No ordre: 2243

THESE

présentée à

L'université des Sciences et Technologies de Lille

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE

Spécialité: Productique: Automatique et Informatique Industrielle

par

Magda WOZNIAK
Ingénieur I.S.E.N.

CONTRIBUTION À LA PLANIFICATION TEMPORELLE T.C.L.P, un planificateur temporel à liens causaux

Soutenue le 10 Mars 1998 devant le jury composé de:

Jean-Claude GENTINA	Président
Marie-Odile CORDIER	Rapporteur
Malik GHALLAB	Rapporteur
Pierre BORNE	Examinateur
Max Dauchet	Examinateur
Patrick TAILLIBERT	Examinateur
Philippe VANHEEGHE	Examinateur

Thèse dirigée par M. P. VANHEEGHE et M. P. BORNE, préparée au Laboratoire d'Automatique et d'Informatique Industrielle de Lille (L.A.I.L U.R.A. - C.N.R.S. D1440), Ecole Centrale de Lille, I.S.E.N.

Remerciements

Je remercie Monsieur Jean Claude Gentina, Directeur de l'Ecole Centrale de Lille, d'avoir bien voulu présider le jury de cette thèse.

Que Monsieur Philippe Vanheeghe, Responsable du département Signaux et Systèmes de l'Institut Supérieur d'Electronique du Nord, soit remercié pour les conseils prodigués pendant ce travail.

Je remercie Monsieur Pierre Borne, Directeur Scientifique de l'Ecole Centrale de Lille, pour sa participation à la direction de ces travaux.

Je suis très reconnaissante envers Madame Marie-Odile Cordier, Professeur à l'IRISA Rennes et Monsieur Malik Ghallab, Directeur de recherche au LAAS Toulouse, d'avoir accepté d'être les rapporteurs de cette thèse.

Je remercie la société Dassault Electronique pour m'avoir confié les travaux de recherche à l'origine de cette thèse.

Mes plus vifs remerciements vont à Patrick Taillibert qui m'a encadrée et guidée durant toute cette thèse. Les discussions, très animées, souvent tumultueuses, que nous avons eues, son esprit de synthèse et ses interminables critiques, ont été d'un grand apport dans cette recherche. Enfin, je ne saurai jamais assez le remercier pour son soutien moral et son optimisme à toute épreuve, pour la confiance qu'il a portée sur ce travail, et qu'il s'est efforcée de me transmettre durant ces trois années.

Je remercie Monsieur Max Dauchet, Professeur à l'université de Lille 1, pour avoir accepté de faire partie de ce jury.

Un grand merci à Corinne qui a su me supporter pendant trois ans grâce à un calme face à toutes les situations, et pour toutes les discussions que nous avons pu avoir.

Merci également à Frédérick Garcia, qui m'a donné de nombreux conseils sur mon travail, avant, pendant et après les conférences, par téléphone ou par e-mail.

Je remercie le Conseil régional du Nord-Pas de Calais et la fondation Norbert Ségard pour le financement de mes travaux.

J'ai enfin été touchée par la confiance que m'a accordée Monsieur Jean-Nöel Decarpigny, Directeur de l'I.S.E.N., en m'acceptant dans son établissement.

Que Monsieur Michel Lannoo, Directeur de la Recherche à l'I.S.E.N., soit associé à ces remerciements.

Table des matières

Chapitre 1

Introduction

Nous décrivons ici le contexte du travail qui nous a conduits à développer l'étude décrite dans ce mémoire.

Notre objectif est d'utiliser la planification comme un moyen de simuler la prise de décision.

Dans la première partie de cette introduction, nous expliquons tout d'abord en quoi l'aide à la décision peut être nécessaire. Puis nous rappelons l'utilité de la simulation dans la conception, la compréhension, la mise au point et l'apprentissage de systèmes divers. Enfin, nous expliquons comment la simulation peut être vue comme une aide à la prise de décision.

La prise de décision résultant en une suite d'actions à appliquer pour résoudre le problème posé, nous présentons dans la deuxième partie diverses méthodes de construction de plans d'actions. L'étude de ces approches nous a conduits à considérer la planification en Intelligence Artificielle comme un moyen de simuler la prise de décision.

Nous présentons alors quelques applications de la littérature utilisant également la planification pour simuler la prise de décision, dont l'observation nous a conduits à porter notre intérêt sur la modélisation des actions et notamment leur composante temporelle. Après avoir rapidement présenté les approches existantes qui répondent à ce problème, nous présentons notre approche et les raisons pour lesquelles nous l'avons adoptée.

Enfin, nous achevons cette introduction par la présentation du plan du mémoire.

1.1 Simuler la prise de décision

Prendre une décision pour résoudre un problème donné, c'est choisir une solution au problème parmi un nombre de solutions potentiellement infini. Souvent, trouver une solution repose sur plusieurs décisions, et envisager à priori la meilleure solution dépend d'un grand nombre de facteurs: connaissance du problème, connaissance du domaine dans lequel le problème est posé, évaluation des conséquences de ces décisions, critères de sélection d'une solution parmi plusieurs... Lorsque le problème est complexe, il

devient difficile de tout maîtriser. Alors un système d'aide à la décision peut être utile.

Or *simuler un système* permet de corriger son comportement au vu des résultats de la simulation, de mieux comprendre son fonctionnement grâce à sa modélisation ou de tester sa robustesse en fonction de perturbations sans altérer le système lui-même. La simulation semble donc être un moyen de prédire le comportement d'un système.

La simulation étant un moyen de reproduire le comportement d'un système, elle peut être utilisée en apprentissage. Ainsi une personne peut tester ses connaissances grâce à la simulation (les résultats de la simulation serviront à l'évaluation), ou apprendre le fonctionnement du système en observant son comportement simulé.

La simulation peut encore être vue comme une aide à la décision. En effet, prendre une décision résulte en une action (plus ou moins complexe) dont l'application représente les effets de la prise de décision. Simuler cette action permet donc d'envisager ses résultats avant de l'exécuter réellement, et donc d'évaluer les conséquences de la prise de décision. La simulation peut servir à choisir la meilleure solution en fonction des résultats obtenus. Nous nous intéressons ici, à la simulation comme une aide à la décision pour un être humain. En effet, lorsque l'homme doit prendre des décisions complexes pour répondre à un problème, il doit rassembler un grand nombre de connaissances sur le domaine, et envisager la suite des étapes nécessaires pour atteindre l'objectif fixé. Cette étape est une projection dans le temps des actions à réaliser et de leurs effets prévus, et il semble évident que tout envisager n'est pas possible.

Prendre des décisions pour répondre à un problème résultant en une suite d'actions à appliquer, l'élaboration d'un système capable de simuler la prise de décision semble donc passer par l'élaboration d'un module de construction de plans d'actions.

Le paragraphe suivant résument diverses méthodes de construction de plans d'actions communément utilisées.

1.2 Planifier

La planification permet de contrôler le comportement d'un système (vivant ou artificiel). En effet, planifier, c'est construire une suite d'actions permettant d'atteindre un but fixé à partir d'une situation donnée, en utilisant des moyens disponibles. Planifier c'est donc prédire l'évolution du monde sur lequel agit le système. Cette prédiction est d'autant plus utile que la situation est complexe et qu'il est difficile de prévoir à long terme son évolution possible. Ce n'est pas le seul moyen de contrôle.

Certains problèmes ne nécessitent pas ou peu de planification; ce sont en général des problèmes bien identifiés et connus, simples dans la mesure où ils sont facilement discernables parmi un ensemble de problèmes, pour lesquels la réponse à fournir est connue. Dans ce cas, le problème consiste à identifier le problème et à appliquer le plan prédéfini pour répondre à la situation. Certaines approches se sont inspirées de ces mécanismes situation / réponse pour construire des modules de contrôle destinés à rendre intelligents des systèmes. Ces approches consistent à apprendre et modéliser des plans connus, répondant à diverses situations. Lorsque le système reconnaît une situation, il recherche le plan adéquat et l'applique. Ces plans sont appelés *plans réactifs*, dans le

sens où ils sont appliqués en réaction à la situation rencontrée. Le comportement du système associé est dans ce cas plus réactif que prédictif car le système n'a qu'une vision locale de la situation. Celle-ci peut conduire à un comportement global incohérent par rapport au but à atteindre.

Cette approche peut être comparée aux *arbres de décision.* En effet, les choix à faire ont également été définis à l'avance en fonction de la situation, et ont été codés dans un arbre. Chaque noeud de l'arbre représente un choix possible et les choix sont réalisés en fonction de règles et de contraintes du domaine, de la situation envisagée et du but à atteindre. Dans ces deux approches (plans réactifs et arbres de décision), dès qu'une nouvelle situation est envisagée se pose le problème de savoir quelles décisions prendre pour y répondre au mieux, sachant que la situation n'a pas été codée. De ce fait, si, à la différence des plans réactifs, les arbres de décisions sont capables de prédire le comportement global d'un système, ils peuvent aussi conduire à des comportements inappropriés.

La planification est alors un moyen de remédier à ce problème, en construisant *pour chaque problème* un plan qui tient compte de la situation courante, du but à atteindre et des moyens disponibles. Cependant elle est confrontée à des *problèmes de représentation* (actions, plans, situations, contraintes du domaine d'application) directement liés à des problèmes d'*efficacité de construction des plans*: construire un plan c'est faire des choix d'actions, en fonction d'un but à atteindre et de contraintes imposées par le domaine, c'est faire des choix sur l'ordre dans lequel elles seront exécutées. Plus la modélisation est complexe (lorsqu'elle cherche à représenter de façon détaillée la réalité) plus la génération des plans est délicate car les critères à prendre en compte sont d'autant plus nombreux. Il semble donc important de faire un *compromis entre pouvoir d'expression et efficacité de génération des plans.*

L'efficacité de la construction d'un plan passe par le contrôle de l'exploration de l'arbre de recherche que parcourt le générateur pour trouver une solution. Cet arbre, ou espace de recherche, peut avoir une taille importante gênante puisqu'elle est liée au temps que met le système pour planifier avant d'agir. Or cette exploration reproduit les choix faits par le générateur de plans pour atteindre le but fixé. Contrôler cette exploration revient donc à contrôler les choix faits, et donc à poser des contraintes sur la génération des plans. Ces contraintes peuvent être indépendantes du domaine et liées à des principes du générateur utilisé: le problème est alors un *problème d'algorithmique.* Elles peuvent aussi être liées au domaine, et traduisent alors des connaissances plus ou moins subjectives sur la façon dont on peut répondre à un problème dans un domaine donné. Il est alors nécessaire de pouvoir modéliser des stratégies ou *heuristiques* dépendant du domaine.

La génération des plans est souvent couplé à un module d'exécution. Ce module peut être un robot ou un agent qui applique réellement les actions, ou un module qui simule l'exécution pour vérifier si le plan généré est exécutable. Très souvent l'environnement dans lequel doit évoluer l'agent est dynamique et si l'exécution est simulée, elle doit tenir compte de cette caractéristque. Dans ce cas, il est nécessaire de contrôler l'exécution non seulement pour savoir quelles sont les actions à exécuter et quand il faut les exécuter, mais aussi pour vérifier si elles le sont correctement. En effet, un monde dynamique peut évoluer indépendamment des actions de l'agent. Dans ce cas, les actions prédites peuvent ne pas être exécutables si leurs conditions d'application ne

sont plus vérifiées au moment de l'exécution, ou peuvent échouer si un effecteur tombe en panne par exemple. Alors le plan prédit peut devenir inapplicable. Le contrôle de l'exécution sert à détecter ces problèmes. Plusieurs solutions permettent de bénéficier des retours d'exécution pour modifier le plan.

- **les étapes de planification et d'exécution sont complètement séparées**
  1- le plan est complètement abandonné et une nouvelle étape de planification est relancée pour tenir compte de la nouvelle situation.

  2- des réparations du plan ont été prévues et sont déclenchées en fonction des résultats d'exécution pour tenter de modifier localement le plan afin d'en conserver la majeure partie.

Dans les deux cas, le temps passé pour planifier peut s'avérer inutile si le plan est devenu obsolète.

- **les étapes de planification et d'exécution sont entrelacées: planification réactive**
  3- dès qu'une ou plusieurs actions sont exécutables elles sont exécutées.

L'engagement est plus important que celui des approches précédentes, mais les retours d'exécution sont pris en compte plus tôt.

- **L'agent agit par réaction, il n'y a pas de planification ([GEO 87])**
  4- des plans réactifs sont déclenchés en fonction des situations rencontrées.

Nous venons de présenter les besoins soulevés par la simulation de la prise de décision et les problèmes auxquels répond la planification. La partie suivante résume en quoi la planification peut être utilisée pour simuler la prise de décision.

1.3 Simuler la prise de décision grâce à la planification

La planification peut être utilisée à différentes étapes de la simulation de prise de décision.

En effet, un planificateur fait des choix pour construire un plan. L'homme fait également des choix pour trouver une solution à un problème donné. Les choix du planificateur peuvent reproduire les choix de l'homme: la planification peut être considérée comme la simulation de la prise de décision.

De plus, le résultat fourni par un planificateur est une suite d'actions. Le résultat de décisions est également une suite d'actions. Un plan peut donc être vu comme le résultat

de prises de décision.

Par ailleurs, évaluer la pertinence d'une décision, c'est évaluer ses conséquences. Exécuter un plan, c'est suivre l'évolution du monde due à l'application des actions. L'exécution (réelle ou simulée) d'un plan permet donc de connaître les effets des prises de décision.

Nous nous sommes donc intéressés à la planification comme un moyen d'aider l'homme à prendre des décisions.

Diverses approches de la planification ont été utilisées dans des applications réelles, soit en vue d'aider l'homme à prendre des décisions, soit en vue de rendre un système autonome. Nous présentons ici quelques exemples.

1.4 Quelques applications

1.4.1 Diagnostic et gestion de dysfonctionnement dans des applications temps réel

F. Ingrand et M. Georgeff ([ING 92]) proposent l'utilisation de plans réactifs pour contrôler les systèmes de propulsion d'une navette spatiale. Le système doit produire un raisonnement orienté (but à atteindre) et avoir des réactions en temps réel pour faire face à des événements imprévus.

En effet, le système reçoit des alarmes des divers propulseurs de la navette et contrôle son mouvement en gérant les forces propulsives. Les alarmes sont des signaux asynchrones qui représentent des compte rendus ou des pannes. Le système doit alors identifier les pannes, les localiser et déterminer les actions à appliquer pour réparer les composants défectueux ou rétablir une situation normale. Cette application met en évidence la nécessité d'un compromis entre un comportement réactif (répondre à une alarme) et un comportement orienté (réparer une panne, c'est-à-dire, appliquer plusieurs actions dans un but fixé).

Pour cela, et afin d'optimiser le temps de réaction, les actions à effectuer dans une situation précise sont décrites de façon procédurale. Ces règles sont encore appelées plans réactifs: ils définissent les suites d'actions à appliquer lorsqu'une situation particulière est identifiée. Le système ne construit pas globalement les plans nécessaires pour répondre à une alarme car le temps de planification serait trop important devant les temps de réaction demandés. Il est cependant capable de composer plusieurs plans réactifs pour obtenir un plan complet répondant au problème et donc tenir compte du but fixé.

1.4.2 Aider les garde-côtes américains à planifier leurs interventions d'urgence

Le planificateur SIPE-2 ([WIL 94]) est utilisé pour la planification des interventions d'urgence des garde-côtes en cas de marée noire ([KNO 96]). Il n'est pas utilisé en temps réel pour construire rapidement un plan en réponse à l'accident, mais en tant que simulateur pour répondre à des scénarios catastrophes destinés à l'entraînement des gardes-côtes.

Le système sert à évaluer le nombre d'équipements de nettoyage nécessaires et à déterminer leur localisation, pour répondre à un accident donné. Les plans peuvent être construits de façon interactive ou automatique. Lorsque la construction est automatique, il est alors nécessaire de modéliser des heuristiques liées au domaine, non seulement pour pouvoir reproduire le comportement des garde-côtes (on ne cherche pas à les remplacer), mais aussi pour contrôler la combinatoitre de la génération des plans.

