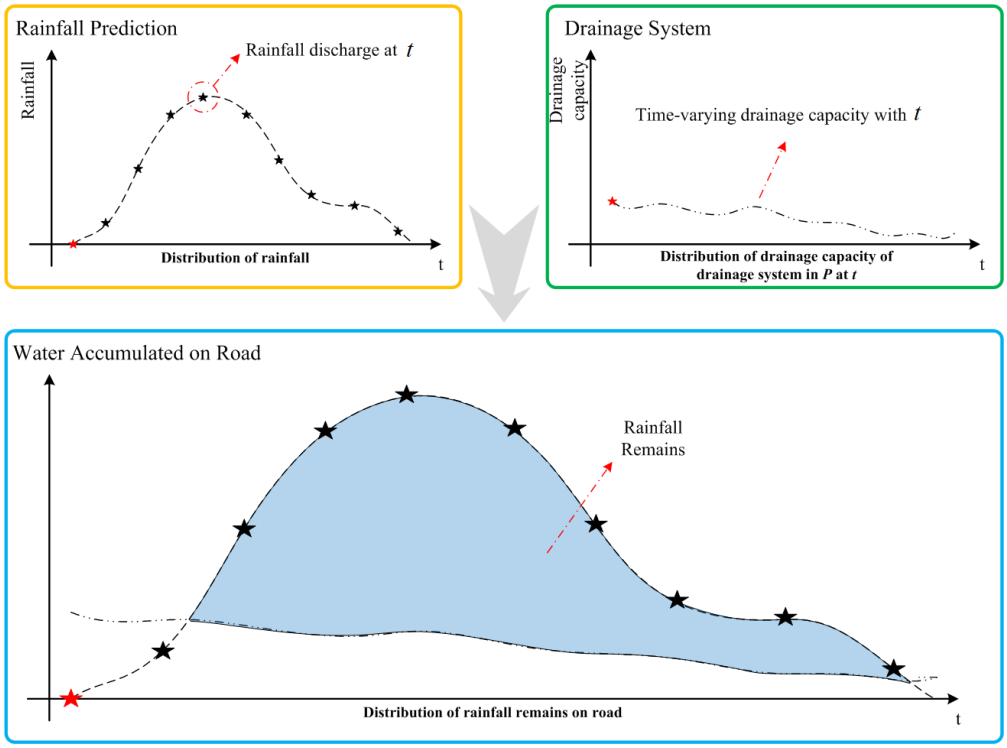


Figure 6.11: Example of Flood scenario estimation in resilience planning.

6.3.2.1 An Example of data collection and prediction framework: Flash Flood

Floods are often caused by heavy rainfall, rapid snowmelt or a storm surge from a tropical cyclone or tsunami in coastal areas. In this example, we only consider the floods caused by heavy/excessive rainfall (Flash Flood). Point out that the collection of rainfall data set is essential in the flash flood predictor training, which requires two types of data: the amount of rainfall and the drainage capacity. It's known to all that the amount of rainfall is time-varying and is dependent with location (the operating site $\boldsymbol{os_m}$).

This example 6.11 focus on the flash flood in the urban area. For one fixed operating site $\boldsymbol{os_m}$, the yellow block in Fig.6.11 shows the evolution of rainfall prediction with time $\boldsymbol{t}$ that obtained from the historical data. Considering the urban drainage systems, the drainage capacity of different operating site $\boldsymbol{os_m}$ depends on the capacity and amounts of related drains and ditches. In addition, a heavy rain may result in accumulation of leaves and silt in drains and ditches, which can decrease the drainage capacity with time. Meanwhile, improvements and updates of urban drainage systems will increase the drainage capacity. Therefore, the drainage capacity also varies with operating site $\boldsymbol{os_m}$ and time $\boldsymbol{t}$. The green block in Fig.6.11 shows the drainage capacity with time $\boldsymbol{t}$, in one fixed operating site $\boldsymbol{os_m}$. In the blue block, we draw the trajectories of rainfall discharge prediction and drainage capacity together, in which the area colored blue represents the integral region to

estimate the water accumulation on road.

Rebuild raw meteorological data with multi-scale time series Meteorological data is a typical time series, which has periodic and seasonal characteristics. Thus, simply increasing the amount of data cannot improve the prediction accuracy.

Different scales of time series will present distinct behaviors, which reveals the selection of scales plays an important role in the estimation. In this regard, the *multi-scale time series* is applied to determine the data size that involved in the rainfall prediction. Multi-scale time series analysis allows consistent modelling of time series at different levels of resolution (e.g., daily or monthly aggregates of meteorological [Plocoste 2019] or financial [Jiang 2017] data).

An hybrid rainfall prediction model In this work, we propose the following hybrid prediction model based on multi-scale time series to estimate the rainfall, which can be divided into the following two parts:

Step 1. A raining forecasting model (i.e., a classification model) to predict whether it rains or not;

Step 2. Only for those predicted rainy days in Step 1, estimate the related rainfalls in terms of the data of historical rainy days based on the multi-scale time series.

The reason for performing the first step in advance is to improve the prediction accuracy. If we directly perform the rainfall prediction in Step 2 without the classification model in Step 1, the rainfall may present in those days without rains in prediction.

Let $\boldsymbol{\Delta}^{\boldsymbol{m}}_{\boldsymbol{max}} \in \mathbb{N}^{+}$ denote the maximal scale of prediction at operating site $\boldsymbol{os_m}$. The sampling period is denoted as $\boldsymbol{T}$. Besides, let $\mathrm{MS}^{\boldsymbol{m}}$ represent the scale of data related to operating site $\boldsymbol{os_m}$. Based on the multi-scale time series, the data used in both classification (Step 1) and regression (Step 2) for operating site $\boldsymbol{os_m}$ is presented in Table 6.1, which is corresponding to the multi-scale time series $\boldsymbol{t - aT, a \in \{0, 1, 2, ..., \mathrm{MS}^m\}}$.

In Step 1, we use the above historical data to train the classification model (i.e., the raining forecasting model), on the basis of which we can obtain it's predicted to rain or not for operating site $\boldsymbol{os_m}$ at time $\boldsymbol{t + \Delta \times T}$, $\boldsymbol{\Delta \in \{1, 2, ..., \Delta^m_{max}\}}$. Denote the set of predicted rainy lags at operating site $\boldsymbol{os_m}$ as $\boldsymbol{\Omega^m_r}$, which implies it's predicted to rain for site $\boldsymbol{os_m}$ at $\boldsymbol{t + \Delta \times T}$ if and only if $\boldsymbol{\Delta \in \Omega^m_r}$.

In step 2, only for those predicted rainy days $\boldsymbol{t + \Delta \times T}$ at operating site $\boldsymbol{os_m}$, $\boldsymbol{\Delta \in \Omega^m_r}$. We perform the following multi-variable regression to predict the rainfall discharge based on the historical data in Table 6.1.

$$\begin{aligned}\mathrm{Rainfall}^{\boldsymbol{m}}_{\boldsymbol{t+\Delta\times T}} = \boldsymbol{F^m_\Delta}(&\mathrm{Tmin}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \mathrm{Tmax}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \mathrm{Tavg}^{\boldsymbol{m}}_{\boldsymbol{t-aT}},\\ &\mathrm{RH}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \mathrm{SS}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \mathrm{Wmax}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \mathrm{Wd}^{\boldsymbol{m}}_{\boldsymbol{t-aT}},\end{aligned}$$

Table 6.1: Historical data related to multi-scale time series used in classification and regression

Feature	Description
$\mathrm{Tmin}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Minimal temperature ($^{\circ}\boldsymbol{C}$)
$\mathrm{Tmax}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Maximal temperature ($^{\circ}\boldsymbol{C}$)
$\mathrm{Tavg}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Average temperature ($^{\circ}\boldsymbol{C}$)
$\mathrm{RH}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Average humidity (%)
$\mathrm{RF}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Rainfall (mm)
$\mathrm{SS}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Duration of sunshine (hour)
$\mathrm{Wmax}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Maximal wind speed (m/s)
$\mathrm{Wd}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Wind direction at maximal speed ($^{\circ}$)
$\mathrm{Wavg}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}$	Average wind speed (m/s)

$$\mathrm{Wavg}^{\boldsymbol{m}}_{\boldsymbol{t-aT}}, \boldsymbol{a = 0, 1, 2, ...,} \mathrm{MS}^{\boldsymbol{m}}) + \boldsymbol{\varepsilon}, \tag{6.17}$$

where $\mathrm{Rainfall}^{\boldsymbol{m}}_{\boldsymbol{t+\Delta \times T}}$ represents the predicted rainfall at operating site $\boldsymbol{os_m}$ at time $\boldsymbol{t+\Delta \times T}$. $\boldsymbol{\Delta \times T}$ represents the time interval between the final sampling point and the predicted point. Besides, function $\boldsymbol{F^m_\Delta(\cdot)}$ signifies the multi-variable regression function utilized in the rainfall prediction at time $\boldsymbol{t+\Delta \times T}$ for operating site $\boldsymbol{os_m}$. In this work, we select $\boldsymbol{F^m_\Delta(\cdot)}$ as the polynomial regression function. Additionally, $\boldsymbol{\varepsilon}$ represents the residual error.

6.3.3 Optimization for resilience planning based on disaster prediction

Based on the data-driven based scenario prediction in Section 6.3.2, the optimization of operating hours for each $\boldsymbol{PCS_i}$ under operating site $\boldsymbol{m}$ during the whole resilience planning time horizon $\boldsymbol{T^{total}}$ is given below:

$$\min_{\boldsymbol{T^p_{i,m,s}}} \ \sum_{\boldsymbol{i}} \sum_{\boldsymbol{m}} \boldsymbol{C^u_{i,m} T^p_{i,m,s}} \tag{6.18}$$

$$\text{s.t. } \boldsymbol{0 \le \sum_i I(v_i, os_m) \le \mathcal{CA}^{max}_m} \qquad \boldsymbol{\forall os_m \in E_{os}} \tag{6.19}$$

$$\boldsymbol{0 \le \sum_i I(v_i, s) \le |V_P|} \tag{6.20}$$

$$\boldsymbol{\frac{\sum_i \sum_m L^u_{i,m} T^p_{i,m,s}}{\sum_i \sum_m L^u_{i,m} T^d_{i,m}} \ge Per^{lb}} \tag{6.21}$$

$$\mathbf{0} \leq \sum_{m} \boldsymbol{T}^{\boldsymbol{p}}_{\boldsymbol{i,m,s}} \leq \frac{\mathbf{48}}{\mathbf{7}} * \lceil \frac{\boldsymbol{T}^{\boldsymbol{total}}}{\mathbf{24}} \rceil \qquad \forall \boldsymbol{v_i} \in \boldsymbol{V}^{\text{human}}_{\boldsymbol{P}} \quad (6.22)$$

$$\mathbf{0} \leq \sum_{m} \boldsymbol{T}^{\boldsymbol{p}}_{\boldsymbol{i,m,s}} \leq \boldsymbol{T}^{\boldsymbol{total}} \qquad \forall \boldsymbol{v_i} \in \boldsymbol{V}^{\text{machine}}_{\boldsymbol{P}} \quad (6.23)$$

where the vertex sets $\boldsymbol{V}^{\text{human}}_{\boldsymbol{P}}$ and $\boldsymbol{V}^{\text{machine}}_{\boldsymbol{P}}$ can be depicted by

$$\boldsymbol{v_i} \in \begin{cases} \boldsymbol{V}^{\text{human}}_{\boldsymbol{P}}, & \text{if } \boldsymbol{PCS_i} \text{ is human operator,} \\ \boldsymbol{V}^{\text{machine}}_{\boldsymbol{P}}, & \text{if } \boldsymbol{PCS_i} \text{ is machine.} \end{cases}$$

In the scenario $\boldsymbol{s}$, the objective function (6.18) aims at minimizing the total cost of operation (quantified as $\sum_i \sum_m \boldsymbol{C}^{\boldsymbol{u}}_{\boldsymbol{i,m}} \boldsymbol{T}^{\boldsymbol{p}}_{\boldsymbol{i,m,s}}$), by the **planning** of operating hour $\boldsymbol{T}^{\boldsymbol{p}}_{\boldsymbol{i,m,s}}$ for PCS$_i$ works in operating site $\boldsymbol{os_m}$ during the whole time horizon $\boldsymbol{T}^{\boldsymbol{total}}$.

Observe that inequality (6.19) reveals the number of PCSs working in each operating site $\boldsymbol{os_m}$ cannot exceed its maximal capacity. Besides, (6.20) suggests the number of PCSs utilized in the resilience planning is no greater than its total number. In inequality (6.21), the numerator represents the total desirable workload of PCSs among all operating sites in normal scenario during time horizon $\boldsymbol{T}^{\boldsymbol{total}}$, while the denominator stands for the sum of workload in scenario $\boldsymbol{s}$. Their ratio is the operational performance of SoS in scenario $\boldsymbol{s}$ and it's required to be no less than the lower bound $\boldsymbol{Per}^{\boldsymbol{lb}}$. Inequality (6.21) implies the degradation of performance compared with that of normal scenario must be constrained. Inequalities (6.22) and (6.23) pose requirements on the maximal operating hours. For the PCSs as human operators, (6.22) means their total working hours should respect the maximum weekly working time in EU, i.e., 48 hours per week. For the PCSs as machines, their working hours are not effected by above constrains. (6.23 indicates the sum of operating hours for PCS$_i$ at operating site $\boldsymbol{m}$ should not exceed the time horizon $\boldsymbol{T}^{\boldsymbol{total}}$ minus the corresponding inoperable hours $\boldsymbol{t}^{\boldsymbol{a}}_{\boldsymbol{i,m,s}}$, where $\boldsymbol{t}^{\boldsymbol{a}}_{\boldsymbol{i,m,s}}$ is obtained based on the data-driven scenario prediction in Section 6.3.2.

6.3.4 Qualitative and quantitative tests

In the simulation, we consider the resilience planning with flood scenario.

6.3.4.1 Rainfall prediction

For the rainfall prediction, we use the open source dataset from Kaggle (https://www.kaggle.com/) to verify the methodology we proposed in Section 6.3.2. The dataset is about Indonesia daily climate data from 2010 to 2022 which contains 173 sampling locations among 34 provinces in Indonesia.

In Step 1, the random forest based classifier shows good performance with 98.8% training accuracy and 79.1% testing accuracy.

In Step 2, the raw daily climate data is rebuilt based on the methodology we proposed in subsection 6.3.2.1. The lag $\boldsymbol{\Delta}$ is in range $\mathbf{[1, 7]}$ and the scale $\boldsymbol{a}$ is in

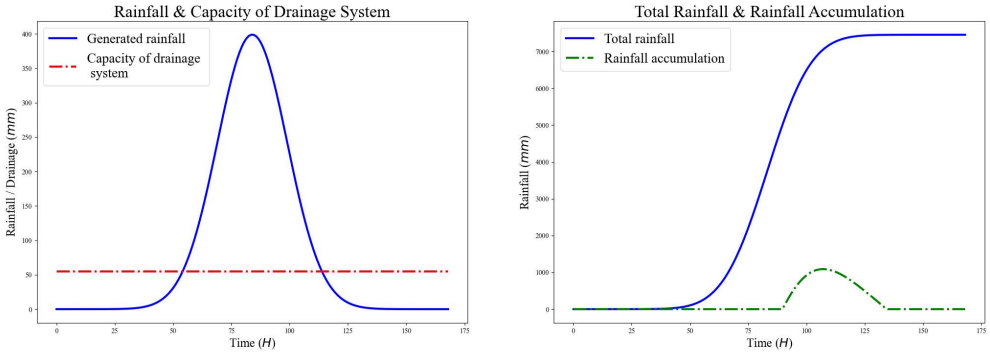


Figure 6.12: Simulated rainfall and its corresponding capacity of drainage system.

range $[\mathbf{1}, \mathbf{9}]$.

In table 6.2, a part of different sampling locations and their related best pairs $(\boldsymbol{\Delta}, \boldsymbol{a})$ to ensure the minimal percentage error are presented, from which we can observe that for distinct sampling locations, the corresponding best pair $(\boldsymbol{\Delta}, \boldsymbol{a})$ varies.

Table 6.2: Results of Rainfall Prediction

Position ID	$\boldsymbol{\Delta}$	$\boldsymbol{a}$	$\boldsymbol{y_{test}}$	$\boldsymbol{y_{pred}}$	% Error
96001	7	6	5.85	6.04	3.31%
96017	4	7	3.63	4.09	12.45%
96087	2	7	9.32	8.84	5.15%
96247	6	6	100.3	102.11	1.80%
96249	1	7	17.4	17.66	1.47%
96581	3	6	30.5	32.37	6.13%
96633	6	6	20.1	20.13	0.15%

6.3.4.2 Water accumulation prediction

Based on the rainfall prediction, for each operating site $\boldsymbol{os_m}$, we are able to know the starting and ending date of rainy days in terms of the classification model in Step 1, which is noted as $\boldsymbol{t_0^m}$ and $\boldsymbol{t_f^m}$. Moreover, one obtains the predicted rainfall $\text{Rainfall}_{\boldsymbol{t}}^{\boldsymbol{m}}$ during $\boldsymbol{t_0^m}$ and $\boldsymbol{t_f^m}$ according to the prediction.

For operating site $\boldsymbol{os_m}$, let $\text{drainage-capacity}_{\boldsymbol{t}}^{\boldsymbol{m}}$ and $\text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}}$ denote the drainage capacity and water accumulation at time $\boldsymbol{t}$. The estimation of $\text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}}$ will be divided into the following three stages regarding the natural evolution of rainfall in Fig.6.11:

1. At the beginning of rainy days, the rainfall is no greater than the drainage ca-

pacity. Denote $\boldsymbol{t_g^m}$ as the first point such that $\text{Rainfall}_{\boldsymbol{t_g}}^{\boldsymbol{m}} \leq \text{drainage-capacity}_{\boldsymbol{t_g^m}}^{\boldsymbol{m}}$. It therefore follows that

$$\text{Rainfall}_{\boldsymbol{t}}^{\boldsymbol{m}} \leq \text{drainage-capacity}_{\boldsymbol{t}}^{\boldsymbol{m}}, \boldsymbol{t} \in [\boldsymbol{t_0^m}, \boldsymbol{t_g^m}] \tag{6.24}$$

In this case, the rate of drainage is greater than the rate of accumulation, which implies there is no water accumulated on the road, that is

$$\text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}} = \boldsymbol{0}, \boldsymbol{t} \in [\boldsymbol{t_0^m}, \boldsymbol{t_g^m}]. \tag{6.25}$$

2. For a time period after $\boldsymbol{t_g^m}$, the water accumulation is quicker than the drainage rate and the water will accumulate on the road. The water accumulation can be calculated by

$$\begin{aligned} \text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}} = \boldsymbol{\max\{0,} \int_{\boldsymbol{t_g^m}}^{\boldsymbol{t}} (\text{Rainfall}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \\ - \text{drainage-capacity}_{\boldsymbol{\tau}}^{\boldsymbol{m}}) \boldsymbol{d\tau\}}, \boldsymbol{t} \in (\boldsymbol{t_g^m}, \boldsymbol{t_f^m}] \end{aligned} \tag{6.26}$$

where the integral is approximated by

$$\int_{\boldsymbol{t_g^m}}^{\boldsymbol{t}} \text{Rainfall}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \boldsymbol{d\tau} \approx \sum_{\boldsymbol{\tau}=\boldsymbol{t_g^m}}^{\boldsymbol{t}} \text{Rainfall}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \times \boldsymbol{T}, \boldsymbol{t} \in (\boldsymbol{t_g^m}, \boldsymbol{t_f^m}]. \tag{6.27}$$

In addition, $\text{drainage-capacity}_{\boldsymbol{\tau}}^{\boldsymbol{m}}$ is treated as a constant $\text{drainage-capacity}^{\boldsymbol{m}}$ obtained by empirical data. Thus, $\int_{\boldsymbol{t_g^m}}^{\boldsymbol{t}} \text{drainage-capacity}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \boldsymbol{d\tau} \approx \text{drainage-capacity}^{\boldsymbol{m}} \times (\boldsymbol{t} - \boldsymbol{t_g^m})$.

3. For $\boldsymbol{t} > \boldsymbol{t_f^m}$, if $\text{water-accumulation}_{\boldsymbol{t_f^m}}^{\boldsymbol{m}} > \boldsymbol{0}$, then there are still water accumulated on the road in the following days although there is no rain after $\boldsymbol{t_f^m}$, which is called the drainage period. In this stage, the water accumulation can be calculated by

$$\begin{aligned} \text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}} = \boldsymbol{\max\{0,} \text{water-accumulation}_{\boldsymbol{t_f^m}}^{\boldsymbol{m}} - \\ \int_{\boldsymbol{t_f^m}}^{\boldsymbol{t}} \text{drainage-capacity}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \boldsymbol{d\tau\}}, \boldsymbol{t} > \boldsymbol{t_f^m}, \end{aligned} \tag{6.28}$$

where $\int_{\boldsymbol{t_f^m}}^{\boldsymbol{t}} \text{drainage-capacity}_{\boldsymbol{\tau}}^{\boldsymbol{m}} \boldsymbol{d\tau} \approx \text{drainage-capacity}^{\boldsymbol{m}} \times (\boldsymbol{t} - \boldsymbol{t_f^m})$.

6.3.4.3 resilience planning

The considered machines are of three types, truck, freight train and ship which are noted as type I, II and III respectively. The related inoperable duration after $\boldsymbol{t_0^m}$ for the trucks, freight train and ship in the port can be estimated in terms of the predicted $\text{water-accumulation}_{\boldsymbol{t}}^{\boldsymbol{m}}$. The related inoperable duration for machine $\text{PCS}_{\boldsymbol{i}}$

can be shown as

$$t^{a}_{i,m,s} = \begin{cases} t^{\mathrm{I}}_{m,s}, & v_i \in V_P^{m-\mathrm{I}} \\ t^{\mathrm{II}}_{m,s}, & v_i \in V_P^{m-\mathrm{II}} \\ t^{\mathrm{III}}_{m,s} = 0, & v_i \in V_P^{m-\mathrm{III}}, \end{cases} \tag{6.29}$$

where $V_P^{m-\mathrm{I}}$, $V_P^{m-\mathrm{II}}$ and $V_P^{m-\mathrm{III}}$ represent the vertex set of trucks, freight train and ship of machine PCSs respectively. Due to the operation of ship is not affected by the water accumulation, the value of $t^{\mathrm{III}}_{m,s} = 0$. In addition, $V_P^{\mathrm{machine}} = V_P^{m-\mathrm{I}} \cup V_P^{m-\mathrm{II}} \cup V_P^{m-\mathrm{III}}$. Besides, each type of machine requires a certain number of operators, which are noted as n_{I}, n_{II} and n_{III}.

If the human operator $\mathrm{PCS}_i, v_i \in V_P^{\mathrm{human}}$ is assigned to the machine $\mathrm{PCS}_j, v_j \in V_P^{\mathrm{machine}}$, then

$$t^{a}_{i,m,s} = t^{a}_{j,m,s} \in \{t^{\mathrm{I}}_{m,s}, t^{\mathrm{II}}_{m,s}, 0\} \tag{6.30}$$

which implies the inoperable duration of human operator after t_0 is the same to the assigned machines.

According to the predicted water-accumulation$^{m}_{t}$, the inoperable duration of type I, II are related to their wading depth. Both vehicles and trains are designed to operate in a variety of weather conditions, including heavy rain and flooding. Therefore, the maximum wading depth are introduced below:

- **Type I:** Maximum wading depth of freight trucks are various by the size of the trucks. Normally, the values is from 450 mm to 1,200 mm. It is very dangerous for a vehicle to wade into deep water, which can often lead to serious damage to the engine. It can also lead to water intrusion into the gearbox and damage to the electronics. Therefore, we select half of the minimum value as the maximum water wading depth of freight trucks, which is 225 mm.

- **Type II:** The depth of water that a train can safely drive through depends on various factors such as the design of the track, the type of train, and the specific conditions of the water, such as current and debris. As a general rule, trains can typically operate in water up to the level of the rails, but it's essential for train operators to assess the safety of the tracks and surrounding conditions before attempting to drive through flooded areas. Safety is always the top priority in such situations.

  In Europe, the most common rail sizes are 46E1 and 54E1. Both rails are 8.27 inches tall and 5.91 inches wide at the base. Considering the safety of train operation, we accept half the tall value as the maximum water wading depth of freight trains in our simulation, which is 4.135 inches, namely, around 105 mm.

Unfortunately, there is no related extreme climate data (flooding) matching both the historical climate data and the drainage system. As consequence, we apply the generated data as the rainfall data and the constant value as the capacity of the

drainage system. The designed resilience planning time horizon $\boldsymbol{T^{total}}$ is 14 days which contain 7 raining days.

Based on the simulated climate data and the drainage system, the inoperable duration of Type I and II can be calculated by the above simulated data. The simulated rainfall and corresponding capacity of drainage system can be seen in Fig.6.12. In the above simulated data, the sampling period $\boldsymbol{T}$ equals to 1 hour. Therefore, based on equation 6.28, the inoperable duration can be calculated. The inoperable duration for type I and II are within time range $\mathbf{[93, 130]}$ and $\mathbf{[92, 132]}$ in Fig.6.12, namely, 38 and 41 hours respectively.

The conditions of the resilience planning optimization is introduced in detail in Table 6.3.

It is important to note that without disaster prediction and resilient planning, the system's performance can only reach **48%** of normal operation under the impact of the flooding disaster. In such a scenario, it would take **2** days to reorganize operations once the water level meets the criteria for resuming production.

Furthermore, with disaster prediction but without resilient planning, the system's performance can achieve **57%** of normal operation. This indicates that while the manager reorganizes operations once the water level meets the criteria for resuming production, no proactive measures are taken to mitigate the degradation in system performance.

Based on the optimization what we proposed in Section 6.3.3, the $\boldsymbol{T^{p}_{i,m,s}}$ is predicted related to different $\boldsymbol{Per^{lb}}$ and $\boldsymbol{\mathcal{CA}^{max}_{m}}$. The results of the optimization is given in Table 6.4. Due to the fact that the system's performance can achieve **57%** of normal operation with disaster prediction but without resilient planning. In the optimization starts considering the minimum target performance from **60%**.

6.3.5 Validation of resilience planning of digital twin via Flexsim

In this section, a digital twin of the real smart port is generated via Flexsim. FlexSim is a powerful, user-friendly simulation software used for modeling, analyzing, and optimizing LSSs in various industries, including manufacturing, logistics, healthcare, and supply chain management. It is a discrete-event simulation tool that allows users to create 3D models of real-world processes, visualize how these processes function over time, and test various scenarios to improve efficiency and decision-making. It's widely used both in academia and industry due to its ability to model complex systems realistically and its focus on enhancing operational performance. It supports academic research in fields like operations research, systems engineering, and industrial management by providing a versatile platform for experimenting with real-world scenarios.

In this chapter, based on disaster prediction, resilient planning optimization is conducted to allocate the operating hours of PCSs at operating sites, ensuring that performance remains above the specified threshold while minimizing costs. Digital twin is applied to validate the strategy (see in Fig. 6.13), which involves validating the robustness, adaptability, and efficiency of the optimization.

Table 6.3: Parameters used in the resilience planning optimization

$\boldsymbol{PCS_i}$ $\boldsymbol{(v_i)}$		Machine			Human operator		
		Type-I	Type-II	Type-III	Type-I	Type-II	Type-III
Maximum available PCSs		50	10	5	152	69	152
PCSs operate in normal case		20	5	2	20	10	20
$\boldsymbol{\mathcal{CA}_m^{max}}$	$\boldsymbol{m=1}$	$\boldsymbol{\mathcal{CA}_{1,TypeI}^{max}}$	0	0	-	-	-
	$\boldsymbol{m=2}$	0	$\boldsymbol{\mathcal{CA}_{2,TypeII}^{max}}$	0	-	-	-
	$\boldsymbol{m=3}$	0	0	$\boldsymbol{\mathcal{CA}_{3,TypeIII}^{max}}$	-	-	-
Mean cost (€/hour)		100	800	1280	20	23	25
Mean workload (/hour)		100	1000	2000	1	1.2	1.4

Table 6.4: Results of resilience planning optimization

PCS_i (v_i)			Machine			Human operator		
			Type-I	Type-II	Type-III	Type-I	Type-II	Type-III
Conditions I	$\mathcal{CA}_m^{max}$	$m = 1$	40	0	0	-	-	-
		$m = 2$	0	8	0	-	-	-
		$m = 3$	0	0	3	-	-	-
Results for different Per^{lb}	Per^{lb}	60%	0	1643	894	0	3286	8940
		70%	0	2216	894	0	4433	8940
		80%	4265	2360	894	4265	4720	8940
		90%	9955	2360	894	9955	4720	8940
		100%	**No solution**					
Conditions II	$\mathcal{CA}_m^{max}$	$m = 1$	40	0	0	-	-	-
		$m = 2$	0	5	0	-	-	-
		$m = 3$	0	0	4	-	-	-
Results for different Per^{lb}	Per^{lb}	60%	0	1044	1192	0	2089	11920
		70%	1417	1475	1192	1417	2950	11920
		80%	7106	1475	1192	7106	2950	11920
		90%	**No solution**					
		100%						

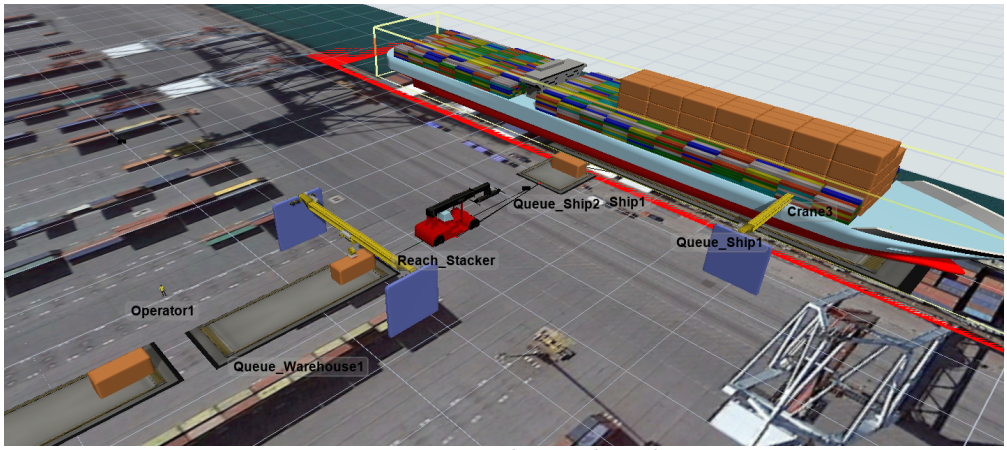


Figure 6.13: Digital twin by Flexsim.

The validation process consists of four sets of experiments: one control group and three experimental groups, each designed to evaluate the effects of resilient planning on system performance. The control group, representing the normal operation scenario, serves as a baseline, showcasing the system's standard performance without any disruptions or resilience strategies applied. This allows for a clear benchmark against which the experimental outcomes can be compared.

The first experimental group simulates operations facing disruptive disasters without resilient planning, providing insight into the system's vulnerability and response to disruptions in the absence of strategic interventions. This scenario is crucial for understanding the baseline level of risk and performance degradation when resilience measures are not in place.

The second and third experimental groups incorporate resilient planning strategies under different constraints (the available backup systems), reflecting varying operational conditions and limitations. These groups are optimized to test the effectiveness of resilience measures, with each constraint representing different levels of resource availability, response time, or other operational factors that influence system behavior. By comparing these scenarios, the experiments aim to identify which resilience strategies are most effective in maintaining system performance and minimizing costs under specific conditions.

In short, these experimental setups enable a comprehensive evaluation of the impact of resilient planning on overall system performance and resilience. The findings help determine the optimal balance between resource allocation, operational costs, and performance stability, offering valuable insights into the development of robust, adaptive strategies for managing disruptions in complex systems.

Before the comparative test, the definition of the four types of experimental groups are given below:

- **1. Control group 0:** Simulate the normal operation scenario, which means all the CSs operate without any faults, external disturbance/noises and network service issue (see in Section 5.3).

- **2. Experimental group I:** Simulate the operation facing disruptive disasters without disaster prediction and optimization.
- **3. Experimental group II:** Simulate the operation facing disruptive disasters with disaster prediction and optimization under the condition I (see in Table 6.4).
- **4. Experimental group III:** Simulate the operation facing disruptive disasters with disaster prediction and optimization under the condition II (see in Table 6.4).

Assumptions and conditions of the digital twins are given as below:

- The capacities of vehicles, trains, and ships are **1**, **50**, and **90**, respectively (container(s)).
- Each loading or unloading operation takes approximately **90** seconds per container.
- All containers handled are of standard dimensions of **12** meters (**40** feet).

In the following four subsections, the results of validation is given in detail.

6.3.5.1 Control group 0

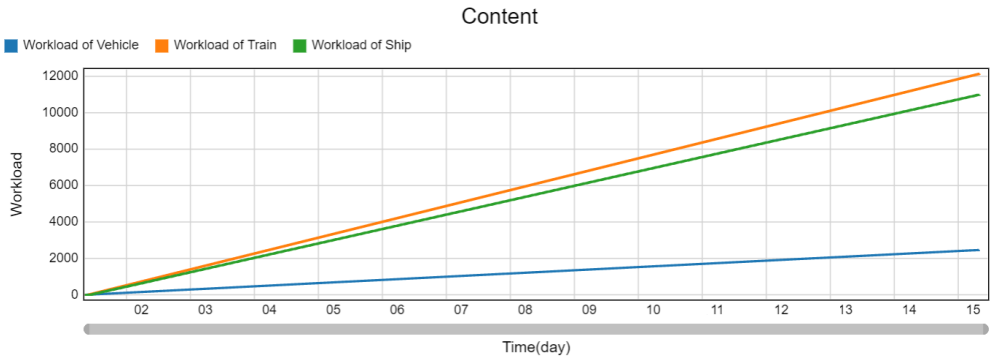


Figure 6.14: Workloads of vehicle, train and ship in Control group **0**

Firstly, in the normal scenario, the workloads of vehicles, trains, and ships, as obtained via the Flexsim, will be illustrated for Control Group 0. In this baseline scenario, the smart port operates under optimal conditions without any disasters, disruptions or disturbances. Specifically, there are **20** vehicles, **5** trains, and **2** ships involved in the port's operations.