SIPE-2 est un planificateur dit classique car il construit des plans en fonction d'un but à atteindre, d'une situation initiale et de moyens d'action; il ne fait pas appel à une bibliothèque de plans prédéfinis. Il est par ailleurs capable de corriger les plans générés au moment de leur exécution, pour faire face à des événements inattendus ou à des échecs d'action. L'exécution est couplée à un module type PRS (plans réactifs) ([GEO 87]) capable de corriger localement les plans en appliquant des réparations prédéfinies.

Dans cette application, la construction des plans est incrémentale et est réalisée pour chaque problème posé. La réparation des plans est, elle, procédurale.

1.4.3 La simulation de combats

EAGLE-AP ([SAL 93]) est un simulateur de combats intégrant un module de planification. EAGLE est le simulateur, AP (Adversarial Planning) est le planificateur. Au niveau stratégique de la simulation de combats se pose le problème de la construction de plans d'actions pour répondre à une mission. A chaque niveau de la hiérarchie, les ordres venant du niveau supérieur sont les objectifs à atteindre du niveau considéré. A ce niveau, un plan est construit et est traduit en ordres qui sont répartis vers le niveau inférieur. Une première solution nécessitait l'intervention humaine pour l'élaboration des plans à un niveau hiérarchique donné. Chaque individu construisant un plan pour une partie d'une mission globale, le système devait alors faire face à un manque de coordination globale de haut niveau.

Le système de planification AP a donc été rajouté pour simuler la construction des plans à un niveau hiérarchique relativement élevé. Alors, les plans générés faisant intervenir l'ensemble des agents nécessaires à la mission, leurs actions sont coordonnées pour atteindre un but global. De plus, ces plans sont conditionnels car ils sont construits en tenant compte du raisonnement de l'adversaire: des données terrain permettent d'évaluer les plans d'actions ennemis les plus probables. Pour chaque plan ennemi est alors généré un contre-plan ami qui sera déclenché lorsque la situation sera rencontrée.

AP contrôle également l'exécution des plans et peut donc savoir si ce qui a été prédit est correctement exécuté. Dans le cas où cette prédiction n'a pas été suffisante, le plan prévu peut échouer. Le système de planification est alors capable de modifier le plan pour le réparer en vue de réaliser le but global fixé d'une autre manière, à deux niveaux différents: soit le plan peut être corrigé localement (à bas niveau) et dans ce cas il y a conservation du reste du plan, soit il devient impossible de conserver le plan, alors il est abandonné et un nouveau plan est généré.

CARNEADE ([MAR 92]) (Combat Aéroterrestre Représenté Numériquement pour l'Etude, L'aide à la Décision et l'Entrainement) est un ensemble d'outils de simulation de bataille aéroterrestre conçu pour satisfaire 3 types de besoins: l'entraînement des officiers, l'étude de systèmes de forces et l'aide à la décision.

Ce système offre la possibilité de représenter les différents niveaux hiérarchiques de commandement et constitue un système complet de simulation. A un niveau élevé de

la hiérarchie (niveau stratégique), il s'agit de simuler les prises de décisions pour construire un plan en réponse à une mission (ordre) donnée. Au plus bas niveau (niveau exécution), il s'agit de simuler les actions sur le terrain (niveau tactique). Pour un niveau considéré dans la hiérarchie, le niveau supérieur est vu comme le niveau stratégique, le niveau inférieur sert à évaluer les décisions qui ont été prises. Plus le niveau hiérarchique est bas, moins la prise de désision est importante: ainsi, au niveau exécution, les ordres reçus sont des plans complètement spécifiés n'autorisant plus de choix.

Nous décrivons ci-dessous les 3 utilisations possibles de CARNEADE.

- **Entraînement des officiers**

Un officier reçoit un ordre; il traduit cet ordre en sous-ordres qui représentent autant de buts à atteindre par le niveau inférieur. Le simulateur fournit alors des plans possibles et les exécute. L'officier évalue si les décisions qu'il a prises, c'est-à-dire les ordres qu'il a donnés, sont corrects en fonction de critères fixés par le niveau supérieur.

- **Etude de systèmes de force, de doctrines et de systèmes d'armes majeurs**

L'officier founit des objectifs à atteindre, ainsi que les nouveaux moyens d'actions (nouvelles méthodes, nouvelles armes) et observe les résultats de la simulation pour évaluer ces nouveaux moyens et déterminer les situations dans lesquelles ils peuvent être utilisés efficacement.

- **Aide à la décision opérationnelle**

En temps de paix, CARNEADE sert à la planification opérationnelle pour l'évaluation de nouvelles stratégies, face à de nouvelles situations. Le principe d'utilisation est le même que précédemment. En temps de guerre, il est utilisé pour l'évaluation des hypothèses établies à partir de la situation militaire courante.

Quelle que soit l'utilisation du simulateur, se pose le besoin de construire des plans d'actions. Les plans sont stratégiques si le niveau hiérarchique est élevé; il s'agit alors de prendre des décisions. Ils sont tactiques et très détaillés à bas niveau et servent à simuler précisément les actions sur le terrain. Actuellement, les plans sont construits à l'aide d'un système expert, à partir d'arbres de décision représentant les réponses possibles à des situations particulières. Les plans sont des plans conditionnels, comme ceux générés par EAGLE-AP: des situations et actions ennemies sont envisagées à divers points dans le plan ami. Celui-ci contient alors des branches conditionnelles en réponse à d'éventuelles situations ennemies. Les réponses proposées sont des plans partiellement spécifiés qui sont complétés si la situation est réellement rencontrée.

1.4.4 Des problèmes communs

Les systèmes présentés précédemment utilisent un module qui leur fournit des plans. Or un plan est un ensemble d'actions situées temporellement les unes par rapport aux autres, dont l'application permet d'atteindre une situation finale (des objectifs) à partir d'une situation initiale. Ces systèmes sont donc confrontés à des problèmes:

- de représentation du changement
- de représentation de situations
- de modélisation et de gestion de contraintes temporelles

Par ailleurs, toutes ces applications mettent en évidence le besoin d'élaborer un plan en réponse à chaque problème posé, ceci afin d'éviter les comportements évoqués dans le paragraphe 1.2, lorsque les plans sont construits à partir d'arbres de décision. Utiliser la planification semble donc être justifié.

Ces systèmes ont également montré la necéssité de remettre en cause les plans construits en fonction de la situation courante. Ce problème a été abordé suivant différentes approches:

- utilisation de plans réactifs pour réagir rapidement ou pour corriger des plans en utilisant des réparations standard.
- utilisation de plans conditionnels pour envisager des réponses à des événements extérieurs probables
- la planification réactive peut également être envisagée comme une solution.

1.5 Nos motivations

Ce travail est issu des besoins de planification soulevés par la simulation de combats au niveau stratégique, au sein du système CARNEADE. Nous nous sommes demandés si l'utilisation des techniques de planification pour fournir les plans d'actions au simulateur permettrait d'éviter les problèmes posés par la construction de plans à l'aide d'arbres de décision ([WOZ 95]).

Nous avons étudié l'adéquation de la planification pour la synthèse de ces plans en utilisant des techniques de planification classique:

- les actions du domaine sont representées par des opérateurs de transformation instantanés de type STRIPS ([FIK 88]). Nous appelons planificateurs traditionnels les planificateurs suivant cette hypothèse.
- le planificateur est non linéaire afin de bénéficier de la représentation synthétique des plans et de la diminution de l'espace de recherche et
- il utilise des liens causaux afin d'éviter l'utilisation de critère de vérité (de type TWEAK) souvent trop coûteux ([CHA 87]).

Nous avons alors remarqué qu'il est facile de traduire un ordre verbal, ou mission, en un ensemble de buts à atteindre. Par ailleurs, au niveau stratégique de la simulation le problème revient à construire des plans d'actions (choix des actions et de l'ordre d'application) et la plupart des concepts manipulés sont ***symboliques***. Par exemple, le général impose le déroulement d'une action pendant celui d'une autre sans tenir compte des contraintes temporelles numériques entre ces deux actions. Celles-ci seront gérées à un niveau de raisonnement plus bas (lors de l'exécution par exemple).

Cette étude a mis en évidence les besoins suivants:

- la ***description détaillée du déroulement des actions*** pour pouvoir modéliser des actions dont les effets ne sont perceptibles que pendant leur application. Par

exemple, l'action "la troupe T1 soutient la troupe T2" nécessite la description du déroulement de l'action car le soutien n'est réalisé que pendant l'application de l'action.
- l'expression de la ***persistance d'un fait*** pour pouvoir distinguer un fait établi à un instant, d'un fait établi à un instant et maintenu sur un intervalle, ou une condition nécessaire à un instant et une condition à vérifier sur un intervalle
- la prise en compte et l'expression de ***contraintes externes*** temporelles symboliques
- la prise en compte de ***contraintes sur des états***: le fait FAIT ne peut être vrai entre deux instants.

Ces caractéristiques ne peuvent être prises en compte sans considérer la ***durée des actions***. Cette durée n'est pas exprimée par une valeur numérique, mais représente le fait que l'action n'est pas instantanée. Nous parlons alors de ***durée symbolique***.

1.6 Notre proposition

Nous avons étudié différentes techniques de planification et différentes modélisation du temps et de l'action, dont nous résumons les caractéristiques et dégageons avantages et inconvénients ci-après. Nous présentons ensuite notre approche.

- **La planification traditionnelle**

Un plan est dit traditionnel lorsque les actions qui le constituent sont des actions instantanées et sont ordonnées par la seule relation de précédence. TWEAK, SNLP ([MCA 91]) et UCPOP ([PEN 92]) sont des planificateurs traditionnels.

Des hypothèses précédentes découlent 2 restrictions.

Premièrement, les actions ne peuvent être décrites que par leurs conditions d'application et leurs résultats puisque la transformation du monde, due à l'application d'une action, est supposée instantanée. Il est alors impossible de décrire le déroulement d'une action.

Deuxièmement, les relations temporelles entre les actions étant restreintes à la seule relation de précédence, une seule action ne peut être appliquée à la fois. Dans ce cas, il est impossible de décrire le déroulement parallèle ou simultané de plusieurs actions, de résoudre des problèmes pour lesquels l'entrelacement de plusieurs actions est nécessaire afin de réaliser un objectif. L'utilisation de tels planificateurs pour des applications réelles semble donc inadaptée.

Décrire la durée des actions permet alors, non seulement de modéliser les conditions d'applications, le déroulement et les résultats de l'action, mais aussi d'étendre les relations temporelles possibles entre elles: puisqu'elle se déroulent sur des intervalles de temps, on peut parler de chevauchement d'intervalles.

Utiliser un modèle d'action instantanée, et donc des planificateurs traditionnels ne permet pas de construire des plans répondant aux problèmes précédents. Cependant, ces planificateurs contrôlent parfaitement la gestion du temps car elle est incluse dans leurs mécanismes de génération des plans,et en comparaison aux planificateurs temporels que nous présentons ci-après, leur efficacité est reconnue. Or dans le cadre de notre application, il semble nécessaire d'utiliser des algorithmes de planification efficaces. C'est pourquoi, tout au long de ce travail nous avons essayé de conserver au maximum les caractéristiques de ces planificateurs.

- **Planification traditionnelle et recherche de parallélisme**

Cette approche consiste à générer des plans grâce à la méthode précédente puis d'optimiser ces plans en recherchant les actions exécutables en parallèle.

Ainsi, dans [VEL 90] et [REG 91] ont été définis des critères de parallélisation permettant de mettre en parallèle des actions à partir d'un plan traditionnel. Un algorithme permet de convertir des plans totalement ordonnés en plan dont les actions non ordonnées sont exécutables en parallèle. Même si cette approche concerne le problème de parallélisme, elle ne répond pas au problème de l'interaction de plusieurs actions pour répondre à un but global, car le principe de construction des plans ne le permet pas. Nous ne nous attarderons donc pas sur cet aspect.

- **La planification temporelle**

Les planificateurs temporels incluent une représentation explicite du temps dans la modélisation des actions et gèrent des relations temporelles complexes leur permettant d'être utilisés dans des applications réelles. On peut discerner deux grandes approches de modélisation du temps. L'une considère l'intervalle comme unité temporelle alors que l'autre se base sur l'instant et décrit un intervalle comme un couple ordonné d'instants.

Une représentation naturelle consiste à décrire la durée des actions par un intervalle. De la même façon, il est possible de décrire la persistance d'un fait sur un intervalle, en associant au fait l'intervalle sur lequel il est vrai. Il est alors possible de modéliser une action par un ensemble de persistances positionnées les unes par rapport aux autres grâce aux relations temporelles d'Allen ([ALL 83a]). La modélisation d'une action passe donc par l'identification des intervalles sur lesquels persistent les faits et de supposer que ces intervalles représentent la durée maximale pendant laquelle les faits concernés sont vrais. Même si cette approche offre des capacités d'expression importante, elle conduit à une modélisation complexe des actions (il est plus naturel d'identifier des changements que des persistances et il est nécessaire de décrire toutes les relations temporelles possibles entre les intervalles).

Allen et Koomen ([ALL 83b]), et plus tard E. Tsang ([TSA 86]), ont construit un planificateur sur cette algèbre d'intervalles. Un problème de planification est alors traduit en un problème de contraintes temporelles: le planificateur communique à un résolveur de contraintes temporelles la ou les nouvelles contraintes à ajouter à l'ensemble existant, suite à une modification du plan (par exemple l'établissement d'un fait). Le résolveur propage l'information et détecte les incohérences. Une incohérence conduit le planificateur à faire un autre choix.

Cependant la richesse d'une telle algèbre conduit à des algorithmes de propagation dont la complexité combinatoire est exponentielle ou qui sont incomplets si on veut réduire cette complexité. Ceci limite leur utilisation dans le cas de problématique réelle.

Des approches ont tenté de diminuer cette combinatoire en diminuant la richesse de l'algèbre d'intervalles tout en conservant un pouvoir de représentation plus riche que celui des planificateurs traditionnels. De telles approches considèrent l'instant comme unité de base, définissent un intervalle comme un couple d'instants totalement ordonnés et décrivent non plus des relations entre intervalles mais entre instants. Par ailleurs une action est désormais considérée comme une ensemble de changements instantanés et non plus comme un ensemble de persistances.

Le planificateur IxTeT ([GHA 94]) utilise une représentation de l'action et du temps basée sur une algèbre d'instants (ou algèbre d'intervalles restreinte). Dans ce cadre, les algorithmes de propagation sont complets pour une complexité polynômiale et sont donc utilisables dans le cas d'applications réelles. Cependant la diminution de la complexité se fait aux dépens du pouvoir d'expression. En effet, il n'est pas possible de représenter toutes les relations temporelles exprimables dans l'algèbre d'intervalles. Ainsi, il n'est plus possible de contraindre 2 actions à être disjointes (les intervalles représentant leur durée ne se chevauchent pas) car l'expression de cette contrainte entre intervalles en contraintes temporelles entre instants n'est pas possible dans l'algèbre d'instants.

Les besoins soulevés par les applications présentées précédemment en ce qui concerne la représentation et gestion du temps, l'étude de la planification traditionnelle et les solutions proposées par la planification temporelle nous ont conduits à:

- utiliser un modèle d'action permettant de décrire leur durée
- utiliser le contrôle des planificateurs traditionnels pour pouvoir exprimer des contraintes sur la génération des plans
- limiter les relations temporelles possibles pour éviter la complexité induite par l'algèbre d'intervalles.

Nous avons donc:

- utilisé une représentation du temps basée sur les instants
- établi un modèle d'actions basé sur les instants permettant de décrire leur déroulement et de tenir compte de leur durée pendant la synthèse des plans.
- défini un moyen d'exprimer la persistance d'un fait
- limité les relations temporelles entre actions à un sous-ensemble des relations d'Allen: ce sous-ensemble contient toutes les relations temporelles exprimables à partir des 6 primitives d'Allen ne contenant pas l'égalité.

La suite de ce mémoire se décompose en 3 parties.

Dans la première partie, nous définissons la représentation des actions. Notre modèle est défini comme une extension du modèle STRIPS. Cette extension nous permet de considérer la durée symbolique des actions lors de la synthèse des plans, de décrire leur déroulement. Nous appelons ***d-action*** toute action possédant une durée ([WOZ 97]).Une d-action est un ensemble partiellement ordonné d'opérateurs instantanés, tous situés entre un instant de début et un instant de fin. Une description hiérarchique des actions nous permet en outre de décrire des actions complexes dont les composantes (d-

actions ou opérateurs instantanés) peuvent être positionnés par des relations d'Allen ou la relation de précédence entre instants. Par ailleurs, afin de décrire la persistance d'un fait sur un intervalle, nous définissons la notion de ***maintien d'un fait entre deux instants***. Ces extensions ont été réalisées en conservant les caractéristiques des planificateurs traditionnels non linéaires à liens causaux.

La deuxième partie est consacrée à l'étude des relations temporelles que nous sommes capables d'exprimer entre deux d-actions dans le cadre des planificateurs traditionnels non linéaires à liens causaux ([WOZ 98]). Nous proposons une table de correspondance entre un langage de haut niveau permettant de poser des contraintes temporelles symboliques entre intervalles, et un langage compréhensible des planificateurs traditionnels à liens causaux. Ce dernier langage est basé sur l'utilisation de la relation de précédence entre instants, les maintiens et les réparations des conflits qui peuvent apparaître dans de tels plans. Cette table de correspondance nous permet d'étendre facilement les planificateurs à liens causaux pour tenir compte de la durée symbolique des actions et décrire des relations temporelles disjonctives entre intervalles.

Dans la troisième partie, nous présentons un planificateur implémentant cette approche: TCLP pour Planificateur Temporel à Liens Causaux. L'utilisation de la table de correspondance pour décrire des relations entre d-actions conduit à une augmentation sensible du nombre de conflits dans les plans. Nous nous attachons dans un premier temps à la spécification des conflits dans TCLP, puis nous présentons les algorithmes de gestion des conflits construits pour diminuer l'engagement du planificateur, le taux de retour arrière. Nous terminons enfin par un récapitulatif de la démarche suivie tout au long de ce travail, et examinons les problèmes de mise en oeuvre rencontrés.

PARTIE I

Représentation des actions

Introduction

Un plan est un ensemble d'actions situées temporellement les unes par rapport aux autres. Appliquer un plan modifie le monde pour passer d'une situation initiale à une situation finale décrivant les buts à atteindre. La description d'un plan passe donc par la description des actions qui transforment le monde.

Nous voulons expliciter la durée des actions dans les plans afin de construire des plans les plus réalistes possibles. Nous décrivons la durée des actions par un intervalle représenté par un couple d'instants correspondant au début et à la fin de l'intervalle.

Nous proposons alors une représentation temporelle de l'action comme extension du modèle STRIPS: une action n'est plus représentée par un ensemble de changements associés à un instant, mais par un ensemble de changements associés à plusieurs instants. Les changements sont modélisés par des opérateurs de transformation de type STRIPS; une action est constituée d'un ensemble d'opérateurs STRIPS, partiellement ordonnés entre son début et sa fin.

L'extension de la représentation des actions nous autorise la description de fait maintenu pendant le déroulement de l'action. Nous avons donc introduit la notion de maintien pour exprimer la persistance d'un fait entre deux instants.

Le chapitre 2 concerne la modélisation du changement indépendamment du temps, et des problèmes liés à cette modélisation.

Le chapitre 3 concerne la prise en compte du temps dans la représentation des actions. Nous étudions alors plusieurs approches de la littérature pour en dégager avantages et inconvénients.

L'étude réalisée dans ces deux chapitres, ainsi que les besoins mis en évidence dans l'introduction nous ont conduits à faire des choix de représentation que nous expliquons dans le chapitre 4.

Chapitre 2

Modéliser le changement

Modéliser une action revient à modéliser les évolutions du monde provoquées par son application. De la même manière se pose le problème de la modélisation du monde. Plus la modélisation est fidèle à la réalité, plus la gestion des transformations est délicate, et plus la construction de plan est complexe. Il s'agit donc de faire un compromis afin de choisir une modélisation nous permettant de représenter des caractéristiques importantes du monde et ses changements, sans alourdir la construction des plan.

La modélisation la plus courante en planification considère l'action comme une fonction d'un état du monde à un autre. Un état est une photographie du monde à un instant donné. Très souvent un état est décrit par un ensemble de formules d'un langage. Une action est caractérisée par ses conditions d'application (les préconditions) et par ses effets (les post-conditions).

Il existe alors trois principaux problèmes liés à la détermination des préconditions et post-conditions d'une action:

- le problème de *rémanence* (ou *frame problem*) concerne la description de tous les faits qui ne changent pas après application d'une action. Par exemple, est-il nécessaire de décrire, dans une action de déplacement depuis le tableau qui se trouve dans la pièce p vers la porte de sortie, que le tableau se trouve toujours dans la pièce p à la fin du déplacement?
- le problème de *qualification* concerne la définition des conditions d'application de l'action. Quelles sont toutes les préconditions nécessaires à l'application correcte du déplacement précédent?
- le problème de *ramification* concerne la description de tout ce qui est modifié après application de l'action. Quels sont tous les faits du monde qui ont été modifiés suite au déplacement précédent?

2.1 Approche logique

Le calcul de situation ([MCC 69]) est un premier moyen de représenter l'action. La logique situationnelle est un langage où les états et les actions sont des termes du langage. Le seul prédicat *Holds* décrit une propriété p vraie dans un état e: *Holds(p,e)*. La seule fonction *Result* permet le passage d'un état e1 à un état e2, par application d'une action a: *e2 = Result(a,e1)*. Ainsi, l'action qui consiste à déplacer un cube x d'un cube y

vers un cube z est décrite par:

$$\forall(x, y, z), \text{Holds}(\text{sur}(x, y) \wedge \text{libre}(z), e1) \rightarrow$$
$$\text{Holds}(\text{sur}(x, z) \wedge \neg\text{libre}(z) \wedge \neg\text{sur}(x, y), \text{Result}(\text{déplacer\_x\_y\_z}, e1))$$

Cependant cette représentation ne caractérise pas ce qui n'a pas été transformé par l'action. Elle ne résout donc pas le frame problem. J. Mc Carthy et P.J. Hayes ([MCC 69]) ont proposé d'introduire des axiomes du décor pour décrire les faits inchangés après application d'une action. Ainsi, dire qu'un fait f, vrai dans l'état e1, reste vrai dans l'état e2 après application de l'action a, se traduit par: $holds(f, e1) \rightarrow holds(f, Result(a, e1))$. Dès que le nombre de faits décrivant le monde devient important, cette approche devient inutilisable.

Elle résout cependant facilement le problème de ramification, puisqu'il suffit de rajouter à la description du monde des lois du domaine permettant de déduire toutes les modifications du monde engendrées par les effets de l'action. Ainsi, dans les post-conditions de l'action seront considérés uniquement les effets principaux de l'action, et les effets secondaires seront déduits grâce aux lois du domaine.

Le problème de qualification reste entier: à notre connaissance, aucun auteur de la littérature n'a tenté de résoudre ce problème dans le cadre du calcul situationnel.

2.2 Approche par opérateur de transformation

Une autre approche consiste à modéliser les actions par opérateurs de transformation.

2.2.1 Le modèle STRIPS

Un opérateur STRIPS Op ([FIK 88]) décrit le passage d'un état du monde e1, représenté par un ensemble de formules, à un état du monde e2. L'action est caractérisée par ses conditions d'applications et ses effets. Un tel opérateur est décrit par:

- ***préconditions***(Op): la liste de préconditions décrivant les formules qui doivent être vraies dans l'état e1
- ***effets-add***(Op): la liste des faits qui deviennent vrais dans l'état e2 après application de Op. Les effets-add sont ajoutés à la description du monde après l'application de l'action Op.
- ***effets-del***(Op): la liste des faits qui deviennent faux dans l'état e2. Les effets-del sont retirés de la description du monde après application de l'action.

La figure 1 donne une représentation graphique d'un tel opérateur de transformation. Nous conserverons cette notation par la suite.

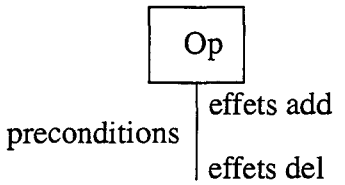


Figure 1: Représentation graphique d'un opérateur de transformation

L'état e2 est calculé à partir de e1, des effets ajoutés et retirés:
e2 = e1 + effest add(Op) - effets del(Op).

Tout fait de e1 absent de la liste des retraits reste inchangé après application de Op. Cette hypothèse, connue sous le nom d'*hypothèse de STRIPS*, permet de résoudre le problème de rémanence.

Lorsqu'un fait f de la liste des ajouts effets-add est déjà vrai dans la description du monde M, alors f n'est pas ajouté "une deuxième fois" à la description du monde. Il en est de même lorsqu'un fait f de la liste des retraits effets-del est déjà faux dans la description du monde.

Afin de résoudre facilement le problème de rémanence, ***nous avons choisi de représenter les changements par des opérateurs de transformation de type STRIPS.***

Se pose également le problème de cohérence des changements décrits. Le monde dans lequel une action est appliquée est supposé cohérent, c'est-à-dire qu'il est possible d'atteindre physiquement la situation correspondant à la description. Le monde résultant de l'application d'une action doit également être cohérent: les listes des ajouts et des retraits doivent respecter la sémantique du domaine; la liste des préconditions doit également respecter cette sémantique, pour qu'il soit possible d'atteindre au moins une situation où toutes les préconditions de l'action soient vérifiées simultanément.

Par exemple, l'action déplacer un cube x d'un cube y vers un cube z est incohérente si les effets retirés ne contiennent pas le fait sur(x,y). dans ce cas, et d'après le modèle de la figure 2, après application de l'action le cube x est à la fois sur le cube y et sur le cube z.

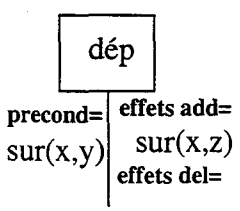


Figure 2: Description incohérente de l'action déplacer-x-y-z

Vérifier la cohérence d'un modèle est du ressort de la représentation des connaissances. Dans ce travail, nous nous fierons au bon sens de l'utilisateur (ici, celui qui décrit les actions du problèmes) pour respecter la cohérence des actions.

V. Lifschitz ([LIF 86]) a travaillé sur la sémantique des opérateurs STRIPS, de façon à la clarifier et à restreindre la liste des effets d'une action aux faits essentiels du domaine.

Certaines approches ([WIL 88], [GAR 93]) ont pour but, à la fois de simplifier la tâche de l'utilisateur, mais aussi de résoudre le problème de ramification, en ajoutant à la description du monde des règles du domaine permettant de déduire des faits (faits secondaires ou faits déduits) à partir de la connaissance d'autres faits (faits primaires ou de base). L'utilisation des règles du domaine simplifie la description des effets d'une action, car elle ne nécessite pas l'évaluation de tous ses effets. Nous décrivons le principe de ces approches dans le paragraphe suivant.

Remarquons ici, que le problème de qualification a été attaqué de la même façon par Ginsberg ([GIN 88]): il a défini un ensemble de règles du domaine permettant de calculer toutes les préconditions d'une action à partir d'un petit nombre de préconditions.

2.2.2 Le modèle déductif

D. Wilkins ([WIL 88]) étend le modèle STRIPS en scindant la description d'une action. Seuls les effets principaux de l'action seront décrits dans ses post-conditions, les effets secondaires seront déduits à l'aide de règles du domaine: des règles causales déclenchées sur un changement d'état, et des règles d'état déclenchées sur un état.

Dans [WIL 88], D. Wilkins présente un problème du monde des cubes où une modélisation par simple opérateur STRIPS nécessite deux modèles de l'action déplacer, alors que l'utilisation de règles du domaine conduit à l'écriture d'un seul modèle.

Dans cet exemple, les cubes sont susceptibles de supporter 1 ou 2 autres cubes.

- L'action "déplacer c de x à y" déplace un cube c d'un cube x à un cube y.
- Si le cube x ne supportait que le cube c, alors x ne supporte plus aucun cube après l'application de l'action.
- Si le cube x supportait un autre cube, alors x n'est pas complètement libre après application de l'action.

L'utilisation du modèle STRIPS conduit au deux modèles d'action suivants:

déplacer 1 :: préconditions sur(c, x), –sur(z, x), libre(y)
post-conditions sur(c, y), libre(x), –sur(c, x), –libre(y)

déplacer 2 :: préconditions sur(c, x), sur(z, x), libre(y)
post-conditions sur(c, y), –sur(c, x), –libre(y)

Remarque:

- la notation –p en post-condition signifie que p est retiré de la description du monde (effet-del de l'action).
- la notation –p dans une expression logique est équivalente à ¬p

D. Wilkins définit une seule représentation de l'action associée à des règles du domaine:

déplacer :: préconditions sur(c, x), libre(y)
post-conditions sur(c, y)

Une règle causale:

$\forall(x, y, z)$, holds(sur(x, y), e2) $\wedge$ holds(sur(x, z), e1) $\rightarrow$ holds(–sur(x, z), e2)

Une règle d'état:

$\forall(x, y, z), \neg holds(sur(x, y), e2) \wedge \neg holds(sur(z, y), e2) \rightarrow holds(libre(y), e2)$

Si une telle représentation simplifie la description des opérateurs, elle peut conduire à des problèmes de non-déterminisme se traduisant par une augmentation de la combinatoire au moment de la construction d'un plan. En effet, si la description du monde ne permet pas de savoir si x supportait ou pas un cube y avant l'application de l'action déplacer, il est impossible de déterminer exactement les effets de l'action. Dans ce cas, il faudra envisager toutes les situations possibles (le cas où un cube se trouvait sur x et le cas où il n'y en avait pas) et considérer toutes ces solutions pour construire un plan.

Le but de ce travail n'étant ni la mise en place d'un modèle de représentation, ni la recherche d'algorithme de mise à jour efficaces, nous allons utiliser le modèle d'action le plus simple vis à vis du planificateur, c'est-à-dire celui qui offre le maximum d'informations vis à vis des changements provoqués par l'application des actions.

Nous supposons donc que ***tous les changements provoqués par une action seront complètement spécifiés dans la liste de ses postconditions***. Notre modèle d'action est donc *déterministe*.

2.3 Conclusion: Du modèle instantané à la représentation temporelle

Les 2 approches précédentes supposent que les actions sont instantanées. Dans des planificateurs utilisant ces modélisations, la représentation du temps est basée sur l'instant et les relations temporelles entre les actions sont réduites à la seule relation de précédence entre actions. Or de nombreuses situations font intervenir des actions se déroulant en parallèle ou interagissant. Le fait de considérer uniquement la relation de précédence entre actions instantanées nous interdit de décrire de telles situations.

De plus, l'instantanéité des actions ne correspond pas à la réalité, car toute action possède une durée et certaines actions ne peuvent pas être décrites uniquement par leurs conditions d'application et leurs résultats. Ces sont toutes les actions dont des effets sont visibles uniquement pendant leur application. Elles ne peuvent être décrites que s'il est possible de décrire leur déroulement. Ainsi, une porte battante est *maintenue* ouverte uniquement pendant l'application de l'action "ouvrir la porte". La description de cette action nécessite la capacité d'exprimer la constance (ou le maintien) d'une propriété pendant un intervalle de temps. Or l'utilisation de modèles qui supposent que les actions sont instantanées rend impossible cette description.

Soit l'opérateur de transformation Op, qui a p en précondition, qui ajoute l'effet a et retire l'effet d (figure 3).

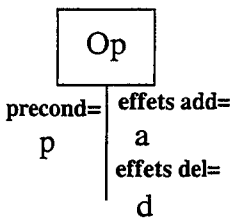


Figure 3: Description de l'opérateur Op

Cet opérateur traduit le passage d'un monde Mt (description du monde à l'instant t) où p est vrai, au monde Mt+1 où a est vrai et d est faux (figure 4).