In this context, the normal scenario assumes that there are no faults, external disturbances, noise, or network service issues that could potentially disrupt the operations. As a result, no disruptions are observed in the operating hours of the vehicles, trains, or ships due to external factors such as flooding. The port's logistics function as intended, with all transport modes maintaining their operational efficiency.

The evolution of the workloads corresponding to vehicles, trains, and ships under this normal scenario is illustrated in Fig 6.14. From this figure, it can be clearly observed that the absence of external influences allows the workloads of the vehicles, trains, and ships to increase steadily over time. Each mode of transportation exhibits a consistent and gradual rise in its workload, reflected by the uniform slope in the graph.

This steady increase in workload is attributed to the uninterrupted operation of the vehicles, trains, and ships, as none of them are impacted by external disruptions. Consequently, all **20** vehicles, **5** trains, and **2** ships are able to contribute their stable capacity throughout the entire time period, showcasing the performance of the smart port under ideal conditions.

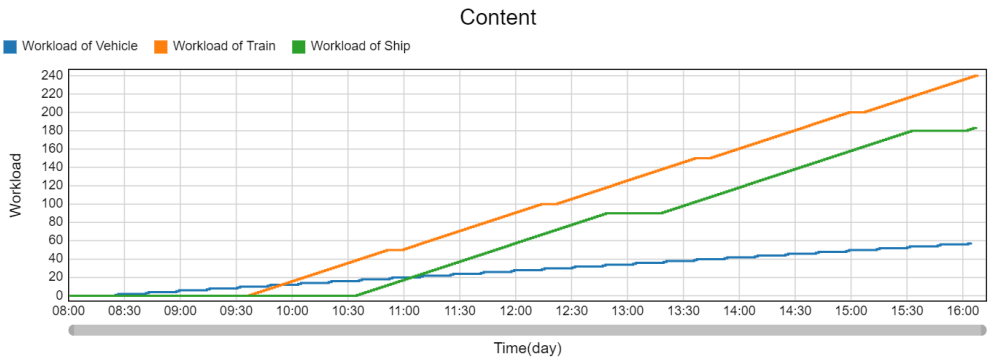


Figure 6.15: Workloads of vehicle, train and ship in Control group **0** in one day

Notice that, in Fig 6.14, The workload of ships increases in a stepwise manner over time. To better explain this phenomenon, we have plotted the changes in workload throughout the day in Fig 6.15. It can be observed that the workloads of vehicles, trains and ships exhibit stepwise growth. This occurs because the workload remains constant during the transport process of vehicles, trains, and ships, and only increases during unloading. Since the transport time for vehicles is the shortest, followed by trains, and ships have the longest transport time, this corresponds to the shortest platform period for vehicles, the next longest for trains, and the longest for ships in Fig 6.14. Because the load capacity of ships is the largest, followed by trains, and vehicles have the smallest capacity, the growth period for ships is the longest, followed by trains, and the shortest for vehicles.

6.3.5.2 Experimental group I

To show the effectiveness of the proposed disaster prediction and resilient planning optimization in mitigating performance degradation in smart port, the workloads of vehicle, train and ship facing disruptive disasters without disaster prediction and resilient planning optimization have been given in Fig 6.16. Due to the absence of disaster predictions, the manager of the smart port cannot determine the expected duration negative impact of disaster. As a result, the manager can only notify and arrange for employees to return to work after the flood has receded. In order to

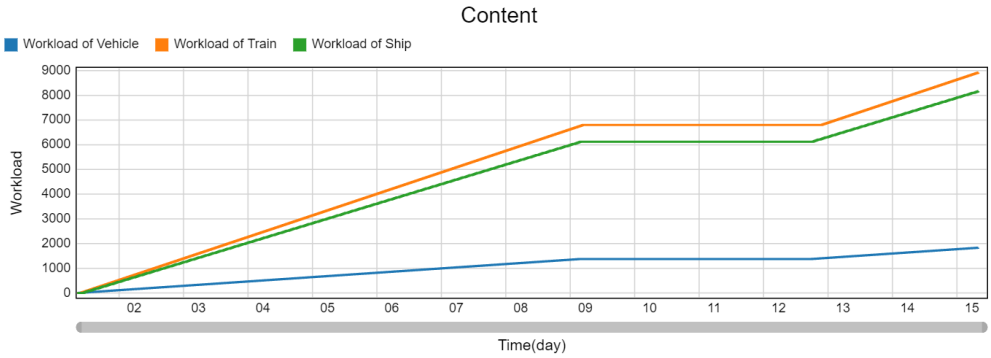


Figure 6.16: Workloads of vehicle, train and ship in Experimental group I

organize the resumption of operations, the restart time is set for two days after the water level drops below the safety line. Additionally, the manager is also unable to determine the number of backups required. Thus, the numbers of vehicles, trains and ships operating in the smart port are the same to that of normal scenario. During the flood, vehicles and trains cannot operate in the smart port. Additionally, to ensure the safety of operators, we also halted ship operations. During the period when the flood occurs and recedes, there is a prolonged plateau in the workloads of vehicles, trains and ships. This delay is caused by the lack of flood prediction and the absence of corresponding advance planning. Due to different requirements for water levels to resume operation, the timing of workload increases for vehicles, trains, and ships varies. From Fig 6.16, we see that since trains require a lower water level to resume operation, their recovery time is the latest.

6.3.5.3 Experimental group II

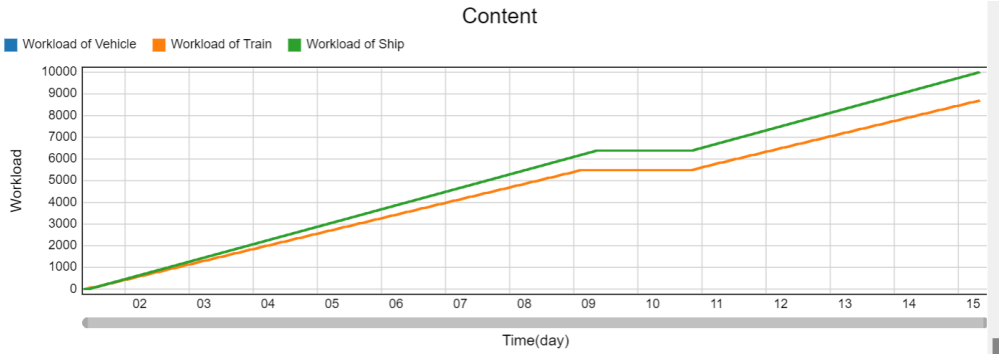


Figure 6.17: Workloads of vehicle, train and ship in Experimental group II with $\boldsymbol{Per^{lb} = 60\%}$

The operation facing flood with the proposed disaster prediction and resilient planning optimization under condition I with different lower bound of operational performance ratios $\boldsymbol{Per^{lb}}$ is presented in this part. In Experimental group II, extra backup vehicles, trains and ships are provided, where the total numbers of available vehicles, trains and ships are **40**, **8** and **3** respectively.

In the Case 1, the lower bound of operational performance ratio is set as $\boldsymbol{Per^{lb} = 60\%}$. According to the prediction-based resilient planning optimization, the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.17. Since the unit workload cost is lowest for ships, followed by trains, and highest for vehicles, it is seen from Fig 6.17 that only ships and trains are involved in the operation based on optimization in Case 1. It's seen that there are plateaus in the workload of train and ships respectively. Due to the flood, trains will stop service when the water level is too high. At the same time, to ensure the safety of operators, ships will also cease operations. However, due to the disaster prediction strategy and resilient planning, the trains and ships are able to return to work immediately once the flood has ended. Consequently, the plateau periods of trains and ships in Fig 6.17 are significantly shorter compared to that of Fig 6.16 of the Experimental Group I. Due to the disaster prediction-based resilient planning, the workload in Fig 6.17 is larger than that of Fig 6.16, which reveals the effectiveness of the disaster prediction and resilient planning optimization.

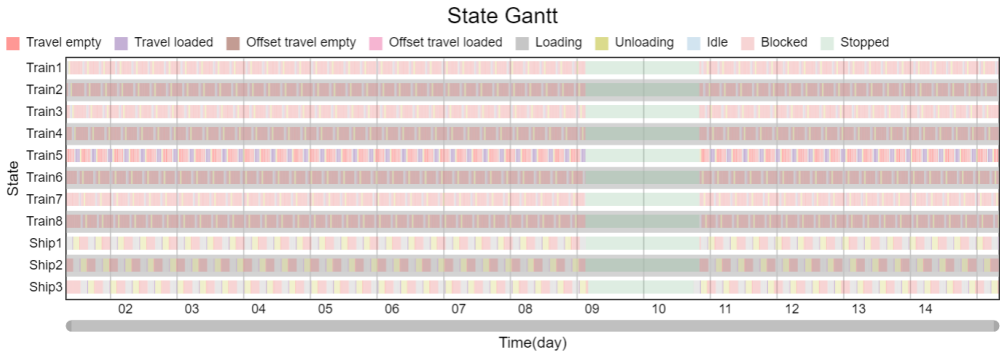


Figure 6.18: State Gantt of trains and ships in Experimental group II with $\boldsymbol{Per^{lb} = 60\%}$

The State Gantt of trains and ships has been shown in Fig 6.18. Note that, in the resilient planning optimization, we can only obtain the sums of operating hours for trains and ships respectively. The scheduling of **8** trains and **3** ships actually have various solutions. To better show the tendency of workloads of operating hours, we average the total operating hours of trains and ships across each train and ship, as well as over each day. From Fig.6.18, we can see that all trains and ships stop service during flood. But However, trains cease operation for a longer period since trains require a lower water level.

In the Case 2, the lower bound of operational performance ratio is set to be $\boldsymbol{Per^{lb} = 70\%}$. According to the prediction-based resilient planning optimization, the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.19. Due to the increase of the lower bound of operational performance ratio, the slope of the workload of trains versus time in Fig 6.19 is higher than that in Fig 6.17, which indicates trains contribute more capacities compared with that of Case 1. The plateaus in the workload of train and vehicle have also been witnessed in Fig 6.19 during flood. As a result of disaster prediction, the plateau periods of

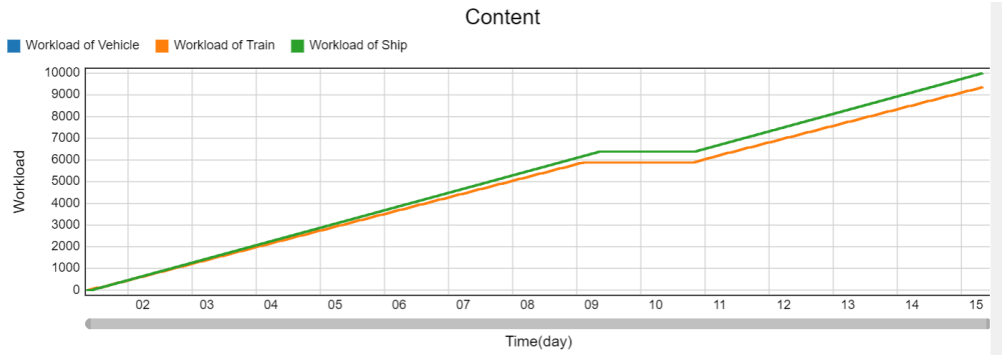


Figure 6.19: Workloads of vehicle, train and ship in Experimental group II with $\boldsymbol{Per^{lb} = 70\%}$

trains and ships in Figure 6.19 are also significantly shorter compared to the that of Figure 6.16 of the Experimental Group I.

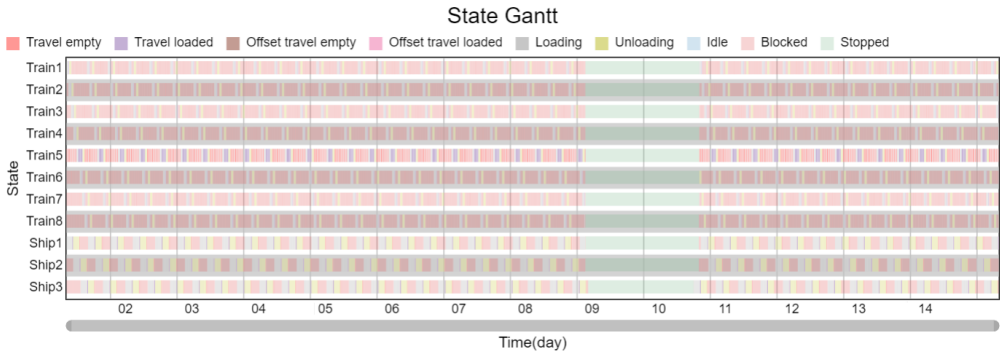


Figure 6.20: State Gantt of trains and ships in Experimental group II with $\boldsymbol{Per^{lb} = 70\%}$

The corresponding State Gantt chart for trains and ships is displayed in Fig. 6.20, illustrating the average operating hours for both modes of transport. As shown in Fig. 6.20, all trains and ships stop service during the flood. However, trains experience a longer downtime as they require lower water levels to safely resume service.

In the Case 3, the lower bound of operational performance ratio is set to be $\boldsymbol{Per^{lb} = 80\%}$. According to the prediction-based resilient planning optimization, the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.21. Due to the increase of the lower bound of operational performance ratio, the slope of the workload of trains versus time in Fig 6.21 is higher than that in Fig 6.19, which indicates trains contribute more capacities compared with that of Case 1. Besides, not only all ships and trains contribute their full capacities based on optimization, but also vehicles are involved in Case 3. It's seen that there also exists plateau in the workload of vehicle, as vehicle cannot operate in the smart port due to the rising water level. However, due to the disaster prediction strategy, the vehicles, trains and ships are able to return to work immediately once the flood

Figure 6.21: Workloads of vehicle, train and ship in Experimental group II with $\boldsymbol{Per^{lb} = 80\%}$

has ended. Consequently, the plateau periods in Figure 6.21 are also significantly shorter compared to the plateau periods in Figure 6.16 of the Experimental Group I.

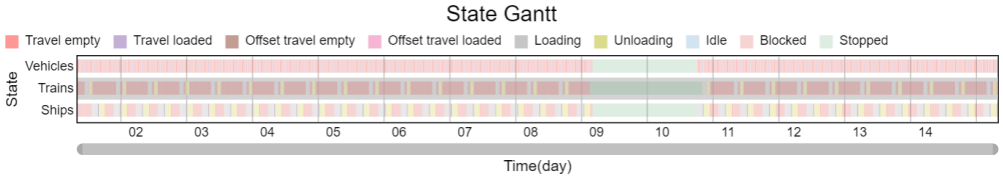


Figure 6.22: State Gantt of trains and ships in Experimental group II with $\boldsymbol{Per^{lb} = 80\%}$

The related State Gantt of vehicles, trains and ships has been shown in Fig 6.22. It presents the average operating hours for vehicles, trains and ships respectively. Figure 6.22 shows that all vehicles, trains and ships stop service during the flood. However, trains stop for a longer period because they require a lower water level to resume service.

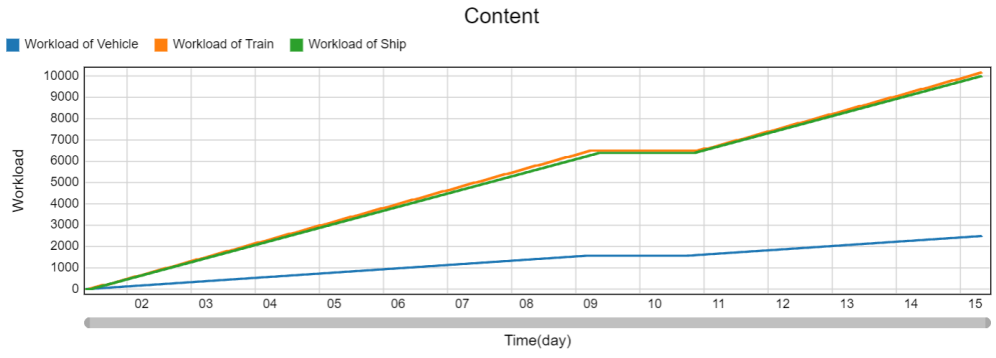


Figure 6.23: Workloads of vehicle, train and ship in Experimental group II with $\boldsymbol{Per^{lb} = 90\%}$

In the Case 4, the lower bound of operational performance ratio is set to be $\boldsymbol{Per^{lb} = 90\%}$. According to the prediction-based resilient planning optimization,

the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.23. Due to the increase of the lower bound of operational performance ratio, the slope of the workload of vehicles versus time in Fig 6.23 is higher than that in Fig 6.21, which indicates vehicles contribute more capacities compared with that of Case 3. Due to the disaster prediction strategy, the vehicles, trains and ships are able to return to work immediately once the flood has ended. Consequently, the plateau periods in Figure 6.23 are also significantly shorter compared to the plateau periods in Figure 6.16 of the Experimental Group I.

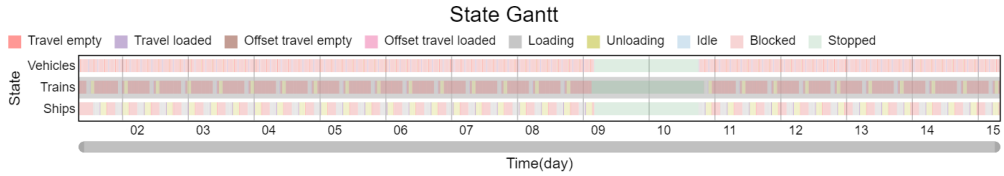


Figure 6.24: State Gantt of trains and ships in Experimental group II with $\boldsymbol{Per^{lb} = 90\%}$

The related State Gantt of vehicles, trains and ships has been shown in Fig 6.24. It present the average operating hours for vehicles, trains and ships respectively. Figure 6.24 shows that all vehicles, trains and ships stop service during the flood. Besides, the downtime of trains is longer than vehicles and ships during flood because they require a lower water level to resume service.

6.3.5.4 Experimental group III

The operation facing flood with the proposed disaster prediction and resilient planning optimization under condition II with different lower bound of operational performance ratios $\boldsymbol{Per^{lb}}$ is presented in this part. In Experimental group III, the total numbers of available vehicles, trains and ships are **40**, **5** and **4** respectively.

In the Case 1, the lower bound of operational performance ratio is set as $\boldsymbol{Per^{lb} = 60\%}$. According to the prediction-based resilient planning optimizations with $\boldsymbol{Per^{lb} = 60\%}$, the corresponding evolution of workloads corresponding to ships are given in Fig 6.25. Given that ships have the lowest unit workload cost,

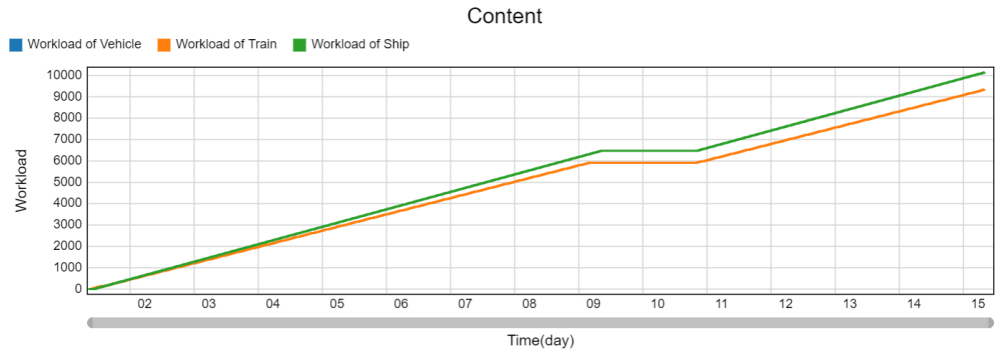


Figure 6.25: Workloads of ship in Experimental group III with $\boldsymbol{Per^{lb} = 60\%}$

followed by trains, with vehicles having the highest, Fig 6.25 reflect that only ships and trains are involved in the operation in Cases 1 and 2. It's seen that due to the presence of flood, there are plateaus in the workload of ships and trains during flood. Due to the disaster prediction strategy, the trains and ships are able to return to work immediately once the flood has ended. Consequently, the plateau periods in Figure 6.25 are also significantly shorter compared to the plateau periods in Figure 6.16 of the Experimental Group I. In addition, due to the disaster prediction-based resilient planning, the workload in Fig 6.25 is larger than that of Fig 6.16, which reveals the effectiveness of the disaster prediction and resilient planning optimization.

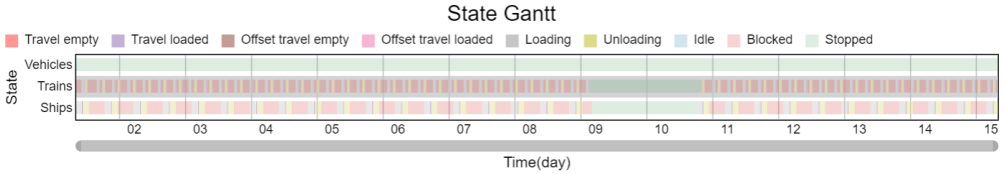


Figure 6.26: State Gantt of trains and ships in Experimental group I with $\boldsymbol{Per^{lb} = 60\%}$

In Cases 1, the related State Gantt of vehicles, trains and ships haven been drawn in Fig 6.26. It present the average operating hours for vehicles, trains, and ships respectively. It's seen from Fig 6.26 that only trains and ships operate in the Case 1. Besides, trains stop for a longer period than ships during flood because they require a lower water level to resume service.

In the Case 2, the lower bound of operational performance ratio is set as $\boldsymbol{Per^{lb} = 70\%}$. According to the prediction-based resilient planning optimizations with $\boldsymbol{Per^{lb} = 70\%}$, the corresponding evolution of workloads corresponding to ships are given in Fig 6.27. As a result of the increase of the lower bound of operational performance ratio, the slope of the workload of trains versus time in Fig 6.27 is higher than that in Fig 6.25, which indicates trains contribute more capacities compared with that of Case 1. In addition, it is seen from Fig 6.27 that not only ships and trains are involved in the operation in the Case 2, but also vehicles contribute their capacities, which is different from Case 2 in Experimental group II. This is because there are only **5** trains in the group III, much less than that of group

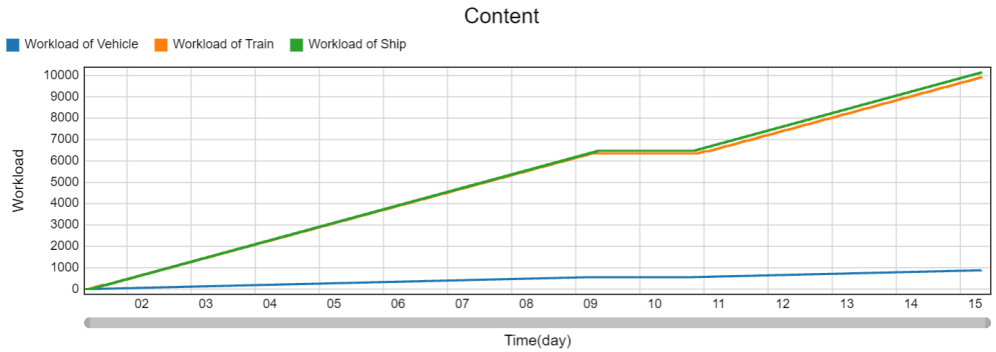


Figure 6.27: Workloads of ship in Experimental group III with $\boldsymbol{Per^{lb} = 70\%}$

II. It's seen that due to the presence of flood, there is also plateau in the workload of vehicles during flood. Due to the disaster prediction strategy, the vehicles, trains and ships are able to return to work immediately once the flood has ended. Consequently, the plateau periods in Figure 6.27 are also significantly shorter compared to the plateau periods in Figure 6.16 of the Experimental Group I.

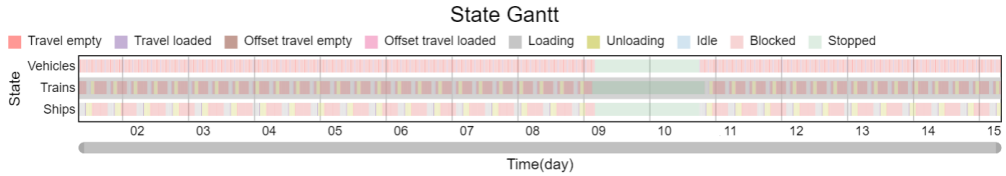


Figure 6.28: State Gantt of trains and ships in Experimental group I with $\boldsymbol{Per^{lb} = 70\%}$

In Case 2, the related State Gantts of vehicles, trains and ships haven been drawn in Fig 6.26 and Fig 6.28. It present the average operating hours for vehicles, trains, and ships respectively. It's seen from Fig 6.20 that vehicles, trains and ships all operate in the Cases. Besides, the operating hours of trains in Fig 6.20 are less than vehicles and ships as a result of the lower water level to resume service.

In Case 3, the lower bound of operational performance ratio is set to be $\boldsymbol{Per^{lb} = 80\%}$. According to the prediction-based resilient planning optimization, the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.29. Due to the increase of the lower bound of operational performance ratio, the slope of the workload of vehicles versus time in Fig 6.29 is higher than that in Fig **??**, which indicates vehicles contribute more capacities compared with that of Case 2. Due to the disaster prediction strategy, the vehicles, trains and ships are able to return to work immediately once the flood has ended. Consequently, the plateau periods in Figure 6.29 are also significantly shorter compared to the plateau period in Figure 6.16 of the Experimental Group I.

The related State Gantt of vehicles, trains and ships of Case 3 has been shown in Fig 6.30. It present the average operating hours of vehicles, trains, and ships respectively. It's seen from Fig 6.20 that vehicles, trains and ships all operate in the Cases, while the downtime of trains in Fig 6.20 is longer than vehicles and ships as

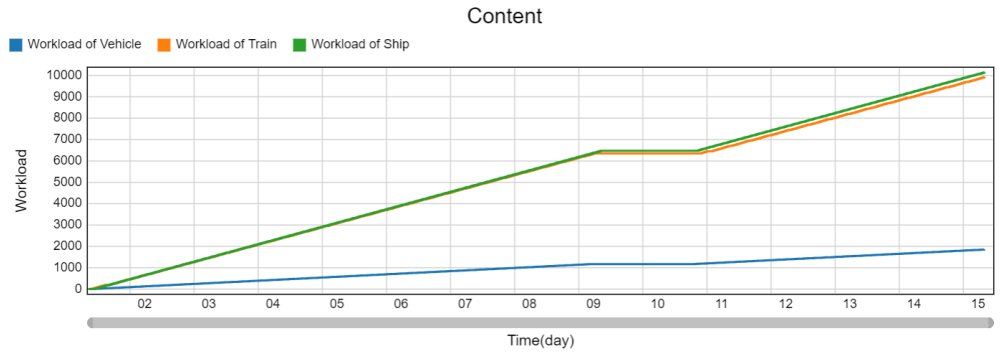


Figure 6.29: Workloads of ship in Experimental group III with $\boldsymbol{Per^{lb} = 80\%}$

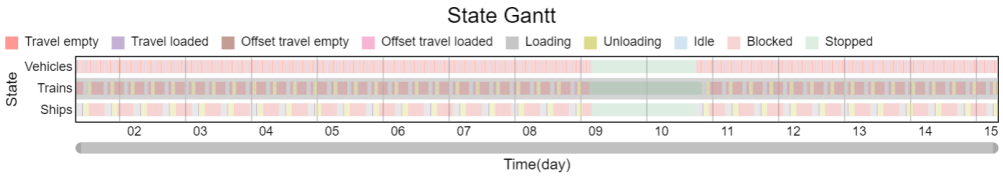


Figure 6.30: State Gantt of trains and ships in Experimental group III with $\boldsymbol{Per^{lb} = 80\%}$

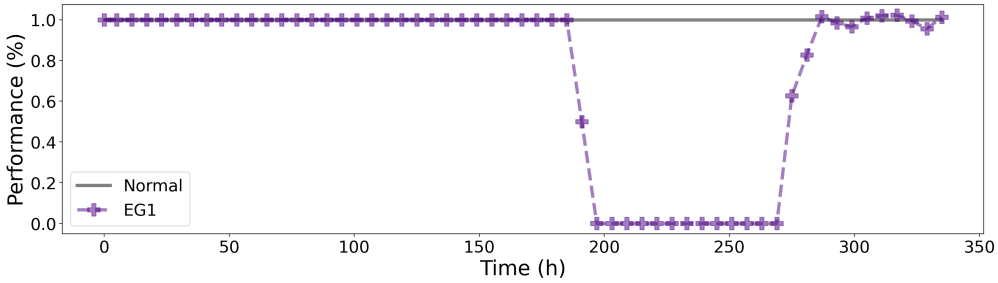


Figure 6.31: Operating performance between control group 0 and experimental group I during the whole operating time horizon.

a result of the lower water level to resume service.

6.3.6 Comparison and analysis

As a comparison, the overall operating performance of the whole SoS with time in Control group 0 (normal status without disaster) and Experiment group I (without disaster prediction and resilient planning) has been shown in Fig.6.31. To unify the performance metrics normal status, the performance of SoS in other groups is expressed as a ratio relative to the SoS performance in the control group 0.

From Fig 6.31, one can see that the SoS performance decrease rapidly in the presence of flood. In Experiment Group I, due to the lack of disaster predictions, the manager of the smart port is unable to estimate the duration of the negative impacts caused by the flood. Consequently, the manager can only notify and arrange for operators to return to work after the flood has completely receded. During the flood period, the SoS performance remains at a low level, reflecting the complete halt in operations of vehicles, trains, and ships in the smart port. This is shown as a prolonged plateau in performance in Fig 6.31. This extended low-performance phase is caused by the lack of flood prediction. After the flood recedes, the SoS performance gradually returns the to normal level. Due to the absence of resilient planning, the manager is also unable to determine the number of backups required. Thus, the numbers of vehicles, trains and ships operating in the smart port are the same to that of normal scenario, which hinders the quick restoration of performance.

In Experimental Group II, in addition to disaster prediction and resilient planning, extra backup vehicles, trains, and ships are provided to enhance operational resilience. The total numbers of available vehicles, trains, and ships in this scenario are **40**, **8**, and **3**, respectively. The overall operating performance of the entire SoS

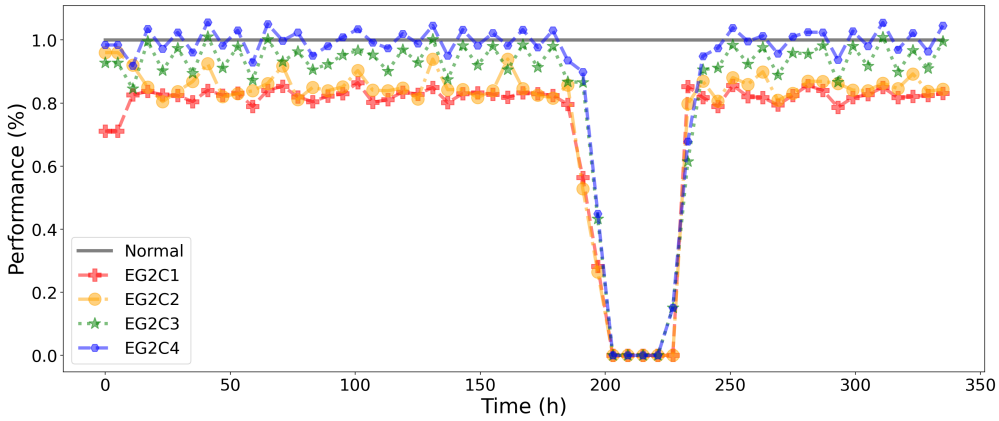


Figure 6.32: Operating performance between control group 0 and experimental group II (concluding 4 cases) during the whole operating time horizon.

over time for four different cases, each with varying lower bounds of performance ratios $\boldsymbol{Per^{lb} = 60\%, 70\%, 80\%, 90\%}$, is depicted in Figure 6.33.

As shown in Figure 6.33, in all four cases, the SoS performance drops rapidly in response to the flood. This is due to vehicles, trains and ships all cease operations during the flood. However, due to the implementation of disaster prediction and resilient planning, the manager can estimate the flood's duration and preemptively allocate the necessary operators. This foresight allows vehicles, trains, and ships to resume operations immediately once the flood has receded. As a result, the low-performance phases in all four cases of Experimental Group II are significantly shorter ompared to Experimental Group I, as seen in Figure 6.31.

In the performance recovery stage, the gradient of the performance increase in Figure 6.33 is steeper compared with that of Figure 6.31. Extra backup vehicles, trains and ships ensure a quicker performance restoration compared with that of Experimental Group I. Furthermore, the performance curves in the figure clearly align with the respective lower bounds of performance ratios set in the optimization process, ranging from **60%** in Case 1 to **90%** in Case 4. This consistency highlights the effectiveness of the resilient planning in maintaining operational performance despite the presence of the flood.

In Experimental Group III, extra backup vehicles, trains, and ships are also compensated, in which the total numbers of available vehicles, trains, and ships in this scenario are **40**, **5**, and **4**, respectively. The overall operating performance of the entire SoS over time for three different cases, each with varying lower bounds of performance ratios $\boldsymbol{Per^{lb} = 60\%, 70\%, 80\%}$, is depicted in Figure 6.33.

One can obtain from Figure 6.31 that the SoS performance drops sharply facing with flood, since all vehicles, trains and ships stop service during flood. based on the disaster prediction and resilient planning, the manager is able to estimate the duration of flood and allocate the operators in advance. Thus, vehicles, trains and ships are able to back to work immediately when the floodwaters have receded. To this end, the low-performance phases in all three cases of Experimental Group III

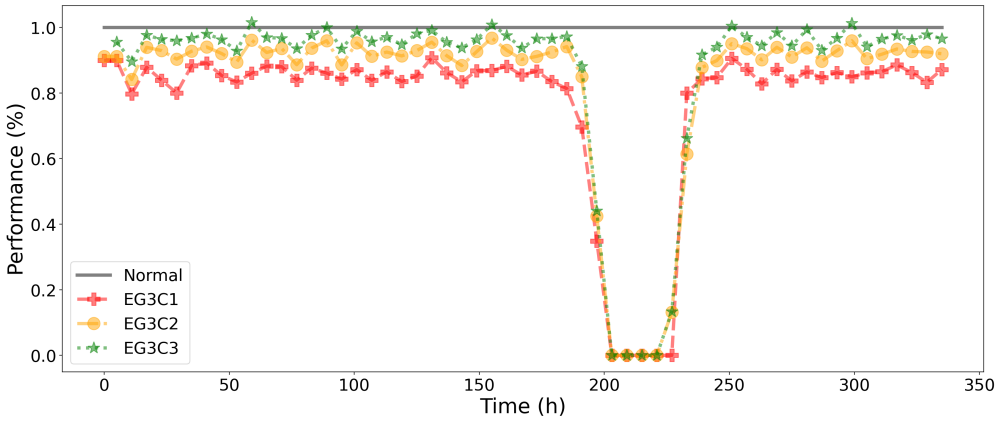


Figure 6.33: Operating performance between control group 0 and experimental group III (concluding 3 cases) during the whole operating time horizon.

are also shorter compared to that of Experimental Group I in Figure 6.31.