Une propriété est vraie dans le monde Mt associé à l'instant t si la propriété appartient à la description de ce monde.

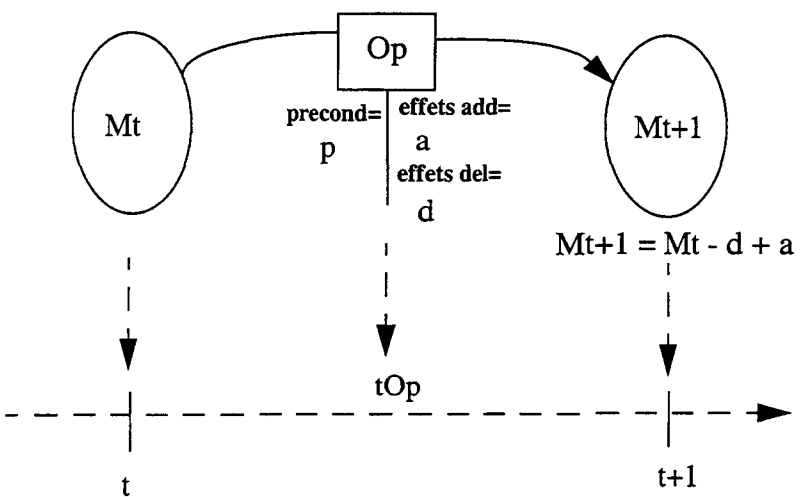


Figure 4: L'opérateur Op appliqué à tOp tranforme Mt en Mt+1;
entre t et tOp le monde ne change pas
les changements réalisés en tOp sont pris en compte en t+1

L'instant où a est vrai (t+1) représente la borne inférieure de l'intervalle sur lequel a est vrai. La borne supérieure n'est pas connue: on suppose que a reste vrai tant que rien ne vient dire le contraire. Ce problème est connu sous le nom de ***problème de persistance***.

L'instant où p *doit être vrai* (t) correspond à la borne supérieure de l'intervalle sur lequel p *doit être vrai*: il n'y a pas de contrainte sur le début de cet intervalle. Si p appartient aux effets retirés de l'opérateur alors p est faux en t+1, si p n'est pas détruit par l'opérateur alors on suppose que p reste vrai tant que rien ne vient affirmer le contraire.

Il est impossible de spécifier des conditions vraies sur un intervalle ou des effets produits sur un intervalle de temps minimal parce qu'on ne possède pas de moyen pour figer les deux bornes de l'intervalle correspondant.

Ainsi nous désirons expliciter la durée des actions pour pouvoir répondre aux problémes posés par les modèles d'action instantanés: description du déroulement des actions avec conditions ou effets vrais sur un intervalle minimal, relations temporelles

entre actions plus complexes que la seule relation de précédence.

Nous ne nous intéressons qu'à l'aspect structurel du temps: nous ne parlons pas de mesure, de date, de durée numérique. Ainsi, la durée que nous associons aux actions est ***symbolique***, au même titre que les relations temporelles que nous manipulons. Nous opposons le terme symbolique au terme numérique.

Le chapitre suivant est consacré à l'introduction du temps dans la représentation des actions.

Chapitre 3

Prise en compte du temps

Pour décrire la durée d'une action, nous lui associons un intervalle décrit par un couple d'instants totalement ordonnés: son début et sa fin. Nous supposons donc que nos actions ont une *durée finie*.

Associer une durée à une action rend possible la description de son déroulement. On peut lui associer des conditions d'applications qui doivent être vraies:

- au début de l'action ou pendant l'action
- à un instant donné ou pendant un intervalle de temps.

De la même façon, il est possible de décrire plus précisément:

- à quel instant pendant le déroulement de l'action sont réalisés les effets de l'action
- sur quel intervalle ils sont maintenus.

Il est donc nécessaire d'associer des objets atemporels (actions, fait, changement) à des objets temporels. Des relations entre objets temporels permettent alors de situer dans le temps des objets atemporels. On peut distinguer deux principales modélisations du temps:

- l'une considère l'intervalle comme objet temporel ([ALL 83a]).
- l'autre considère l'instant ([MCD 82], [SHO 87]).

Dans ce chapitre, nous résumons les deux approches précédemment évoquées, présentons leurs points forts et leurs points faibles de façon générale, mais également en étudiant leur utilisation dans le cadre de la planification.

3.1 Une modélisation du changement basée sur l'intervalle

Une représentation naturelle de la durée d'une action et de la persistance d'un fait consiste à associer à l'action (au fait) un intervalle temporel. Il représente la durée de l'action à laquelle il est associé, l'intervalle maximal sur lequel persiste le fait associé.

J.F. Allen considère l'intervalle comme unité temporelle. Il construit une représentation de l'action et du monde sur l'intervalle. Une action est un événement se déroulant sur un intervalle. Une propriété décrivant un fait du monde est vraie sur un

intervalle ([ALL 84]).

Le temps est donc structuré en intervalles et les relations temporelles sont des relations temporelles entre intervalles. J.F. Allen définit dans [ALL 83a] 13 primitives temporelles qui permettent de décrire toutes les positions relatives possibles entre deux intervalles. Ces relations et leurs abréviations sont décrites en annexe A.1.

J.F. Allen et A. Koomen pour le système Timelogic ([ALL 83b]), ou encore D. Joslin avec le planificateur temporel Descartes ([JOS 96]), proposent une modélisation de l'action basée sur le formalisme temporel défini par Allen. Ainsi, ils associent à chaque propriété représentant les préconditions et post-conditions de l'action un intervalle. Cette association qualifie temporellement l'assertion concernée. Les intervalles sont alors positionnés entre eux par des primitives d'Allen.

- *Soit l'action est instantanée*: on ne décrit que ce qui est vrai avant et après l'action.

Le schéma de la figure 5 représente l'action déplacer_x_y_z sous cette hypothèse. Ii (i=1 à 5) sont les intervalles associés à chaque propriété décrivant l'action. Une flèche représente une contrainte sur une borne d'un intervalle: par exemple, I2 s'achève au moment où I3 et I5 débutent.

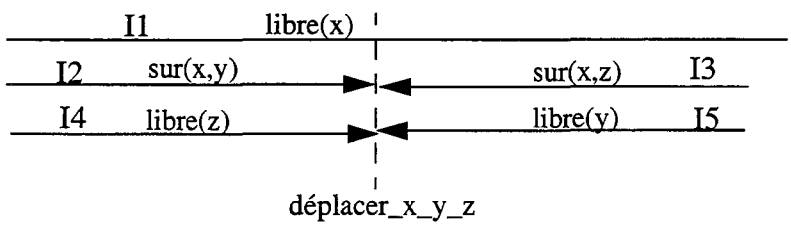


Figure 5: Description instantanée de l'action déplacer_x_y_z

- *Soit la durée de l'action est explicitée et non nulle*: il est alors possible de décrire son déroulement.

La représentation peut être étendue pour tenir compte de la durée de l'action. Soit Ia l'intervalle représentant cette durée. Cette action est modélisée dans la figure 6.

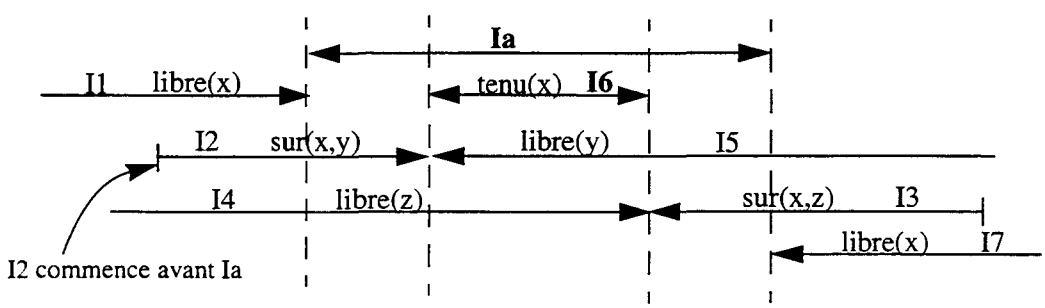


Figure 6: Description de l'action déplacer_x_y_z, en considérant sa durée

Deux hypothèses sont faites:

- l'intervalle associé à une assertion est l'intervalle maximal sur lequel l'assertion est vraie. Ceci implique la description de toutes

les relations temporelles possibles entre deux intervalles, et donc la complétude de l'information temporelle.

- le monde est supposé clos. Alors, une assertion associée à un intervalle signifie que l'assertion est vraie sur l'intervalle et fausse en dehors.

Une action est décrite:

- en associant un intervalle à chaque propriété persistante (I1 à I7)
- en positionnant chacun de ces intervalles entre eux et par rapport à l'intervalle décrivant la durée de l'action (Ia) grâce aux primitives d'Allen (le tableau 1 résume les relations entre les assertions de l'actions et l'action).

	I1	I2	I3	I4	I5	I6	I7
Ia	***mi***	***oi***	***o***	***oi si di***	***o fi di***	***di***	***m***

Tableau 1 : Positionnement des propriétés décrivant l'action déplacer_x_y_z par rapport à Ia

Les relations temporelles associées à l'action déplacer_x_y_z sont données dans [ALL 91] et sont précisées en décomposant l'action en prendre / tenir / poser. Remarquons que cette décomposition hiérarchique n'est pas admise dans Descartes.

- L'action *prendre_x_y* est définie par:

prendre_x_y est associée à l'intervalle Iprendre_x_y.
libre(x) est associé à I1 et I1 m Iprendre_x_y
sur(x,y) est associé à I2 et I2 fi Iprendre_x_y
libre(y) est associé à I3 et I3 mi Iprendre_x_y
tenu(x) est associé à I4 et I4 mi Iprendre_x_y

- L'action *poser_x_z* est définie par:

poser_x_z est associée à l'intervalle Iposer_x_z
tenu(x) est associé à I1 et I1 m Iposer_x_z
libre(z) est associé à I2 et I2 m Iposer_x_z
sur(x,z) est associé à I3 et I3 si Iposer_x_z

- Alors on définit hiérarchiquement l'action déplacer_x_y_z (associée à Ia) par:

prendre_x_y est associée à Iprendre_x_y et Iprendre_x_y s Ia
tenu(x) est associé à I1 et I1 mi Iprendre_x_y
poser_x_z est associé à Iposer_x_z et Iposer_x_z f Ia.

Cette décomposition permet de déduire toutes les relations temporelles entre chaque couple d'intervalles à l'aide de la transitivité des primitives d'Allen (annexe A.2). Nous pouvons par exemple dire que I2 m I6, avec I2 et I6 définis dans la figure 6.

Ce formalisme offre un pouvoir d'expression riche pour décrire l'action: en effet,

le fait d'associer une durée à l'action (Ia) nous a permis de préciser le déroulement de l'action.

Cependant, cette modélisation soulève plusieurs problèmes:

- d'une part, elle tend plus à modéliser la persistance d'une propriété sur un intervalle que le changement. Or une action symbolise le changement.
- la description d'une action devient rapidement illisible et complexe car il est nécessaire d'identifier toutes les relations temporelles entre les intervalles. Cette difficulté s'accentue lorsqu'il n'est pas possible de décomposer une action complexe en un ensemble d'actions plus simples positionnées temporellement les unes par rapport aux autres.
- l'utilisation de la logique temporelle d'Allen conduit à une gestion complexe des relations temporelles.

Or nous cherchons à représenter l'action comme un ensemble de changements, tout en évitant des complexités combinatoires trop importantes.

Dans le paragraphe suivant, nous décrivons le passage d'une représentation baséee sur l'intervalle à une représentation basée sur l'instant.

3.2 Changement, intervalle et instant

3.2.1 Relation entre changement et instant

L'approche précédente associe un intervalle à une propriété (Ipté dans la figure 7). Cet intervalle décrit la durée pendant laquelle la propriété persiste; ailleurs elle est fausse (Iavant et Iaprès dans la figure 7). On peut alors en déduire que le changement se fait à la frontière entre Iavant et Ipté, et à nouveau à la frontière entre Ipté et Iaprès; le changement est supposé discret.

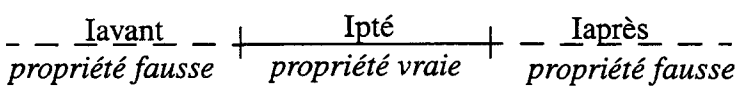


Figure 7: Une propriété est vraie sur l'intervalle qui lui est associé, fausse ailleurs

Un changement discret semble donc lié à un instant plutôt qu'à un intervalle. Ainsi, il est possible d'identifier quatre instants caractéristiques dans la représentation de l'action déplacer_x_y_z (figure 8).

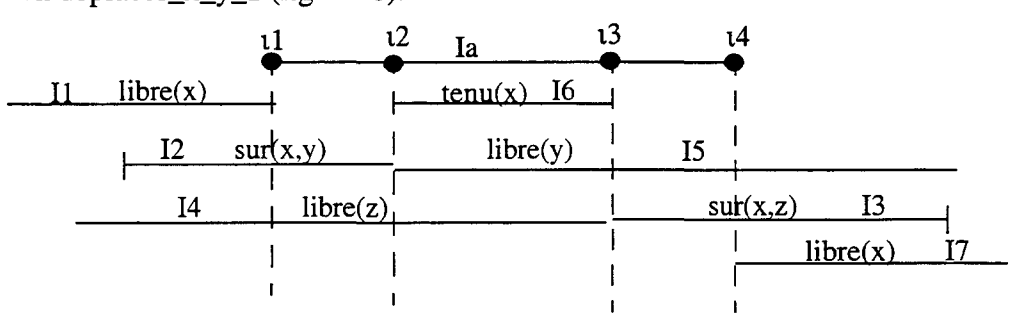


Figure 8: Identification des instants dans l'action déplacer_x_y_z

Il est alors possible d'expliciter les changements de valeur de vérité associés à chaque instant. A l'instar du modèle STRIPS, il est possible de décrire ce qui est vrai, ce qui devient vrai et ce qui devient faux à un instant donné.

- "*p est vrai en i*" (avec p propriété et i instant) signifie que p est vrai dans le monde juste avant l'instant i où se produit un changement. Si le changement ne concerne pas p, alors p est encore vrai juste après i (hypothèse de STRIPS).

- "*p devient faux en i*" signifie qu'à l'instant j juste après i, on peut dire "p est faux en j".

- "*p devient vrai en i*" signifie qu'à l'instant j juste après i, on peut dire "p vrai en j".

Le tableau 2 résume ces changements.

instants	est vrai	devient faux	devient vrai
i1	***libre(x)*** ***sur(x,y)***	***libre(x)***	
i2	***sur(x,y)***	***sur(x,y)***	***tenu(x)*** ***libre(y)***
i3	***tenu(x)*** ***libre(z)***	***libre(z)*** ***tenu(x)***	***sur(x,z)***
i4	***sur(x,z)***		***libre(x)***

Tableau 2 : Description des changements dans déplacer_x_y_z

Dans le tableau 1 nous avons I4 o s d Ia: cette disjonction temporelle traduit le fait que l'intervalle sur lequel libre(z) est vrai doit se terminer pendant l'intervalle Ia (ie, l'instant i3). Elle ne spécifie aucune contrainte sur le début de l'intervalle I4. De plus I4 oi si di Ia: ce qui signifie que libre(z) devient faux avant la fin de l'action. Nous traduisons ces propriétés par:

- la propriété libre(z) doit être vraie (au moins) en i3, peut importe l'instant auquel la propriété a été établie
- la propriété libre(z) devient fausse juste après i3.

3.2.2 Des difficultés liées à la gestion de relations temporelles entre intervalles

La représentation des actions proposée par Allen est directement inspirée de sa logique temporelle: une logique temporelle réifiée basée sur l'intervalle. Le prédicat holds associe une propriété p à un intervalle I: holds(p,I) signifie que p est vrai sur I, et est défini par "p est vrai sur tout sous-intervalle de I".

En outre, prendre en compte le temps nécessite non seulement un modèle de représentation, mais aussi la *mise à jour des connaissances* temporelles, la *vérification de leur cohérence*. Par exemple, si on sait holds(p,I) et si on apprend holds(-p,J), il est

alors nécessaire de contraindre I et J à être disjoints pour rétablir la cohérence de l'ensemble des connaissances. Cette contrainte est traduite par une nouvelle relation temporelle I < > m mi J; elle doit alors être propagée à tous les intervalles en relation avec I et J. Or cette propagation nécessite, ou des algorithmes dont la combinatoire exponentielle ne correspond pas à nos attentes, ou à des algorithmes polynômiaux dont l'incomplétude ne permet pas de savoir si un ensemble de relations temporelles est globalement cohérent suite à l'ajout d'une nouvelle relation ([ALL 83a], [VIL 86]).

Trois raisons nous ont poussés à utiliser les instants plutôt que les intervalles pour modéliser une action:

- Un changement discret est associé à un instant.
- La description des actions utilisant des intervalles et basée sur la persistance est complexe
- La gestion de relation entre intervalles est complexe.

3.3 Une représentation de l'action basée sur l'instant

Dans ce paragraphe, nous étudions les modèles d'actions proposées par cinq planificateurs dont la représentation est basée sur l'instant. Cette étude nous permet de mettre en avant les avantages et les inconvénients des modèles utilisés, afin de mieux faire notre choix de représentation, choix que nous présentons dans le chapitre 4.

La modélisation du temps que nous présentons ici utilise l'instant comme unité temporelle de base ([MCD 82], [SHO 87]). Le temps est un continuum d'instants et à chaque instant correspond un état du monde. Un fait est vrai dans un état, et donc à un instant. Un événement a lieu entre deux états. Dans ce cas, l'objet temporel est l'instant, les relations temporelles sont les relations entre instants (<, >, =). Une action est alors vue comme un ensemble de changements discrets et instantanés provoqués à divers moments durant son déroulement.

Nous présentons dans ce paragraphe plusieurs représentations de l'action basées sur l'instant, utilisées dans les planificateurs suivants:

- ***DEVISER***, premier planificateur à introduire la notion de temps dans les plans. Une action est vue comme une fenêtre temporelle dont la longueur varie en fonction des relations temporelles avec les autres actions.
- ***Triptic*** propose de représenter des actions possédant une durée par un ensemble d'opérateurs instantanés de type STRIPS.
- ***ZENO*** est un planificateur capable de tenir compte de la durée des actions et de modéliser le changement continu. Il est adapté pour gérer des contraintes numériques.
- ***SIPE-2*** étend entre autres les capacités de description de SIPE en considérant la durée des actions, mais n'étend pas les relations temporelles exprimables entre deux actions. Il propose également une description hiérarchique des actions.
- ***IxTeT*** utilise un modèle d'action directement issu d'une logique temporelle réifiée. Les effets et conditions d'application d'une action sont décrits par des

prédicats de la logique exprimant persistance et changements. Une action peut être décrite de façon hiérarchique.

Dans cette étude, nous ne nous attachons qu'au pouvoir d'expression supportée par la représentation adoptée. Afin de comparer de façon détaillée les modèles d'actions de ces planificateurs, nous avons construit plusieurs problèmes (décrits en annexe B) et tentons de les modéliser un à un en utilisant le modèle du planificateur étudié.

3.3.1 DEVISER

S.A. Vere a été le premier à considérer le temps dans le système de planification DEVISER ([VER 83]). Une action est vue comme une activité se déroulant sur un intervalle de temps, ou fenêtre temporelle, délimité par une borne inférieure et une borne supérieure. Les actions sont définies par des préconditions qui doivent être vraies tout au long de l'action et des effets qui sont réalisés à la fin de l'action. Cette restriction empêche donc la description de conditions d'application nécessaires à un instant intermédiaire entre le début et la fin de l'action, ainsi que la description d'une action où les effets produits ne sont visibles que pendant son application (et donc faux à la fin). Par ailleurs les relations entre les fenêtres ne sont descriptibles qu'à travers les relations temporelles entre les instants de début et de fin, puisque ce sont les seuls instants "significatifs" associés à une action.

3.3.2 Triptic

Triptic ([RUT 93]) est un planificateur utilisant des opérateurs de transformation sans règles de déduction. La représentation des actions, associées à des intervalles représentés par un couple d'instants, est plus souple que celle de DEVISER: les préconditions doivent être vraies au début de l'action comme pour DEVISER, mais les effets peuvent être produits à un instant intermédiaire entre le début et la fin de l'action. Ils doivent cependant être vrais à la fin de l'action. Il est donc à nouveau impossible de décrire une action produisant un effet uniquement pendant le temps où elle est appliquée.

3.3.3 ZENO

3.3.3.1 Caractéristiques

Dans ZENO ([PEN 94]), les actions possèdent des préconditions et des effets numériques (métriques). Il est possible de décrire des changements continus qui seront traités comme contraintes supplémentaires lors de la construction des plans.

Le temps est modélisé par des instants.

Une action se déroule sur un intervalle de temps décrit par un couple d'instants; cette durée peut être numérique. Des effets et préconditions des actions peuvent être vrais sur des intervalles de temps: le fait est vrai à t, pour tout t compris entre l'instant de début et l'instant de fin de l'intervalle.

Il est possible de poser des contraintes sur les variables apparaissant dans la description d'une action (temps, variables caractérisant une propriété du monde).

Les préconditions sont décrites par des conjonctions ou disjonctions de littéraux

(f(t,x1, xn) = c, p(X) ou ¬p(X)). Les disjonctions sont interdites dans les effets des actions.

La syntaxe de description est donnée dans la figure 9.

```
Schema NOM (arguments)
at-time: liste des instants

precondition:
    liste des preconditions associées à des instants

constraints:
    contraintes numériques sur les variables
    conntraintes temporelles numériques /symboliques

effect:
```

Figure 9: Syntaxe de description des actions dans ZENO

3.3.3.2 Application aux exemples

- **Une macro-action composée de A et B telles que A d di o oi B (annexe B.1)**

La contrainte temporelle entre les actions A et B s'exprime par une conjonction de contraintes de précédence entre le début et la fin des deux actions: dA < fB et dB < fA avec dA < fA et dB < fB.

La macro-action se décrit donc par 6 instants: le début et la fin de la macro, et les 4 instants délimitant le début et la fin des actions A et B.

Nous donnons dans la figure 10 la description d'une telle action en suivant la syntaxe de description de ZENO.

```
action1
    at-time: ts, da, fa, db, fb, te

    precondition:
    at(da,a1)   ∧   at(db,b1)

    constraints:
    ts < da< te,
    ts < fa < te,
    ts < db < te,
    ts < fb < te,
    da<fa
    db<fb
    da <fb, db <fa

    effect:
    at(da,a2)  ∧   at(fa,a3)  ∧   at(db,b2)  ∧   at(fb,b3)
```

Figure 10: Description de l'action correspondant au cas 1

L'absence de description hiérarchique fait disparaître la structure de l'action: les composantes A et B sont remplacées par 4 instants quelconques.

ZENO utilise l'algorithme du Simplexe pour traiter les contraintes temporelles. Le Simplexe est capable de déterminer les solutions numériques (si elles existent) d'un système d'inéquations linéaires. Il est donc nécessaire de traduire les contraintes symboliques en contraintes numériques. Le système de contraintes linéaires suivant correspond aux contraintes temporelles entre A et B:

$$\begin{cases} dA - fB < 0 \\ dB - fA < 0 \\ dA - fA < 0 \\ dB - fB < 0 \end{cases}$$

- **Appeler les secours (annexe B.2).**

Appeler les secours est composé de deux actions A (pour appeler la police) et B (pour appeler les pompiers). La relation temporelle entre les actions A et B (A < > B) se traduit par la disjonction de relations de précédence entre instants $fA < dB \vee fB < dA$

S'il est possible de décrire des contraintes disjonctives dans le champ *constraints* d'une action alors cette macro-action est décrite dans la figure 11.

```
action2
    at-time: ts, da, fa, db, fb, te

    precondition:
    at(da,a1)   ∧   at(db,b1)

    constraints:
    ts < da< te,
    ts < fa < te,
    ts < db < te,
    ts < fb < te,
    fB < dA   ∨   fA < dB

    effect:
    at(da,a2)   ∧   at(fa,a3)   ∧   at(db,b2)   ∧   at(fb,b3)
```

Figure 11: Description de l'action du cas 2

Le simplexe ne sait traiter que des conjonctions de contraintes puiqu'il est adapté pour résoudre des ***systèmes*** d'inéquations linéaires. Chaque fois qu'une disjonction apparaît, il est alors nécessaire de scinder le problème global en autant de sous-problèmes qu'il y a de membres dans la disjonction. Chaque sous-problème est un système que le simplexe doit résoudre. Dans notre cas, les deux sous-problèmes sont décrits dans la figure 12.

Sous-problème 1:	Sous-problème 2:
ts < da< te, ts < fa < te, ts < db < te, ts < fb < te, ***fB < dA***	ts < da< te, ts < fa < te, ts < db < te, ts < fb < te, ***fA < dB***
L'ensemble des contraintes est équivalent à B < A	L'ensemble des contraintes est équivalent à A < B

Figure 12: Décomposition d'un problème disjonctif entre deux sous-problèmes et un point de choix

Si l'un des deux systèmes ne possède pas de solution alors ZENO ne conserve que la solution correcte.

Si les deux systèmes possèdent une solution alors ZENO doit faire un choix et considère l'une et l'autre des voies successivement.

- **Contraintes externes (annexe B.3).**

  - Le voyage

Nous cherchons ici à modéliser un plan initial contenant 3 buts non ordonnés tels que l'un des buts (B3) soit réalisé avant ou après les deux autres (B1, B2).

ZENO décrit le plan partiel initial de la même façon que toute autre action. Les buts à atteindre sont décrits comme des conditions qui doivent être vraies à un instant marquant la fin du plan (tfin), la situation initiale par les effets d'un instant marquant le début du plan (tdébut).

Il est possible de poser des contraintes sur les variables qui décrivent des propriétés du monde et les variables associées au temps. Ainsi, il est possible d'imposer une durée maximale à l'exécution du plan par tdébut < tfin < tdébut + 10: le plan doit s'exécuter en moins de 10 heures.

Cependant, les contraintes autorisées dans la description d'une action ne semble pas permettre la description de la contrainte que nous cherchons à exprimer, qui fait intervenir une disjonction. Il faut donc envisager deux plans partiels initiaux distincts et tenter de construire des plans en partant des deux sous-problèmes décrits dans la figure 13.

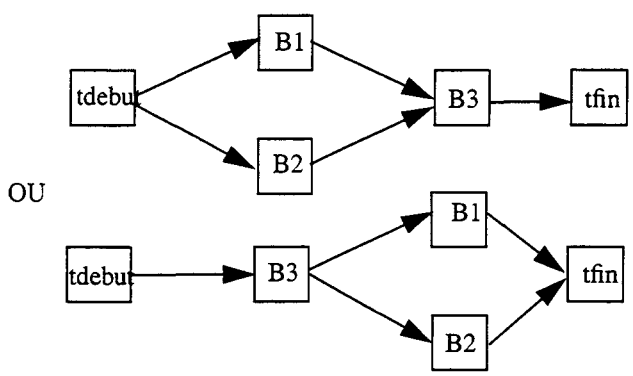


Figure 13: Les deux plans partiels initiaux équivalents au plan initial du voyage

Si les deux plans conduisent tous les deux à une solution alors le travail du planificateur aura été beaucoup plus important que si la disjonction avait été conservée (nécessité de faire tourner le générateur deux fois au lieu d'une).

Si les deux générations mènent à une impasse, là encore le travail du générateur aura été doublement inutile.

Si le premier choix mène à une impasse le générateur tente de construire une deuxième solution seulement après s'être rendu compte du premier échec.

- Interdiction d'obtenir un état particulier

Soit en(piece_x, objet_y) le fait interdit lors de la construction du plan.

La situation où en(piece_x, objet_y) et –en(piece_x, objet_y) sont vrais en même temps est incohérente.

Il suffit donc de décrire le plan initial en précisant que ¬en(piece_x, objet_y) est vrai en t avec $tdebut \leq t \leq tfin$ pour interdire toute action de produire l'effet en(piece_x, objet_y). La description de l'action fictive correspondante (ppi) est donnée dans la figure 14.

ppi
at-time: tdebut, tfin

precondition:
at(tfin, but)

constraints:
[]

effect:
$\neg at(tdebut, en(piece\_x, objet\_y)) \wedge \forall t\ t \in [tdebut, tfin] \wedge \neg at(t, en(piece\_x, objet\_y))$

Figure 14: Description d'un plan partiel initial avec état interdit

Remarques:

- nous avons décrit l'interdiction par l'intermédiaire d'un effet de l'action ppi. Cependant, la sémantique associée aux effets d'une action correspond à la production d'un état après application de l'action. Peut-on associer au concept de production celui d'interdiction?
- si l'interdiction a lieu entre deux instants non ordonnés, alors il est impossible de la décrire puisque l'effet ¬at(T, en(piece_x,objet_y)) ne peut être attribué à l'un ou l'autre des instants.

3.3.3.3 Conclusion

ZENO est capable de décrire des actions possédant une durée.

La représentation du temps est basée sur les instants. La persistance d'un fait sur un intervalle est traduite par la vérité du fait à tout instant compris entre l'instant de début et l'instant de fin de l'intervalle.

ZENO est adapté pour traiter des durées numériques puisqu'il fait appel au

Simplexe pour déterminer l'existence d'une solution à l'ensemble des contraintes temporelles issues du plan. Si les contraintes temporelles sont symboliques, l'utilisation du Simplexe nécessite le passage à des contraintes numériques. En outre, la complexité combinatoire de cet algorithme peut aller à l'encontre des critères d'efficacité recherchés dans le cadre d'une utilisation pratique des techniques de planifications. Par ailleurs, seules les contraintes temporelles symboliques appartenant à l'algèbre d'instants sont considérées. Toute disjonction, si elle est exprimable, se traduit par un point de choix dans la recherche.

3.3.4 SIPE-2

3.3.4.1 Caractéristiques

SIPE-2 ([WIL 90]) est un planificateur directement issu de SIPE ([WIL 88]). SIPE est un planificateur gérant également l'exécution des plans générés; il est capable de modifier les plans construits en fonction des retours d'exécution.

SIPE-2 étend les capacités de description de SIPE en vue de pouvoir être appliqué à des problèmes réels tels que l'entraînement des garde-côtes américains en cas de marée noire ([WIL 94]), ou la production à la chaîne de produits à partir de matières premières. Les extensions portent sur une amélioration de:

- la prise en compte de durée numérique et du raisonnement temporel
- la gestion de ressources
- la gestion des "critics" (comparables aux conflits).

L'un des objectifs de SIPE-2 est la recherche d'un équilibre entre efficacité, expressivité et flexibilité. L'amélioration majeure porte sur la gestion des conflits dans les plans partiels.

Nous décrivons ici les caractéristiques de SIPE-2 vis à vis de la description des actions, des contraintes qu'il accepte.

- **Les actions**

SIPE-2 utilise une représentation hiérarchique des actions, qui permet une meilleure compréhension du rôle des actions lorsqu'elles sont complexes, et une amélioration de l'efficacité de la construction des plans, en travaillant à divers niveaux de détails.

La syntaxe de description des actions est la suivante:

Operator: NOM
Arguments: liste d'arguments de l'action.
Purpose: ensemble des buts que réalise l'action.
Precondition: ensemble des conditions d'application de l'action (peut contenir des disjonctions).

Il est possible de spécifier des préconditions vraies sur un intervalle de temps (protect-until). Par défaut les préconditions sont vraies durant toute l'action (ie: jusque l'application de la dernière action décrite dans le plot).

Plot: ensemble de *buts* et d'*actions* qui compose l'***operator*** NOM et qui permettent d'accomplir les buts de ***purpose***. Ces composantes sont soit en parallèle, soit

séquentielles.