Apart from this, the gradient of the performance recover in Figure 6.33 is steeper compared with that of Figure 6.31 . The performance curves of the **3** cases in the figure adhere to the varying lower bounds of performance ratios set in the optimization, ranging from **60%** to **80%** for Case 1 to Case 3. This demonstrates the effectiveness of the resilient planning.

6.3.7 An extended disaster scenario

In the previous **5** sections (Section 6.3.2 to Section 6.3.6), a comprehensive process is given including disaster prediction, optimization for resilience planning, qualitative and quantitative tests as well as the corresponding validation using digital twin.

The key to using disaster prediction data is to match the relationships between the predicted values with the performance of various port equipment. These relationships are collected and summarized through real-world observations, experiments, or operational experiences, and are referred to as empirical data.

In this section, an extended disaster scenario is given to show how to extend the scope of applications of our proposed method.

6.3.7.1 Wind scenario

The wind disaster scenario is generated as what we did in Section 6.3.4.3. The designed resilience planning time horizon $\boldsymbol{T^{total}}$ is also 14 days which contain 7 windy days. Before the data generation, the discussion about the wind speed level and its corresponding performance degradation is introduced below.

Different models and structures of cranes will experience varying degrees of performance reduction due to wind speed. By comparing the technical data of Mobile Crane[3] and Rubber Tyre Gantry Crane[4], wind speed level is categorized into these

[3] https://heavyequipmentcollege.edu/wind-speed-guide-for-mobile-crane-operation/
[4] Technical Description Rubber Tyre Gantry Crane

five groups:

- **Very Calm or Still:** wind speeds $\boldsymbol{v < 2m/s}$ (4.47mph).
- **Calm:** wind speeds between $\boldsymbol{2 \leq v < 5m/s}$ (4.47 to 11.16mph).
- **Low:** wind speeds between $\boldsymbol{5 \leq v < 10m/s}$ (11.17mph to 22.36mph).
- **Medium / Caution:** wind speeds between $\boldsymbol{10 \leq v < 15m/s}$ (22.36mph to 33.55mph).
- **High / Risk:** wind speeds between $\boldsymbol{10 \leq v < 20m/s}$ (33.55 to 44.74mph).
- **Over High:** wind speeds $\boldsymbol{v \geq 20m/s}$ (over 44.74mph).

The impact of wind speed on crane performance is not only dependent on the wind speed itself, but also on the crane's condition and the operator. As consequence, the following assumptions are given based on the five wind speed levels:

- **Under Medium**, for wind speed level 'Very calm', 'Calm' and 'Low', the performance of Crane equals **100%**.
- **Medium**, for wind speed level 'Medium', the performance of Crane equals **80%**.
- **High**, for wind speed level 'high', the performance of Crane equals **60%**.
- **Over High**, for wind speed level higher than 'high', shut down due to safety concerns, namely, the performance of Crane equals **0%**.
- To ensure safety and efficient operations, speed limits in ports are more strictly enforced. Therefore, in this scenario, we keep the same performance degradation as Cranes. Which ensures operational safety while also reducing computational complexity.

So the generated wind disaster is labelled as shown in Fig.6.34. The detailed range of each type is: Uner Medium: $\boldsymbol{[0, 42] \cup (126, 168]}$, Medium $\boldsymbol{: (42, 52] \cup (116, 126]}$, High $\boldsymbol{: (52, 61] \cup (107, 116]}$ and Over High $\boldsymbol{: (61, 107]}$.

For the resilience planning, three cases are given to show the results of digital twin simulation. Which follows the Experimental Group I, Experimental Group II with $\boldsymbol{Per^{lb} = 90\%}$ and Experimental group III with $\boldsymbol{Per^{lb} = 60\%}$.

6.3.7.2 Experimental group I

The workloads of vehicle, train and ship facing disruptive disasters without wind prediction and resilient planning optimization have been given in Fig 6.35. Due to the absence of wind predictions, the manager of the smart port cannot determine the expected duration and negative impact of disaster. As a result, the manager can only notify and arrange for employees to return to work after the wind speed falls

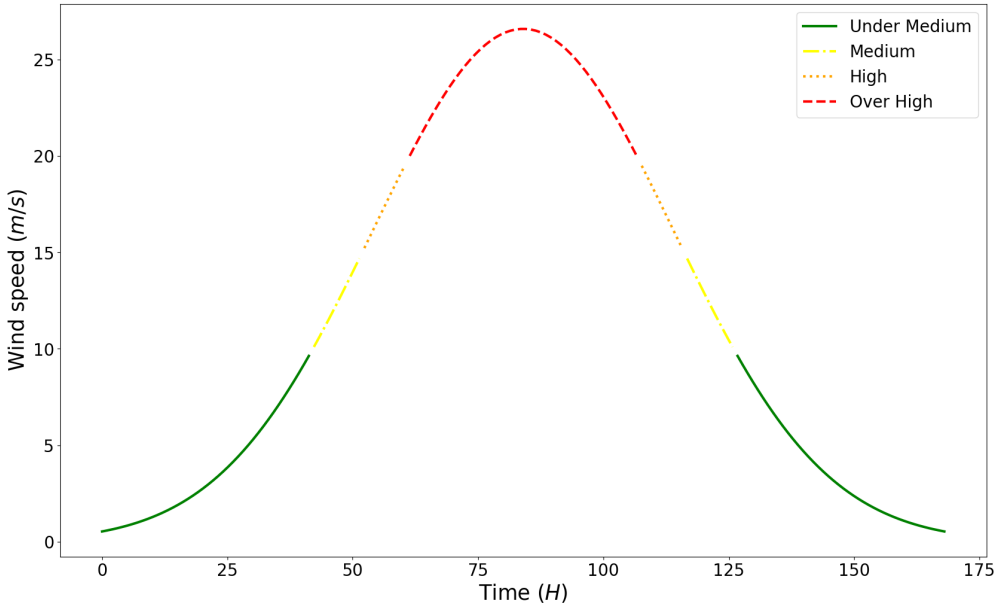


Figure 6.34: Generated wind disaster scenario.

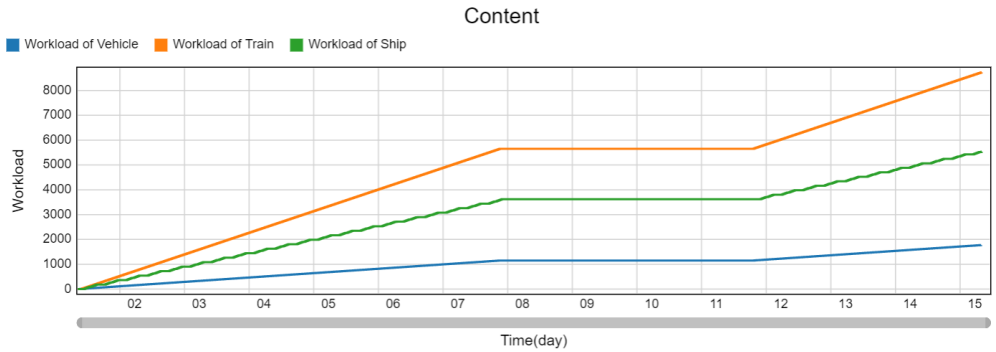


Figure 6.35: Workloads of vehicle, train and ship in Experimental group I under wind disaster.

below the safety value. In order to organize the resumption of operations, the restart time is set for two days after the wind speed drops below $\mathbf{20}\boldsymbol{m/s}$. Additionally, the manager is also unable to determine the number of backups required. Thus, the numbers of vehicles, trains and ships operating in the smart port are the same to that of normal scenario in Control group 0, which is the same to that of the above flood scenario. During the period of very high wind speeds, the performance of crane equals to **0%**, which implies vehicles, trains and ships in the smart port cannot load or unload in the smart port. During the period when the wind speed is very high, there is a prolonged plateau in the workloads of vehicles, trains and ships. This prolonged delay is caused by the lack of wind prediction and the absence of corresponding advance planning. Since the vehicles vehicles, trains, and ships begin operating when the cranes are back to work, the timing of workload increases for vehicles, trains, and ships are almost the same in Fig 6.35.

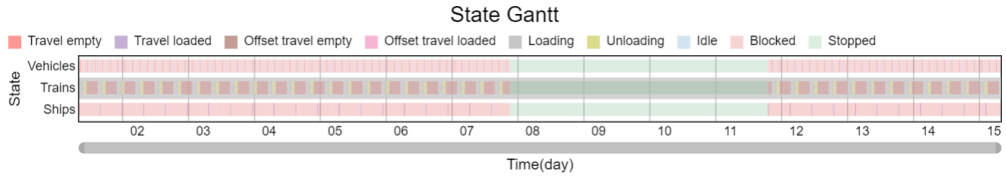


Figure 6.36: State Gantt of trains and ships in Experimental group I under wind disaster.

The State Gantt of trains and ships has been shown in Fig 6.36. It presents the average operating hours for vehicles, trains and ships respectively. Fig 6.36 reveals the prolonged downtime due to lack of wind prediction and resilient planning. Besides, the downtime period for vehicles, trains, and ships are almost the same in Fig 6.36. This is because the vehicles vehicles, trains, and ships can operate only when the cranes are back to work.

6.3.7.3 Experimental group II

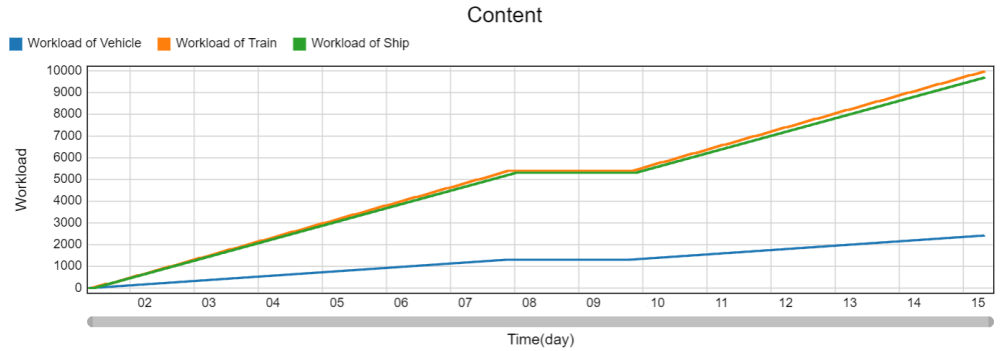


Figure 6.37: Workloads of vehicle, train and ship in Experimental group II with $\boldsymbol{Per^{lb}} = \mathbf{90\%}$ under wind disaster.

The operation facing wind under condition I with lower bound of operational

performance ratios $\boldsymbol{Per^{lb} = 90\%}$ is presented in this part. In Experimental group II, extra backup vehicles, trains and ships are provided, where the total numbers of available vehicles, trains and ships are **40**, **8** and **3** respectively.

According to the disaster prediction and resilient planning, the evolution of workloads corresponding to vehicles, trains and ships are given in Fig 6.37. It's seen that there are plateaus in the workload of vehicles, train and ships respectively. This is because the performance of crane equals to **0%** during the period of very high wind speeds, which results in that vehicles, trains and ships in the smart port cannot load or unload in the smart port. In addition, due to the disaster prediction, the cranes, vehicles, trains and ships are able to return to work immediately once the speed of wind is less than $\boldsymbol{20m/s}$. Thus, the workload plateaus of vehicles, trains and ships in Fig 6.37 are much shorter then that of Experimental Group I in Fig 6.35. The ratio of overall workload in Fig 6.37 is over **90%**, larger than that of Experimental Group I in Fig 6.35, which reveal the effectiveness of the disaster prediction and resilient planning,

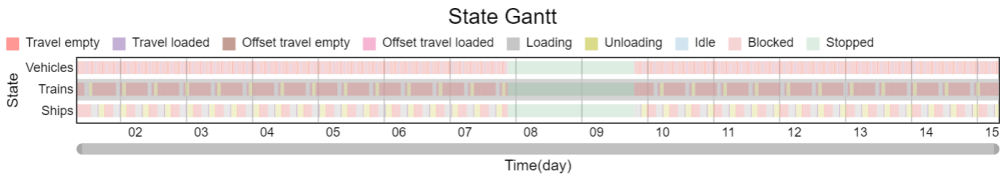


Figure 6.38: State Gantt of trains and ships in Experimental group II with $\boldsymbol{Per^{lb} = 90\%}$ under wind disaster.

The State Gantt of vehicles, trains and ships of Experimental group II has been shown in Fig 6.38. It presents the average operating hours for vehicles, trains and ships respectively. From Fig 6.38, we can obtain that the downtime of vehicles, trains and ships in Experimental group II is much shorter than that of Experimental group I in Fig 6.37. This is because, thanks to the wind prediction, the cranes, vehicles, trains and ships are able to return to work immediately once the speed of wind is less than $\boldsymbol{20m/s}$.

6.3.7.4 Experimental group III

The operation facing wind with the proposed disaster prediction and resilient planning optimization under condition II with lower bound of operational performance ratio $\boldsymbol{Per^{lb} = 60\%}$ is presented in this part. In Experimental group III, the total numbers of available vehicles, trains and ships are **40**, **5** and **4** respectively.

According to the wind prediction and resilient planning with $\boldsymbol{Per^{lb} = 60\%}$, the corresponding evolution of workloads corresponding to trains and ships are given in Fig 6.39. Given that ships have the lowest unit workload cost, followed by trains, with vehicles having the highest, Fig 6.39 reflect that only ships and trains are involved in the operation. It's seen that due to the presence of very high wind speed, there are plateaus in the workload of ships and trains when the cranes stop services in the period of wind with very high speed. Due to the wind prediction

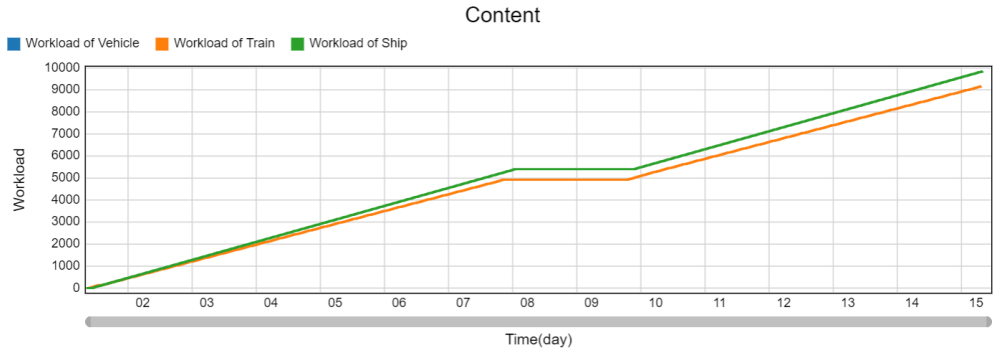


Figure 6.39: Workloads of vehicle, train and ship in Experimental group III with $\boldsymbol{Per^{lb} = 60\%}$ under wind disaster.

strategy, the cranes, trains and ships are able to return to work immediately once the the wind speed is lower than $\boldsymbol{20m/s}$. Consequently, the plateau periods in Figure 6.39 are also significantly shorter compared to the plateau periods in Figure 6.35 of the Experimental Group I. In addition, due to the disaster prediction-based resilient planning, the workload in Fig 6.39 is larger than that of Fig 6.35, which reveals the effectiveness of the disaster prediction and resilient planning.

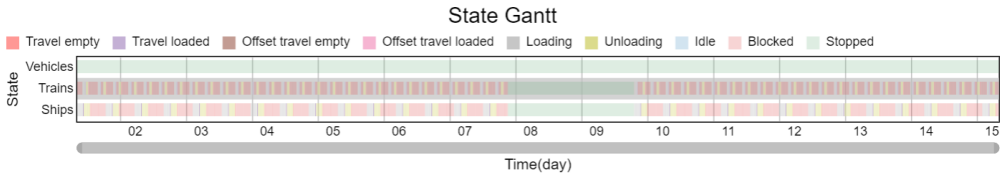


Figure 6.40: State Gantt of trains and ships in Experimental group III with $\boldsymbol{Per^{lb} = 60\%}$ under wind disaster.

The related State Gantt of vehicles, trains and ships of Experimental group III haven been drawn in Fig 6.26. It present the average operating hours for vehicles, trains, and ships respectively. It's seen from Fig 6.26 that only trains and ships operate in Experimental group III . Besides, trains stop for a longer period than ships during flood because they require a lower water level to resume service. Besides, the downtime of trains and ships in Experimental group III is also much shorter than that of Experimental group I in Fig 6.37 due to the wind prediction.

6.3.8 Comparison and analysis

As a comparison, the overall operating performance of the whole SoS with time in Control group 0 (normal status without wind) and Experiment group I (without disaster prediction and resilient planning) has been shown in Fig.6.41. To unify the performance metrics normal status, the performance of SoS in other groups is expressed as a ratio relative to the SoS performance in the control group 0.

From Fig 6.41, one can see that the SoS performance decrease rapidly during the wind with very high speed since all cranes stop service. In Experiment Group

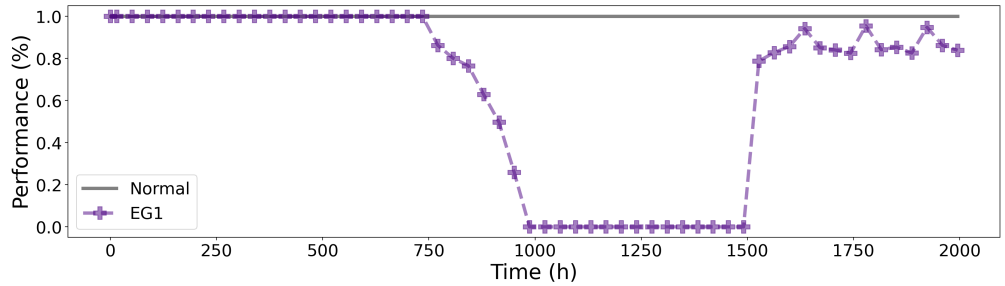


Figure 6.41: Operating performance between control group 0 and experimental group I under wind disaster during the whole operating time horizon.

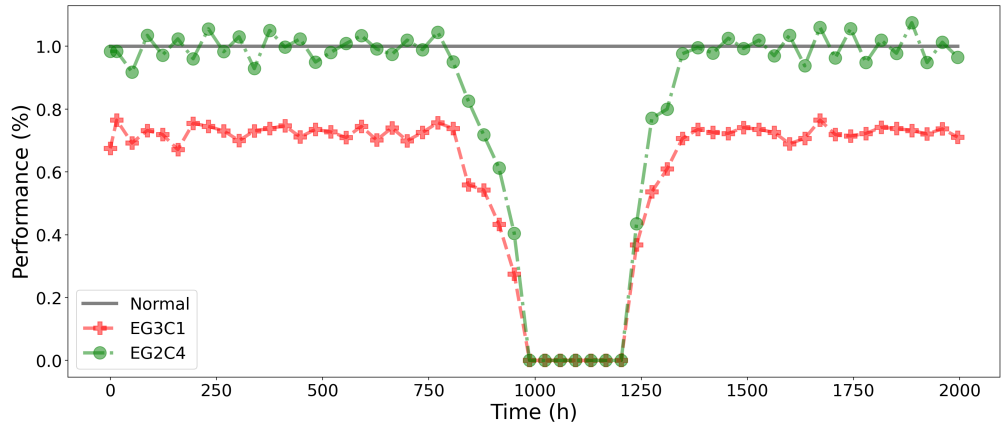


Figure 6.42: Operating performance between control group 0, experimental group II and experimental group III under wind disaster during the whole operating time horizon.

I, due to the lack of wind prediction, the manager of the smart port is unable to estimate the duration of the negative impacts caused by the wind. Consequently, the manager can only notify and arrange for operators to return to work after the wind speed is less than $\mathbf{20}\boldsymbol{m/s}$ after two days. During the wind period, the SoS performance remains at a low level, reflecting the complete halt in operations of cranes, vehicles, trains, and ships in the smart port. This is shown as a prolonged plateau in performance in Fig 6.41. This extended low-performance phase is caused by the lack of wind prediction. After two days of the wind speed is less than $\mathbf{20}\boldsymbol{m/s}$, the SoS performance gradually returns the to normal level. Due to the absence of resilient planning, the manager is also unable to determine the number of backups required. Thus, the numbers of vehicles, trains and ships operating in the smart port are the same to that of normal scenario, which hinders the quick restoration of performance.

In Experimental Groups II and III, in addition to disaster prediction and resilient planning, extra backup vehicles, trains, and ships are provided to enhance operational resilience. The overall operating performance of the entire SoS over time of the two groups, has been depicted in Figure 6.42.

As shown in Figure 6.42, in both two groups, the SoS performance drops rapidly

in response to the wind. This is due to vehicles, trains and ships all fail to load and unload when the cranes cease operation in very high speed wind. However, due to the implementation of wind prediction and resilient planning, the manager can estimate the wind duration and preemptively allocate the necessary operators. This foresight allows vehicles, trains, and ships to resume operations immediately once the wind speed is lower than $\mathbf{20}\boldsymbol{m/s}$. As a result, the low-performance phases in all both Experimental Groups II and III are significantly shorter compared to Experimental Group I, as seen in Fig 6.42 and Fig 6.41.

In the performance recovery stage, the gradient of the performance increase in Figure 6.42 is steeper compared with that of Figure 6.41. Extra backup vehicles, trains and ships ensure a quicker performance restoration compared with that of Experimental Group I. Furthermore, the performance curves in the figure clearly align with the respective lower bounds of performance ratios in the resilient planning in two groups, $=\mathbf{60\%}$ in Experimental Group III and $\mathbf{90\%}$ in in Experimental Group II. This consistency highlights the effectiveness of the resilient planning in maintaining operational performance despite the presence of the wind.

6.3.9 Conclusion

Facing disruptive disasters, the resilience planning of SoS in advance has been investigated in this work. The complex SoS facing various disasters is depicted by a scenario-dependent hypergraph model, in which the multi-way interaction between PCSs and operating sites are described by the attributes of edge-dependent weight function. The disaster prediction is performed via a hybrid estimation model with data-driven AI and multi-scale time series analysis. Take flash flood caused by excessive rainfall as an instance of disasters, the rainfall prediction is divided into two steps. The first step utilizes the data-driven AI classification model to predict the rainy days and the second step estimate the amount of rainfall for those predicted rainy days based on the multi-scale time series analysis. Indonesia daily climate data from 2010 to 2022 including 173 sampling locations among 34 provinces have been utilized to test the prediction model, which presents high estimation accuracy. On the basis of the disaster prediction, the resilient planning optimization has been addressed to allocate the operating hours PCSs in operating sites to ensure the performance no lower than the given bound and meanwhile guarantee the minimal cost. Finally, the case study focuses on the digital twin of smart port via Flexsim, in which the operating hours of human operators, trucks, freight trains and ships at various operating sites are calculated based on the proposed planning optimization strategy.

6.4 SoS Resilient Recovery Time Estimation

This section focuses on optimization-based resilient recovery time estimation for SoS with diverse resources and capacities. Regarding the crew dispatch in a large-scale SoS, it's natural to see distinct crews possess different capacities and performance

in the repair of alternative PCSs. Facing with inevitable malfunctions of PCSs, the proposed optimization strategy incorporates not only the crew dispatch and PCS restoration, but also the heterogeneous capacity distribution that reflects the cooperation inside the SoS. The designed multi-objective optimization function minimizes the crew travel time, malfunction lost and potential risk. The estimation of recovery time is calculated based on the solution to the optimization.

6.4.1 Recovery time estimation of SoS

The SoS considered in this section consists of various CSs, which can be divided into diverse PCSs and crews. While a part of PCSs are facing with intolerable malfunctions/damages, it is a important responsibility for the manager to dispatch the crews for repairing to restore the SoS. The assignment and routing problem of crews are solved by a multi-objective optimization to minimize the travel time, potential risk as well as the lost during restoration. The estimation of the recovery time is calculated based on the solution of optimization problem.

6.4.1.1 Problem formulation

Assumptions and conditions of our Recovery Time Estimation (RTE) problem are given as below:

- The failures behavior of PCSs is mutually independent.
- Each PCS operates with Normal status in the beginning of each navigation.
- Preventive Maintenance (PM) is scheduled in non-operating period which means we tend not to consider the PM in our RTE.
- After the Corrective Maintenance (CM), PCSs will back to Normal status.
- All crews and backup PCSs are in the same depot $\boldsymbol{d}$, they start at the depot at the beginning of the operational horizon, and back to depot $\boldsymbol{d}$ after finished the restoration.
- The travel time of crews and backup PCSs is only related to the travel distance which is presented by weighted graph.
- Technicians are considered heterogeneous which means each technician can deal with specific failures and for the same failure event the recovery time varies with technicians.

6.4.1.2 Routing of repair crews with capacity distribution

The routing of repair crews can be divided into the following aspects. The first aspect aims at repair all PCSs with malfunctions in diverse scenarios with the distribution

of heterogeneous capacities of crews, which can be formulated as follows:

$$\sum_{c \in C} g_{i,c,s} * h_{i,c,s} = b_{i,s}, \forall i \in V_P, \forall s \in \mathcal{S}. \tag{6.31}$$

Observe that equation (6.31) reveals that, if PCS_i is subject to the malfunction of scenario s, there must exist a suitable c capable of repairing it in scenario s (i.e., $\mathcal{H}_{i,c,s} = 1$) and it's also assigned to PCS_i for the sake of restoration (i.e., $g_{i,c,s} = 1$).

The second aspect focus on the routing of crews, which can be formulated as the network flow problem with the help of graph theory. Let $G = (V_L, E)$ denote a directed graph with node set V_L and edge set E, where $V_L = V_p \cup \{d\}$ represents the locations including all PCSs and the depot. The edge set $E = \{(m, m'), m/m' \in V_L\}$ contains edge (m, m') that connects two locations.

$$\begin{aligned} (\sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s})(\sum_{m \in V_L} x_{d,m,c}) &= \sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s}, \\ 0 \leq \sum_{m \in V_L} x_{d,m,c} &\leq \sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s}, \forall c \in \mathcal{C} \end{aligned} \tag{6.32}$$

$$\begin{aligned} (\sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s})(\sum_{m \in V_L} x_{m,d,c}) &= \sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s}, \\ 0 \leq \sum_{m \in V_L} x_{m,d,c} &\leq \sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s}, \forall c \in \mathcal{C} \end{aligned} \tag{6.33}$$

$$\sum_{m \in V_L \setminus \{i\}} x_{i,m,c} - \sum_{m \in V_L \setminus \{i\}} x_{m,i,c} = 1, \forall c \in \mathcal{C}, i \in V_P \tag{6.34}$$

Notice that $\sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s} > 0$ implies crew c is dispatched for repairing. In this case, equation (6.32) ensures those dispatched crews start from the depot, i.e.,

$$\sum_{m \in V_L} x_{m,d,c} = 1, \text{ if } g_{i,c,s} > 0.$$

Otherwise, $\sum_{i \in V_P} \sum_{s \in \mathcal{S}} g_{i,c,s} = 0$ implies crew c is not dispatched and stays in the depot, that is

$$\sum_{m \in V_L} x_{m,d,c} = 1, \text{ if } g_{i,c,s} = 0.$$

Similarly, equation (6.33) guarantees all dispatched crews will finally go back to the depot. For instance, $x_{d,1,1} = 1$ implies the crew 1 travels from the depot to PCS_1. $x_{3,d,2}$ reveals the crew 2 goes back to the depot from PCS_3. Additionally, equation (6.34) reveals the locations of PCSs are not the source for nodes in the network flows, which is known as the flow conservation constraints. Every crew travels through the locations of PCSs and then moves to the next location rather

than stays there.

Besides, the following constrain is proposed to ensure that if there exists malfunction in $\boldsymbol{PCS_i}$, then only one crew will be sent to $\boldsymbol{PCS_i}$ for repairing.

$$\begin{aligned} \boldsymbol{b_{i,s}} \sum_{\forall c \in \mathcal{C}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} &= \boldsymbol{b_{i,s}}, \\ \mathbf{0} \leq \sum_{\forall c \in \mathcal{C}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} &\leq \sum_{s \in \mathcal{S}} \boldsymbol{b_{i,s}}, i \in \boldsymbol{V_P} \end{aligned} \tag{6.35}$$

If $\boldsymbol{b_{i,s}} = \mathbf{1}$, i.e., $\boldsymbol{PCS_i}$ is subject to malfunction $\boldsymbol{s}$, then (6.35) implies

$$\sum_{\forall c \in \mathcal{C}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{1}, \text{ for } \boldsymbol{b_{i,s}} = \mathbf{1},$$

that is only one crew will be sent to $\boldsymbol{PCS_i}$ for repairing. Otherwise, $\boldsymbol{b_{i,s}} = \mathbf{0}$, i.e., $\boldsymbol{PCS_i}$ is not subject to malfunction $\boldsymbol{s}$, then one crew might be dispatched to $\boldsymbol{PCS_i}$ for other scenarios or/not. If for all $\boldsymbol{s} \in \mathcal{S}$, $\boldsymbol{b_{i,s}} = \mathbf{0}$, then $\sum_{s \in \mathcal{S}} \boldsymbol{b_{i,s}} = \mathbf{0}$ and no crew is allocated to $\boldsymbol{PCS_i}$ with

$$\sum_{\forall c \in \mathcal{C}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{0}, \forall \boldsymbol{s} \in \mathcal{S}, \boldsymbol{b_{i,s}} = \mathbf{0}.$$

Otherwise, there exists at least one $\boldsymbol{s'} \in \mathcal{S}$ such that $\boldsymbol{b_{i,s'}} = \mathbf{1}$, according to (6.35) with scenario $\boldsymbol{s'}$,

$$\sum_{\forall c \in \mathcal{C}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{1}, \exists \boldsymbol{s'} \in \mathcal{S} \setminus \{\boldsymbol{s}\}, \boldsymbol{b_{i,s'}} = \mathbf{1}.$$

For simplicity, we do not consider the situation that two crews are assigned to the same $\boldsymbol{PCS_i}$. Additionally, to link the routing problem with the capacity distribution, the following constraint is proposed.

$$\begin{aligned} \boldsymbol{g_{i,c,s}} \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} &= \boldsymbol{g_{i,c,s}}, \\ \mathbf{0} \leq \sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} &\leq \sum_{s \in \mathcal{S}} \boldsymbol{g_{i,c,s}}, \forall \boldsymbol{c} \in \mathcal{C}, \boldsymbol{i} \in \boldsymbol{V_P}, \boldsymbol{s} \in \mathcal{S}. \end{aligned} \tag{6.36}$$

Similar to (6.35), (6.36) reveals if crew $\boldsymbol{c}$ is assigned to $\boldsymbol{PCS_i}$ for repairing malfunction $\boldsymbol{s}$, i.e., $\boldsymbol{g_{i,c,s}}$, then it must travel through $\boldsymbol{PCS_i}$ once, i.e., $\sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{1}$, for $\boldsymbol{g_{i,c,s}} = \mathbf{1}$. Otherwise crew $\boldsymbol{c}$ might be dispatched to $\boldsymbol{PCS_i}$ for other scenarios or/not. If for all $\boldsymbol{s} \in \mathcal{S}$, $\boldsymbol{g_{i,c,s}} = \mathbf{0}$, then crew $\boldsymbol{c}$ is not allocated to $\boldsymbol{PCS_i}$ with $\sum_{s \in \mathcal{S}} \boldsymbol{g_{i,c,s}} = \mathbf{0}$ and $\sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{0}, \forall \boldsymbol{s} \in \mathcal{S}, \boldsymbol{g_{i,c,s}} = \mathbf{0}$. Otherwise, there exists at least one $\boldsymbol{s'} \in \mathcal{S}$ such that $\boldsymbol{g_{i,c,s'}} = \mathbf{1}$, according to (6.36) with scenario $\boldsymbol{s'}$, $\sum_{m \in V_L \setminus \{i\}} \boldsymbol{x_{m,i,c}} = \mathbf{1}, \exists \boldsymbol{s'} \in \mathcal{S} \setminus \{\boldsymbol{s}\}, \boldsymbol{g_{i,c,s'}} = \mathbf{1}$.

Remark 6.4.1 *Compared with the classical routing problem in the frame of flow*

network [Arif 2018, Chen 2018], the proposed constrains further consider the heterogeneous capacities distribution of crews in (6.31) and (6.36), which reflects the cooperation inside the SoS structure. Besides, the difference in repair time $\boldsymbol{T^R_{i,c,s}}$ *represents the heterogeneous performance of diverse crews in the repair of distinct PCSs and scenarios. Equation (6.31) ensures the malfunctions of each PCS to be repaired by the crews with related capacities. Equation (6.36) further relates the capacities distribution with the conventional network flow. Furthermore, (6.35) admits there exist PCSs without degradation, which implies there is no need to visit those PCSs.*

6.4.1.3 PCS restoration with crew capacity distribution

The PCS restoration can be related to the crew routing problem in virtue the arrival time of crews at PCSs. Firstly, if a crew arrives at a PCS starting from the depot then the arrival time only depends on the travel time, that is,

$$\boldsymbol{a_{i,c} = d_{d,c} + T_{d,i},} \text{ if } \boldsymbol{x_{d,i,c} = 1, \forall i \in V_p, c \in \mathcal{C}.} \tag{6.37}$$

Otherwise, the calculation of arrival time depends on the repairing time of the preceding PCS and the travel time between the adjacent PCSs. If crew $\boldsymbol{c}$ travels from $\boldsymbol{PCS_i}$ to location $\boldsymbol{m}$ (including depot $\boldsymbol{d}$ and $\boldsymbol{PCS_{i'}}$, $\boldsymbol{i \neq i'}$), then

$$\begin{aligned} \boldsymbol{a_{i,c} + \sum_{s\in\mathcal{S}} g_{i,c,s} T^R_{i,c,s} + T_{i,m} = a_{m,c},} \\ \text{if } \boldsymbol{x_{i,m,c} = 1, \forall c \in \mathcal{C}, i \in V_P, m \in V_L \setminus \{i\},} \end{aligned} \tag{6.38}$$

where the term $\sum_{s\in\mathcal{S}} \boldsymbol{g_{i,c,s} T^R_{i,c,s}}$ stands for the sum of time for crew $\boldsymbol{c}$ to repair all assigned scenarios. If crew $\boldsymbol{c}$ is not distributed to repair $\boldsymbol{PCS_i}$ in scenario $\boldsymbol{s}$, then $\boldsymbol{g_{i,c,s} T^R_{i,c,s} = 0}$. That is, the repairing time $\boldsymbol{T^R_{i,c,s}}$ will be added into arriving time $\boldsymbol{a_{m,c}}$ only if it's assigned to repair $\boldsymbol{PCS_i}$ in scenario $\boldsymbol{s}$.