Les actions qui s'appliquent en séquence sont décrites dans l'ordre d'application, les actions s'appliquant en parallèle sont repérées par "parallel - end parallel". Un exemple est donné dans la figure 15.

```
operator: action
    .
    .
    plot
        a1: ...
        a2: ...
        parallel
            a3: ...
            a4: ...
        end-parallel
    .
    .
```

Figure 15: Exemple d'action composée d'action en parallèle

Une telle action est représentée par le plan partiel de la figure 16. Le noeud "S" indique le début d'un ensemble d'actions non ordonnées (en parallèle), le noeud "J" indique la fin d'un ensemble d'actions non ordonnées.

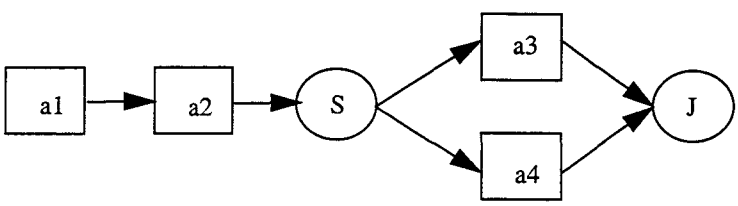


Figure 16: Représentation graphique de l'action de la figure 15

Les *actions* peuvent utiliser des ressources; ce sont des arguments particuliers déclarés comme ressources. Une ressource peut être produite (produce), consommée (consume), on peut connaître son niveau (level). Les actions possèdent des effets. Il est possible de leur imposer une durée.

Action:	NOM-ACTION
Argument:	liste des arguments
Ressources:	liste des arguments pris comme ressource
Effects:	liste des effets (les effets peuvent décrire des consommations de ressource)
Duration:	durée numérique.

- **Des contraintes**

SIPE-2 accepte des contraintes externes, qui ne sont pas réalisées par le plan, mais qui doivent être respectées par le plan.

Soit l'action M "terminer le mur" et l'action E "installer l'électricité".
M est composée de deux actions non ordonnées:

- Me pour la finition des murs extérieurs
- Mi pour la finition des murs intérieurs.

E est également constituée de deux actions non ordonnées:
- Ee pour "Electricité extérieure" .
- Ei pour "Electricité interne" .

M E

Figure 17: Description des actions M et E

Il est alors possible de décrire une contrainte externe du type "si M et E sont réalisées sur un laps de temps communs, alors l'installation de l'électricité interne (Ei) doit être terminée avant le début de la finition des murs intérieurs (Mi)" (Ei < Mi).

Les informations recueillies sur SIPE-2 ne nous permettent pas de connaître les types des contraintes externes acceptées, ni la façon dont elles sont traitées.

3.3.4.2 Applications aux exemples

- **Une macro-action composée de A et B telles que A d di o oi B (annexe B.1)**

Il ne semble pas possible de décrire la macro-action composée des actions A et B avec la relation temporelle A d di o oi B.

En effet, les seules relations autorisées dans le formalisme de description des actions sont la relation de précédence ou le parallélisme. Or le parallélisme signifie que les actions en parallèle peuvent s'exécuter dans n'importe quel ordre: les actions A et B peuvent vérifier n'importe laquelle des 13 primitives d'Allen.

De plus, il est impossible de dire à quel moment sont produits les effets d'une action durant son déroulement; il est donc impossible de dire que l'action A produit a2 en début d'action et a3 en fin d'action. Le seul moyen serait de considérer que dA, fA, dB et fB sont elles-mêmes des actions et de les modéliser comme telles. Dans ce cas, trois difficultés apparaissent:
- l'utilisateur ne voit plus la hiérarchie des actions: les actions A, B et la macro-action disparaissent au profit de 4 composantes qui n'ont pas de sens pour lui.
- la syntaxe de description des actions dans SIPE-2 ne lui offre pas le moyen de mettre en évidence les relations temporelles existantes (figures 18 et 19). L'utilisateur doit décrire les relations entre les composantes qui traduisent les relations entre les actions A et B.
- il n'est pas possible de différencier changements instantanés et actions se déroulant sur un intervalle. Ainsi, dans SIPE-2, dA, fA, dB et fB peuvent être considérées comme des actions possédant une durée.

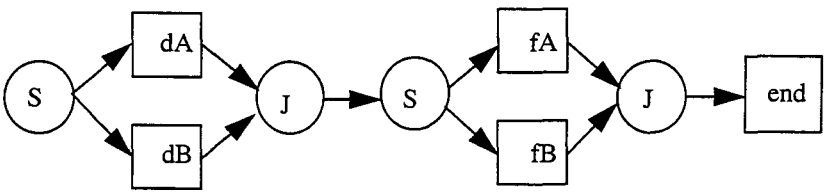


Figure 18: Représentation graphique de macro-action dans SIPE-2

```
operator:       macro-action
argument:       []
purpose:        a
precondition:   a1      (protect until a2)
                b1      (protect until b2)
plot:
parallel
        process
        action:         dA
                argument:       []
                effects:        a2

        action:         dB
                argument:       []
                effects:        b2
end-parallel
parallel
        process
        action:         fA
                argument:       []
                effects:        a3
        action:         fB
                argument:       []
                effects:        b3
end-parallel
process
action:         end
        argument:       []
        effects:        a
```

Figure 19: Description de la macro-action dans le formalisme SIPE-2

- **Appeler les secours (annexe B.2).**

A et B sont les deux actions qui composent la macro-action. Ce sont les mêmes actions que précédemment mais la relation temporelle qu'elles vérifient est A < > B.

Soit A et B sont considérées comme deux actions, alors il n'est pas possible de traduire la relation temporelle dans le formalisme SIPE-2 car les seules relations temporelles qu'il est possible de décrire sont précédence et parallélisme. On retrouve les mêmes problèmes que pour l'exemple précédent.

Soit les débuts et fins d'actions sont considérés. Or la relation temporelle A < > B

ne peut être exprimée par une conjonction de relations entre instants (en admettant que dA, fA, dB et fB soient des instants). Les seules relations acceptées sont celles qui appartiennent à l'algèbre d'instants.

L'action appeler-les-secours ne peut donc pas être représentée suivant le formalisme de SIPE-2.

- **Contraintes externes (annexe B.3)**

  - Le voyage

Le problème est d'imposer la réalisation de B3 avant ou après (B1 et B2). Est-il possible de décrire un plan partiel initial partiellement spécifié dans SIPE-2?

Supposons que la description d'un plan partiel initial soit possible. Dans SIPE la description de la contrainte temporelle entre les sous-buts est immédiate puisque seul les sous-plans (parties de plans délimitées par deux noeuds S et J) peuvent être ordonnés entre eux. Le plan décrit dans la figure 20 correspond à la description du problème suivant le formalisme de SIPE.

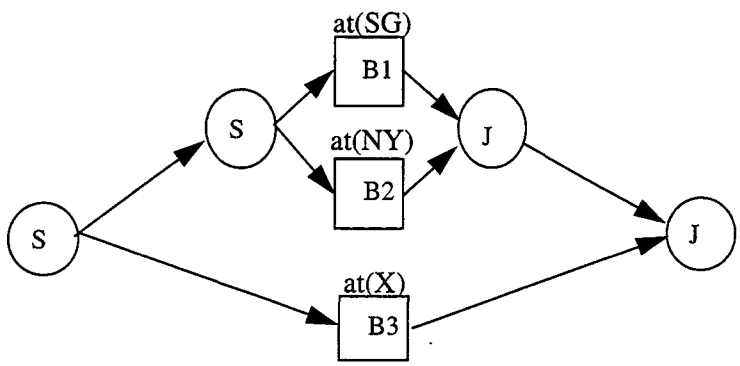


Figure 20: Description SIPE du plan partiel initial:
B3 est avant ou après (B1 et B2)

Dans SIPE-2, les actions peuvent être ordonnées les unes par rapport aux autres de façon individuelle: B3 peut donc être positionnée entre B1 et B2. Il est par ailleurs impossible de décrire la contrainte temporelle à l'aide de conjonctions de relations de précédence car la relation entre B1, B2 et B3 ne peut être décrite dans l'algèbre d'instants.

Il est peut être possible d'utiliser des contraintes externes pour imposer à SIPE-2 les contraintes temporelles adéquates entre les buts. Par exemple: si B3 < B1 alors B3 < B2; si B3 > B1 alors B3 > B2. Les caractéristiques que nous avons récupérées à propos de SIPE-2 ne nous permettent pas de conclure sur cet exemple.

  - Interdiction d'obtenir un état particulier

En supposant que le problème soit traduit en une pseudo-action dont les effets représentent la situation initiale et les buts décrits dans le "plot" de l'action, et en supposant que l'interdiction soit une précondition de réalisation du plan, protégée

jusqu'à la réalisation de tous les buts, la représentation du plan initial dans SIPE-2 correspond à:

operator: plan-initial
argument: []
precondition: ¬en(piece_x, objet_y)
protect_until(b1,b2...)
plot:
goals:
b1
b2
.
.

3.3.4.3 Conclusion

SIPE-2 propose un modèle d'action déductif et hiérarchique. Ce modèle simplifie la tâche de l'utilisateur pour qui le nombre d'actions à modéliser est réduit grâce aux règles du domaine, et pour qui la compréhension des actions est facilitée grâce à leur structure hiérarchique.

Dans SIPE-2 toute action possède une durée, même les actions primitives. Cette durée peut être symbolique, ou numérique. Le temps n'est pas explicitement représenté. Les actions sont positionnées par la seule relation de précédence. Des actions non ordonnées sont en parallèle: elles peuvent être appliquées dans n'importe quel ordre, y compris simultanément.

Cependant, il n'est pas possible de poser des contraintes "plus fines" sur les relations temporelles entre les actions. Ainsi, il est impossible de spécifier qu'une action doit se dérouler pendant l'application d'une autre action. Les relations d'Allen ne sont pas prises en compte dans la description des actions et du plan.

Par ailleurs, il n'est pas possible de spécifier à quels moments de l'application d'une action sont produits ses effets. Ce qui interdit la description de certaines actions. Les actions sont davantage perçues comme des process dont on connaît les effets de manière générale, que comme un ensemble de changements positionnés dans le temps, permettant de raisonner plus précisément sur le déroulement de l'action.

3.3.5 IxTeT

3.3.5.1 Caractéristiques

IxTeT ([LAR 94], [GHA 94]) est un planificateur temporel. La prise en compte du temps est explicite: la modélisation du temps est basée sur l'instant, les relations temporelles autorisées sont les relations entre instants.

IxTeT sépare clairement les contraintes temporelles, la description des états du monde et les ressources, ce qui lui permet de gérer les trois concepts de façon indépendante et de développer des algorithmes spécifiques.

La représentation des actions (appelées ***tâches*** dans IxTeT) et des états du monde est basée sur une logique temporelle réifiée. Deux prédicats servent à décrire assertions (hold) et changements (event) définis par:

- ***hold***(attribut : val, (t1. t2)) signifie que l'attribut prend la valeur val pour tout

instant t tel que $t1 \leq t < t2$

- ***event***(attribut : (val1,val2), t) signifie que l'attribut passe de la valeur val1 à la valeur val2 à l'instant t.

  event(attribut : (val1,val2), t) ≡

  $\exists t1 \exists t2 (t1 < t < t2) \wedge$ hold(attribut : val1, (t1.t)) $\wedge$ hold(attribut : val2, (t.t2))

Un changement est supposé instantané.

Les états sont décrits par un ensemble d'***attributs*** prenant des valeurs dans un domaine fini. Par exemple, dans le monde des cubes: l'attribut cube désigne un cube. Si trois cubes a, b, c sont considérés alors l'attribut cube prendra ses valeurs dans l'ensemble {a, b, c}. Si on(X) est l'attribut désignant la position du cube X, alors on(X) pourra prendre ses valeurs dans { a, b, c, table, hand }. La représentation du monde par attribut-valeur permet de gérer naturellememt des contraintes du domaine telle que la non ubiquité, en contraignant le domaine de valeurs des attributs.

Une tâche est représentée par un ensemble d'assertions et de changements. Elle peut être représentée par un ensemble d'instants partiellement ordonné et peut donc être assimilée à un plan partiel.

IxTeT autorise la description hiérarchique des tâches pour simplifier leur écriture et améliorer leur compréhension.

Une tâche possède des conditions d'applications, des effets, des composantes, des contraintes temporelles et des contraintes d'instanciation. La syntaxe de description d'une tâche est donnée dans la figure 21, et un exemple est donné dans la figure 22.

```
Task Nom_tâche {

        args = variables externes de la tâche (d'instanciation, temporelles);

        timepoints liste des points temporels de la tâche;
        variables liste des variables internes de la tâche;

        conditions = { hold1; ... ;holdn };

        effects = {event1; ...; eventp };

        subtasks = liste des sous-tâches;

        contraints = liste des contraintes temporelles
                        et d'instanciation sur les variables de la tâche
}
```

Figure 21: Syntaxe de description d'une tâche dans IxTeT

Les conditions d'application d'une tâche sont réparties dans les conditions et dans les effets puisque les effets sont définis à partir du prédicat event qui est lui même défini à l'aide du prédicat hold. Il traduit le passage de la valeur val1 à la valeur val2 d'un attribut à un instant donné, ce qui implique que l'attribut doit avoir la valeur val1 sur un intervalle se terminant en l'instant de changement. Ainsi, des conditions d'application de l'action déplacer_x_y_z (figure 22) sont par exemple:

état(X): libre pour débuter l'action

état(Z): libre et sur(X): main pour effectuer le changement en t3

```
Task déplacer_x_y_z {

        args = (X, Y, Z) ( début, fin );
        timepoints t2, t3;
        variables

        event ( état(X) : (libre, non-libre), début );
        event ( état(X) : (non-libre,libre), fin );
        event ( état(Z) : (libre, non-libre), t3 );
        event ( état(Y) : (non-libre,libre), t2 );
        event ( sur(X) : ( Y, main), t2 );
        event ( sur(X) : (main, Z), t3 );
        holds( état(X) : non-libre, (début. fin) );
        holds( sur(X): main, (t2. t3) );

        subtasks
        contraints = { début<t2; t2 < t3; t3 < fin }
}
```

Figure 22: Description du déplacement d'un cube X de Y à Z.

La définition du changement semble interdire la description d'actions modifiant des attributs dont la valeur initiale est inconnue. Par exemple, l'action "la personne X prévient la personne Y du fait FAIT" ne peut pas être modélisée dans le formalisme IxTeT sans supposer que forcément Y ne connaît pas le fait FAIT avant que X ne la prévienne. Cette action peut être simplement modélisée dans le formalisme de TCLP (figure 23). Nous donnons les caractéristiques de ce formalisme dans le chapitre suivant. L'interprétation d'une telle action est alors:

si X connaissait déjà le fait FAIT alors rien n'est modifié.

si X ne connaissait pas le fait alors X le connaît.

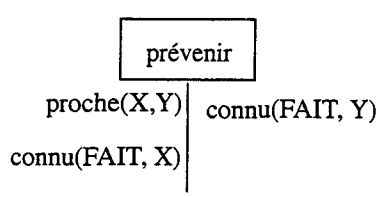


Figure 23: Représentation TCLP de l'action prévenir

Les contraintes sur les variables sont des contraintes d'appartenance à un domaine, de différenciation ou d'instanciation.

Les contraintes temporelles sont soit symboliques soit numériques. Les contraintes symboliques sont les relations temporelles qu'il est possible de poser entre des instants, composées des primitives <, > et =. Seules les relations entre instants sont admises. Les contraintes numériques portent sur l'intervalle de temps entre deux instants. La donnée de sa valeur minimale et de sa valeur maximale permet l'expression de contraintes imprécises entre les instants. Les contraintes numériques ne peuvent pas être disjonctives. Elles sont de la forme i1 - i2 in [min, max].

Les composantes d'une action (ou sous-tâches) ont la même structure que l'action.

Toute tâche est transformée en un IxTeT (Indexed Time Table):
- un ensemble d'événements
- un ensemble d'assertions
- un ensemble de contraintes entre instants
- un ensemble de contraintes sur les variables.

3.3.5.2 Application aux exemples

- **Une macro-action composée de A et B telles que A d di o oi B (annexe B.1).**

IxTeT accepte une hiérarchie de description. Il est donc possible de décrire la macro-action à partir de la description indépendante des actions A et B.

La relation A d di o oi B est une relation qui appartient à l'algèbre d'instants car elle peut être décrite par une conjonction de relations binaires entre instants: dA < fB et dB < fA avec dA, dB les débuts des actions et fA, fB leurs fins.

La description des actions A et B est donnée dans la figure 24, celle de la macro-action est donnée dans la figure 25. Nous utilisons de façon générique "att" pour désigner les attributs..

```
Task A {
    arg () (begin, end)

    hold( att: a1, (?T, begin) )

    event( att: (?X, a2), begin) )
    event( att: (?Y, a3), end) )
    }
```

```
Task B {
    arg () (begin, end)

    hold( att: b1, (?T, begin) )

    event( att: (?X, b2), begin) )
    event(att: (?Y, b3), end) )
    }
```

Figure 24: Description de A et B

```
Task macro-action {
    arg () (begin, end)

    time-point (dA, fA, dB, fB)

    event (att: (?X, a): end)

    subtasks = {        task ( A (), (dA, fA) );
                        task ( B (), (dB,fB) ) }

    contraintes = {     dA < fB;
                        dB < fA }
```

Figure 25: Description de la macro-action

Le fait de décrire uniquement des relations entre instants peut diminuer la lisibilité et la compréhension des actions. Ainsi, au lieu de décrire la relation A o B, il est nécessaire de décrire l'ensemble des contraintes: dA < dB, dB < fA, fA < fB.

- **Appeler les secours (annexe B.2).**

A et B sont les mêmes actions que précédemment. La contrainte temporelle est A < > B. Cette contrainte n'appartient pas à l'algèbre d'instants car elle fait intervenir une disjonction de relations entre plus de 2 instants: $fB < dA \vee fA < dB$. Il n'est donc pas possible d'exprimer cette contrainte, ni de la traiter en utilisant les algorithmes de propagation des contraintes temporelles.

H. Laruelle remarque que de telles contraintes sont gérées dans deux branches distinctes de l'arbre de recherche ([LAR 94]). Il ne spécifie cependant pas comment elles sont exprimées.

- **Contraintes externes (annexe B.3)**

  - Le voyage

Dans IxTeT le plan initial est une tâche à l'instar des tâches utilisables par le plnaificateur pour construire les plans.

La disjonction temporelle issue de ce problème n'appartient pas à l'algèbre d'instants. Il n'est donc pas possible de la décrire dans les contraintes du plan initial (le plan initial étant décrit comme une tâche). Le problème est alors scindé en deux problèmes l'un avec B3 avant (B1 et B2), l'autre avec B3 après (B1 et B2).

```
Task init1 {
    arg () (begin, end)

    time-point (b1, b2, b3)

    explained ... // description de la
                  // situation initiale

    hold (at(SG), (b1, ?T1) )
    hold (at(NY), (b2, ?T2) )
    hold (at(X), (b3, ?T3) )

    contraintes { b3 < b1; b3 < b2 }
    }
```

```
Task init2 {
    arg () (begin, end)

    time-point (b1, b2, b3)

    explained ... // description de la
                  // situation initiale

    hold (at(SG), (b1, ?T1) )
    hold (at(NY), (b2, ?T2) )
    hold (at(X), (b3, ?T3) )

    contraintes { b3 > b1; b3 > b2 }
    }
```

Figure 26: Description des deux plans initiaux correspondant au problème du voyage

Comme pour ZENO, deux générations sont lancées pour chaque sous-problème; il est donc possible de tirer les mêmes conclusions que celles que nous avons tirées en page page 41.

  - Interdiction d'obtenir un état particulier

On désire construire un plan pour répondre à un problème sans obtenir un résultat particulier: ici, on ne veut pas placer l'objet y dans la pièce x. Un moyen de traduire cette contrainte dans IxTeT est d'utiliser les capacités associées au modèle attribut - valeur.

Le problème est décrit de façon classique par une tâche *init* contenant la description des buts et de la situation initiale. Des règles du domaine peuvent être ajoutées afin de restreindre les valeurs des attributs (figure 27).

```
constant PIECE = {x, z, w}
constant OBJET = {y, t, u}

// une contrainte spécifiant que l'objet x ne peut pas être dans la pièce y
attribut en (?X) {
                    ?X = y
                    ?value in PIECE - {x}
                    }
// une contrainte spécifiant que tout objet différent de y se situe dans
   les pieces PIECE
attribut en (?X) {
                    ?X in OBJET - {y}
                    ?value in PIECE
                    }
```

Figure 27: Description de l'interdiction d'obtenir un résultat précis dans le formalisme IxTeT

Cette description semble moins naturelle que la seule description du fait interdit. Cependant les attributs valués permettent de restreindre facilement les valeurs possibles d'un attribut, et donc d'énoncer des contraintes naturellement. Ainsi, si un attribut prend deux valeurs différentes au même instant, il y a incohérence. Par exemple, si l'attribut en(y) = z et en(y) = w alors il y a incohérence; la non-ubiquité des objets est traitée automatiquement sans règle du domaine.

3.3.5.3 Conclusion

IxTeT possède de grandes capacités de description:

- l'utilisation d'une description hiérarchique lui permet de décrire plus facilement et de mieux comprendre les tâches complexes.
- l'utilisation des instants lui permet de décrire facilement les actions, comparativement à l'utilisation des intervalles et des assertions temporelles.
- les définitions distinctes des assertions et des changements lui permettent de décrire des changements ponctuels persistants ou non.

Une action est un ensemble d'assertions et de changements positionnés temporellement sur toute la durée de l'action (décrite par un couple d'instants). Ainsi, le déroulement d'une action peut être spécifié de façon détaillée (conditions maintenues, effets intermédiaires...).

IxTeT autorise des contraintes numériques et symboliques. Il est donc capable d'imposer une durée maximale au plan.

L'utilisation d'attributs multivalués facilite l'expression de certaines contraintes (non ubiquité), mais peut rendre artificielle l'expression d'autres contraintes (cas 3.2: interdiction d'obtenir un résultat particulier). Il ne semble pas possible de décrire des règles liant deux sous-domaines de deux attributs en fonction de la valeur d'un autre attribut sans devoir lister tous les cas possibles.

La définition du changement à partir de celle de l'assertion nécessite la connaissance de la valeur de l'attribut avant le changement: il n'est donc pas possible de

décrire des actions qui peuvent être réalisées sans connaître exactement la valeur de l'attribut qu'elles modifient.

L'utilisation des relations temporelles entre instants pour décrire des relations temporelles entre intervalles limite les capacités de description: seules les relations appartenant à l'algèbre d'instants (ou algèbre d'intervalles restreinte) peuvent être décrites et gérées par les algorithmes de propagation de contraintes. Par ailleurs la description de telles relations peut être alourdie (une relation entre intervalle est souvent représentée par une conjonction de relations entre les débuts et fins des intervalles).

3.4 Conclusion

La prise en compte de la durée des actions semble nécessaire pour la résolution de problèmes réels. L'utilisation d'intervalles pour représenter les actions et le monde conduit à des descriptions complexes et à une gestion du temps faisant intervenir des algorithmes de forte complexité combinatoire. Afin d'éviter cette complexité, pour faciliter la description des actions, et parce que nous désirons conserver les principes des planificateurs traditionnels, nous allons baser notre représentation de l'action sur l'instant.

Les diverses représentations étudiées dans ce chapitre ne nous satisfont pas car elles ne permettent pas toutes de répondre aux problèmes posés dans les annexes B.1, B.2 et B.3 (relations temporelles de l'algèbre d'intervalles, persistance entre deux instants non ordonnés, hiérarchie de description, effets et conditions d'application intermédiaires, facilité de description pour l'utilisateur et techniques de planification classique). Aussi nous proposons dans le chapitre suivant un modèle d'action répondant aux besoins que nous avons identifiés dans l'introduction de ce mémoire et dans les annexes.

Chapitre 4

Choix de la représentation des opérateurs

Notre modèle d'action nous permet de décrire des actions possédant une durée. Nous les nommons des ***d-actions***. Il nous permet également de décrire le ***déroulement*** des actions:

- conditions d'application intermédiaires
- effets produits pendant l'action
- conditions vraies sur un intervalle
- effets maintenus sur un intervalle.

Pour décrire les deux derniers concepts, nous introduisons la notion de ***maintien***.

Nous mettons en évidence des propriétés associées aux conditions d'application et aux effets des actions, qui permettent de mieux les spécifier et de guider par la suite les effectués par le planificateur.

Nous donnons enfin la syntaxe de description des d-actions et proposons une ***description hiérarchique*** des actions complexes facilitant leur écriture et leur compréhension. Les composantes d'une macro-action sont des d-actions positionnées temporellement à l'aide des ***64 relations d'Allen ne faisant pas intervenir l'égalité entre instants***.

4.1 La durée des actions

Nous désirons inclure la durée des actions dans leur représentation. Pour cela, nous associons à chaque action un intervalle temporel fini. Nous rappelons que cet intervalle ne possède pas de valeur numérique. Nous ne manipulons que des concepts symboliques et désirons uniquement tenir compte du fait que les actions ne sont pas instantanées.

Les actions modélisent des changements du monde. Pour des raisons de complexité, et parce que l'utilisation d'instants pour modéliser le changement semble plus naturelle et plus simple que l'utilisation d'intervalles, nous modélisons l'intervalle associé à la durée d'une action par un couple d'instants totalement ordonné représentant le début et la fin de l'action.

Par ailleurs, nous tentons de construire un modèle d'action qui puisse s'intégrer facilement dans le cadre des planificateurs traditionnels. Nous rappelons qu'un planificateur traditionnel considère que les actions sont instantanées et que seule la relation de précédence peut être utilisée pour positionner les actions les unes par rapport aux autres. Nous allons utiliser ces caractéristiques pour décrire des actions possédant une durée.

Un plan traditionnel non linéaire est un ensemble d'actions instantanées partiellement ordonné. C'est donc un ensemble d'instants partiellement ordonné, chaque instant caractérisant des changements du monde. En s'inspirant de l'association action/ instant, nous allons associer à l'instant de début et à l'instant de fin de l'intervalle représentant la durée de l'action, un opérateur de transformation instantané de type STRIPS (figure 28). Nous supposons de ce fait que tous les changements associés à un instant sont décrits dans les effets de l'opérateur.

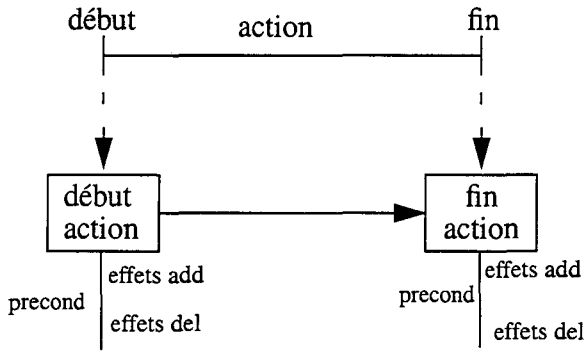


Figure 28: Représentation de la durée d'une action par un couple d'instants associés à des opérateurs de transformation

Les préconditions, effets ajoutés et retirés sont décrits par ***des conjonctions de prédicats de la logique du premier ordre***. Nous n'acceptons pas d'effets ou de préconditions disjonctives pour ne pas augmenter la complexité de la génération des plans: les accepter nécessiterait l'évaluation de tous les mondes possibles suite à l'application d'une action, et l'évaluation de toutes les situations possibles dans lesquelles pourrait être appliquée une action.

Cependant, se contenter de cette description (couples d'opérateurs de transformations totalement ordonnés) n'est pas suffisante pour représenter correctement le déroulement des actions. Il est en effet impossible de décrire des effets ou des préconditions intermédiaires.

4.2 Description des conditions d'application, du déroulement et des effets des actions

L'utilisation d'opérateurs de transformation nous permet de décrire facilement les changements provoqués par l'action en effets ajoutés et effets retirés. Ainsi, l'opérateur de début (ou de fin) de l'action peut modéliser des transformations réalisées par l'action. Nous remarquons ici que la matérialisation de la durée de l'action par des opérateurs de début et de fin nous permet déjà de distinguer les effets ayant lieu au début de l'action des effets ayant lieu à la fin.

L'extension de cette représentation à n instants nous permet de décrire le déroulement de l'action.

Le ***déroulement*** d'une action est constitué d'un ensemble de changements intermédiaires instantanés partiellement ordonnés et de maintiens de propriétés entre

deux instants.

Nous définissons une ***d-action*** par:
- un couple d'instants totalement ordonnés représentant son début et sa fin
- un ensemble d'instants intermédiaires partiellement ordonnés entre le début et la fin de l'action
- un ensemble de maintiens entre certains couples d'instants parmi ceux préalablement cités

Ces instants intermédiaires autorisent la description du déroulement de l'action. Ce déroulement peut être conditionné ponctuellement ou sur un intervalle de temps.

4.2.1 Les effets

Les changements provoqués à un instant sont représentés par les effets de l'opérateur de transformation associé à cet instant.

Ainsi, pour décrire les changements de l'action déplacer_x_y_z, nous allons associer à chaque instant caractéristique identifié (i1, i2, i3 et i4 dans la figure 8) un opérateur de transformation (figure 29). Ce qui devient vrai à l'instant considéré est décrit dans les effets ajoutés de l'opérateur; ce qui devient faux est décrit dans les effets retirés de l'opérateur.

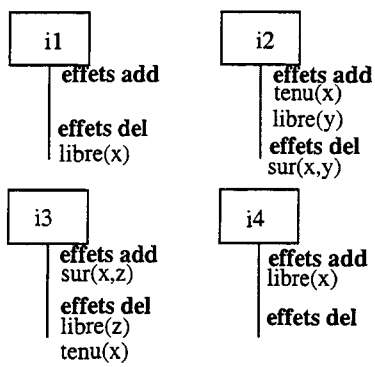


Figure 29: Modélisation des effets de l'action déplacer_x_y_z

L'utilisation d'opérateurs instantanés et de la relation de précédence nous permet de situer temporellement les changements provoqués par l'action les uns par rapport aux autres: i1 < i2, i2 < i3 et i3 < i4 (figure 30), et i1 et i4 sont les opérateurs de début et fin de l'action.

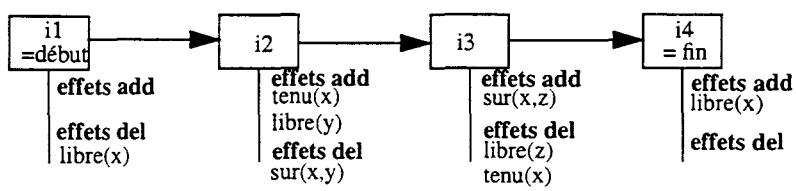


Figure 30: Positionnement temporel des effets de l'action

4.2.2 Les conditions

Les effets d'une action peuvent être conditionnés: ils sont produits seulement si des conditions sont vérifiées. Ainsi, une action ne peut pas être appliquée si toutes ses conditions d'application ne sont pas vraies dans le monde où elle doit être appliquée. De même pour les changements intermédiaires: certains changements ne peuvent avoir lieu que si certaines conditions sont vérifiées. Par exemple, les changements décrits par l'opérateur i3 dans la figure 29 ne peuvent avoir lieu que si les propriétés libre(z) et tenu(x) sont vraies juste avant (Tableau 2, page 39). ***Les conditions de réalisation des changements seront décrites dans les préconditions de l'opérateur associé*** (figure 31).

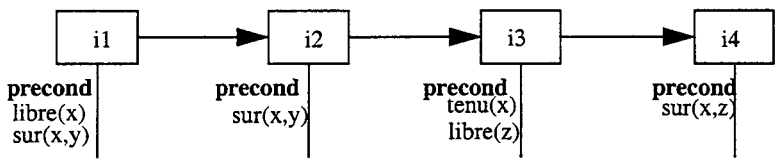


Figure 31: Description des conditions de réalisation des changements

Ayant associé un opérateur de transformation au début de l'action, les conditions d'application de l'action sont décrites à travers les préconditions de cet opérateur (precond(début-action) dans la figure 28).