To formulate the constrains on arrival time combing multiple occasions ($\boldsymbol{x_{d,i,c} = 0}$ or $\boldsymbol{1}$; $\boldsymbol{x_{i,m,c} = 0}$ or $\boldsymbol{1}$)

$$\begin{aligned} \boldsymbol{a_{m',c} + \sum_{s\in\mathcal{C}} g_{m',c,s} T^R_{m',c,s} + T_{m',m}} \\ \boldsymbol{-(1 - x_{m',m,c})\Delta \leq a_{m,c}} \\ \boldsymbol{a_{m',c} + \sum_{s\in\mathcal{C}} g_{m',c,s} T^R_{m',c,s} + T_{m',m}} \\ \boldsymbol{+(1 - x_{m',m,c})\Delta \geq a_{m,c}} \\ \boldsymbol{\forall s \in \mathcal{S}, c \in \mathcal{C}, m \in V_L \setminus \{m'\}} \end{aligned} \tag{6.39}$$

in which disjunctive constraints are used involving the implementation of the binary

$\boldsymbol{x_{m',m,c}}$. In addition, the arrival time satisfies

$$\mathbf{0} \leq \boldsymbol{a_{i,c}} \leq \boldsymbol{\Delta} \sum_{m \in V_L \backslash i} \boldsymbol{x_{i,m,c}}, \forall \boldsymbol{c} \in \boldsymbol{\mathcal{C}}, \boldsymbol{i} \in \boldsymbol{V_P}, \tag{6.40}$$

which reveals if crew $\boldsymbol{c}$ doesn't go through $\boldsymbol{PCS_i}$, i.e., $\boldsymbol{PCS_i}$ is not assigned to crew $\boldsymbol{c}$ for repairing, then the related arrival time $\boldsymbol{a_{i,c}} = \mathbf{0}$. The role of $\boldsymbol{\Delta}$ is similar to the $\boldsymbol{M}$ in Big-M reformulation. Notice that when $\boldsymbol{m'} = \boldsymbol{d}$ and $\boldsymbol{x_{d,i,c}} = \mathbf{1}$, inequalities (6.39) degrade to equation (6.37) due to $\boldsymbol{g_{d,c,s}} = \mathbf{0}$ for all $\boldsymbol{c} \in \boldsymbol{\mathcal{C}}$, $\boldsymbol{s} \in \boldsymbol{\mathcal{S}}$. Additionally, if $\boldsymbol{m'} = \boldsymbol{i'} \in \boldsymbol{V_P}$ and $\boldsymbol{x_{i',m,c}} = \mathbf{1}$, (6.39) degrade to equation (6.38) exactly.

To determine the restoration status of each PCS over the whole time horizon, the following constrain is proposed,

$$\sum_{t \in \mathcal{T}} \boldsymbol{f_{i,t}} = \mathbf{1} \tag{6.41}$$

If all scenarios of $\boldsymbol{PCS_i}$ is repaired at time $\boldsymbol{t} = \mathbf{2}$, then $\boldsymbol{f_{i,2}} = \mathbf{1}$ and $\boldsymbol{f_{i,t}} = \mathbf{0}, \boldsymbol{t} \neq \mathbf{2}$.

In addition, the following inequalities relate the arrival time $\boldsymbol{a_{i,c}}$ with the binary variable $\boldsymbol{f_{i,t}}$,

$$\begin{aligned}
\sum_{t \in \mathcal{T}} \boldsymbol{t f_{i,t}} \leq & \sum_{c \in \mathcal{C}} (\boldsymbol{a_{i,c}} + \sum_{s \in \mathcal{S}} \boldsymbol{g_{i,c,s}} \boldsymbol{T^R_{i,c,s}} \sum_{m \in V_L \backslash \{i\}} \boldsymbol{x_{i,m,c}}) \\
\sum_{t \in \mathcal{T}} \boldsymbol{t f_{i,t}} \leq & \sum_{c \in \mathcal{C}} (\boldsymbol{a_{i,c}} + \sum_{s \in \mathcal{S}} \boldsymbol{g_{i,c,s}} \boldsymbol{T^R_{i,c,s}} \sum_{m \in V_L \backslash \{i\}} \boldsymbol{x_{i,m,c}}) \\
& + \mathbf{1} - \boldsymbol{\tau}, \forall \boldsymbol{i} \in \boldsymbol{V_P}
\end{aligned} \tag{6.42}$$

Due to we consider the discrete time horizon $\boldsymbol{\mathcal{T}}$, the time $\boldsymbol{t}$ is considered as integer time step in this work. In addition, $\sum_{t \in \mathcal{T}} \boldsymbol{t f_{i,t}}$ represents the restoration instant of $\boldsymbol{PCS_i}$. For example, if $\boldsymbol{PCS_i}$ is repaired at time $\boldsymbol{t} = \mathbf{3}$, then only $\boldsymbol{f_{i,3}} = \mathbf{1}$ while $\boldsymbol{f_{i,t}} = \mathbf{0}, \boldsymbol{t} \neq \mathbf{3}$. This reveals $\sum_{t \in \mathcal{T}} \boldsymbol{t f_{i,t}} = \mathbf{3} * \boldsymbol{f_{i,3}} + \mathbf{0} * \sum_{t \neq 3} \boldsymbol{t} = \mathbf{3}$, which equals to the restoration instant. Besides, the restoration instant equals to the arrival time and repair time of the assigned crew. Due to only one crew is assigned to a specific PCS, there only exists one $\boldsymbol{c} \in \boldsymbol{\mathcal{C}}$ such that $\boldsymbol{a_{i,c}} \neq \mathbf{0}$. Moreover, due to only one crew will go through $\boldsymbol{PCS_i}$ once. There only exists one $\boldsymbol{c} \in \boldsymbol{\mathcal{C}}$ and one $\boldsymbol{m} \in \boldsymbol{V_L \backslash \{i\}}$ such that $\boldsymbol{x_{i,m,c}} = \mathbf{1}$. Thus, $\sum_{c \in \mathcal{C}} (\boldsymbol{a_{i,c}} + \sum_{s \text{ satisfying } g_{i,c,s}=1} \boldsymbol{T^R_{i,c,s}} \sum_{m \in V_L \backslash \{i\}} \boldsymbol{x_{i,m,c}})$ is the arrival time plus the repair time. Inequalities (6.42) actually perform the round up function on $\sum_{c \in \mathcal{C}} (\boldsymbol{a_{i,c}} + \sum_{s \text{ satisfying } g_{i,c,s}=1} \boldsymbol{T^R_{i,c,s}} \sum_{m \in V_L \backslash \{i\}} \boldsymbol{x_{i,m,c}})$ to ensure restoration time $\sum_{t \in \mathcal{T}} \boldsymbol{t f_{i,t}}$ to be an integer.

6.4.1.4 Optimization-based repair time estimation

The repair time estimation is performed on the basis of a multi-objective optimization with the above mentioned routing and restoration problem. Firstly, to minimize

the travel time of crews as well as the lost caused by PCS degradation in diverse scenarios, the following multi-objective optimization problem is proposed based on the above discussion.

Minimum travel time and lost optimization with variable vectors $\boldsymbol{\delta} = (\boldsymbol{x_{m,m',c}}, \boldsymbol{g_{i,t,s}})$, $\boldsymbol{\omega} = (\boldsymbol{a_{i,c}}, \boldsymbol{f_{i,t}})$:

$$\min_{\boldsymbol{\delta}} \sum_{\boldsymbol{c}\in\boldsymbol{\mathcal{C}}} \sum_{\boldsymbol{m}\in\boldsymbol{V_L}} \sum_{\boldsymbol{m'}\in\boldsymbol{V_L}\setminus\{\boldsymbol{m}\}} \boldsymbol{T^L_{m,m'}} \boldsymbol{x_{m,m'c}} \tag{6.43}$$

$$\min_{\boldsymbol{\delta},\boldsymbol{\omega}} \sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \sum_{\boldsymbol{s}\in\boldsymbol{\mathcal{S}}} \boldsymbol{b_{i,s}} \boldsymbol{l_{i,s}} \sum_{\boldsymbol{t}\in\boldsymbol{\mathcal{T}}} \boldsymbol{tf_{i,t}} + (\boldsymbol{a_{d,c}} - \boldsymbol{d_{d,c}}) \sum_{\boldsymbol{c}\in\boldsymbol{\mathcal{C}}} \boldsymbol{\omega_c} \sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \boldsymbol{x_{d,i,c}} \tag{6.44}$$

s.t. (6.31)-(6.36), (6.39)-(6.42)

$$\boldsymbol{x_{m,m',c}}, \boldsymbol{g_{i,t,s}}, \boldsymbol{f_{i,t}} \in \{\boldsymbol{0}, \boldsymbol{1}\} \tag{6.45}$$

$$\boldsymbol{a_{i,c}}, \boldsymbol{tf_{i,t}} \in \boldsymbol{\mathcal{T}}, \tag{6.46}$$

$$\sum_{\boldsymbol{c}\in\boldsymbol{\mathcal{C}}} \sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \boldsymbol{x_{d,i,c}} \leq \sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \sum_{\boldsymbol{s}\in\boldsymbol{\mathcal{S}}} \boldsymbol{b_{i,s}} \tag{6.47}$$

The first objective functions (6.43) minimizes the total travel time cost for all crews, in which $\sum_{\boldsymbol{m'}\in\boldsymbol{V_L}\setminus\{\boldsymbol{m}\}} \boldsymbol{T^L_{m,m'}} \boldsymbol{x_{m,m',c}}$ represents the sum of travel time of crew $\boldsymbol{c}$ among all PCSs and the depot $\boldsymbol{d}$. If crew $\boldsymbol{c}$ does not travel from location $\boldsymbol{m}$ to location $\boldsymbol{m'}$ then $\boldsymbol{T^L_{m,m'}} \boldsymbol{x_{m,m',c}} = \boldsymbol{0}$ as $\boldsymbol{x_{m,m',c}} = \boldsymbol{0}$. Function (6.43) aims to achieve a minimum travel cost in human resources by minimizing the travel time.

Meanwhile, the second optimization function (6.44) minimizes the sum of the lost caused by malfunctions of PCSs in multiple scenarios before restoration and the potential risk generated by the absence of crew $\boldsymbol{c}$ in the depot. Note that $\sum_{\boldsymbol{t}\in\boldsymbol{\mathcal{T}}} \boldsymbol{tf_{i,t}}$ represents the restoration duration of $\boldsymbol{PCS_i}$. On the basis of it, $\boldsymbol{b_{i,s}} \boldsymbol{l_{i,s}} \sum_{\boldsymbol{t}\in\boldsymbol{\mathcal{T}}} \boldsymbol{tf_{i,t}}$ stands for the lost of $\boldsymbol{PCS_i}$ in scenario $\boldsymbol{s}$ during restoration. If $\boldsymbol{PCS_i}$ is not subject to scenario $\boldsymbol{s}$, then $\boldsymbol{b_{i,s}} \boldsymbol{l_{i,s}} \sum_{\boldsymbol{t}\in\boldsymbol{\mathcal{T}}} \boldsymbol{tf_{i,t}} = \boldsymbol{0}$ owing to $\boldsymbol{b_{i,s}} = \boldsymbol{0}$. Besides, $(\boldsymbol{a_{d,c}} - \boldsymbol{d_{d,c}})$ represents the duration of the absence of crew $\boldsymbol{c}$ in the depot. If crew $\boldsymbol{c}$ is allocated out, there exists potential risk that malfunctions in the future may not be handled in time due to the absence of crew $\boldsymbol{c}$. As per $\sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \boldsymbol{x_{d,i,c}} = \boldsymbol{1}$ if crew $\boldsymbol{c}$ is allocated for repairing, $(\boldsymbol{a_{d,c}} - \boldsymbol{d_{d,c}}) \sum_{\boldsymbol{c}\in\boldsymbol{\mathcal{C}}} \boldsymbol{\omega_c} \sum_{\boldsymbol{i}\in\boldsymbol{V_P}} \boldsymbol{x_{d,i,c}}$ reveals the total potential risk lost related to the absence of all dispatched crews.

The optimization is subject to the constrains (6.31)-(6.36) concerned about the routing problem with crew capacities distribution, together with the PCS restoration constrains (6.37)-(6.42). Moreover, inequality (6.47) indicates the number of assigned crews is no greater than the number of PCSs with malfunctions, which avoids the waste of excessive human resource consumption.

With the help of linear scalarization, the multi-objective optimization problem

is transformed into the following single optimization problem:

$$\begin{aligned}\min_{\boldsymbol{\delta},\boldsymbol{\omega}}\ & \boldsymbol{\lambda_1}\sum_{c\in\mathcal{C}}\sum_{m\in V_L}\sum_{m'\in V_L\setminus\{m\}} \boldsymbol{T^L_{m,m'}x_{m,m'c}} \\ & + \boldsymbol{\lambda_2}[\sum_{i\in V_P}\sum_{s\in\mathcal{S}} \boldsymbol{b_{i,s}l_{i,s}}\sum_{t\in\mathcal{T}} \boldsymbol{tf_{i,t}} \\ & + (\boldsymbol{a_{d,c}-d_{d,c}})\sum_{c\in\mathcal{C}}\boldsymbol{\omega_c}\sum_{i\in V_P}\boldsymbol{x_{d,i,c}}] \qquad (6.48)\end{aligned}$$

s.t. (6.31)-(6.36), (6.39)-(6.42), **(6.45) − (6.47)**

Let $\boldsymbol{\delta^*} = (\boldsymbol{x^*_{m,m'c}}, \boldsymbol{g^*_{i,t,s}})$ and $\boldsymbol{\omega^*} = (\boldsymbol{a^*_{i,c}}, \boldsymbol{f^*_{i,c}})$ be the solution to optimization problem with objective functions (6.43)-(6.43) are subject to constrains (6.31)-(6.42). In terms of the solution to optimization problem, the estimation of $\boldsymbol{PCS_i}$ recovery time is given as

$$\boldsymbol{\hat{t}^R_i} = \sum_{t\in\mathcal{T}} \boldsymbol{f^*_{i,t}}. \qquad (6.49)$$

Moreover, the estimation of SoS recovery time is shown as

$$\boldsymbol{\hat{t}^R} = \boldsymbol{\max\{\sum_{t\in\mathcal{T}} f^*_{i,t}, i\in V_P\}}. \qquad (6.50)$$

Notice that $\boldsymbol{\hat{t}^R}$ also equals to the latest time for all crews to start to go back to the depot, that is,

$$\boldsymbol{\hat{t}^R} = \boldsymbol{\max\{a^*_{d,c} - \sum_{i\in V_P} x^*_{i,d,c}T^L_{i,d}, c\in\mathcal{C}\}}. \qquad (6.51)$$

Due to recovery of SoS requires the restoration of all composed PCSs, the recovery time of SoS depends on the latest recovery PCS.

6.4.2 Qualitative and quantitative tests

In the simulation, all PCSs operate under a given operating area, with the depot $\boldsymbol{d}$ at the center to better provide support and coverage to all areas. We consider an event $\mathscr{E}$ that 9 inoperable (broken) PCSs waiting for recovery. To simplify, the position of each PCS is generate randomly inside the operating area and the travel time of crews is only related to the distance. Therefore, we can get a digraph $\boldsymbol{G}$ with 10 nodes (1 depot and 9 inoperable PCSs). The weight of edge between two nodes means the travel time with two directions. Due to arbitrary two nodes are reachable for each other, the digraph $\boldsymbol{G}$ is a strongly connected graph. If the paths between two nodes are not unique, the weight of edge between two nodes is set be the minimum travel time. There are 8 available crews in depot $\boldsymbol{d}$ when the event $\mathscr{E}$ happen.

In the simulation, 8 types of faults with different repair times (testing time

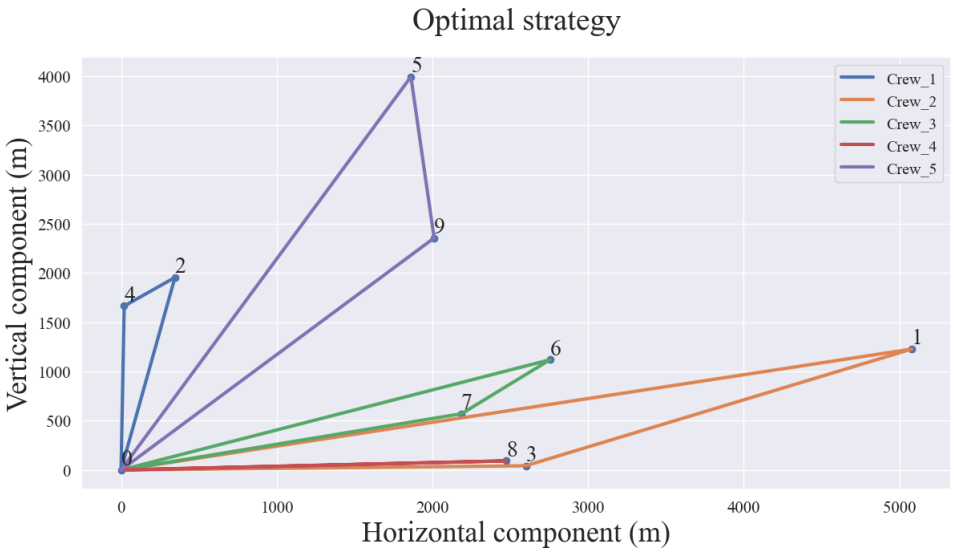


Figure 6.43: Recovery with 6 crews

included) are considered. The mean repair time (from type 1 to type 8) is set to $\mathbf{[10, 15, 20, 25, 50, 60, 70, 80]}$ (minutes).

Therefore, based on what we proposed in Section 6.4.1, the recovery strategies with different number of crews are calculated which can be seen in Table 6.5.

As we mentioned before, the optimization strategy encourages more crews to participate in the recovery to solve the incident event faster while taking into account the minimum travel time. At the same time, excessive participation will be punished to reduce the situation that there is no crew in the depot.

Consequently, as the crew increases, the loss of the objective function shows a trend of decreasing and then increasing. In the best strategy **5 crews** are involved in the recovery operation which is highlight with bold in Table 6.5. The total operating duration (including the travel time back to depot) for the five crews are $\left[\boldsymbol{a}^*_{d,1}, \boldsymbol{a}^*_{d,2}, \boldsymbol{a}^*_{d,3}, \boldsymbol{a}^*_{d,4}, \boldsymbol{a}^*_{d,5}\right] = \mathbf{[181, 171, 105, 130, 200]}$ (minutes). Based on calculation, the operating duration without travel time back to depot for the five crews are $\mathbf{[164, 119, 83, 105, 157]}$ (minutes). In virtue of (6.51), the estimated SoS recovery time $\hat{\boldsymbol{t}}^{\boldsymbol{R}} = \mathbf{164}$ (minutes).

6.4.3 Validation of RTE via Flexsim

According to the crew dispatch optimization results presented, the routing and scheduling of the five crews are simulated in the digital twin of smart port using Flexsim. This simulation allows us to observe how crews are dispatched to each PCS in the smart port over time, aiming to minimize the traveling time, lost and potential risk and optimize the overall restoration of SoS.

The workload progression of each PCS over time is illustrated in Fig 6.44. In this figure, the workload of each PCS starts from zero and gradually increases, signifying that the PCS has undergone repairs and is back in operation. The step-like shape of

Table 6.5: Results of the optimization

Crews in $\mathscr{E}$		Recovery strategy for each crew	Loss
1 Crew	$\boldsymbol{c_1}$	0 -> 6 -> 7 -> 8 -> 9 -> 5 -> 2 -> 4 -> 1 -> 3 -> 0	1095.1
2 Crews	$\boldsymbol{c_1}$	0 -> 3 -> 1 -> 0	1052.6
	$\boldsymbol{c_2}$	0 -> 6 -> 7 -> 8 -> 9 -> 5 -> 2 -> 4-> 0	
3 Crews	$\boldsymbol{c_1}$	0 -> 3 -> 1 -> 0	1037.0
	$\boldsymbol{c_2}$	0 -> 8 -> 7 -> 6 -> 0	
	$\boldsymbol{c_3}$	0 -> 9 -> 5 -> 2 -> 4 -> 0	
4 Crews	$\boldsymbol{c_1}$	0 -> 2 -> 4 -> 0	1025.7
	$\boldsymbol{c_2}$	0 -> 3 -> 1 -> 0	
	$\boldsymbol{c_3}$	0 -> 8 -> 7 -> 6 -> 0	
	$\boldsymbol{c_4}$	0 -> 9 -> 5 -> 0	
5 Crews	$\boldsymbol{c_1}$	0 -> 2 -> 4 -> 0	**1020.0**
	$\boldsymbol{c_2}$	0 -> 3 -> 1 -> 0	
	$\boldsymbol{c_3}$	0 -> 6 -> 7 -> 0	
	$\boldsymbol{c_4}$	0 -> 8 -> 0	
	$\boldsymbol{c_5}$	0 -> 9 -> 5 -> 0	
6 Crews	$\boldsymbol{c_1}$	0 -> 1 -> 0	1020.6
	$\boldsymbol{c_2}$	0 -> 2 -> 4 -> 0	
	$\boldsymbol{c_3}$	0 -> 3 -> 0	
	$\boldsymbol{c_4}$	0 -> 6 -> 7 -> 0	
	$\boldsymbol{c_5}$	0 -> 8 -> 0	
	$\boldsymbol{c_6}$	0 -> 9 -> 5 -> 0	
7 Crews	$\boldsymbol{c_1}$	0 -> 1 -> 0	1026.3
	$\boldsymbol{c_2}$	0 -> 2 -> 0	
	$\boldsymbol{c_3}$	0 -> 3 -> 0	
	$\boldsymbol{c_4}$	0 -> 4 -> 0	
	$\boldsymbol{c_5}$	0 -> 6 -> 7 -> 0	
	$\boldsymbol{c_6}$	0 -> 8 -> 0	
	$\boldsymbol{c_7}$	0 -> 9 -> 5 -> 0	
8 Crews	$\boldsymbol{c_1}$	0 -> 1 -> 0	1035.7
	$\boldsymbol{c_2}$	0 -> 2 -> 0	
	$\boldsymbol{c_3}$	0 -> 3 -> 0	
	$\boldsymbol{c_4}$	0 -> 4 -> 0	
	$\boldsymbol{c_5}$	0 -> 6 -> 0	
	$\boldsymbol{c_6}$	0 -> 7 -> 0	
	$\boldsymbol{c_7}$	0 -> 8 -> 0	
	$\boldsymbol{c_8}$	0 -> 9 -> 5 -> 0	

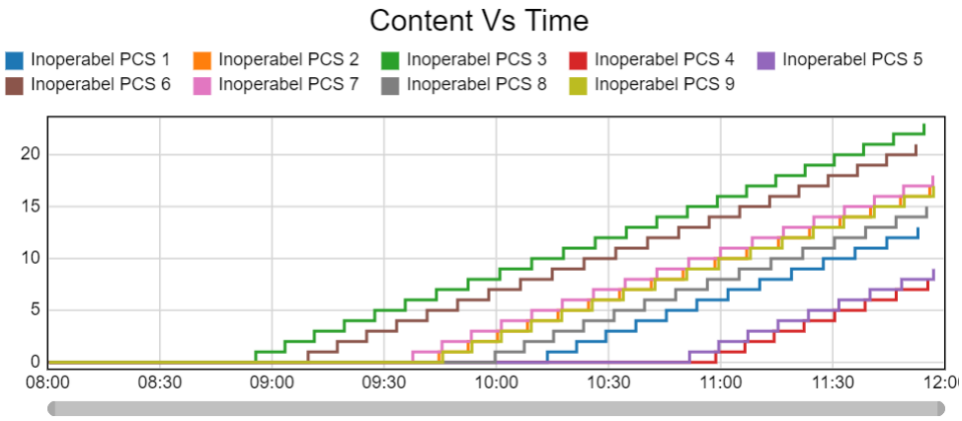


Figure 6.44: Workloads of inoperable PCSs with the optimized 5 crews recovery strategy.

the workload increase indicates the duration of each PCS transporting cargo within the smart port, highlighting the discrete nature of operational periods and task completion times. This growth in workload reflects the successful restoration of each system to its functional state, validating the effectiveness of the crew dispatch strategy.

Furthermore, the recovery order of the PCSs, as depicted in Fig 6.44, aligns perfectly with the optimized dispatch results shown in Table 6.5. This consistency confirms that the simulated crew deployment follows the optimal sequence determined by the optimization algorithm, ensuring that maintenance efforts are efficiently prioritized to achieve the best possible recovery performance. From Fig. 6.44, we can observe that the **4**th PCS, which is the last to recover, is repaired at approximately **164** minutes. This observation is consistent with the overall SoS recovery time estimation obtained from the optimization results.

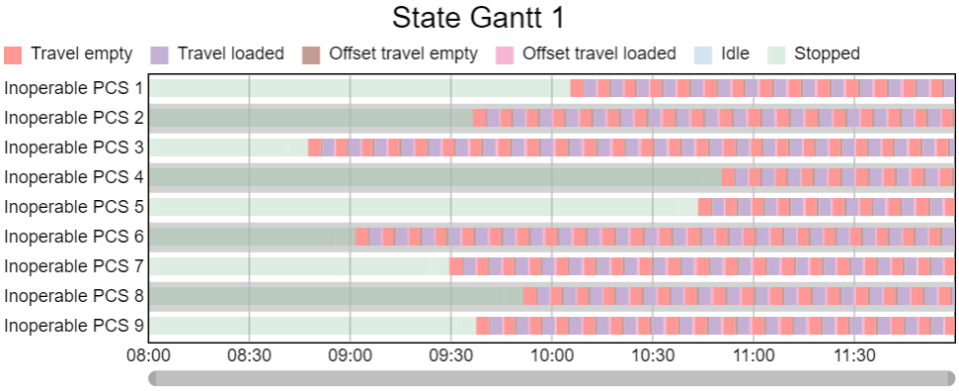


Figure 6.45: State Gantt of inoperable PCSs with the optimized 5 crews recovery strategy.

In addition to the workload progression illustrated in Fig. 6.44, the State Gantt has been created in Fig 6.45 to provide a clearer visualization of the PCS recovery

process over time. The State Gantt complements the Fig. 6.44 by illustrating the operational status of each PCS as they transition from being out of service to becoming fully operational.

In Fig 6.45, each PCS is represented by a horizontal bar that indicates its state over time. The bars change colorto reflect different states, such as travel empty, travel loaded, stopped or idle. The transitions on the Gantt chart correspond directly to the points where the workload of each PCS starts to increase in Fig. 6.44, providing a visual timeline of the recovery process.

The State Gantt allows us to observe the exact moments when each PCS returns to service, with the **4**th PCS being restored at around **164** minutes, as previously noted. It also highlights the sequential nature of the repairs, illustrating how the optimization strategy was executed in real-time, with each PCS being addressed in the order that maximizes overall SoS recovery.

This visualization not only reinforces the consistency between the State Gantt and the optimization results but also provides an intuitive overview of the recovery process, making it easier to understand the effectiveness and timing of the crew dispatch decisions.

6.4.4 Conclusion

Considering the diverse resources and capacities in a large scale SoS, this work incorporates the capacity distribution into the crew dispatch and restoration optimization problem facing multiple kinds of malfunctions in various PCSs. The proposed optimization strategy is more applicable in practical implementation as different crews are natural to have distinct capacities and performance. Not only the travel time and malfunction lost, but also the potential risk caused by crew absence have been included in the design of multi-objective function. The recovery time estimation can be calculated based on the optimal solution to the optimization problem. The proposed strategy is supported finally with a case study on a SoS with **9** PCSs and **8** crews, where **8** types of malfunctions are considered.

6.5 Conclusion

This chapter contributes to the integrated design in the concept of the decision-making, where the resilience reconfiguration strategy has been devised over three aspects. The first part presented a resilience implementation framework with bottom-up monitoring and top-down management, enabling SoS to proactively detect faults, adjust operating modes and optimise the performance. Subsequently, data-driven prediction has been employed to estimate the disruptive disasters for allocation and scheduling CSs in advance to mitigate the performance degradation. Moreover, facing with unpredictable disasters, a crew dispatch optimization algorithm with heterogeneous capacity distribution has been proposed to manage the repair of PCSs and restoration the overall SoS. The proposed SoS resilience reconfiguration

strategy reflects the capacity of SoS to remain operational, quickly response, predict and prepare in advance, and optimize the repair process confronting disruptive disasters.

Chapter 7

Conclusion and prospective

Contents

7.1 Summary of main results 164
7.2 Future works 167
7.2.1 Controllability analysis 167
7.2.2 Performance analysis 168
7.2.3 Reconfiguration strategy 168

7.1 Summary of main results

This thesis aims to establish a comprehensive and integrated framework that enhances the overall resilience of the SoS. The proposed design to strengthen SoS resilience can be divided into three key aspects.

First, we investigate the controllability of the SoS under dynamic networked interactions. In the event of malfunctions or faults, certain PCSs may become uncontrollable. Understanding the overall controllability of the SoS is essential for resilient structural analysis, enabling the system to remain operational despite local disruptions.

The second aspect centers on system diagnosis. We have developed a performance evaluation framework for the SoS, extracting features from various perspectives, including the dynamic performance of PCSs, the effectiveness of MCSs in executing supervised missions, communication quality, and top-down management efficiency. The framework facilitates the classification of SoS performance into distinct status types, assisting managers in identifying necessary reconfigurations for system restoration. This lays the ground for the resilience reconfigurations.

Finally, based on the structural analysis and system diagnosis, we propose a resilience reconfiguration strategy for decision-making within the SoS. This strategy encompasses a resilience implementation approach featuring bottom-up monitoring and top-down management, advance resilience planning informed by data-driven disaster predictions, and recovery time estimation through crew dispatch in response to unpredictable disruptive events.

To implement an integrated design for resilient SoS, we first propose formulations for collaborative heterogeneous systems across multiple levels, each tailored

to different aspects. To ensure that each MCS effectively manages its specific supervisory scope, we introduce a management decomposition strategy that restricts interactions to adjacent levels. This approach streamlines the overall management process and reduces the risk of information overload.

Building on this management decomposition, we introduce three types of topologies to represent the communication and organization among CSs within the SoS model. In the context of controllability analysis, it is essential to illustrate the pairwise interactions among CSs to accurately capture the dynamics of the system.

To achieve this, we propose a graph-based topology that captures both the organization and point-to-point interactions within physical dynamics, thereby facilitating the analysis of the dynamic response of the SoS. Additionally, our design for resilience reconfiguration emphasizes the multi-way relationships among CSs across different levels. Relying solely on pairwise edges from graph theory can result in an overwhelming number of connections, so we introduce a hypergraph-based topology that encapsulates multi-way relationships within a single hyperedge.

However, to create a comprehensive performance evaluation framework for SoS, it is essential to consider both the dynamic response of PCSs and the organization among CSs across different levels. Therefore, we propose a hybrid graph/hypergraph-based topology. In this framework, dynamic interactions among PCSs are represented through a graph, while hypergraphs illustrate the interconnections across various levels, providing a more nuanced view of the system.

The first part of integrated design was emerged on the resilient structure analysis for SoS. Due to the presence of local disruptions and malfunctions, only a portion of CSs are subject to external input. Thus, controllability analysis has been provided to ensure the overall SoS is controllable based on the networked interactions while a part of CSs are out of control. The controllability of multi-level SoS has been explored over directed graph-based topology. The considered CSs are of high-degree heterogeneity including different dimensions, heterogeneous dynamics and nonidentical interactions. For a two layer heterogeneous SoS, we utilized the modified Sylvester equation, which is widely applied in cooperative output regulation, to mitigate the computational complexity. The results have been thereafter extended to multi-level SoS in virtue of the proposed two-layer decomposition approach, which decompose sophisticated higher-level SoS into various lower-layer components from a "divide-and-conquer" perspective. Some necessary and/or sufficient conditions have been derived to ensure the SoS controllability with less computational cost. Examples of some typical graphs in the communication and organization of SoS, such star, chain, circle and tree have been tested to validate the effectiveness of the theoretical results.

The second part built a comprehensive evaluation framework of the SoS performance, to classify them for system diagnosis. The diagnosis helps the manager to determine the reconfiguration to restore the SoS in different statuses facing with disturbances, communication issue and malfunctions. Various performance features have been extracted from multiple perspectives, such as PCSs performance, PCSs performance, the inter/intra-layer communication quality and top-down manage-

ment efficiency. These features offer a detailed overview of the system's performance, including task achievements, dynamic response, communication and management. By employing AI-based classifiers on the extracted features, the SoS performance was classified into five status types: Normal, Robust, Fault-tolerance, Resilience and Breakdown. The classification results reflect the severity of degradation and the urgency or need for a resilient reaction, guiding engineers in the choice of appropriate measures to restore SoS performance. The simulation of a three-level SoS with three MCSs and eight PCSs (UAVs) has been performed. Multiple classification algorithms have been employed in the simulation, showing high accuracy across both the training and test sets. The results of the performance classification inform the manager of the degree of degradation, helps in determining the necessary resilient actions for recovery in the following resilience reconfiguration.

Subsequently, the resilience reconfiguration strategy for decision making to prevent and recover system from degradation has been developed. A resilience implementation approach was introduced on the basis of bottom-up monitoring and top-down reconfiguration, which can proactively detect the existence of faults and adjust the operating modes. In virtue of the FDI of CPSs in the bottom level, the supervisers in higher levels are able to receive the fault information leveraging the bottom-up monitoring. Additionally, the top-down reconfiguration enables supervisors to send necessary instructions to CSs at lower levels to recover and meanwhile optimize the performance of the overall SoS. The reconfiguration framework was devised from various aspects, including communication, operation and mission, which is more comprehensive compared with previous studies. The case study considered the physical twin of the smart port with several scenarios, which validated the effectiveness of the designed reconfiguration algorithm.

Moreover, disaster prediction and advance planning are vital for minimizing impact and loss, improving response efficiency, and enhancing SoS resilience. To this end, the resilience planning in advance has been explored for SoS under predictable disasters. The first step was emerged on the data-driven disaster prediction. Using flash floods resulting from excessive rainfall as a case study, a hybrid rainfall prediction model has been proposed, incorporating data-driven AI techniques with multi-scale time series analysis. In the simulation, the prediction model has been tested using daily climate data from Indonesia, spanning 2010 to 2022, which includes data from 173 sampling locations across 34 provinces. The resilience planning optimization has been developed on the basis of disater prediction, scheduling the operating of PCS at distinct operating sites in advance. It minimizes the operation cost and restricts the performance degradation to a certain extent. A case study on the digital twin of smart port via Flexsim subject to flash flood has been presented, allocating the operating hours of human operators, trucks, freight trains and ships in port. An extended disaster scenario, the wind scenario has also been presented. In both scenarios, the groups utilizing disaster prediction-based resilient planning performed better than those without it, demonstrating the effectiveness of the proposed strategy.

Note that a number of disasters still remain unpredictable. These disasters often

strike suddenly and unexpectedly with little or no warning, making them difficult to foresee and prepare for in advance. Facing with those unpredictable disasters, the crew dispatch is crucial in the repairing of damaged CSs to restore the overall SoS. Additionally, the scheduling and routing of crews enables the estimation of SoS recovery time, which enhances the system resilience. Therefore, the optimization-based recovery time estimation for SoS facing unpredictable disasters has been explored. The crew dispatch optimization considered the heterogeneous capacities and performance in the repair of alternative of PCSs. The developed optimization strategy incorporated the crew dispatch, PCS restoration and heterogeneous capacity distribution, presentign the cooperative behavior of CSs in SoS. The devised multi-objective optimization function minimizes the the crew travel time, malfunction lost and potential risk. The recovery time estimation was derived from the results of optimization solution. A case study has been performed on the digital twin of smart port using Flexsim, with nine inoperable PCSs, eight crews and eight types of faults with heterogeneous repair times, showing the validity of the developed strategy.

In summary, the proposed resilient integrated design has provided: controllability analysis on the overall SoS for a resilient SoS structure, performance evaluation and classification for SoS diagnosis, resilience reconfiguration including implementations, proactive preparation, crew dispatch for repairing. These approaches enable a more structured (controllability analysis), reactive (performance evaluation), efficient (reconfiguration) and overall resilient SoS, guaranteeing that the SoS is better equipped to restore its normal status after disruptive events.

7.2 Future works

7.2.1 Controllability analysis

The controllability analysis in this work mainly focus on the SoS subject graph-based communication and organization topology. The dynamics of PCSs, the virtual dynamics of MCSs and the interactions among CSs are supposed to be linear. Although we have provided linear approximation in Chapter 3 for PCSs under nonlinear dynamics, there still exists a part of systems that cannot be accurately described using linear approximation. Many natural and engineered systems inherently exhibit nonlinear behaviours. Nonlinear dynamics allows for a more accurate representation of these systems, capturing phenomena such as turbulence, chaos, bifurcations, and other complex behaviours that linear models cannot. To this end, we further expect to explore the controllability of SoS composed of CSs with nonlinear dynamics as a future research direction. On the one hand, nonlinear dynamics can describe a wider range of physical phenomena.

On the other hand, unlike linear systems, where controllability is well-understood and can often be determined using relatively straightforward mathematical criteria, nonlinear systems lack a general, universally applicable theorem. Even for the nonlinear single systems, each system may require a tailored approach, often involving sophisticated and complex mathematical tools like Lie algebra and Lie bracket.

Moreover, to the authors' knowledge, there are little research considering the controllability of networked nonlinear systems. Some works defined the controllability of nonlinear systems as the connectivity of the attractor network, which is not related to our topic. A part of works discussed the analysis the controllability of some specific nonlinear dynamics with experiments. Almost no work can give a general sufficient condition to ensure the controllability of NSs with nonlinear dynamics. For a multi-level SoS, the theoretical analysis tends to be more complicated due to the heterogeneous dynamics of diverse PCSs and nonidentical interactions among CSs across multiple levels. Thus, there still remain technical difficulties in the SoS with nonlinear dynamics.

Additionally, the controllability analysis of SoS subject to hypergraph-based communication and organization is also a interesting topic, since hypergraph provides a clearer and simpler description for multi-way relationship. Nevertheless, similar to the analysis on controllability of nonlinear systems, even for single systems, there are very little results on the controllability analysis using hypergraph. The analysis was mainly emerged on systems with specific nonlinear dynamics, such as homogeneous polynomial systems. Therefore, technical challenges persist in the controllability analysis of SoS under hypergraph-based topology.

7.2.2 Performance analysis

In the performance analysis of SoS, although we have already provided a comprehensive feature construction, the current five SoS statuses can be further refined. In Chapter 5, we have addressed five SoS performance classes, including Normal, Robust, Fault-tolerance, Resilience and Breakdown.

As a future research direction, we aim to improve the classification by refining the classes. For example, in the robust class, we want to pinpoint the types of external disturbances or noises with their degree of negative impact. For fault-tolerance class, we want to refine the class identifying the types of faults and malfunctions and meanwhile evaluating the extent of their negative impact. For resilience class, we expect to determine the resilience reactions to restore the SoS performance. By breaking down broad categories into more specific subclasses, we can achieve greater accuracy in identifying and differentiating between different types of instances, leading to more precise analysis.

Furthermore, we aim to develop an online API platform capable of generating more advanced algorithms by selecting diverse feature selection strategies, enabling the implementation of more adaptable classification approaches.

7.2.3 Reconfiguration strategy

For example, the types and quantities of CSs within the simulated SoS could be expanded to further evaluate the effectiveness of the proposed resilience implementation algorithm.

In the data-driven prediction of disaster, only flash flood has been estimated in

Chapter 6.3. We further expect to perform the prediction of more kinds of disasters to prove the efficacy of the disaster prediction model.

Additionally, facing with predictable and unpredictable disasters, we plan to test the proposed reconfiguration strategy in the real port, for instance, within the project SAFARI[1], which is dedicated to developing a digital platform to enhance the resilience of port infrastructure, ensuring operational efficiency and safety during extreme weather events.

With the aide of the application in practice, we can not only test the effectiveness of resilience planning in port management based on the data-driven disaster prediction, but also explore the validity of the crew dispatch optimization algorithm with heterogeneous capacity distribution.

[1] https://www.safariports.eu/

Appendices

Appendix A

Preliminaries

In Preliminaries, there are four main parts introducing the theories and methodologies which are involved in the thesis.

A.1 Graph and Hypergraph

Graph and Hypergraph are both mathematical structures used to model relationships between objects. However, they differ significantly in how they represent these relationships. Graphs are simpler structures suitable for modelling pairwise relationships, while Hypergraphs are more general structures capable of representing complex multi-way relationships. The choice between using a graph or a hypergraph depends on the specific nature of the relationships being modeled in the application at hand.

Therefore, the Graph and Hypergraph are introduced separately in this section.

A.1.1 Graph

Graph theory is a branch of mathematics that studies the properties of graphs, which are structures made up of nodes (vertices) connected by edges. It has applications in various fields, including computer science, biology, social sciences, and more. Here are the basic concepts and terminology in graph theory.

A Graph $\mathcal{G}$ is an ordered pair $(\mathcal{V}, \mathcal{E})$ where:

- $\mathcal{V}$ is a non-empty set of vertices (or nodes).
- $\mathcal{E}$ is a set of edges, which are unordered pairs of vertices.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent a graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is a finite vertex, in which vertex v_i represents the i-th vertex in graph $\mathcal{G}$. $\mathcal{E} = \{e_1, e_2, \ldots, e_m\} \subseteq \mathcal{V} \times \mathcal{V}$ signifies the edge set, which describes the connections among $\mathcal{G}$.

In graph theory, different types of graphs are characterized by the properties and constraints imposed on their vertices and edges. The types of Graphs are given below to have a basic understanding of different types of Graphs:

- **Simple Graph:** A graph with no loops (edges connecting a vertex to itself) and no multiple edges between the same pair of vertices.
- **Multigraph:** A graph that can have multiple edges (parallel edges) between the same pair of vertices.

- **Directed Graph (Digraph):** A graph where each edge has a direction, going from one vertex to another.
- **Weighted Graph:** A graph where edges have weights (or costs) associated with them.
- **Complete Graph (K_n):** A graph in which every pair of distinct vertices is connected by a unique edge.
- **Bipartite Graph:** A graph whose vertices can be divided into two disjoint sets such that no two vertices within the same set are adjacent.
- **Tree:** An acyclic connected graph.
- **Forest:** A disjoint set of trees.
- ...

Connectedness in graph theory refers to the property of a graph where there exists a path between any pair of vertices. This concept is fundamental in understanding the structure and behavior of graphs. Here is a detailed look at common used graphs:

- **Connected Graph (undirected graph):** An undirected graph is connected if there is a path between any two vertices.
- **Disconnected Graph (undirected graph):** An undirected graph is disconnected if there is at least one pair of vertices that do not have a path between them.
- **Strongly Connected Graph (directed graph):** A directed graph is strongly connected if there is a path from any vertex to every other vertex.
- **Weakly Connected Graph (directed graph):** A directed graph is weakly connected if replacing all directed edges with undirected edges makes the graph connected.

The connectedness of Graph is widely used in network design, pathfinding and navigation, cluster analysis, robustness and reliability, social networks, etc. Connectedness is a crucial property in graph theory with wide-ranging applications across various fields. It ensures the reachability and coherence of a network, enabling effective communication, transportation, data analysis, and more.

Adjacency Matrix is a square matrix used to represent a graph, where the element at row $\boldsymbol{i}$ and column $\boldsymbol{j}$ represents the presence (and possibly the weight) of an edge between vertices $\boldsymbol{i}$ and $\boldsymbol{j}$.

$\boldsymbol{A} = [\boldsymbol{a_{ij}}]$ is the adjacency matrix related to graph $\boldsymbol{\mathcal{G}}$, where

$$\boldsymbol{a_{ij}} = \begin{cases} \mathbf{1}, & \text{if } (\boldsymbol{v_i}, \boldsymbol{v_j}) \in \boldsymbol{\mathcal{E}}, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{A.1}$$

A.1.2 Hypergraph

Hypergraph is an extension of graph theory, where edges (called hyperedges) can connect more than two vertices. They are useful in modelling complex relationships that involve multiple entities simultaneously. The preliminaries of hypergraph theory are given as follows.

A hypergraph $\mathcal{H}$ is an ordered pair $(\mathcal{V}, \mathcal{E})$ where:

- $\mathcal{V}$ is a non-empty set of vertices.
- $\mathcal{E}$ is a set of non-empty subsets of $\mathcal{V}$ called hyperedges.

Here, $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ represents a finite vertex set composed of vertex v_i. $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ signify the hyperedge set whose elements $e_j, j = 1, 2, ..., m$ are the subsets of V. Note that the hyperedge e_j satisfies $e_j \neq \emptyset$ and $\bigcup_{j=1}^{m} e_j = V$.

The main types of Hypergraph are introduced below:

- **k-Uniform Hypergraph:** A hypergraph where every hyperedge connects exactly k vertices. For example, in a 3-uniform hypergraph, every hyperedge is a 3-element subset of $\mathcal{V}$.
- **Simple Hypergraph:** A hypergraph with no repeated hyperedges.
- **Regular Hypergraph:** A hypergraph where each vertex has the same degree.

A hyperpath HP_{v_s,v_t} of $\mathcal{E}$ between vertices v_s and v_t is a sequence of hyperedges $\{e_{j1}, e_{j2}, ..., e_{jq}\}, e_{jg} \in \mathcal{E}, g = 1, 2, ..., q$ such that $v_s \in e_{j1}$, $v_t \in e_{jq}$ and $e_{jg} \cap e_{j(g+1)} \neq \emptyset, g = 1, 2, ..., q-1$. Besides, vertex v_i is isolated if $\forall j, v_i \notin e_j$. A weak deletion of hyperedge e_j removes a hyperedge e_j. A weak deletion of vertex v_i removes v_i from $\mathcal{V}$ and v_i from all e_j satisfying $v_i \in e_j$ in $\mathcal{E}$. A weighted hypergraph can be given by $H = (V, E, W)$, where extra term $W = [\omega_{i,j}] \in R^{n \times m}$ is the weight matrix of hypergraph H. $\omega_{i,j}$ represents the weight of vertex v_i in hyperedge e_j.

The indicator function with respect to the relationship between vertex and hyperedge is defined as [Zhang 2018]:

$$I(v_i, e_j) = \begin{cases} 1, if\ v_i \in e_j, \\ 0, otherwise. \end{cases} \tag{A.2}$$

Let δ_j denotes the weight of hyperedge e_j as follows:

$$\delta_j = \frac{1}{|e_j|} \sum_{v_i \in e_j} w_{i,j} \tag{A.3}$$

where $|e_j|$ represents the cardinality of a hyperedge e_j, i.e., the number of nodes composed in the hyperedge e_j [Guo 2016].

The dual of weighted hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ is a hypergraph $\mathcal{H}^* = (\mathcal{V}^*, \mathcal{E}^*, \mathcal{W}^*)$. The vertex set $\mathcal{V}^*$ of dual hypergraph $\mathcal{H}^*$ is $\mathcal{V}^* =$

$\{v_1^*, v_2^*, \ldots, v_m^*\}$, the hyperedge set $\mathcal{E}^*$ is defined as

$$\begin{aligned} \mathcal{E}^* &= \{e_1^*, e_2^*, ..., e_n^*\} \\ e_i^* &= \{v_j^*, \text{if } v_i \in e_j \text{ in } \mathcal{H}\}. \end{aligned} \tag{A.4}$$

The weighted matrix $\mathcal{W}^* \in \mathbb{R}^{m \times n}$ of dual hypergraph $\mathcal{H}^*$ is defined as $\mathcal{W}^* = [w_{ij}^*]$, where $w_{ij}^* = w_{ji}$.

Connected Hypergraph means that a hypergraph is connected if there is a path between any pair of vertices, considering paths through hyperedges.

Hypergraphs generalize Graphs by allowing edges to connect more than two vertices, making them suitable for modelling complex multi-way relationships. They extend the concepts of vertices and edges to vertices and hyperedges, where hyperedges can include any number of vertices. The fundamental concepts of hypergraphs, such as incidence matrices, paths, cycles, and connectedness, provide a rich framework for various applications in computer science, engineering, and data analysis.

A.2 Basic concepts in control theory

Controllability, observability and cooperative output regulation are basic concepts in control theory which are involved in the research of the SoS's structure analysis. Consequently, those three concepts are introduced in detail in the following subsections.

A.2.1 Controllability

As a fundamental concept in control theory, controllability, pertains to the ability to move a system from any initial state to any desired final state within a finite time span, using appropriate inputs. Controllability means that by applying suitable control inputs, one can drive the state of the system to any desired state within finite time. If a system is not controllable, it means that there are some states that cannot be reached, no matter what control input is applied. It is crucial for designing and analyzing control systems. Here is an overview of the preliminary concepts of controllability in the following.

A dynamical system can often be represented in state-space form below:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{A.5}$$

$$y(t) = Cx(t) \tag{A.6}$$

- $x(t)$: the state vector.
- $u(t)$: the control input vector.
- $y(t)$: the output vector.
- A: the state matrix.

- $\boldsymbol{B}$: the input matrix.
- $\boldsymbol{C}$: the output matrix.

The controllability of a system can be determined using the controllability matrix $\boldsymbol{\mathcal{C}}$:

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{AB} & \boldsymbol{A^2B} & \dots & \boldsymbol{A^{n-1}B} \end{bmatrix}$$

where $\boldsymbol{n}$ is the number of states.

A system is said to be controllable if the controllability matrix $\boldsymbol{\mathcal{C}}$ has full rank. That is, $\boldsymbol{\mathcal{C}}$ should have rank $\boldsymbol{n}$, where $\boldsymbol{n}$ is the number of states in the system. If the rank of $\boldsymbol{\mathcal{C}}$ is less than $\boldsymbol{n}$, the system is not completely controllable.

For a discrete-time system, the state-space will be represented by:

$$\boldsymbol{x[k+1] = A_d x[k] + B_d u[k]} \tag{A.7}$$

$$\boldsymbol{y[k] = D_d x[k]} \tag{A.8}$$

The corresponding controllability matrix is given below:

$$\boldsymbol{\mathcal{C}_d} = [\boldsymbol{B_d} \ \boldsymbol{A_d B_d} \ \boldsymbol{A_d^2 B_d} \ \dots \ \boldsymbol{A_d^{n-1} B_d}]$$

Another way to test controllability is using the controllability Gramian $\boldsymbol{W_c}$.

For continuous-time systems:

$$\boldsymbol{W_c} = \int_0^\infty \boldsymbol{e^{A\tau} B B^T e^{A^T \tau} d\tau}$$

For discrete-time systems:

$$\boldsymbol{W_c} = \sum_{\boldsymbol{k=0}}^{\infty} \boldsymbol{A_d^k B_d B_d^T (A_d^T)^k}$$

A system is controllable if the controllability Gramian is positive definite (i.e., all its eigenvalues are positive).

The Popov-Belevitch-Hautus (PBH) controllability and observability tests allow us to test for both controllability and observability. In this part, the PBH controllability test is introduced firstly, and the PBH observability test will given in Subsection A.2.2.

Lemma A.2.1 (Popov-Belevitch-Hautus (PBH) controllability test) *[Hespanha 2018] For a linear time-invariant (LTI) system* $\boldsymbol{\Sigma}$*:* $\boldsymbol{\dot{x}(t) = Ax(t) + Bu(t)}$*, where* $\boldsymbol{A \in \mathbb{R}^{n \times n}}$*,* $\boldsymbol{B \in \mathbb{R}^{n \times m}}$*, the following statements are equivalent:*

- *Pair* $\boldsymbol{(A, B)}$ *is controllable (i.e., LTI system* $\boldsymbol{\Sigma}$ *is controllable);*
- *For arbitrary* $\boldsymbol{s \in \mathbb{C}}$*,* $\boldsymbol{rank \begin{bmatrix} sI - A & B \end{bmatrix} = n}$*;*

- *For arbitrary* $\boldsymbol{s \in \sigma(A)}$, $\boldsymbol{rank\begin{bmatrix} sI - A & B \end{bmatrix} = n}$;

where $\boldsymbol{\sigma(A)}$ *represents the eigenvalue set of matrix* $\boldsymbol{A}$.

The PBH controllability test allows us to identify uncontrollable modes. The mode associated with right and left eigenvectors $\boldsymbol{v}$, $\boldsymbol{w}$ is

- **uncontrollable** if $\boldsymbol{w^T B = 0}$.

Controllability is a crucial concept in control theory that ensures the feasibility of state regulation through input manipulation. The rank condition of the controllability matrix and the properties of the controllability Gramian are fundamental tools to determine the controllability of a system.

A.2.2 Observability

Observability pertains to the ability to infer the internal state of a system from its external outputs. Observability means that by observing the outputs over time, one can reconstruct the entire state vector. If a system is not observable, it means that some aspects of the system's state cannot be determined from the outputs. In the following part, the preliminary concepts of observability will be introduced detailly.

A dynamical system can be represented in state-space form as what we proposed in equation A.5.

The controllability of a system can be determined using the controllability matrix $\boldsymbol{\mathcal{C}}$:

$$\boldsymbol{\mathcal{C} = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix}}$$

where $\boldsymbol{n}$ is the number of states.

A system is said to be controllable if the controllability matrix $\boldsymbol{\mathcal{C}}$ has full rank. That is, $\boldsymbol{\mathcal{C}}$ should have rank $\boldsymbol{n}$, where $\boldsymbol{n}$ is the number of states in the system. If the rank of $\boldsymbol{\mathcal{C}}$ is less than $\boldsymbol{n}$, the system is not completely controllable.

For a discrete-time system have the form of equation A.7. The observability matrix is:

$$\boldsymbol{\mathcal{O}_d = \begin{pmatrix} C_d \\ C_d A_d \\ C_d A_d^2 \\ \vdots \\ C_d A_d^{n-1} \end{pmatrix}}$$

Another way to test observability is using the observability Gramian $\boldsymbol{W_o}$.

For continuous-time systems:

$$\boldsymbol{W_o = \int_0^\infty e^{A^T \tau} C^T C e^{A\tau} d\tau}$$

For discrete-time systems:

$$W_o = \sum_{k=0}^{\infty} (A_d^T)^k C_d^T C_d A_d^k$$

A system is observable if the observability Gramian is positive definite (i.e., all its eigenvalues are positive).

Observability is the dual of controllability. This means that many results and methods in controllability have corresponding counterparts in observability. For instance:

- The controllability matrix and observability matrix are duals.
- The controllability Gramian and observability Gramian are duals.

Lemma A.2.2 (Popov-Belevitch-Hautus (PBH) observability test) *[Hespanha 2018] For a linear time-invariant (LTI) system* $\boldsymbol{\Sigma}$*:* $\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{A}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{B}\boldsymbol{u}(\boldsymbol{t})$*,* $\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{C}\boldsymbol{x}(\boldsymbol{t})$*, where* $\boldsymbol{A} \in \mathbb{R}^{n\times n}$*,* $\boldsymbol{C} \in \mathbb{R}^{q\times n}$*, the following statements are equivalent:*

- *Pair* $(\boldsymbol{C}, \boldsymbol{A})$ *is observable (i.e., LTI system* $\boldsymbol{\Sigma}$ *is observable);*
- *For arbitrary* $\boldsymbol{s} \in \mathbb{C}$*,* $\boldsymbol{rank} \begin{bmatrix} \boldsymbol{sI} - \boldsymbol{A} \\ \boldsymbol{C} \end{bmatrix} = \boldsymbol{n}$*;*
- *For arbitrary* $\boldsymbol{s} \in \boldsymbol{\sigma}(\boldsymbol{A})$*,* $\boldsymbol{rank} \begin{bmatrix} \boldsymbol{sI} - \boldsymbol{A} \\ \boldsymbol{C} \end{bmatrix} = \boldsymbol{n}$*.*

The PBH observability test allows us to identify unobservable modes. The mode associated with right and left eigenvectors $\boldsymbol{v}$, $\boldsymbol{w}$ is

- **unobservable** if $\boldsymbol{Cv} = \boldsymbol{0}$**.**

In control theory, observability ensures the feasibility of state estimation through output measurements. The rank condition of the observability matrix and the properties of the observability Gramian are fundamental tools to determine the observability of a system. Understanding observability is essential for designing effective observers and state estimators.

A.2.3 Cooperative output regulation

Cooperative output regulation is an advanced topic in control theory, especially in the context of multi-agent systems. This concept extends the classical control theory to networks of dynamical systems (agents) working together to achieve a common goal, such as synchronizing their outputs or following a leader's trajectory.

Before introducing the cooperative output regulation, the Sylvester equation is discussed in the first. Due to the fact that Sylvester equation lays the foundation for the following discussion.

The Sylvester equation is $\boldsymbol{AX}+\boldsymbol{XB}=\boldsymbol{C}$, where $\boldsymbol{A}\in\mathbb{R}^{n\times n},\boldsymbol{B}\in\mathbb{R}^{n\times m},\boldsymbol{C}\in\mathbb{R}^{q\times n},\boldsymbol{X}\in\mathbb{R}^{n\times n}$. Expressing as $\boldsymbol{S}(\boldsymbol{X})=\boldsymbol{C}$, where $\boldsymbol{S}$ is the linear function $\boldsymbol{S}(\boldsymbol{X})=\boldsymbol{AX}+\boldsymbol{XB}$ ($\boldsymbol{S}$ maps $\mathbb{R}^{n\times n}$ into $\mathbb{R}^{n\times n}$). We refer to $\boldsymbol{S}(\boldsymbol{X})$ as the Sylvester operator.

$\boldsymbol{S}$ is singular if and only if there exists a nonzero $\boldsymbol{X}$ with $\boldsymbol{S}(\boldsymbol{X})=\boldsymbol{0}$. This means $\boldsymbol{AX}+\boldsymbol{XB}=\boldsymbol{0}$, so $\boldsymbol{AX}=\boldsymbol{X}(-\boldsymbol{B})$, which means $\boldsymbol{A}$ and $-\boldsymbol{B}$ share at least one eigenvalue (since $\boldsymbol{X}\neq\boldsymbol{0}$).

So we have: if $\boldsymbol{S}$ is singular, then $\boldsymbol{A}$ and $-\boldsymbol{B}$ have a common eigenvalue. Let's show the converse: if $\boldsymbol{A}$ and $-\boldsymbol{B}$ share an eigenvalue, $\boldsymbol{S}$ is singular suppose

$$\boldsymbol{Av}=\boldsymbol{\lambda v},\boldsymbol{w^T B}=-\boldsymbol{\lambda w^T},\boldsymbol{v},\boldsymbol{w}\neq\boldsymbol{0}$$

then with $\boldsymbol{X}=\boldsymbol{vw^T}$ we have $\boldsymbol{X}\neq\boldsymbol{0}$ and

$$\boldsymbol{S}(\boldsymbol{X})=\boldsymbol{AX}+\boldsymbol{BX}=\boldsymbol{Avw^T}+\boldsymbol{vw^T B}=(\boldsymbol{\lambda v})\boldsymbol{w^T}+\boldsymbol{v}(-\boldsymbol{\lambda w^T})=\boldsymbol{0}$$

which shows $\boldsymbol{S}$ is singular.

So, Sylvester operator is singular if and only if $\boldsymbol{A}$ and $-\boldsymbol{B}$ have a common eigenvalue, or Sylvester operator is non-singular if and only if $\boldsymbol{A}$ and $-\boldsymbol{B}$ have no common eigenvalues.

After the discussion of Sylvester equation, we will introduce the cooperative output regulation as follows.

A large scale system consists of multiple dynamical systems (PCSs) that interact with each other. Each PCS can be represented by:

$$\begin{aligned}\boldsymbol{\dot{x}_i}&=\boldsymbol{A_i x_i}+\boldsymbol{B_i u_i}\\ \boldsymbol{y_i}&=\boldsymbol{C_i x_i}\end{aligned}$$

where $\boldsymbol{x_i}$ is the state vector of the $\boldsymbol{i}$-th PCS, $\boldsymbol{u_i}$ represents the control input of the $\boldsymbol{i}$-th PCS, $\boldsymbol{y_i}$ means the output of the $\boldsymbol{i}$-th PCS. $\boldsymbol{A_i},\boldsymbol{B_i},\boldsymbol{C_i}$ are the system matrices for the $\boldsymbol{i}$-th PCS.

For cooperative output regulation, the design typically involves solving a set of regulator equations. These equations ensure that the desired output regulation is achieved despite the presence of disturbances and model uncertainties.

$$\begin{aligned}\boldsymbol{A_i X_i}+\boldsymbol{B_i U_i}&=\boldsymbol{X_i S}\\ \boldsymbol{C_i X_i}&=\boldsymbol{Q}\end{aligned}$$

where $\boldsymbol{S}$ means the system matrix of the exosystem generating the reference trajectory or disturbance. $\boldsymbol{Q}$ represents the desired regulated output.

Cooperative output regulation involves controlling a network of PCSs to achieve a collective goal, leveraging the interactions between PCSs. It combines concepts from classical control theory, network theory, and distributed algorithms. By ensuring that PCSs cooperate, we can achieve robust and scalable control solutions for

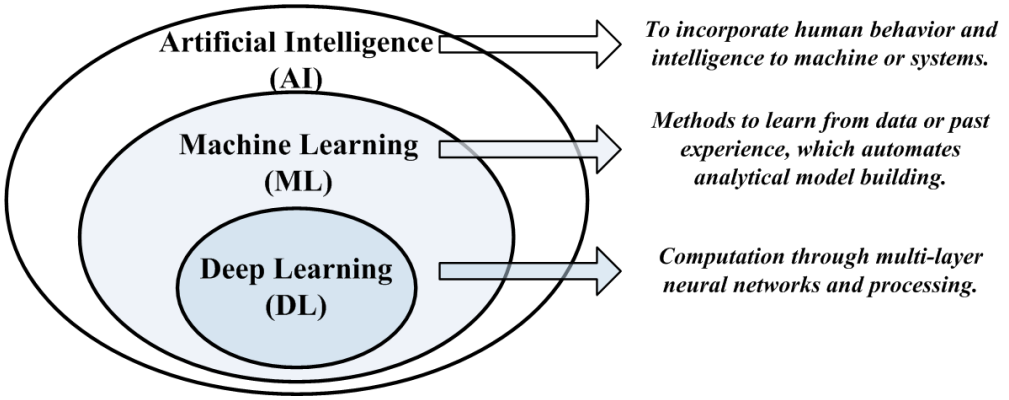


Figure A.1: The relationship among AI, ML and DL.

large scale systems.

A.3 AI-based classification and prediction

AI, ML, and DL are closely related fields that often overlap, but they each have distinct definitions and roles. A brief discussion among AI, ML and DL is given below.

AI is a broad field that encompasses any technique that enables machines to mimic human intelligence. This includes reasoning, problem-solving, learning, perception, language understanding, and more. As a subset of AI, ML involves the development of algorithms and statistical models that enable computers to improve their performance on a specific task through experience (data). DL is a specialized subset of ML that uses NNs with many layers. These Deep Neural Networks (DNNs) are capable of learning complex patterns and representations from large amounts of data. The relationship among AI, ML and DL can be seen in Fig.A.1.

AI-based classification and prediction are essential components in ML and AI, used to categorize data into predefined classes or predict future outcomes based on historical data. The following subsections will describe the AI algorithms involved in detail.

A.3.0.1 AI-based classification

In this subsection, several commonly used algorithms are introduced in two parts, ML and DL, respectively.

Commonly used classification algorithms in ML:

1. **Logistic Regression:** A statistical method for binary classification that models the probability of a binary outcome, where the output can take one of two possible values: $\boldsymbol{y = 1}$ (positive class); $\boldsymbol{y = 0}$ (negative class).

   The core idea of logistic regression is to model the probability of the positive class ($\boldsymbol{y = 1}$) using the sigmoid function (also known as the logistic function,

see in Fig.A.2[1]). The sigmoid function is defined as:

$$\boldsymbol{\sigma(z) = \frac{1}{1 + e^{-z}}}$$

where $\boldsymbol{z}$ is the linear combination of input features. The sigmoid function maps any real-valued number into the range (0, 1), making it suitable for probability estimation.

In logistic regression, the input features $\boldsymbol{x}$ are linearly combined using weights $\boldsymbol{w}$ and a bias term $\boldsymbol{b}$:

$$\boldsymbol{z = \mathbf{w}^T \mathbf{x} + b.}$$

The probability of the positive class is then given by:

$$\boldsymbol{P(y = 1|\mathbf{x}) = \sigma(z) = \frac{1}{1 + e^{-(\mathbf{w}^T \mathbf{x} + b)}}.}$$

.

Conversely, the probability of the negative class is:

$$\boldsymbol{P(y = 0|\mathbf{x}) = 1 - \sigma(z).}$$

The decision boundary is the set of points where the model predicts a probability of 0.5 for both classes. For logistic regression, this boundary is a linear hyperplane defined by:

$$\boldsymbol{\mathbf{w}^T \mathbf{x} + b = 0}$$

Data points on one side of the boundary are classified as one class, and points on the other side are classified as the other class.

To train the logistic regression model, we need a cost function that measures how well the model's predictions match the actual labels. The cost function for logistic regression is derived from the likelihood of the data:

$$\boldsymbol{Cost(\mathbf{w}, b) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} log(\hat{y}^{(i)}) log(1 - \hat{(y)}^{(i)}) \right]}$$

where $\boldsymbol{m}$ represents the number of training examples, $\boldsymbol{y^{(i)}}$ means the actual label for the $\boldsymbol{i}$-th training example, $\boldsymbol{\hat{(y)}^{(i)}}$ is the predicted probability for the $\boldsymbol{i}$-th training example.

This cost function is also known as the binary cross-entropy or log-loss.

[1]Source: https://towardsai.net/p/machine-learning/logistic-regression-with-mathematics

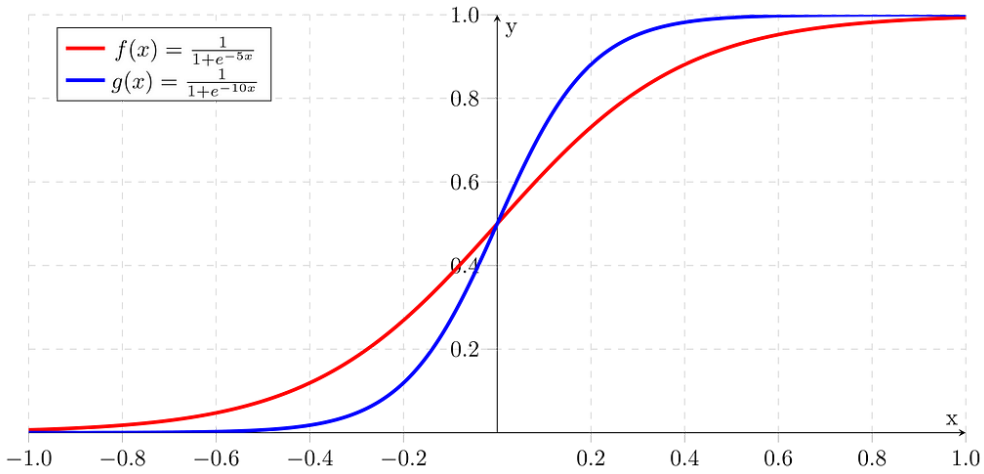


Figure A.2: Examples of Logistic function (Sigmoid function)

2. **Decision Tree (DT):** DT is a popular and intuitive method for classification tasks in ML. It works by recursively partitioning the data space and making classifications based on the majority class in each partition.

   A decision tree consists of three types of nodes:

   1. **Root Node (Decision node):** The topmost node where the first split occurs (typically represented by squares).
   2. **Internal Node (Chance node):** Nodes that represent decisions or tests on an attribute (typically represented by circles).
   3. **Leaf Node (Terminal/End Node):** Nodes that represent the final classification or decision (typically represented by triangles).

   Edge is the branch from one node to another, representing the outcome of a decision. And path is a sequence of edges from the root to a leaf node.

   The core of a decision tree algorithm is how it splits the data at each node. Some commonly used criteria are given below:

   – **Gini Impurity:** Measures the impurity of a node. It is calculated as:

   $$\boldsymbol{Gini(D) = 1 - \sum_{i=1}^{C} p_i^2, i \in \{1, 2, \ldots, C\}}$$

   where $\boldsymbol{p_i}$ is the proportion of samples belonging to class $\boldsymbol{i}$ in the dataset $\boldsymbol{D}$.

   – **Information Gain:** Based on the concept of entropy from information theory. The entropy of a dataset $\boldsymbol{D}$ is:

   $$\boldsymbol{Entropy(D) = -\sum_{i=1}^{C} p_i log_2(p_i).}$$

Information gain for a split is:

$$IG(D, A) = Entropy(D) - \sum_{v \in values(A)} \frac{|D_v|}{|D|} Entropy(D_v).$$

where D_v is the subset of D for which attribute A has value v.

– **Chi-Square:** Measures the statistical significance of a split.

Recursive Partitioning is the process of splitting the dataset into subsets, then recursively splitting those subsets.

Firstly, select the best attribute to split the data based on the chosen criterion (e.g., Gini impurity, information gain).

Then, split the data into subsets based on the selected attribute.

Finally, repeat the process for each subset until a stopping condition is met (e.g., maximum depth, minimum number of samples per node, or pure nodes). An example of two leaves DT is given in Fig.A.3[2].

Pruning helps to avoid overfitting by removing branches that have little importance. There are two main types of pruning:

1. **Pre-pruning (Early Stopping):** Stop growing the tree before it perfectly classifies the training set by setting constraints (e.g., maximum depth, minimum samples per leaf).
2. **Post-pruning (Pruning After Tree Construction):** Grow the tree to its maximum size and then remove nodes that do not provide significant improvement. For example: *Cost Complexity Pruning:* Minimize a cost complexity criterion that balances tree size and misclassification error.

DT is a powerful and interpretable method for classification tasks. By recursively splitting the data based on features, it creates a model that can make classifications based on the learned decision rules. Despite their tendency to over-fit, techniques like pruning and ensemble methods (e.g., random forests) can help improve their performance and robustness.

3. **Random Forest (RF):** As an ensemble ML method, RF builds multiple DTs and merges them to get a more accurate and stable prediction. Ensemble learning involves combining multiple models to produce a single stronger model. The idea is that by aggregating the predictions of several models, we can reduce variance and bias, leading to better overall performance.

   Random Forest is based on the bagging technique. Bagging involves training multiple models on different subsets of the same training data, created

[2]Source: https://towardsdatascience.com/the-complete-guide-to-decision-trees-28a4e3c7be14

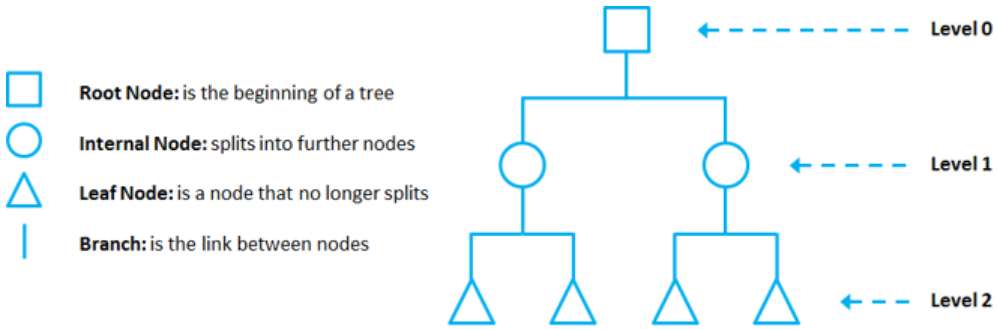


Figure A.3: An example of two leaves DT

by sampling with replacement (bootstrapping), and then aggregating their predictions.

A Random Forest classifier consists of many DTs. Each tree in the forest is built from a different subset of the training data and makes its own prediction. The final prediction is obtained by aggregating the predictions (voting) of all the trees. For classification, the final prediction is the mode (majority vote) of the predictions from all individual trees.

A step-by-step workflow for using Random Forest for classification can be seen below.

1. **Data Preparation:** Preprocess the data, including handling missing values, encoding categorical variables, and feature scaling if necessary.
2. **Splitting the Data:** Divide the dataset into training and testing sets.
3. **Model Initialization:** Initialize the Random Forest classifier with chosen hyperparameters.
4. **Training:** Train the Random Forest model on the training data.
5. **Prediction:** Use the trained model to make predictions on the test data.
5. **Evaluation:** Evaluate the model performance using metrics like accuracy, precision, recall, F1-score, and Receiver Operating Characteristics - Area Under the Curve (ROC-AUC).

Key hyperparameters that can be tuned to improve the performance of a Random Forest model include:

- The number of trees in the forest.
- The maximum number of features considered for splitting a node.
- The maximum depth of the tree.
- The minimum number of samples required to split an internal node.
- The minimum number of samples required to be at a leaf node.
- Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

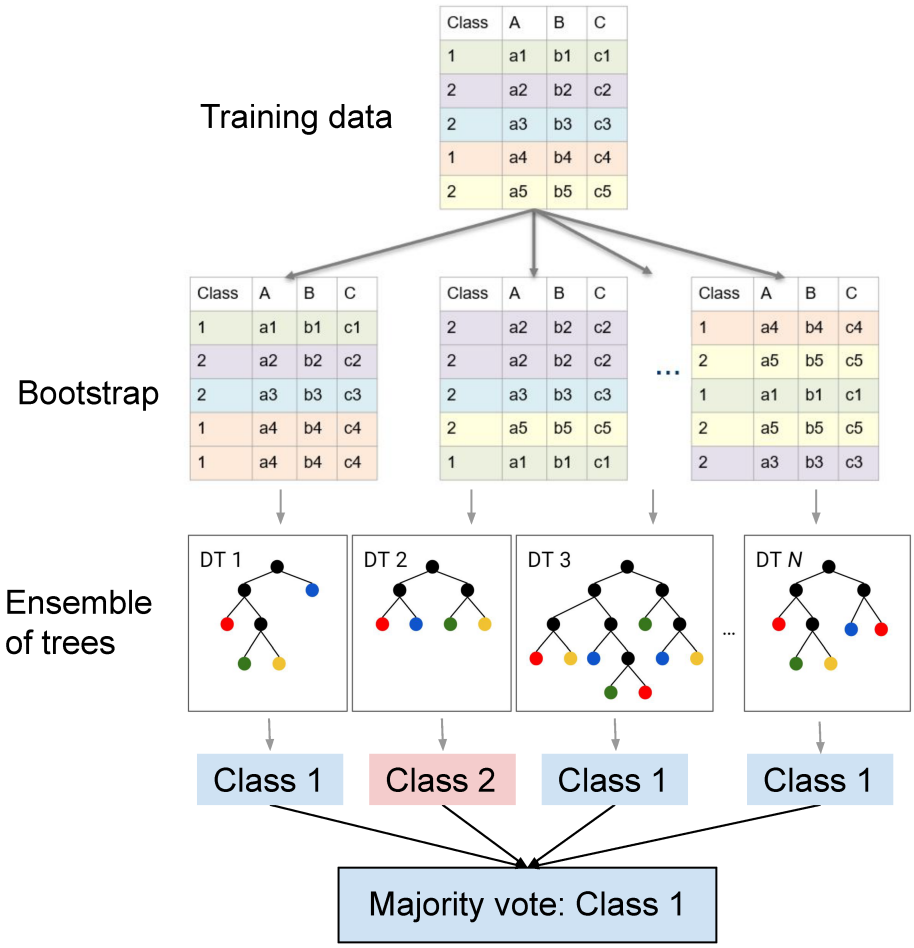


Figure A.4: An example of RF

The schematic structure of the RF algorithm is given in Fig.A.4[3]. RF is a powerful and versatile classification ML technique that combines the predictions of multiple DTs to improve accuracy and robustness.

4. **Support Vector Machine (SVM):** In ML, SVM is a powerful and widely used supervised learning method for classification tasks. SVM is particularly well-suited for binary classification and are known for their effectiveness in high-dimensional spaces.

   Support Vectors are the data points that are closest to the decision boundary (or hyperplane). These points are critical in defining the position and orientation of the hyperplane. Hyperplane is a decision boundary that separates different classes in the feature space. In a two-dimensional space, it's a line; in higher dimensions, it's a hyperplane. Margin represents the distance between

[3]Source: https://pages.cms.hu-berlin.de/EOL/geo_rs/S08_Image_classification2.html

the hyperplane and the nearest data points from either class. SVM aims to maximize this margin.

The primary goal of SVM is to find the hyperplane that best separates the classes by maximizing the margin between the classes' closest points. This results in a model that is more robust and less likely to over-fit.

For linearly separable data, SVM finds a hyperplane defined by $\mathbf{w}^T\mathbf{x}+\boldsymbol{b}=\mathbf{0}$, where $\mathbf{w}$ is the weight vector and $\boldsymbol{b}$ is the bias term. The decision rule for classifying a new point $\mathbf{x}$ is $\boldsymbol{f(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x} + b)}$.

The margin $\boldsymbol{M}$ is defined as the distance from the hyperplane to the closest support vectors. The optimization problem to maximize the margin is:

$$\min_{\mathbf{w},b} \frac{\mathbf{1}}{\mathbf{2}} \|\mathbf{w}\|^2 \tag{A.9}$$

$$\boldsymbol{s.t.}$$

$$\boldsymbol{y_i}(\mathbf{w}^T\mathbf{x}_i + \boldsymbol{b}) \geq \mathbf{1}, \forall \boldsymbol{i} \in \{\mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{n}\} \tag{A.10}$$

for all training samples $(\mathbf{x_i}, \boldsymbol{y_i})$.

When data is not linearly separable, SVM uses a technique called the kernel trick to map the data into a higher-dimensional space where a linear separator might exist. Common kernels include:

- **Linear Kernel:** $\boldsymbol{K}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$
- **Polynomial Kernel:** $\boldsymbol{K}(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T\mathbf{x}' + \boldsymbol{c})^d$
- **Radial Basis Function Kernel:** $\boldsymbol{K}(\mathbf{x}, \mathbf{x}') = \boldsymbol{exp}(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$
- **Sigmoid Kernel:** $\boldsymbol{K}(\mathbf{x}, \mathbf{x}') = \boldsymbol{tanh}(\alpha\mathbf{x}^T\mathbf{x}' + \boldsymbol{c})$

For data that is not perfectly separable, SVM uses a soft margin approach, introducing slack variables $\boldsymbol{\xi_i}$ to allow some misclassifications:

$$\boldsymbol{min}_{\mathbf{w},b,\xi} \frac{\mathbf{1}}{\mathbf{2}} \|\mathbf{w}\|^2 + \boldsymbol{C} \sum_{i=1}^{n} \boldsymbol{\xi}_i \tag{A.11}$$

$$\boldsymbol{s.t.}$$

$$\boldsymbol{y}^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + \boldsymbol{b}) \geq \mathbf{1} - \boldsymbol{\xi_i} \tag{A.12}$$

$$\boldsymbol{\xi_i} \geq \mathbf{0} \tag{A.13}$$

where $\boldsymbol{C}$ is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

SVMs are a robust classification technique capable of handling both linear and non-linear problems. SVMs are widely used due to their high accuracy

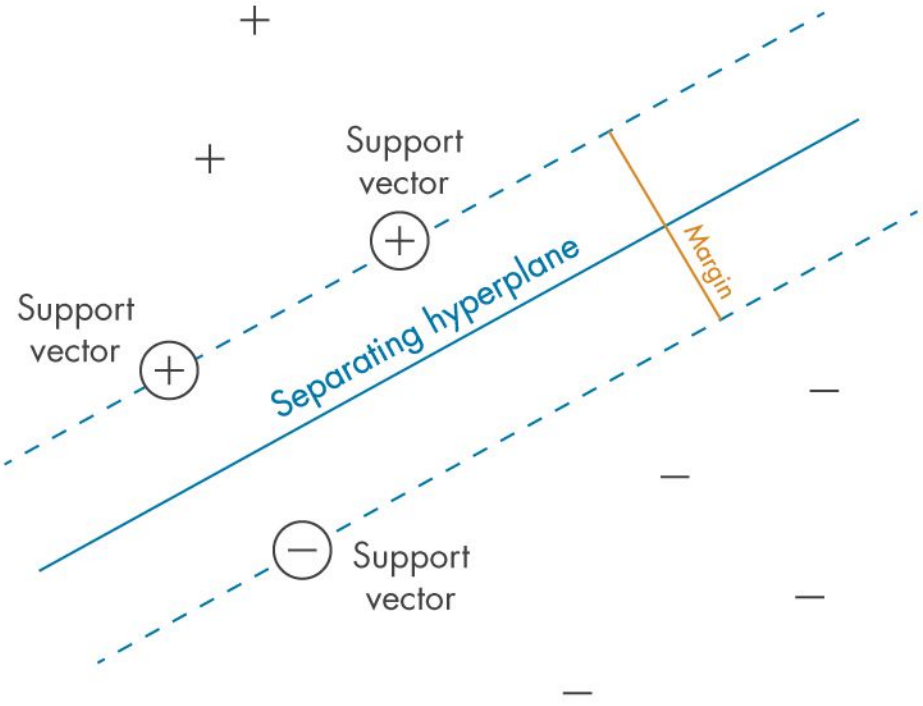


Figure A.5: An example of SVM: the criterion that SVM seeks to optimize.

and ability to handle complex datasets. Fig.A.5[4] shows how a SVM finds the optimal solution to a classification problem.

5. **K-Nearest Neighbor (KNN):** KNN algorithm is a simple, non-parametric classification algorithm that is easy to understand and implement. As an instance-based supervised learning algorithm, KNN does not explicitly learn a model. Instead, it makes decisions based on the instances in the training data. It works by finding the $\boldsymbol{K}$ nearest points in the training dataset and using their categories (classes) to predict the category of a new data point. It can handle complex data and is also easy to implement, which is why KNN has become a popular tool in the field of artificial intelligence.

   The value of $\boldsymbol{k}$ determines how many neighbors will be considered when making a classification decision. The algorithm identifies the $\boldsymbol{k}$ closest data points to the instance being classified.

   KNN relies on a distance metric to identify the nearest neighbors. Common distance metrics include:

[4]Source: https://fr.mathworks.com/discovery/support-vector-machine.html

– **Euclidean Distance:**

$$\boldsymbol{d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}} \tag{A.14}$$

– **Manhattan Distance (L1 norm):**

$$\boldsymbol{d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|} \tag{A.15}$$

– **Minkowski Distance:** Generalization of Euclidean and Manhattan distances:

$$\boldsymbol{d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}} \tag{A.16}$$

– **Cosine Similarity:** Measures the cosine of the angle between two vectors, used for text classification and other high-dimensional datasets.

The basic process (see in Fig.A.6[5]) of classification via KNN is given below:

1. **Determine the Value of neighbors:** Choose the number of neighbors $\boldsymbol{k}$.
2. **Compute Distances:** Calculate the distance between the instance to be classified and all instances in the training set.
3. **Identify Neighbors:** Identify the $\boldsymbol{k}$ instances in the training set that are closest to the instance being classified.
4. **Vote:** The class label of the new instance is determined by the majority class among the $\boldsymbol{k}$ nearest neighbors. This can be a simple majority vote or a weighted vote where closer neighbors have more influence.

The selection of $\boldsymbol{k}$ is critical for KNN classification. A small $\boldsymbol{k}$ may lead to a noisy decision boundary and overfitting; a large $\boldsymbol{k}$ can smooth out the decision boundary but might lead to underfitting. Therefore, $\boldsymbol{k}$ is often chosen as an odd number to avoid ties. Cross-validation can be used to select the optimal $\boldsymbol{k}$.

Since KNN relies on distance metrics, it is sensitive to the scale of the features. Therefore, it is important to normalize or standardize the features so that they have the same scale. Common techniques include:

– **Min-Max Scaling:** Rescale the feature to a fixed range, typically $\mathbf{[0, 1]}$.
– **Standardization:** Rescale the feature to have a mean of $\mathbf{0}$ and a standard deviation of $\mathbf{1}$.

[5] https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4

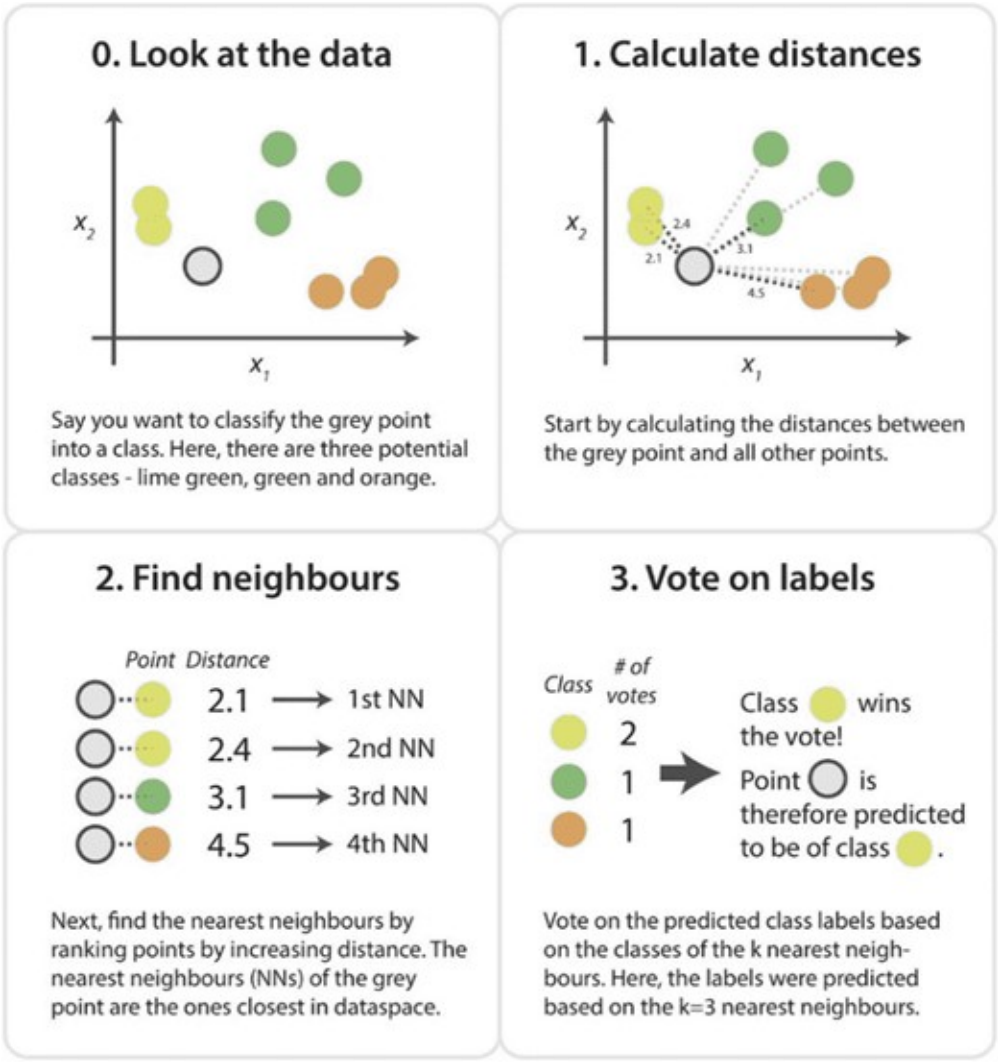


Figure A.6: An example of $\boldsymbol{k = 4}$ to visualize the process of KNN.

KNN is a simple yet effective classification technique that makes predictions based on the majority class of the nearest neighbors. While it is easy to implement and understand, it requires careful consideration of distance metrics, normalization, and the choice of k. Despite its computational and memory limitations, KNN remains a popular choice for a variety of classification tasks.

Commonly used classification algorithms in DL:

- **Neural Network (NN):** NNs are a class of models inspired by the human brain, capable of learning complex patterns and making predictions based on data. They are widely used for classification tasks due to their flexibility and ability to model non-linear relationships. Before the introduction of NN, Fig.A.7[6] visualizes the basic contents in NN learning process which will be discussed in the following.

  The basic unit of a NN, analogous to a biological neuron. It receives input, processes it, and produces an output. NN consists of connected units or nodes called artificial neurons, those neurons aggregated into three types of layers, input layer, hidden layer and output layer, respectively. Input Layer is the first layer that receives the input data. Hidden Layer(s) intermediate layers that process the inputs through weights and activation functions. And output layer is the final layer that produces the output prediction.

  Feedforward Neural Network (FNN) is the simplest type of neural network where connections between the nodes do not form cycles. Information moves in one direction from input to output. MLP is a type of FNN with one or more hidden layers.

  Forward propagation means the inputs are passed through the network layer by layer, applying weights, biases, and activation functions to compute the output. Backpropagation is the process of updating the network's weights and biases to minimize the loss function. It involves:

  - **Calculating Gradients:** Compute the gradient of the loss function with respect to each weight using the chain rule.
  - **Gradient Descent:** Update the weights using the gradients to minimize the loss.

    $$\boldsymbol{w} := \boldsymbol{w} - \eta \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{w}} \tag{A.17}$$

    where η is the learning rate.

  A NN activation function is a function that is applied to the output of a neuron. Activation functions introduce non-linearity into the network, allowing it to learn complex patterns. Common activation functions include:

[6] https://medium.com/data-science-365/overview-of-a-neural-networks-learning-process-61690a502fa

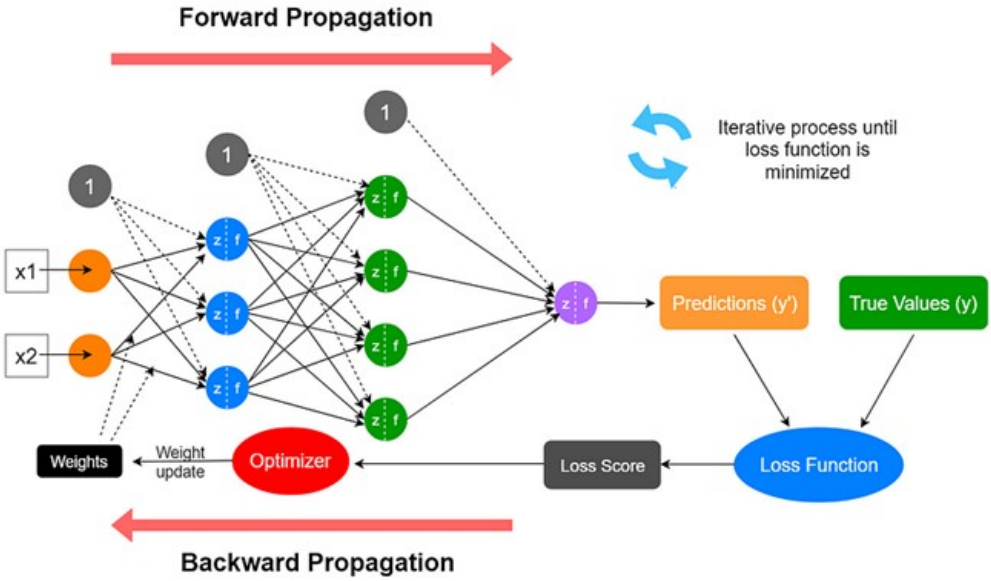


Figure A.7: Overview of a NN's learning process.

– **Sigmoid:** Outputs a value between **0** and **1**.

$$\boldsymbol{\sigma(x) = \frac{1}{1 + e^{-x}}} \tag{A.18}$$

– **Tanh:** Outputs a value between $\boldsymbol{-1}$ and **1**.

$$\boldsymbol{tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}} \tag{A.19}$$

– **Rectified Linear Unit (ReLU):** Outputs the input if positive, otherwise zero.

$$\boldsymbol{ReLU(x) = max(0, x)} \tag{A.20}$$

The loss function measures the discrepancy between the predicted output and the actual target. Cross-Entropy Loss are used for binary and multiclass classification.

$$\boldsymbol{L = -\sum_{i=1}^{N} y_i log(\hat{y}_i)} \tag{A.21}$$

where $\boldsymbol{y_i}$ is the true label and $\boldsymbol{\hat{y}_i}$ is the predicted probability.

NNs are powerful models capable of learning complex patterns for classification tasks. Despite their complexity, NNs are widely used due to their flexibility and ability to model intricate data relationships.

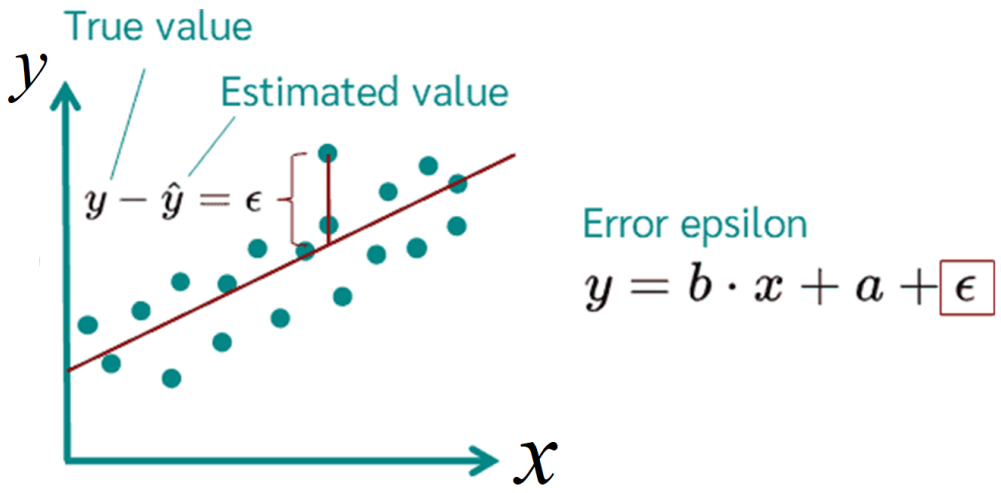


Figure A.8: Linear Regression with one independent variable.

A.3.1 AI-based prediction

In this subsection, some commonly used prediction algorithms introduced in two parts as what subsection A.3.0.1 did. What's more, for the algorithms which are applied for both classification and prediction will mainly focus on emphasizing the difference rather than repeating the algorithm.

Commonly used prediction algorithms in ML:

1. **Linear Regression LR:** As the simplest statistical regression technique in ML, LR provides a linear relationship between an independent variable and one or more independent variables. It's widely used for its simplicity and interpretability.

   Independent variable(s) ($\mathbf{x}$) represents the predictor variable(s) used to make predictions about the dependent variable. The outcome variable that we are trying to predict is called dependent variable ($\mathbf{y}$). Linear relationship (or linear association) is a statistical term used to describe a straight-line relationship between $\mathbf{x}$ and $\mathbf{y}$. An example of LR is given in Fig.A.8[7].

   For a simple linear regression with one independent variable, the model is represented as:

   $$\mathbf{y} = \boldsymbol{\beta_0} + \boldsymbol{\beta_1}\mathbf{x} + \boldsymbol{\varepsilon} \tag{A.22}$$

   $\boldsymbol{\beta_0}$ represents the intercept, $\boldsymbol{\beta_1}$ is the slope (coefficient) and $\boldsymbol{\varepsilon}$ means the error term.

   For multiple linear regression with $\boldsymbol{n}$ independent variables, the model is:

   $$\mathbf{y} = \boldsymbol{\beta_0} + \boldsymbol{\beta_1}\mathbf{x_1} + \cdots + \boldsymbol{\beta_n}\mathbf{x_n} + \boldsymbol{\varepsilon} \tag{A.23}$$

   In linear regression, the most common method for estimating the coefficients

[7]Source: https://datatab.net/tutorial/linear-regression

$(\boldsymbol{\beta_0}, \boldsymbol{\beta_1}, \ldots, \boldsymbol{\beta_n})$ is Ordinary Least Squares (OLS). OLS estimates the coefficients by minimizing the sum of the squared differences between the observed and predicted values:

$$\boldsymbol{min}_{\beta_0,\beta_1,\ldots,\beta_n} \sum_{i=1}^{n} (\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^2 \tag{A.24}$$

where $\boldsymbol{y_i}$ is the real value and $\boldsymbol{\hat{y}_i}$ is the predicted value for the $\boldsymbol{i}$-th observation.

In the following, some common metrics for evaluating a linear regression model are introduced in detail. For all the predictive algorithms, the evaluation metrics are the same. So, these evaluation metrics will not be repeated again in the following introduction.

- **R-squared (Coefficient of Determination):** Measures the proportion of variance in the dependent variable that is predictable from the independent variables.

$$\boldsymbol{R^2} = \mathbf{1} - \frac{\sum_{i=1}^{n} (\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^2}{\sum_{i=1}^{n} (\boldsymbol{y_i} - \boldsymbol{\bar{y}})^2} \tag{A.25}$$

  $\boldsymbol{\bar{y}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y_i}$ is the mean of the observed data.

- **Mean Absolute Error (MAE):** The average of the absolute errors.

$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{i=1}^{n} |\boldsymbol{y_i} - \boldsymbol{\hat{y}_i}| \tag{A.26}$$

- **Mean Squared Error (MSE):** The average of the squared errors.

$$\boldsymbol{MSE} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{i=1}^{n} (\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^2 \tag{A.27}$$

- **Root Mean Squared Error (RMSE):** The square root of the average of the squared errors.

$$\boldsymbol{RMSE} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{n}} \sum_{i=1}^{n} (\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^2} \tag{A.28}$$

To address overfitting and multicollinearity, regularization techniques like Ridge Regression (L2 regularization) and Lasso Regression (L1 regularization) can be used, where $\mathbf{x}$ is a $\boldsymbol{n}$ by $\boldsymbol{p}$ matrix with centered columns and $\mathbf{y}$ is a centered $\boldsymbol{n}$-vector.

- **Ridge Regression:** Adds the sum of squared coefficients to the loss

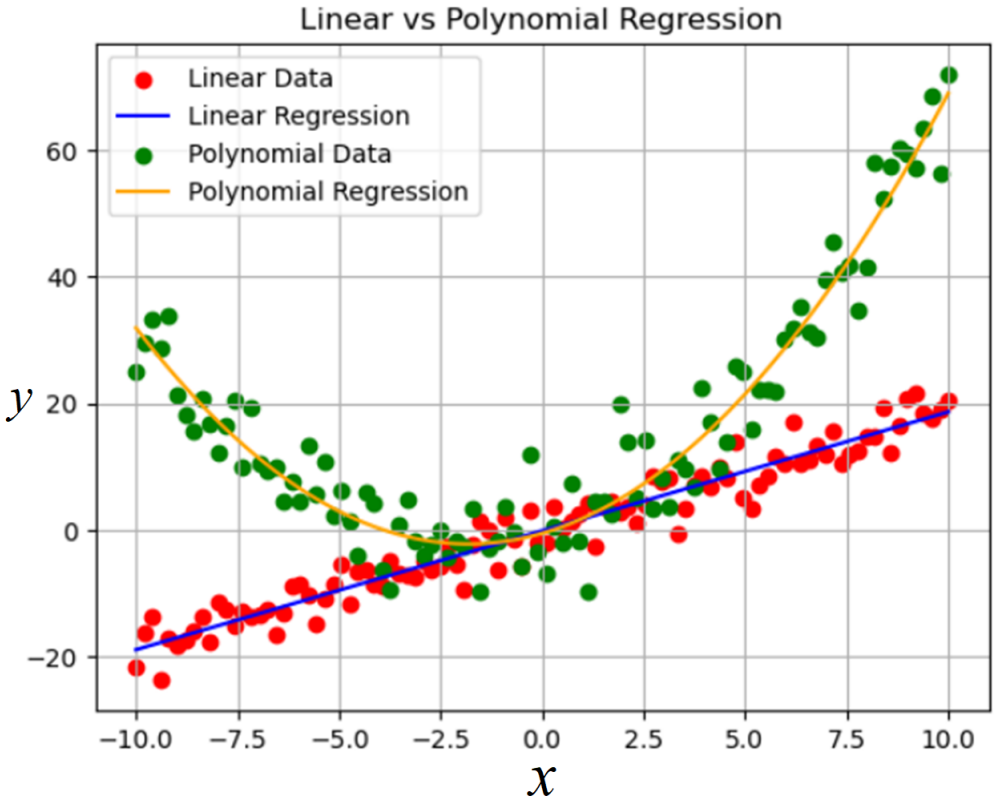


Figure A.9: The difference between Linear Regression and Polynomial Regression.

function.

$$L = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{A.29}$$

– **Lasso Regression:** Adds the sum of absolute values of coefficients to the loss function.

$$L = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{A.30}$$

Despite its simplicity, linear regression can be a powerful tool when used appropriately and with proper data preprocessing and validation techniques.

2. **Polynomial Regression PR:** PR is an extension of linear regression that allows to model the relationship between the independent and dependent variables as an n-degree polynomial (see in Fig.A.9[8]). This approach can capture more complex patterns in the data that a simple linear model might miss.

   PR extends linear regression by adding polynomial terms to the model, allowing for curved relationships. Degree of the Polynomial (n) determines the

[8]Source: https://tahera-firdose.medium.com/understanding-polynomial-regression-603eb25501d

highest power of the independent variable in the model. A n-degree polynomial regression model includes terms up to x^n.

For a single independent variable $\mathbf{x}$, a polynomial regression model of degree n is represented as:

$$\mathbf{y} = \beta_0 + \beta_1 \mathbf{x} + \beta_2 \mathbf{x}^2 + \cdots + \beta_n \mathbf{x}^n \varepsilon \tag{A.31}$$

where $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients, ε is the error term.

PR is essentially a linear model in terms of the transformed input features. The independent variable $\mathbf{x}$ is transformed into $\mathbf{x}, \mathbf{x}^2, \ldots, \mathbf{x}^n$ to fit a linear regression model on these transformed features.

Therefore, the model training of PR contains two steps. The first one is data transformation in which transform the original independent variables into polynomial features. For instance, if you have a single feature $\mathbf{x}$ and you want a n-degree polynomial, you transform it into $[\mathbf{x}, \mathbf{x}^2, \mathbf{x}^n]$. The second step is to fit a linear regression model on the transformed polynomial features.

High-degree polynomials can fit the training data very closely but perform poorly on new data which is called overfitting. Regularization techniques or choosing an appropriate degree n can mitigate this. Increasing the polynomial degree reduces bias but increases variance. It's crucial to find a balance to ensure good generalization to unseen data.

PR provides a flexible approach for modelling non-linear relationships by transforming the input features into polynomial features and applying linear regression. By carefully selecting the polynomial degree and using appropriate evaluation metrics, PR can offer improved predictive performance for complex datasets.

3. **Decision Trees (DT):** DT is a powerful tool for both classification and prediction. For the basic knowledge, it has already discussed in subsection A.3.0.1 which related to the classification tasks.

   The main difference between classification tree and regression tree is the target variable. For the classification tree is used when the target variable is categorical, and the regression tree is applied when the target variable is continuous.

   The criteria used to split the data at each node are also different for both classification and prediction. Different from Gini Impurity and Entropy for classification tasks, MSE is the common used criteria for prediction tasks.

4. **Random Forest RF:** RF is also feasible for both classification and prediction tasks. Hence, the introduction will mainly focus on the difference.

   As what we mentioned in subsection A.3.0.1, RF constructs a multitude of decision trees to vote for getting the final decisions. Therefore, instead of the classification tree the predictive RF are generated by multiple regression trees.

So that the predictive RF will predict a continuous numerical value for new instances based on past observations.

5. **Support Vector Regression SVR:** Unlike the classification SVM which intends to separate different categories by finding a plane or a hyperplane. SVR (which is a type of SVM algorithms and is commonly used for regression analysis) aims to find a function $\boldsymbol{f(\mathbf{x})}$ that deviates from the actual observed $\boldsymbol{y}$ by a value no greater than $\boldsymbol{\varepsilon}$ for each training point, while being as flat as possible.

   For a given training set $\boldsymbol{(\mathbf{x}_i, y_i)}$, the SVM model aims to solve:

$$\boldsymbol{min}_{\mathbf{w},b} \ \frac{\mathbf{1}}{\mathbf{2}} \|\mathbf{w}\|^{\mathbf{2}} \tag{A.32}$$

$$\boldsymbol{s.t.}$$

$$|y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle - b| \leq \varepsilon \tag{A.33}$$

   where $\langle \mathbf{w}, \mathbf{x} \rangle$ is the dot product of $\mathbf{w}$ and $\mathbf{x}$, $\boldsymbol{b}$ is the bias term.

   ε-insensitive loss function ignores errors that are within ε distance of the observed value by treating them as equal to zero. It is a loss function that is used in SVR:

$$\boldsymbol{L(y, f(\mathbf{x})) = max(0, |y - f(\mathbf{x}) - \varepsilon)} \tag{A.34}$$

   where $\boldsymbol{f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b}$.

   The optimization problem can be expressed as:

$$\boldsymbol{min}_{\mathbf{w},\xi,\hat{\xi}} \ \frac{\mathbf{1}}{\mathbf{2}} \|\mathbf{w}\|^{\mathbf{2}} + \boldsymbol{C} \sum_{i}^{N} (\xi_i + \hat{\xi}_i) \tag{A.35}$$

$$\boldsymbol{s.t.}$$

$$y_i \leq f(\mathbf{x}_i, \mathbf{w}) + \varepsilon + \xi_i \tag{A.36}$$

$$y_i \geq f(\mathbf{x}_i, \mathbf{w}) - \varepsilon - \hat{\xi}_i \tag{A.37}$$

$$\xi_i \geq \mathbf{0} \tag{A.38}$$

$$\hat{\xi}_i \geq \mathbf{0} \tag{A.39}$$

$$\boldsymbol{for\ i = 1, \ldots, N}$$

   where ε is a free parameter that serves as a threshold: all predictions have to be within an ε range of the true predictions. Slack variables $\boldsymbol{(\xi_i, \hat{\xi}_i)}$ are usually added into the above to allow for errors and to allow approximation in the case the above problem is infeasible. $\boldsymbol{C}$ is a regularization parameter that determines the trade-off between the flatness of $\boldsymbol{f(\mathbf{x})}$ and the amount up to which deviations larger than ε are tolerated.

6. **k-Nearest Neighbors KNN:** When used for prediction, KNN can help predict continuous values by considering the values of the k-nearest points. The

main goal of KNN regression is to predict the target value for a new data point based on the average of the target values of its k-nearest neighbors.

Commonly used prediction algorithms in DL:

- **Neural network (NN):** NNs are versatile models used for various tasks, including prediction (regression) and classification. Despite their similar structures, the objectives and approaches for prediction and classification tasks differ.

  First of all, the objects are different between the two tasks. For the classification, the goal of a NN classifier is to categorize inputs into discrete classes. But for NN prediction (regression), the goal is to predict continuous numerical values.

  Secondly, the output layers are different. In classification tasks, there are two cases. The first one is the binary classification tasks, which is often a single neuron in output layer with a sigmoid activation function that outputs a probability; the second one is the multi-class classification, which contains multiple neurons in output layer with a softmax activation function, where each neuron represents a class. Typically, NN in prediction (regression) tasks consist of a single neuron that outputs a continuous value. No activation function or a linear activation function is used in the output layer.

  And then, the commonly used loss functions are different. As for classification, Binary Cross-Entropy Loss is used to measure the difference between predicted probabilities and actual class labels in binary classification, and Categorical Cross-Entropy Loss is used in multi-class classification. As for prediction (regression), loss functions measure the difference between predicted and actual continuous values and the common loss functions are MSE, MAE.

  Fourth, the commonly used activation functions are different. For classification tasks, Sigmoid activation function is for binary classification in the output layer. And Softmax activation function is for multi-class classification in the output layer. In hidden layers, the activation functions can be ReLU, tanh, etc. For prediction (regression) tasks, linear or no activation function is applied in the output layer. Hidden layers can use ReLU, tanh, etc.

  Finally, turns to evaluation metrics. In classification tasks, evaluation metrics assess the accuracy of categorical predictions. Common metrics contain accuracy, precision, recall, F1 Score, ROC-AUC. As for prediction (regression), evaluation metrics assess the accuracy of continuous value predictions. The commonly used metrics include MSE, MAE, RMSE and R-squared.

A.4 Multiple Traveling Salesman Problem (mTSP)

The Multiple Traveling Salesman Problem (mTSP) is an extension of the classic Traveling Salesman Problem. It is a combinatorial optimization problem where

multiple salesmen are used to visit a set of cities, with the objective of minimizing the total travel cost while ensuring each city is visited exactly once by one salesman. Depot is the common starting and ending point for all salesmen. And in mTSP, there are multiple salesmen who start and end their tours at a common depot. Cities (also called Nodes) represents the locations that need to be visited once by one salesman. The objective of mTSP is to minimize the total travel cost, which can be the total distance, time, or other cost metrics.

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{V} = v_0, \ldots, v_n$, where node v_0 represents the depot and nodes 1 to n represent the cities and its corresponding index set is $N = \{0, \ldots, n\}$. And $\mathcal{E}$ is the set of edges. Associated with each edge $(i, j) \in \mathcal{E}$ is a cost (or distance) c_{ij} (travelling from node i to node j).

The problem considers m travelling salesmen ($k \in M = \{1, \ldots, m\}$) at the depot. We define a binary variable x_{ij} for each edge $(i, j) \in \mathcal{E}$:

$$x_{ij} = \begin{cases} 1 & \text{a tour goes from city i to city j} \\ 0 & \text{otherwise.} \end{cases} \tag{A.40}$$

For the subtour elimination constraints, we define an auxiliary variable u_i to denote the position of node i in a tour.

The problem can be descried by the following objective function and constrains.

Objective function:

$$min \sum_{k \in M} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{A.41}$$

The corresponding constrains are given below:

1. Each city must be visited exactly once:

$$\sum_{k \in M} \sum_{j \in N} x_{ij} = 1, \ \forall i \in N \tag{A.42}$$

2. Each salesman must depart from depot:

$$\sum_{j \in N} x_{0j} = m, \ \forall k \in M \tag{A.43}$$

3. Each salesman must return to the depot:

$$\sum_{j \in N} x_{j0} = m, \ \forall k \in M \tag{A.44}$$

4. **Flow conservation constraints**: Ensure that each city is visited and left by the same salesman:

$$\sum_{j \in N} x_{ij} = \sum_{j \in N} x_{ji}, \ \forall i \in N, \ \forall k \in M \tag{A.45}$$

5. Eliminate sub-tours using the Miller-Tucker-Zemlin (MTZ) constraints:

$$\begin{gathered} \boldsymbol{u_i - u_j + n \sum_{k \in M} x_{ij} \leq n - 1,\ \forall i \neq j,\ \forall i, j \in N} \\ \boldsymbol{1 \leq u_i \leq n,\ \forall i \in N} \end{gathered} \tag{A.46}$$

Appendix B

Coordinate transformation

Duo to the fact that the directions of Z axis are the same in three coordinate systems, the rotation matrices for rotations around $\boldsymbol{Z}$ axis are given as follows. $\boldsymbol{\theta}$ and $\boldsymbol{\theta'}$ are the angles between two $\boldsymbol{x}$ axis.

$$\boldsymbol{R_1} = \begin{bmatrix} \boldsymbol{\cos\theta} & \boldsymbol{\sin\theta} & \boldsymbol{0} \\ \boldsymbol{-\sin\theta} & \boldsymbol{\cos\theta} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}, \boldsymbol{R_2} = \begin{bmatrix} \boldsymbol{\cos\theta'} & \boldsymbol{\sin\theta'} & \boldsymbol{0} \\ \boldsymbol{-\sin\theta'} & \boldsymbol{\cos\theta'} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}$$

where $\boldsymbol{R_1}$ is the rotation matrix between frame $\boldsymbol{\Re_{base}}$ and $\boldsymbol{\Re_{Tel}}$, and $\boldsymbol{R_2}$ is between frame $\boldsymbol{\Re_{Tel}}$ and $\boldsymbol{\Re_{Tq}}$. The coordinate system conversion of frame $\boldsymbol{\Re_{base}}$ to $\boldsymbol{\Re_{Tel}}$ is given below:

$$\begin{bmatrix} \boldsymbol{x_{Tel}} \\ \boldsymbol{y_{Tel}} \\ \boldsymbol{z_{Tel}} \end{bmatrix} = \boldsymbol{R_1} \left(\begin{bmatrix} \boldsymbol{x_{Op}} \\ \boldsymbol{y_{Op}} \\ \boldsymbol{z_{Op}} \end{bmatrix} - \begin{bmatrix} \boldsymbol{x_{Tel}^{Op}} \\ \boldsymbol{y_{Tel}^{Op}} \\ \boldsymbol{z_{Tel}^{Op}} \end{bmatrix} \right) \tag{B.1}$$

where $\boldsymbol{(x_*, y_*, z_*)^T}$ is the coordinate of object in frame of $\boldsymbol{\Re_*}$ (base/OptiTrack, Tello, T-quad), $\begin{pmatrix} \boldsymbol{x_{Tel}^{Op}} & \boldsymbol{y_{Tel}^{Op}} & \boldsymbol{z_{Tel}^{Op}} \end{pmatrix}$ presents the coordinate of Tello ($\boldsymbol{CS_{0,1}^a}$) in frame $\boldsymbol{\Re_{base}}$. The conversion of frame $\boldsymbol{\Re_{Tel}}$ to frame $\boldsymbol{\Re_{Tq}}$ can be seen as follows:

$$\begin{bmatrix} \boldsymbol{x_{Tq}} \\ \boldsymbol{y_{Tq}} \\ \boldsymbol{z_{Tq}} \end{bmatrix} = \boldsymbol{R_2} \left(\begin{bmatrix} \boldsymbol{x_{Tel}} \\ \boldsymbol{y_{Tel}} \\ \boldsymbol{z_{Tel}} \end{bmatrix} - \begin{bmatrix} \boldsymbol{x_{Tq}^{Tel}} \\ \boldsymbol{y_{Tq}^{Tel}} \\ \boldsymbol{z_{Tq}^{Tel}} \end{bmatrix} \right) \tag{B.2}$$

Bibliography

[Aamir 2019] Muhammad Aamir and Syed Mustafa Ali Zaidi. *DDoS attack detection with feature engineering and machine learning: the framework and performance evaluation.* International Journal of Information Security, vol. 18, no. 6, pages 761–785, 2019. (Cited on page 79.)

[Adam 2020] George Adam, Ladislav Rampášek, Zhaleh Safikhani, Petr Smirnov, Benjamin Haibe-Kains and Anna Goldenberg. *Machine learning approaches to drug response prediction: challenges and recent progress.* NPJ precision oncology, vol. 4, no. 1, page 19, 2020. (Cited on page 26.)

[Adya 1998] Monica Adya and Fred Collopy. *How effective are neural networks at forecasting and prediction? A review and evaluation.* Journal of forecasting, vol. 17, no. 5-6, pages 481–495, 1998. (Cited on page 29.)

[Ahangar-Asr 2011] Alireza Ahangar-Asr, Asaad Faramarzi, N Mottaghifard and Akbar A Javadi. *Modeling of permeability and compaction characteristics of soils using evolutionary polynomial regression.* Computers & Geosciences, vol. 37, no. 11, pages 1860–1869, 2011. (Cited on page 27.)

[Ahmed 2018] Mohamed Ben Ahmed, Farah Zeghal Mansour and Mohamed Haouari. *Robust integrated maintenance aircraft routing and crew pairing.* Journal of Air Transport Management, vol. 73, pages 15–31, 2018. (Cited on page 32.)

[Albert 2000] Réka Albert, Hawoong Jeong and Albert-László Barabási. *Error and attack tolerance of complex networks.* nature, vol. 406, no. 6794, pages 378–382, 2000. (Cited on page 19.)

[Andrade 2022] Sequoia R. Andrade and Daniel E. Hulse. *Evaluation and Improvement of System-of-Systems Resilience in a Simulation of Wildfire Emergency Response.* IEEE Systems Journal, pages 1–12, 2022. (Cited on pages 15, 16 and 79.)

[Arif 2018] Anmar Arif, Shanshan Ma, Zhaoyu Wang, Jianhui Wang, Sarah M Ryan and Chen Chen. *Optimizing service restoration in distribution systems with uncertain repair time and demand.* IEEE Transactions on Power Systems, vol. 33, no. 6, pages 6828–6838, 2018. (Cited on pages 32, 33 and 155.)

[Axelsson 2018] Jakob Axelsson and Stina Nylander. *An analysis of systems-of-systems opportunities and challenges related to mobility in smart cities.* In 2018 13th Annual Conference on System of Systems Engineering (SoSE), pages 132–137. IEEE, 2018. (Cited on page 14.)

[Axelsson 2019] Jakob Axelsson. *Game theory applications in systems-of-systems engineering: A literature review and synthesis.* Procedia Computer Science, vol. 153, pages 154–165, 2019. (Cited on page 17.)

[Barlow 1992] Jewel B Barlow, Moghen M Monahemi and Dianne P O'Leary. *Constrained matrix Sylvester equations.* SIAM Journal on Matrix Analysis and Applications, vol. 13, no. 1, pages 1–9, 1992. (Cited on page 60.)

[Bebis 1994] George Bebis and Michael Georgiopoulos. *Feed-forward neural networks.* Ieee Potentials, vol. 13, no. 4, pages 27–31, 1994. (Cited on page 29.)

[Berge 1984] Claude Berge. Hypergraphs: combinatorics of finite sets, volume 45. Elsevier, 1984. (Cited on page 18.)

[Bhamra 2011] Ran Bhamra, Samir Dani and Kevin Burnard. *Resilience: the concept, a literature review and future directions.* International journal of production research, vol. 49, no. 18, pages 5375–5393, 2011. (Cited on page 19.)

[Bhatia 2010] Nitin Bhatia*et al.* *Survey of nearest neighbor techniques.* arXiv preprint arXiv:1007.0085, 2010. (Cited on page 25.)

[Bhattacharyya 2017] SP Bhattacharyya. *Robust control under parametric uncertainty: An overview and recent results.* Annual Reviews in Control, vol. 44, pages 45–77, 2017. (Cited on page 79.)

[Boccaletti 2014] Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang and Massimiliano Zanin. *The structure and dynamics of multilayer networks.* Physics reports, vol. 544, no. 1, pages 1–122, 2014. (Cited on page 52.)

[Bourne 2018] Mike Bourne, Monica Franco-Santos, Pietro Micheli and Andrey Pavlov. *Performance measurement and management: a system of systems perspective.* International Journal of Production Research, vol. 56, no. 8, pages 2788–2799, 2018. (Cited on page 14.)

[Box 2015] George EP Box, Gwilym M Jenkins, Gregory C Reinsel and Greta M Ljung. Time series analysis: forecasting and control. John Wiley & Sons, 2015. (Cited on page 30.)

[Boyd 2022] Myron Boyd, Thomas Mazzuchi and Shahram Sarkani. *Coevolutionary Network Model for Efficient Collaboration in Systems-of-Systems.* In 2022 17th Annual System of Systems Engineering Conference (SOSE), pages 113–118, 2022. (Cited on page 80.)

[Cao 2003] Li-Juan Cao and Francis Eng Hock Tay. *Support vector machine with adaptive parameters in financial time series forecasting.* IEEE Transactions on neural networks, vol. 14, no. 6, pages 1506–1518, 2003. (Cited on page 31.)

[Carvalho 2003] João Carvalho, Karabi Datta and Yoopyo Hong. *A new block algorithm for full-rank solution of the Sylvester-observer equation.* IEEE Transactions on Automatic Control, vol. 48, no. 12, pages 2223–2228, 2003. (Cited on page 60.)

[Carvalho 2007] Pedro M. S. Carvalho, Fernando J. D. Carvalho and Luis A. F. M. Ferreira. *Dynamic Restoration of Large-Scale Distribution Network Contingencies: Crew Dispatch Assessment.* In 2007 IEEE Lausanne Power Tech, pages 1453–1457, 2007. (Cited on page 32.)

[Cervantes 2020] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua and Asdrubal Lopez. *A comprehensive survey on support vector machine classification: Applications, challenges and trends.* Neurocomputing, vol. 408, pages 189–215, 2020. (Cited on page 25.)

[Chapman 2014] Airlie Chapman, Marzieh Nabi-Abdolyousefi and Mehran Mesbahi. *Controllability and observability of network-of-networks via Cartesian products.* IEEE Transactions on Automatic Control, vol. 59, no. 10, pages 2668–2679, 2014. (Cited on page 52.)

[Chen 2017] Guanrong Chen. *Pinning control and controllability of complex dynamical networks.* International Journal of Automation and Computing, vol. 14, pages 1–9, 2017. (Cited on page 55.)

[Chen 2018] Bo Chen, Zhigang Ye, Chen Chen, Jianhui Wang, Tao Ding and Zhaohong Bie. *Toward a synthetic model for distribution system restoration and crew dispatch.* IEEE Transactions on Power Systems, vol. 34, no. 3, pages 2228–2239, 2018. (Cited on pages 32, 33 and 155.)

[Choi 2018] Tsan-Ming Choi. *A system of systems approach for global supply chain management in the big data era.* IEEE Engineering Management Review, vol. 46, no. 1, pages 91–97, 2018. (Cited on pages 15 and 16.)

[Cigizoglu 2006] Hikmet Kerem Cigizoglu and Murat Alp. *Generalized regression neural network in modelling river sediment yield.* Advances in Engineering Software, vol. 37, no. 2, pages 63–68, 2006. (Cited on page 30.)

[Cimellaro 2010] Gian Paolo Cimellaro, Andrei M Reinhorn and Michel Bruneau. *Framework for analytical quantification of disaster resilience.* Engineering structures, vol. 32, no. 11, pages 3639–3649, 2010. (Cited on page 20.)

[Clauset 2017] Aaron Clauset, Daniel B Larremore and Roberta Sinatra. *Data-driven predictions in the science of science.* Science, vol. 355, no. 6324, pages 477–480, 2017. (Cited on page 30.)

[Cutler 2007] D Richard Cutler, Thomas C Edwards Jr, Karen H Beard, Adele Cutler, Kyle T Hess, Jacob Gibson and Joshua J Lawler. *Random forests*

for classification in ecology. Ecology, vol. 88, no. 11, pages 2783–2792, 2007. (Cited on page 25.)

[Cutler 2012] Adele Cutler, D Richard Cutler and John R Stevens. *Random forests.* Ensemble machine learning: Methods and applications, pages 157–175, 2012. (Cited on page 28.)

[Datta 2011] Karabi Datta and Mohan Thapa. *Unique Full-Rank Solution of the Sylvester-Observer Equation and Its Application to State Estimation in Control Design.* Numerical Linear Algebra in Signals, Systems and Control, pages 185–200, 2011. (Cited on page 60.)

[Davendralingam 2013] Navindran Davendralingam and Daniel DeLaurentis. *A robust optimization framework to architecting system of systems.* Procedia Computer Science, vol. 16, pages 255–264, 2013. (Cited on page 14.)

[De Ville 2013] Barry De Ville. *Decision trees.* Wiley Interdisciplinary Reviews: Computational Statistics, vol. 5, no. 6, pages 448–455, 2013. (Cited on page 27.)

[DeLaurentis 2011] Daniel A DeLaurentis, William A Crossley and Muharrem Mane. *Taxonomy to guide systems-of-systems decision-making in air transportation problems.* Journal of Aircraft, vol. 48, no. 3, pages 760–770, 2011. (Cited on page 52.)

[Desaulniers 1998] Guy Desaulniers, Jacques Desrosiers, Irina loachim, Marius M Solomon, François Soumis and Daniel Villeneuve. *A unified framework for deterministic time constrained vehicle routing and crew scheduling problems.* In Fleet management and logistics, pages 57–93. Springer, 1998. (Cited on page 32.)

[Díaz-Ramírez 2014] Jenny Díaz-Ramírez, José Ignacio Huertas and Federico Trigos. *Aircraft maintenance, routing, and crew scheduling planning for airlines with a single fleet and a single maintenance and crew base.* Computers & Industrial Engineering, vol. 75, pages 68–78, 2014. (Cited on page 32.)

[DiMario 2009] Michael J DiMario, John T Boardman and Brian J Sauser. *System of systems collaborative formation.* IEEE Systems Journal, vol. 3, no. 3, pages 360–368, 2009. (Cited on page 4.)

[Dong 2011] Wenjie Dong. *Tracking control of multiple-wheeled mobile robots with limited information of a desired trajectory.* IEEE transactions on robotics, vol. 28, no. 1, pages 262–268, 2011. (Cited on page 96.)

[Drexl 2013] Michael Drexl, Julia Rieck, Thomas Sigl and Bettina Press. *Simultaneous vehicle and crew routing and scheduling for partial-and full-load long-distance road transport.* Business Research, vol. 6, pages 242–264, 2013. (Cited on page 32.)

[Duque 2016] Pablo A Maya Duque, Irina S Dolinskaya and Kenneth Sörensen. *Network repair crew scheduling and routing for emergency relief distribution problem.* European Journal of Operational Research, vol. 248, no. 1, pages 272–285, 2016. (Cited on page 32.)

[Elshenawy 2018] Mohamed Elshenawy, Baher Abdulhai and Mohamed El-Darieby. *Towards a service-oriented cyber–physical systems of systems for smart city mobility applications.* Future Generation Computer Systems, vol. 79, pages 575–587, 2018. (Cited on pages 14 and 15.)

[Eusgeld 2011] Irene Eusgeld, Cen Nan and Sven Dietz. *"System-of-systems" approach for interdependent critical infrastructures.* Reliability Engineering & System Safety, vol. 96, no. 6, pages 679–686, 2011. (Cited on page 14.)

[Fan 2019] Chao Fan and Ali Mostafavi. *Metanetwork framework for performance analysis of disaster management system-of-systems.* IEEE Systems Journal, vol. 14, no. 1, pages 1265–1276, 2019. (Cited on page 80.)

[Farahnakian 2013] Fahimeh Farahnakian, Tapio Pahikkala, Pasi Liljeberg and Juha Plosila. *Energy aware consolidation algorithm based on k-nearest neighbor regression for cloud data centers.* In 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, pages 256–259. IEEE, 2013. (Cited on page 29.)

[Fawagreh 2014] Khaled Fawagreh, Mohamed Medhat Gaber and Eyad Elyan. *Random forests: from early developments to recent advancements.* Systems Science & Control Engineering: An Open Access Journal, vol. 2, no. 1, pages 602–609, 2014. (Cited on page 24.)

[Fortino 2021] Giancarlo Fortino, Claudio Savaglio, Giandomenico Spezzano and MengChu Zhou. *Internet of Things as System of Systems: A Review of Methodologies, Frameworks, Platforms, and Tools.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 51, no. 1, pages 223–236, 2021. (Cited on page 14.)

[Fraccascia 2018] Luca Fraccascia, Ilaria Giannoccaro and Vito Albino. *Resilience of complex systems: State of the art and directions for future research.* Complexity, vol. 2018, no. 1, page 3421529, 2018. (Cited on page 20.)

[Francis 1976] Bruce A Francis and Walter Murray Wonham. *The internal model principle of control theory.* Automatica, vol. 12, no. 5, pages 457–465, 1976. (Cited on page 60.)

[Francis 2014] Royce Francis and Behailu Bekera. *A metric and frameworks for resilience analysis of engineered and infrastructure systems.* Reliability engineering & system safety, vol. 121, pages 90–103, 2014. (Cited on page 20.)

[Fratello 2018] Michele Fratello, Roberto Tagliaferri*et al. Decision trees and random forests.* Encyclopedia of bioinformatics and computational biology: ABC of bioinformatics, vol. 1, no. S 3, 2018. (Cited on page 24.)

[Ge 2014] Bingfeng Ge, Keith W Hipel, Liping Fang, Kewei Yang and Yingwu Chen. *An interactive portfolio decision analysis approach for system-of-systems architecting using the graph model for conflict resolution.* IEEE Transactions on systems, man, and cybernetics: systems, vol. 44, no. 10, pages 1328–1346, 2014. (Cited on page 17.)

[Gianetto 2013] David A Gianetto and Babak Heydari. *Catalysts of cooperation in system of systems: The role of diversity and network structure.* IEEE Systems Journal, vol. 9, no. 1, pages 303–311, 2013. (Cited on page 14.)

[Gopalakrishnan 2018] T Gopalakrishnan, Ritesh Choudhary and Sarada Prasad. *Prediction of sales value in online shopping using linear regression.* In 2018 4th International Conference on Computing Communication and Automation (ICCCA), pages 1–6. IEEE, 2018. (Cited on page 27.)

[Gorod 2008] Alex Gorod, Brian Sauser and John Boardman. *System-of-Systems Engineering Management: A Review of Modern History and a Path Forward.* IEEE Systems Journal, vol. 2, no. 4, pages 484–499, 2008. (Cited on pages 51 and 52.)

[Grigorian 2019] Mark Grigorian, Abdolreza S Moghadam, Hadiseh Mohammadi and Mozhgan Kamizi. *Methodology for developing earthquake-resilient structures.* The Structural Design of Tall and Special Buildings, vol. 28, no. 2, page e1571, 2019. (Cited on page 20.)

[Grigoroudis 2013] Evangelos Grigoroudis and Yannis A Phillis. *Modeling healthcare system-of-systems: A mathematical programming approach.* IEEE Systems Journal, vol. 7, no. 4, pages 571–580, 2013. (Cited on pages 15 and 16.)

[Guan 2016] Yongqiang Guan, Zhijian Ji, Lin Zhang and Long Wang. *Controllability of heterogeneous multi-agent systems under directed and weighted topology.* International Journal of Control, vol. 89, no. 5, pages 1009–1024, 2016. (Cited on pages 21, 22, 52 and 55.)

[Guan 2017] Yongqiang Guan, Zhijian Ji, Lin Zhang and Long Wang. *Controllability of multi-agent systems under directed topology.* International Journal of Robust and Nonlinear Control, vol. 27, no. 18, pages 4333–4347, 2017. (Cited on pages 21 and 22.)

[Guo 2016] Jin-Li Guo, Qi Suo, Ai-Zhong Shen and Jeffrey Forrest. *The evolution of hyperedge cardinalities and bose-Einstein condensation in hypernetworks.* Scientific reports, vol. 6, no. 1, pages 1–8, 2016. (Cited on page 174.)

[Guo 2022] Junhao Guo, Zhijian Ji, Yungang Liu and Chong Lin. *Unified understanding and new results of controllability model of multi-agent systems.* International Journal of Robust and Nonlinear Control, no. 11, page 32, 2022. (Cited on pages 22 and 52.)

[Haimes 2009] Yacov Y Haimes. *On the definition of resilience in systems.* Risk Analysis: An International Journal, vol. 29, no. 4, 2009. (Cited on page 19.)

[Hammann 2010] Felix Hammann, Heike Gutmann, Nadine Vogt, Christoph Helma and Juergen Drewe. *Prediction of adverse drug reactions using decision tree modeling.* Clinical Pharmacology & Therapeutics, vol. 88, no. 1, pages 52–59, 2010. (Cited on page 27.)

[Han 2012a] Seung Yeob Han, Karen Marais and Daniel DeLaurentis. *Evaluating system of systems resilience using interdependency analysis.* In 2012 IEEE international conference on systems, man, and cybernetics (SMC), pages 1251–1256. IEEE, 2012. (Cited on page 4.)

[Han 2012b] Shui Hua Han, Shui Xiu Lu and Stephen CH Leung. *Segmentation of telecom customers based on customer value by decision tree model.* Expert Systems with Applications, vol. 39, no. 4, pages 3964–3973, 2012. (Cited on page 24.)

[Hand 2012] David J Hand. *Assessing the performance of classification methods.* International Statistical Review, vol. 80, no. 3, pages 400–414, 2012. (Cited on page 80.)

[Hao 2018] Yuqing Hao, Zhisheng Duan and Guanrong Chen. *Further on the controllability of networked MIMO LTI systems.* International Journal of Robust and Nonlinear Control, vol. 28, no. 5, pages 1778–1788, 2018. (Cited on pages 22, 52, 53 and 55.)

[Hao 2022] Yuqing Hao, Qingyun Wang, Zhisheng Duan and Guanrong Chen. *Target Controllability of Networked LTI Systems.* IEEE Transactions on Network Science and Engineering, vol. 9, no. 3, pages 1493–1500, 2022. (Cited on page 55.)

[Hata 2009] Yutaka Hata, Syoji Kobashi and Hiroshi Nakajima. *Human health care system of systems.* IEEE Systems Journal, vol. 3, no. 2, pages 231–238, 2009. (Cited on pages 15 and 16.)

[Haupt 2018] Sue Ellen Haupt, Jim Cowie, Seth Linden, Tyler McCandless, Branko Kosovic and Stefano Alessandrini. *Machine learning for applied weather prediction.* In 2018 IEEE 14th international conference on e-science (e-Science), pages 276–277. IEEE, 2018. (Cited on page 26.)

[Heil 2020] Julia Heil, Kirsten Hoffmann and Udo Buscher. *Railway crew scheduling: Models, methods and applications.* European journal of operational research, vol. 283, no. 2, pages 405–425, 2020. (Cited on page 32.)

[Hela 2021] Kadri Hela, Othman Lakhal, Blaise Conrard and Rochdi Merzouki. *Formal approach to SoS management design.* In 2021 16th International Conference of System of Systems Engineering (SoSE), pages 138–143. IEEE, 2021. (Cited on page 17.)

[Hespanha 2018] Joao P Hespanha. Linear systems theory. Princeton university press, 2018. (Cited on pages 21, 52, 56, 176 and 178.)

[Hosseini 2016] Seyedmohsen Hosseini, Kash Barker and Jose E Ramirez-Marquez. *A review of definitions and measures of system resilience.* Reliability Engineering & System Safety, vol. 145, pages 47–61, 2016. (Cited on page 19.)

[Hou 2016] Baoyu Hou, Xiang Li and Guanrong Chen. *Structural Controllability of Temporally Switching Networks.* IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 63, no. 10, pages 1771–1781, 2016. (Cited on page 52.)

[Huang 2007] Cheng-Lung Huang, Mu-Chen Chen and Chieh-Jen Wang. *Credit scoring with a data mining approach based on support vector machines.* Expert systems with applications, vol. 33, no. 4, pages 847–856, 2007. (Cited on page 25.)

[Ishibuchi 1999] Hisao Ishibuchi, Tomoharu Nakashima and Tadahiko Murata. *Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems.* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 29, no. 5, pages 601–618, 1999. (Cited on page 80.)

[Ismail 2018] Zool Hilmi Ismail, Nohaidda Sariff and E Gorrostieta Hurtado. *A survey and analysis of cooperative multi-agent robot systems: challenges and directions.* Applications of Mobile Robots, vol. 5, pages 8–14, 2018. (Cited on page 21.)

[Jamshidi 2008] MO Jamshidi. *System of systems engineering-new challenges for the 21st century.* IEEE Aerospace and Electronic Systems Magazine, vol. 23, no. 5, pages 4–19, 2008. (Cited on pages 3 and 13.)

[Jaradat 2017] Raed Jaradat, Frank Adams, Sawsan Abutabenjeh and Charles Keating. *The complementary perspective of system of systems in collaboration, integration, and logistics: a value-chain based paradigm of supply chain management.* Systems, vol. 5, no. 4, page 50, 2017. (Cited on page 52.)

[Jena 2022] Biswajit Jena, Gopal Krishna Nayak and Sanjay Saxena. *Convolutional neural network and its pretrained models for image classification and object*

detection: A survey. Concurrency and Computation: Practice and Experience, vol. 34, no. 6, page e6767, 2022. (Cited on page 24.)

[Jeong 2016] Jig Han Jeong, Jonathan P Resop, Nathaniel D Mueller, David H Fleisher, Kyungdahm Yun, Ethan E Butler, Dennis J Timlin, Kyo-Moon Shim, James S Gerber, Vangimalla R Reddy*et al. Random forests for global and regional crop yield predictions.* PloS one, vol. 11, no. 6, page e0156571, 2016. (Cited on page 28.)

[Ji 2021] Xingquan Ji, Ziyang Yin, Yumin Zhang, Bo Xu*et al. Real-time autonomous dynamic reconfiguration based on deep learning algorithm for distribution network.* Electric Power Systems Research, vol. 195, page 107132, 2021. (Cited on page 20.)

[Jiang 2017] Jun Jiang, Pengjian Shang, Zuoquan Zhang and Xuemei Li. *Permutation entropy analysis based on Gini–Simpson index for financial time series.* Physica A: Statistical Mechanics and its Applications, vol. 486, pages 273–283, 2017. (Cited on pages 31 and 123.)

[Jiang 2022] Jun Jiang, Othman Lakhal and Rochdi Merzouki. *A Resilience Implementation Framework of System-of-Systems based on Hypergraph Model.* In 2022 17th Annual System of Systems Engineering Conference (SOSE), pages 487–492. IEEE, 2022. (Cited on pages 6, 18, 35, 52, 53, 54, 55, 79, 80 and 81.)

[Jiang 2023a] Jun Jiang, Yiwen Chen, Othman Lakhal and Rochdi Merzouki. *AI-based SoS performance classification for resilience reaction.* In 2023 18th Annual System of Systems Engineering Conference (SoSe), pages 1–6. IEEE, 2023. (Cited on pages vii, 6, 80, 85, 86 and 96.)

[Jiang 2023b] Yuting Jiang, Chengdi Wang and Shengtao Zhou. *Artificial intelligence-based risk stratification, accurate diagnosis and treatment prediction in gynecologic oncology.* In Seminars in cancer biology. Elsevier, 2023. (Cited on page 24.)

[Jiang 2024a] Jun Jiang, Yiwen Chen, Othman Lakhal and Rochdi Merzouki. *Controllability of heterogeneous multi-agent systems via cooperative output regulation.* Journal of the Franklin Institute, vol. 361, no. 15, page 107133, 2024. (Cited on page 6.)

[Jiang 2024b] Jun Jiang, Yiwen Chen, Othman Lakhal and Rochdi Merzouki. *Event-triggered Reconfiguration Scheduling of Hypergraph-based System-of-systems: the Resilience Reaction.* In 2024 IEEE International Systems Conference (SysCon), pages 1–7, 2024. (Cited on page 18.)

[Jiang 2024c] Jun Jiang, Yiwen Chen, Othman Lakhal and Rochdi Merzouki. *System-of-Systems Resilient Recovery Time Estimation via Heterogeneous*

Capacity Distribution. In 2024 19th Annual System of Systems Engineering Conference (SoSE), pages 288–293. IEEE, 2024. (Cited on page 6.)

[Kasem 2024] Mahmoud SalahEldin Kasem, Mohamed Hamada and Islam Taj-Eddin. *Customer profiling, segmentation, and sales prediction using AI in direct marketing*. Neural Computing and Applications, vol. 36, no. 9, pages 4995–5005, 2024. (Cited on page 24.)

[Kavousi-Fard 2018] Abdollah Kavousi-Fard, Alireza Zare and Amin Khodaei. *Effective dynamic scheduling of reconfigurable microgrids*. IEEE Transactions on Power Systems, vol. 33, no. 5, pages 5519–5530, 2018. (Cited on page 20.)

[Khalil 2012a] Wissam Khalil, Rochdi Merzouki, Belkacem Ould-Bouamama and Hafid Haffaf. *Hypergraph models for system of systems supervision design*. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 42, no. 4, pages 1005–1012, 2012. (Cited on pages 18 and 80.)

[Khalil 2012b] Wissam Khalil, Rochdi Merzouki, Belkacem Ould-Bouamama and Hafid Haffaf. *Hypergraph Models for System of Systems Supervision Design*. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 42, no. 4, pages 1005–1012, 2012. (Cited on page 51.)

[Khalil 2015] Wissam Khalil, Ahmad Koubeissi, Rochdi Merzouki, Blaise Conrard and B Ould-Bouamama. *Contribution to system of systems modeling*. In 2015 10th System of Systems Engineering Conference (SoSE), pages 182–186. IEEE, 2015. (Cited on page 18.)

[Kim 2018] Sungwoo Kim, Youngchul Shin, Gyu M Lee and Ilkyeong Moon. *Network repair crew scheduling for short-term disasters*. Applied Mathematical Modelling, vol. 64, pages 510–523, 2018. (Cited on page 32.)

[Kong 2022] Zhi Kong, Lianqian Cao, Lifu Wang and Ge Guo. *Controllability of Heterogeneous Networked Systems With Nonidentical Inner-Coupling Matrices*. IEEE Transactions on Control of Network Systems, vol. 9, no. 2, pages 867–878, 2022. (Cited on pages 23, 52 and 53.)

[Kotsiantis 2013] Sotiris B Kotsiantis. *Decision trees: a recent overview*. Artificial Intelligence Review, vol. 39, pages 261–283, 2013. (Cited on page 24.)

[Kraft 1977] John Kraft and Arthur Kraft. *Determinants of common stock prices: A time series analysis*. The journal of finance, vol. 32, no. 2, pages 417–425, 1977. (Cited on page 31.)

[Kubera 2010] Yoann Kubera, Philippe Mathieu and Sébastien Picault. *Everything can be Agent!* 2010. (Cited on page 52.)

[Kumar 2017] Pushpendra Kumar, Rochdi Merzouki and Belkacem Ould Bouamama. *Multilevel modeling of system of systems*. IEEE transactions on

systems, man, and cybernetics: systems, vol. 48, no. 8, pages 1309–1320, 2017. (Cited on page 17.)

[Kumar 2018] Pushpendra Kumar, Rochdi Merzouki and Belkacem Ould Bouamama. *Multilevel Modeling of System of Systems*. IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 48, no. 8, pages 1309–1320, 2018. (Cited on page 51.)

[Lautenbacher 2006] Conrad C Lautenbacher. *The global earth observation system of systems: Science serving society*. Space Policy, vol. 22, no. 1, pages 8–11, 2006. (Cited on page 14.)

[LeCun 2015] Yann LeCun, Yoshua Bengio and Geoffrey Hinton. *Deep learning*. nature, vol. 521, no. 7553, pages 436–444, 2015. (Cited on page 26.)

[Lee 2019] Oon Ling Lee, Rick Im Tay, Shing Tsair Too and Alex Gorod. *A smart city transportation system of systems governance framework: a case study of Singapore*. In 2019 14th Annual Conference System of Systems Engineering (SoSE), pages 37–42. IEEE, 2019. (Cited on pages 14 and 15.)

[Lei 2019] Shunbo Lei, Chen Chen, Yupeng Li and Yunhe Hou. *Resilient disaster recovery logistics of distribution systems: Co-optimize service restoration with repair crew and mobile power source dispatch*. IEEE Transactions on Smart Grid, vol. 10, no. 6, pages 6187–6202, 2019. (Cited on pages 32 and 33.)

[Lei 2020] Shunbo Lei, Chen Chen, Yue Song and Yunhe Hou. *Radiality constraints for resilient reconfiguration of distribution systems: Formulation and application to microgrid formation*. IEEE Transactions on Smart Grid, vol. 11, no. 5, pages 3944–3956, 2020. (Cited on page 20.)

[Li 2014] Wenchao Li, Ping Yi, Yue Wu, Li Pan and Jianhua Li. *A new intrusion detection system based on KNN classification algorithm in wireless sensor network*. Journal of Electrical and Computer Engineering, vol. 2014, no. 1, page 240217, 2014. (Cited on page 26.)

[Li 2018] Yi Li, Changfu Zou, Maitane Berecibar, Elise Nanini-Maury, Jonathan C-W Chan, Peter Van den Bossche, Joeri Van Mierlo and Noshin Omar. *Random forest regression for online capacity estimation of lithium-ion batteries*. Applied energy, vol. 232, pages 197–210, 2018. (Cited on page 28.)

[Li 2020] Shuanglin Li, Zujun Ma and Kok Lay Teo. *A new model for road network repair after natural disasters: Integrating logistics support scheduling with repair crew scheduling and routing activities*. Computers & Industrial Engineering, vol. 145, page 106506, 2020. (Cited on page 32.)

[Li 2021] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng and Jun Zhou. *A survey of convolutional neural networks: analysis, applications, and prospects*. IEEE

transactions on neural networks and learning systems, vol. 33, no. 12, pages 6999–7019, 2021. (Cited on page 29.)

[Linkov 2013] Igor Linkov, Daniel A Eisenberg, Kenton Plourde, Thomas P Seager, Julia Allen and Alex Kott. *Resilience metrics for cyber systems*. Environment Systems and Decisions, vol. 33, pages 471–476, 2013. (Cited on page 20.)

[Linkov 2018] Igor Linkov, Benjamin D Trump, Elke Anklam, David Berube, Patrick Boisseasu, Christopher Cummings, Scott Ferson, Marie-Valentine Florin, Bernard Goldstein, Danail Hristozov*et al.* *Comparative, collaborative, and integrative risk governance for emerging technologies*. Environment Systems and Decisions, vol. 38, pages 170–176, 2018. (Cited on page 20.)

[Liu 2023] Bo Liu, Mengjie Hu, Junjie Huang, Yin Chen and Housheng Su. *Controllability of Multi-Agent Systems With Cartesian Product Networks*. IEEE Transactions on Circuits and Systems I: Regular Papers, pages 1–13, 2023. (Cited on page 23.)

[Long 2018] Mingkang Long, Housheng Su and Bo Liu. *Group controllability of two-time-scale multi-agent networks*. Journal of the Franklin Institute, vol. 355, no. 13, pages 6045–6061, 2018. (Cited on page 23.)

[Long 2020] Mingkang Long, Housheng Su and Bo Liu. *Group controllability of two-time-scale discrete-time multi-agent systems*. Journal of the Franklin Institute, vol. 357, no. 6, pages 3524–3540, 2020. (Cited on page 23.)

[Lou 2018] Yang Lou, Lin Wang and Guanrong Chen. *Toward Stronger Robustness of Network Controllability: A Snapback Network Model*. IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 9, pages 2983–2991, 2018. (Cited on page 52.)

[Lu 2020] Zehuan Lu, Zhijian Ji and Zhiqiang Zhang. *Sampled-data based structural controllability of multi-agent systems with switching topology*. Journal of the Franklin Institute, vol. 357, no. 15, pages 10886–10899, 2020. (Cited on page 22.)

[Lughofer 2009] Edwin Lughofer, James E Smith, Muhammad Atif Tahir, Praminda Caleb-Solly, Christian Eitzinger, Davy Sannen and Marnix Nuttin. *Human–machine interaction issues in quality control based on online image classification*. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 39, no. 5, pages 960–971, 2009. (Cited on page 24.)

[Ma 2015] Qian Ma, Shengyuan Xu, Frank L Lewis, Baoyong Zhang and Yun Zou. *Cooperative output regulation of singular heterogeneous multiagent systems*. IEEE Transactions on Cybernetics, vol. 46, no. 6, pages 1471–1475, 2015. (Cited on pages 53 and 60.)

[Madni 2014] Azad M Madni and Michael Sievers. *System of systems integration: Key considerations and challenges.* Systems Engineering, vol. 17, no. 3, pages 330–347, 2014. (Cited on page 3.)

[Mahulkar 2009] Vishal Mahulkar, Shawn McKay, Douglas E Adams and Alok R Chaturvedi. *System-of-systems modeling and simulation of a ship environment with wireless and intelligent maintenance technologies.* IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 39, no. 6, pages 1255–1270, 2009. (Cited on page 17.)

[Maier 1998] Mark W Maier. *Architecting principles for systems-of-systems.* Systems Engineering: The Journal of the International Council on Systems Engineering, vol. 1, no. 4, pages 267–284, 1998. (Cited on pages 14, 48 and 52.)

[Maji 2019] Srabanti Maji and Srishti Arora. *Decision tree algorithms for prediction of heart disease.* In Information and Communication Technology for Competitive Strategies: Proceedings of Third International Conference on ICTCS 2017, pages 447–454. Springer, 2019. (Cited on page 28.)

[Malucelli 2019] Federico Malucelli and Emanuele Tresoldi. *Delay and disruption management in local public transportation via real-time vehicle and crew rescheduling: a case study.* Public Transport, vol. 11, no. 1, pages 1–25, 2019. (Cited on page 32.)

[Martín 2010] Luis Martín, Luis F Zarzalejo, Jesus Polo, Ana Navarro, Ruth Marchante and Marco Cony. *Prediction of global solar irradiance based on time series analysis: Application to solar thermal power plants energy production planning.* Solar Energy, vol. 84, no. 10, pages 1772–1781, 2010. (Cited on page 31.)

[Marvin 2014] Joseph W Marvin and Robert K Garrett Jr. *Quantitative SoS architecture modeling.* Procedia Computer Science, vol. 36, pages 41–48, 2014. (Cited on page 17.)

[Maulud 2020] Dastan Maulud and Adnan M Abdulazeez. *A review on linear regression comprehensive in machine learning.* Journal of Applied Science and Technology Trends, vol. 1, no. 2, pages 140–147, 2020. (Cited on page 27.)

[Mebarki 2017a] Ahmed Mebarki. *Resilience: Theory and metrics–A metal structure as demonstrator.* Engineering Structures, vol. 138, pages 425–433, 2017. (Cited on pages 4 and 20.)

[Mebarki 2017b] Ahmed Mebarki. *Safety of atmospheric industrial tanks: Fragility, resilience and recovery functions.* Journal of Loss Prevention in the Process Industries, vol. 49, pages 590–602, 2017. (Cited on pages 4 and 20.)

[Medsker 1999] Larry Medsker and Lakhmi C Jain. Recurrent neural networks: design and applications. CRC press, 1999. (Cited on page 29.)

[Mei 2020] Xueyan Mei, Hao-Chih Lee, Kai-yue Diao, Mingqian Huang, Bin Lin, Chenyu Liu, Zongyu Xie, Yixuan Ma, Philip M Robson, Michael Chung*et al.* *Artificial intelligence–enabled rapid diagnosis of patients with COVID-19.* Nature medicine, vol. 26, no. 8, pages 1224–1228, 2020. (Cited on page 24.)

[Mienye 2019] Ibomoiye Domor Mienye, Yanxia Sun and Zenghui Wang. *Prediction performance of improved decision tree-based algorithms: a review.* Procedia Manufacturing, vol. 35, pages 698–703, 2019. (Cited on page 28.)

[Mostafavi 2011] Ali Mostafavi, Dulcy M Abraham, Daniel DeLaurentis and Joseph Sinfield. *Exploring the dimensions of systems of innovation analysis: A system of systems framework.* IEEE Systems Journal, vol. 5, no. 2, pages 256–265, 2011. (Cited on page 3.)

[Nawroth 2006] Andreas P Nawroth and Joachim Peinke. *Multiscale reconstruction of time series.* Physics Letters A, vol. 360, no. 2, pages 234–237, 2006. (Cited on page 31.)

[Nazir 2015] Sajid Nazir, Hassan Hamdoun and Jafar Alzubi. *Cyber attack challenges and resilience for smart grids.* European Journal of Scientific Research, 2015. (Cited on page 20.)

[Nowicki 2010] David Nowicki, Wesley S Randall and Alex Gorod. *A framework for performance based logistics: A system of systems approach.* In International Congress on Ultra Modern Telecommunications and Control Systems, pages 681–692. IEEE, 2010. (Cited on pages 15 and 16.)

[Ostertagová 2012] Eva Ostertagová. *Modelling using polynomial regression.* Procedia engineering, vol. 48, pages 500–506, 2012. (Cited on page 27.)

[Pang 2024] Kaiyuan Pang, Chongyu Wang, Nikos D. Hatziargyriou and Fushuan Wen. *Dynamic Restoration of Active Distribution Networks by Coordinated Repair Crew Dispatch and Cold Load Pickup.* IEEE Transactions on Power Systems, vol. 39, no. 2, pages 4699–4713, 2024. (Cited on pages 32 and 33.)

[Park 2013] Jeryang Park, Thomas P Seager, Palakurth Suresh Chandra Rao, Matteo Convertino and Igor Linkov. *Integrating risk and resilience approaches to catastrophe management in engineering systems.* Risk analysis, vol. 33, no. 3, pages 356–367, 2013. (Cited on page 19.)

[Patton 1993] Ron Patton. *Robustness issues in fault-tolerant control.* In IEE Colloquium on Fault Diagnosis and Control System Reconfiguration, pages 1–1. IET, 1993. (Cited on page 79.)

[Peterson 2009] Leif E Peterson. *K-nearest neighbor.* Scholarpedia, vol. 4, no. 2, page 1883, 2009. (Cited on page 25.)

[Plocoste 2019] Thomas Plocoste, Rudy Calif and Sandra Jacoby-Koaly. *Multi-scale time dependent correlation between synchronous measurements of ground-level ozone and meteorological parameters in the Caribbean Basin.* Atmospheric Environment, vol. 211, pages 234–246, 2019. (Cited on page 123.)

[Prati 2011] Ronaldo C Prati, Gustavo EAPA Batista and Maria Carolina Monard. *A survey on graphical methods for classification predictive performance evaluation.* IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 11, pages 1601–1618, 2011. (Cited on page 80.)

[Pumpuni-Lenss 2017] Gloria Pumpuni-Lenss, Timothy Blackburn and Andreas Garstenauer. *Resilience in complex systems: an agent-based approach.* Systems Engineering, vol. 20, no. 2, pages 158–172, 2017. (Cited on page 20.)

[Raghavendra 2006] Cauligi S Raghavendra, Krishna M Sivalingam and Taieb Znati. Wireless sensor networks. Springer, 2006. (Cited on page 21.)

[Rahmani 2009] Amirreza Rahmani, Meng Ji, Mehran Mesbahi and Magnus Egerstedt. *Controllability of multi-agent systems from a graph-theoretic perspective.* SIAM Journal on Control and Optimization, vol. 48, no. 1, pages 162–186, 2009. (Cited on page 21.)

[Rajagopal 2015] Gayathri Rajagopal and Ramamoorthy Palaniswamy. *Performance evaluation of multimodal multifeature authentication system using KNN classification.* The Scientific World Journal, vol. 2015, no. 1, page 762341, 2015. (Cited on page 80.)

[Ray 2018] Ruchira Ray, Prakhar Khandelwal and B Baranidharan. *A survey on stock market prediction using artificial intelligence techniques.* In 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT), pages 594–598. IEEE, 2018. (Cited on page 26.)

[Rehman 2021] Abdul Rehman, Nadeem Akhtar and Omar H Alhazmi. *Formal Modeling, Proving, and Model Checking of a Flood Warning, Monitoring, and Rescue System-of-Systems.* Scientific Programming, vol. 2021, no. 1, page 6685978, 2021. (Cited on pages 15 and 16.)

[Rodriguez-Galiano 2012] Victor Francisco Rodriguez-Galiano, Bardan Ghimire, John Rogan, Mario Chica-Olmo and Juan Pedro Rigol-Sanchez. *An assessment of the effectiveness of a random forest classifier for land-cover classification.* ISPRS journal of photogrammetry and remote sensing, vol. 67, pages 93–104, 2012. (Cited on page 25.)

[Saritas 2019] Mucahid Mustafa Saritas and Ali Yasar. *Performance analysis of ANN and Naive Bayes classification algorithm for data classification.* International Journal of Intelligent Systems and Applications in Engineering, vol. 7, no. 2, pages 88–91, 2019. (Cited on page 80.)

[Schlechtingen 2011] Meik Schlechtingen and Ilmar Ferreira Santos. *Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection.* Mechanical systems and signal processing, vol. 25, no. 5, pages 1849–1875, 2011. (Cited on page 30.)

[Seber 2012] George AF Seber and Alan J Lee. Linear regression analysis. John Wiley & Sons, 2012. (Cited on page 27.)

[Shajin 2023] Francis H Shajin, Salini P, Paulthurai Rajesh and Venu Kadur Nagoji Rao. *Efficient framework for brain tumour classification using hierarchical deep learning neural network classifier.* Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, vol. 11, no. 3, pages 750–757, 2023. (Cited on page 26.)

[Sharma 2016] Himani Sharma, Sunil Kumar*et al. A survey on decision tree algorithms of classification in data mining.* International Journal of Science and Research (IJSR), vol. 5, no. 4, pages 2094–2097, 2016. (Cited on page 24.)

[Sheffi 2005] Yossi Sheffi. The resilient enterprise: overcoming vulnerability for competitive advantage. Pearson Education India, 2005. (Cited on page 19.)

[Silva 2020] Rafael de Amorim Silva and Rosana T. Vaccare Braga. *Simulating Systems-of-Systems With Agent-Based Modeling: A Systematic Literature Review.* IEEE Systems Journal, vol. 14, no. 3, pages 3609–3617, 2020. (Cited on page 80.)

[Sohrabpour 2021] Vahid Sohrabpour, Pejvak Oghazi, Reza Toorajipour and Ali Nazarpour. *Export sales forecasting using artificial intelligence.* Technological Forecasting and Social Change, vol. 163, page 120480, 2021. (Cited on page 26.)

[Song 2017] Yunsheng Song, Jiye Liang, Jing Lu and Xingwang Zhao. *An efficient instance selection algorithm for k nearest neighbor regression.* Neurocomputing, vol. 251, pages 26–34, 2017. (Cited on page 29.)

[Sousa-Poza 2008] Andres Sousa-Poza, Samuel Kovacic and Charles Keating. *System of systems engineering: an emerging multidiscipline.* International Journal of System of Systems Engineering, vol. 1, no. 1-2, pages 1–17, 2008. (Cited on pages 3 and 13.)

[Soyez 2017] Jean-Baptiste Soyez, Gildas Morvan, Rochdi Merzouki and Daniel Dupont. *Multilevel Agent-Based Modeling of System of Systems.* IEEE Systems Journal, vol. 11, no. 4, pages 2084–2095, 2017. (Cited on pages 17, 51 and 80.)

[Su 2011] Youfeng Su and Jie Huang. *Cooperative output regulation of linear multi-agent systems.* IEEE Transactions on Automatic Control, vol. 57, no. 4, pages 1062–1066, 2011. (Cited on page 60.)

[Suksomboon 2022] Patitta Suksomboon and Amarita Ritthipakdee. *Performance Comparison Classification using k-Nearest Neighbors and Random Forest Classification Techniques*. In 2022 3rd International Conference on Big Data Analytics and Practices (IBDAP), pages 43–46, 2022. (Cited on page 80.)

[Tanner 2004] H.G. Tanner. *On the controllability of nearest neighbor interconnections*. In 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601), volume 3, pages 2467–2472 Vol.3, 2004. (Cited on pages 21 and 52.)

[Thacker 2017] Scott Thacker, Raghav Pant and Jim W Hall. *System-of-systems formulation and disruption analysis for multi-scale critical national infrastructures*. Reliability Engineering & System Safety, vol. 167, pages 30–41, 2017. (Cited on page 53.)

[Tian 2021] Lingling Tian, Yongqiang Guan and Long Wang. *Controllability and observability of multi-agent systems with general linear dynamics under switching topologies*. International Journal of Control, vol. 94, no. 5, pages 1355–1367, 2021. (Cited on page 22.)

[Tran 2016] Binh Tran, Bing Xue and Mengjie Zhang. *Genetic programming for feature construction and selection in classification on high-dimensional data*. Memetic Computing, vol. 8, pages 3–15, 2016. (Cited on page 79.)

[Tsilipanos 2012] Kosmas Tsilipanos, Ioannis Neokosmidis and Dimitris Varoutas. *A system of systems framework for the reliability assessment of telecommunications networks*. IEEE Systems Journal, vol. 7, no. 1, pages 114–124, 2012. (Cited on page 52.)

[Uday 2014] Payuna Uday and Karen B Marais. *Resilience-based system importance measures for system-of-systems*. Procedia Computer Science, vol. 28, pages 257–264, 2014. (Cited on page 14.)

[Urban 2018] Gregor Urban, Kevin Bache, Duc TT Phan, Agua Sobrino, Alexander K Shmakov, Stephanie J Hachey, Christopher CW Hughes and Pierre Baldi. *Deep learning for drug discovery and cancer research: Automated analysis of vascularization images*. IEEE/ACM transactions on computational biology and bioinformatics, vol. 16, no. 3, pages 1029–1035, 2018. (Cited on page 26.)

[Uslar 2019] Mathias Uslar, Sebastian Rohjans, Christian Neureiter, Filip Pröstl Andrén, Jorge Velasquez, Cornelius Steinbrink, Venizelos Efthymiou, Gianluigi Migliavacca, Seppo Horsmanheimo, Helfried Brunner*et al. Applying the smart grid architecture model for designing and validating system-of-systems in the power and energy domain: A European perspective*. Energies, vol. 12, no. 2, page 258, 2019. (Cited on page 15.)

[Van Der Woude 2019] Jacob Van Der Woude, Christian Commault and Taha Boukhobza. *A dynamic graph characterisation of the fixed part of the controllable subspace of a linear structured system.* Systems & Control Letters, vol. 129, pages 17–25, 2019. (Cited on page 52.)

[Van Hentenryck 2015] Pascal Van Hentenryck and Carleton Coffrin. *Transmission system repair and restoration.* Mathematical Programming, vol. 151, pages 347–373, 2015. (Cited on page 32.)

[Vanfretti 2020] Luigi Vanfretti and VS Narasimham Arava. *Decision tree-based classification of multiple operating conditions for power system voltage stability assessment.* International Journal of Electrical Power & Energy Systems, vol. 123, page 106251, 2020. (Cited on page 24.)

[Vugrin 2010] Eric D Vugrin, Drake E Warren, Mark A Ehlen and R Chris Camphouse. *A framework for assessing the resilience of infrastructure and economic systems.* Sustainable and resilient critical infrastructure systems: Simulation, modeling, and intelligent engineering, pages 77–116, 2010. (Cited on page 19.)

[Wang 2016] Lin Wang, Guanrong Chen, Xiaofan Wang and Wallace KS Tang. *Controllability of networked MIMO systems.* Automatica, vol. 69, pages 405–409, 2016. (Cited on pages 22, 52, 53 and 55.)

[Wang 2020] Xiaomin Wang, Yuqing Hao and Qingyun Wang. *On the controllability of corona product network.* Journal of the Franklin Institute, vol. 357, no. 10, pages 6228–6240, 2020. (Cited on pages 21 and 23.)

[Wang 2024] Yi Wang, Dawei Qiu, Fei Teng and Goran Strbac. *Towards Microgrid Resilience Enhancement via Mobile Power Sources and Repair Crews: A Multi-Agent Reinforcement Learning Approach.* IEEE Transactions on Power Systems, vol. 39, no. 1, pages 1329–1345, 2024. (Cited on pages 32 and 33.)

[Wauters 2014] Mathieu Wauters and Mario Vanhoucke. *Support vector machine regression for project control forecasting.* Automation in Construction, vol. 47, pages 92–106, 2014. (Cited on page 29.)

[Weigend 2018] Andreas S Weigend. Time series prediction: forecasting the future and understanding the past. Routledge, 2018. (Cited on page 30.)

[Woldaregay 2019] Ashenafi Zebene Woldaregay, Eirik Årsand, Ståle Walderhaug, David Albers, Lena Mamykina, Taxiarchis Botsis and Gunnar Hartvigsen. *Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes.* Artificial intelligence in medicine, vol. 98, pages 109–134, 2019. (Cited on page 30.)

[Wu 2007] Ai-Guo Wu, Guang-Ren Duan and Yu Xue. *Kronecker maps and Sylvester-polynomial matrix equations.* IEEE Transactions on Automatic Control, vol. 52, no. 5, pages 905–910, 2007. (Cited on page 60.)

[Wu 2020] Jie-Ning Wu, Xiang Li and Guanrong Chen. *Controllability of Deep-Coupling Dynamical Networks.* IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 12, pages 5211–5222, 2020. (Cited on pages vii, 22, 52, 53 and 54.)

[Wu 2023] Jie-Ning Wu, Xiang Li and Guanrong Chen. *Controllability of Multilayer Snapback Networks.* IEEE Transactions on Control of Network Systems, vol. 10, no. 1, pages 15–25, 2023. (Cited on pages vii, 53 and 54.)

[Wudka 2020] Björn Wudka, Carsten Thomas, Lennart Siefke and Volker Sommer. *A reconfiguration approach for open adaptive systems-of-systems.* In 2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pages 219–222. IEEE, 2020. (Cited on page 13.)

[Xiang 2019] Linying Xiang, Peiru Wang, Fei Chen and Guanrong Chen. *Controllability of directed networked MIMO systems with heterogeneous dynamics.* IEEE Transactions on Control of Network Systems, vol. 7, no. 2, pages 807–817, 2019. (Cited on pages 23, 52, 53 and 55.)

[Xu 2022] Yong Xu, Zheng-Guang Wu and Ya-Jun Pan. *Observer-based dynamic event-triggered adaptive control of distributed networked systems with application to ground vehicles.* IEEE Transactions on Industrial Electronics, vol. 70, no. 4, pages 4148–4157, 2022. (Cited on page 21.)

[Yaghmaie 2016] Farnaz Adib Yaghmaie, Frank L Lewis and Rong Su. *Output regulation of linear heterogeneous multi-agent systems via output and state feedback.* Automatica, vol. 67, pages 157–164, 2016. (Cited on pages 53 and 60.)

[Yang 2009] Hairong Yang and Dayong Luo. *Regional Crew Scheduling Problem in Public Transit Based on a Modified Ant Colony Algorithm.* In International Conference on Transportation Engineering 2009, pages 2713–2718, 2009. (Cited on page 32.)

[Yao 2006] Zizhen Yao and Walter L Ruzzo. *A regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data.* In BMC bioinformatics, volume 7, pages 1–11. Springer, 2006. (Cited on page 29.)

[Yılmaz 2019] Nevriye Yılmaz and Boran Sekeroglu. *Student performance classification using artificial intelligence techniques.* In International Conference on Theory and Application of Soft Computing, Computing with Words and Perceptions, pages 596–603. Springer, 2019. (Cited on page 24.)

[Zaw 2009] Wint Thida Zaw and Thinn Thu Naing. *Modeling of Rainfall Prediction over Myanmar Using Polynomial Regression.* In 2009 International Conference on Computer Engineering and Technology, volume 1, pages 316–320, 2009. (Cited on page 27.)

[Zhang 2013] Shuo Zhang, Ming Cao and M Kanat Camlibel. *Upper and lower bounds for controllable subspaces of networks of diffusively coupled agents.* IEEE Transactions on Automatic control, vol. 59, no. 3, pages 745–750, 2013. (Cited on pages 22 and 52.)

[Zhang 2018] Hongliang Zhang, Lingyang Song, Zhu Han and Yingjun Zhang. Hypergraph theory in wireless communication networks. Springer, 2018. (Cited on pages 18 and 174.)

[Zhang 2020a] Fan Zhang and Lauren J O'Donnell. *Support vector regression.* In Machine learning, pages 123–140. Elsevier, 2020. (Cited on pages 28 and 29.)

[Zhang 2020b] Gang Zhang, Feng Zhang, Xin Zhang, Ke Meng and Zhao Yang Dong. *Sequential disaster recovery model for distribution systems with co-optimization of maintenance and restoration crew dispatch.* IEEE Transactions on Smart Grid, vol. 11, no. 6, pages 4700–4713, 2020. (Cited on pages 32 and 33.)

[Zhang 2021a] Xiufeng Zhang and Jian Sun. *Almost equitable partitions and controllability of leader-follower multi-agent systems.* Automatica, vol. 131, page 109740, 2021. (Cited on page 52.)

[Zhang 2021b] Yuan Zhang, Yuanqing Xia and Di-Hua Zhai. *Structural controllability of networked relative coupling systems.* Automatica, vol. 128, page 109547, 2021. (Cited on page 55.)

[Zhao 2009] Huimin Zhao, Atish P Sinha and Wei Ge. *Effects of feature construction on classification performance: An empirical study in bank failure prediction.* Expert Systems with Applications, vol. 36, no. 2, pages 2633–2644, 2009. (Cited on page 79.)

[Zhao 2015] Chen Zhao, Wen-Xu Wang, Yang-Yu Liu and Jean-Jacques Slotine. *Intrinsic dynamics induce global symmetry in network controllability.* Scientific reports, vol. 5, no. 1, page 8422, 2015. (Cited on pages 22, 52 and 53.)

[Zhao 2018] Bo Zhao, Xiangjin Wang, Da Lin, Madison M Calvin, Julia C Morgan, Ruwen Qin and Caisheng Wang. *Energy management of multiple microgrids based on a system of systems architecture.* IEEE Transactions on Power Systems, vol. 33, no. 6, pages 6410–6421, 2018. (Cited on page 15.)

[Zheng 2020] Weiye Zheng, Wanjun Huang and David J Hill. *A deep learning-based general robust method for network reconfiguration in three-phase unbalanced*

active distribution networks. International Journal of Electrical Power & Energy Systems, vol. 120, page 105982, 2020. (Cited on page 20.)

[Zheng 2022] Xiangtian Zheng, Nan Xu, Loc Trinh, Dongqi Wu, Tong Huang, S Sivaranjani, Yan Liu and Le Xie. *A multi-scale time-series dataset with benchmark for machine learning in decarbonized energy grids*. Scientific Data, vol. 9, no. 1, page 359, 2022. (Cited on page 31.)

[Zhou 1998] Kemin Zhou and John Comstock Doyle. Essentials of robust control, volume 104. Prentice hall Upper Saddle River, NJ, 1998. (Cited on page 79.)

[Zhou 2021] Yuyu Zhou and Tong Zhou. *A Revisit to the Controllability and Observability of Networked Dynamic Systems*. IEEE Transactions on Control of Network Systems, vol. 8, no. 4, pages 1659–1668, 2021. (Cited on page 52.)

[Zhu 2022] Jiawei Zhu, Linying Xiang, Yanying Yu, Fei Chen and Guanrong Chen. *Average Controllability of Complex Networks With Laplacian Dynamics*. IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 4, pages 1704–1714, 2022. (Cited on page 52.)

[Zou 2021] Bo Zou, Pooria Choobchian and Julie Rozenberg. *Cyber resilience of autonomous mobility systems: cyber-attacks and resilience-enhancing strategies*. Journal of transportation security, pages 1–19, 2021. (Cited on page 20.)