Parfois les conditions de réalisation des changements sont inexistantes; l'utilisation d'instants intermédiaires sert alors à spécifier l'ordre dans lequel sont provoqués les changements ou à distinguer les instants où ils sont provoqués.

La figure 32 décrit les changements réalisés par l'action déplacer_x_y_z à l'aide d'opérateurs de transformation.

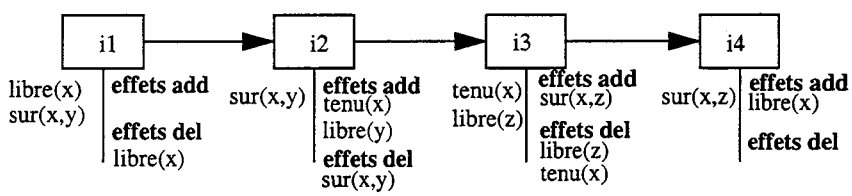


Figure 32: Modélisation des changements de déplacer_x_y_z par opérateurs de transformation

Comme nous l'avons remarqué dans la partie 2.3, il n'est pas possible de spécifier des conditions ou des effets vrais sur un intervalle en utilisant uniquement les opérateurs de transformation. Ainsi, dans la figure 32, rien ne spécifie que la propriété tenu(x) doit rester vraie entre i2 et i3. Une action peut alors être insérée entre i2 et i3 et détruire ce fait. Si le fait est rétabli avant i3 par une autre action, alors i3 peut être appliquée; sinon, l'action déplacer_x_y_z ne peut être poursuivie. Or la représentation de cette même action donnée par Allen indique que cette propriété est vraie sur un intervalle inclus entre le début et la fin de l'action (figure 6: tenu(x) est associé à I6, et I6 d Ia).

Nous ajoutons donc à notre représentation le concept de maintien.

4.2.3 Le maintien d'une propriété sur un intervalle

Nous définissons le maintien d'une propriété p entre deux instants t1 et t2 (associés à des opérateurs instantanés) comme une ***contrainte externe*** qui impose que p soit vraie à tous les instants t entre t1 et t2, sans pour autant connaître l'ordre temporel de ces deux instants.

Définition du maintien:

$maintien(p, t1, t2) \equiv (\forall t)(t < \min(t1, t2) \vee t \geq \max(t1, t2) \vee p(t))$

Le maintien définit l'intervalle ***minimal*** sur lequel est vraie la propriété maintenue.

Le maintien peut être vu comme une abstraction du lien causal utilisé pour protéger les buts établis dans les planificateurs dits à liens causaux. La différence est que le maintien ne fait pas intervenir la notion de production et consommation de propriété. Il est donc possible d'abstraire la relation de précédence entre le producteur et le consommateur. Il rejoint la notion de persistance définie par T. Dean et D. McDermott dans [DEA 87], hormis le fait que pour le maintien les instants ne sont pas obligatoirement ordonnés. Le maintien peut être comparé aux contraintes de préservation d'intervalle (IPC) définis par S. Kambhampati dans [KAM 95]. Une IPC correspond à toute contrainte posée sur le plan qui ne peut être classée ni dans l'ensemble des actions, ni dans l'ensemble des relations temporelles, ni dans l'ensemble des unifications entre variables. IPC(s, p, t) définit la condition p vraie entre s et t. S. Kambhampati utilise l'IPC pour définir le lien de protection (IPC et contrainte de précédence) comme ce que nous allons faire à l'aide du maintien. Cependant, il ne l'utilise pas pour poser des contraintes temporelles, et n'envisage pas le cas où s et t ne sont pas ordonnés. En effet, la réparation associée à un conflit avec une IPC(s, p, s') suppose que s est avant s'([KAM 95], page 196).

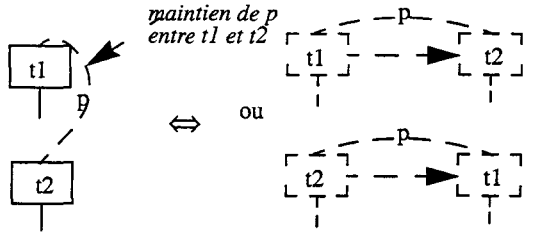


Figure 33: Représentation graphique du maintien

Le maintien (p,t1,t2) est représenté graphiquement par une ligne courbe identifiée par p, joignant les opérateurs de transformation liés à t1 et t2 (figure 33).

Soit $t1^-$ et $t1^+$ (respectivement $t2^-$ et $t2^+$) les instants associés au monde juste avant et juste après t1 (respectivement t2). Le schéma de la figure 34 représente l'intervalle sur lequel la propriété p est vraie lorsqu'elle est maintenue entre t1 et t2: p est vraie en Mt, $\forall t, t1^+ \leq t \leq t2^+$; Mt est la description du monde à l'instant t. Nous supposons que si p est vrai en t1 alors p est vrai en $t1^+$: aucune action ne peut avoir lieu entre ces deux instants.

Le maintien représente une contrainte externe qui interdira l'application de toute action détruisant p entre t1 et t2.

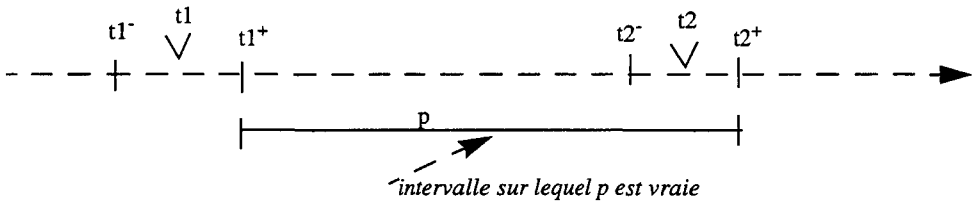


Figure 34: Description du maintien(p, t1, t2):
p est vraie en Mt avec t compris entre $t1^+$ et $t2^+$

Nous décrivons dans le tableau 3, l'utilisation et la signification du maintien entre deux opérateurs (si la composition du maintien avec des préconditions et des effets possède un sens; cohérence dans le tableau).

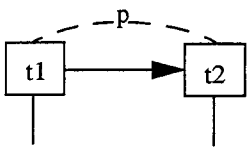
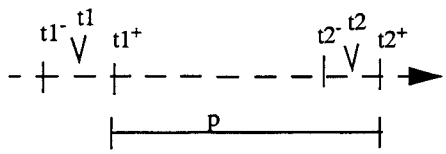
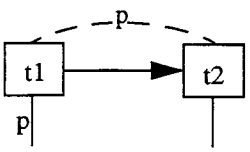
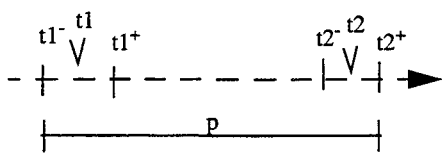
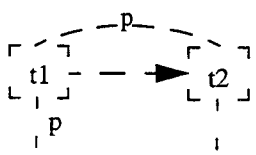
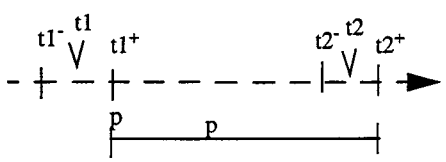
Modèle d'utilisation	Interprétation	
	***Sens***	***p est vrai entre t1 et t2***
	Intervalle	
(0)	***Cohérence***	***oui: contrainte externe sur tous les états entre t1 et t2***
	***Sens***	***p est vrai pour appliquer t1 et entre t1 et t2***
	Intervalle	
(1)	***Cohérence***	***oui: condition d'application maintenue***
	***Sens***	***p est produit en t1 et est maintenu vrai jusque t2***
	Intervalle	
(2)	***Cohérence***	***oui: p est un effet produit maintenu***

Tableau 3 : Les diverses utilisations du maintien

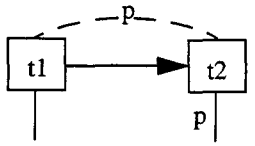
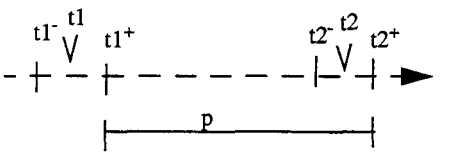
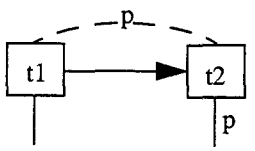
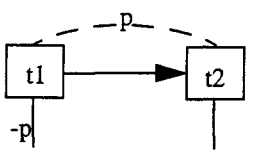
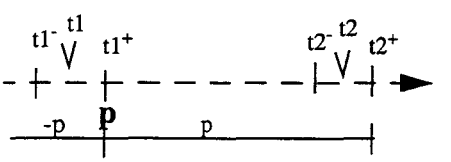
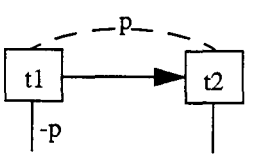
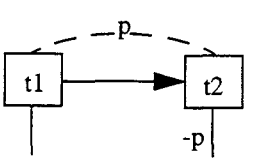
Modèle d'utilisation	Interprétation	
 (3)	*Sens*	***p doit être vrai entre t1 et t2, et p doit être vraie en t1***
	Intervalle	
	Cohérence	***oui, mais ce cas est inclus dans le maintien***
 (4)	*Sens*	***p doit être vrai entre t1 et t2, et p est produit en t2***
	Intervalle	/
	Cohérence	***<u>non:</u> t2 doit vérifier ce qu'il produit or un opérateur de changement ne peut pas produire un effet nécessaire en précondition***
 (5)	*Sens*	***p doit être vrai entre t1 et t2, et -p doit être vrai juste avant t1***
	Intervalle	
	Cohérence	***oui si p est produit en t1*** ***<u>non</u> sinon***
 (6)	*Sens*	***p doit être vrai entre t1 et t2, et p devient faux en t1***
	Intervalle	/
	Cohérence	***<u>non:</u> p est vrai et faux au même instant***
 (7)	*Sens*	***p doit être vrai entre t1 et t2, et p doit être faux en t2***
	Intervalle	/
	Cohérence	***<u>non:</u> p est vrai et faux au même instant***

Tableau 3 : Les diverses utilisations du maintien

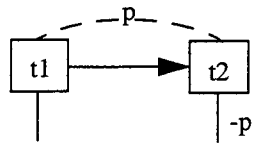
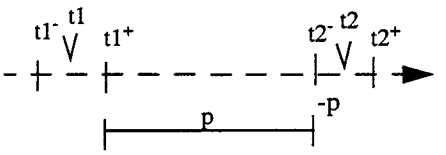
Modèle d'utilisation	Interprétation	
	***Sens***	***p doit être vrai entre t1 et t2, et p est rendu faux en t2***
	Intervalle	
(8)	***Cohérence***	***oui: condition externe de déroulement entre t1 et t2, détruite en t2***

Tableau 3 : Les diverses utilisations du maintien

4.3 Application à l'action déplacer_x_y_z

4.3.1 Application du cas (1)

Dans l'action déplacer_x_y_z la condition sur(x,y) doit être vraie pour débuter l'action (à l'instant i1 = début) mais aussi jusqu'à l'instant intermédaire i2. Ceci est traduit par:

- sur(x,y) est une précondition de i1 et
- sur(x,y) est maintenue entre i1 et i2. D'où la représentation graphique de la figure 35

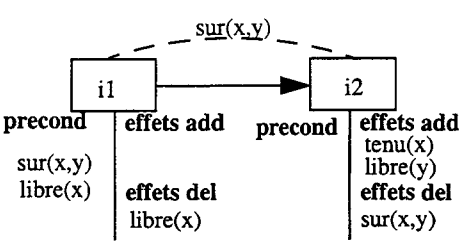


Figure 35: Maintien de la condition sur(x,y) dans l'action déplacer_x_y_z

4.3.2 Application du cas (2)

Dans l'action déplacer_x_y_z, le fait tenu(x) établi en i2 est, selon la description de la figure 8 et du tableau 2, maintenu jusqu'en i3. Cette caractéristique est traduite dans notre modèle par:

- tenu(x) ajouté en i2 et
- maintien(tenu(x), i2, i3). (figure 36)

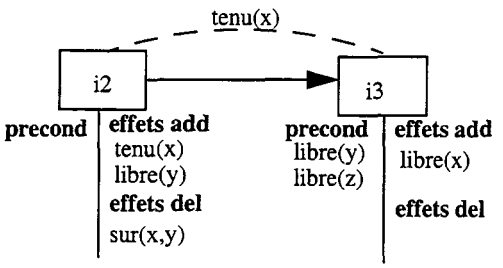


Figure 36: tenu(x) est un effet maintenu entre i2 et i3

Si le maintien n'est pas spécifié, le fait tenu(x) est vrai tant que rien ne permet de dire qu'il est faux, ie, tant qu'aucun changement du monde ne le détruit. Par ailleurs, ne pas spécifier que tenu(x) est vrai entre i2 et i3 peut conduire à des plans où le cube x est alternativement tenu, posé, tenu pendant le déroulement de l'action. Celle-ci ne pourra s'achever que si la condition tenu(x) est vraie. (figure 37)

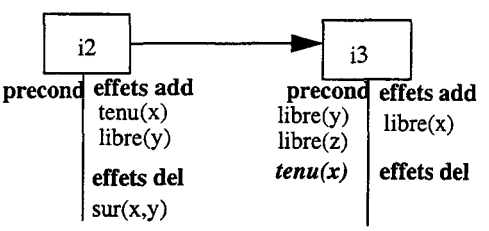


Figure 37: tenu(x) est établie en i2, peut être successivement détruit et rétabli entre i2 et i3, mais doit être vraie pour réaliser les changements décrits en i3.

4.3.3 Modélisation de l'action déplacer_x_y_z

Nous sommes maintenant capable de décrire l'action déplacer_x_y_z suivant la définition donnée dans [ALL 83b] (figure 38).

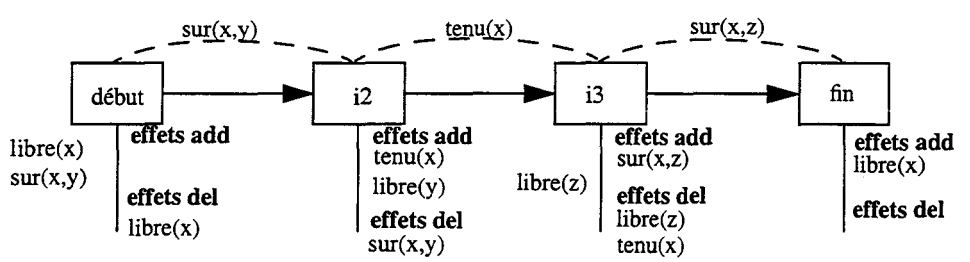


Figure 38: Description de la d-action déplacer_x_y_z

Afin de limiter la combinatoire due aux choix d'actions faits par le planificateur pour atteindre les buts et sous-buts fixés, il est possible de graduer l'importance des effets d'une action et de spécifier ses préconditions.

4.4 Des priorités associées aux préconditions et aux effets des opérateurs

4.4.1 Deux types d'effets

4.4.1.1 Effets primaires

La distinction que nous faisons entre effets primaires et effets secondaires correspond à la distinction entre effets principaux et effets déduits dont nous avons parlée dans la présentation du modèle d'action déductif (paragraphe 2.2.2). Cependant, si cette distinction a été réalisée dans le but de simplifier la représentation des actions (en ne conservant que les effets principaux dans leur description), nous la réalisons dans le but de diminuer le choix des actions lors de la construction d'un plan.

Ainsi, ***nous limitons le choix des nouveaux établisseurs d'un but ou d'un sous-but*** (ie, les effets ajoutés d'une action qui ne se trouve pas dans le plan), ***aux effets primaires de l'action.***

Par exemple, si dans l'action déplacer_x_y_z, on choisit de modéliser l'utilisation d'un bras de robot pour manipuler les cubes, on peut dire que le bras doit être libre pour pouvoir être utilisé par l'action, qu'il est occupé pendant toute l'action et qu'il est rendu libre lorsque l'action est terminée (figure 39).

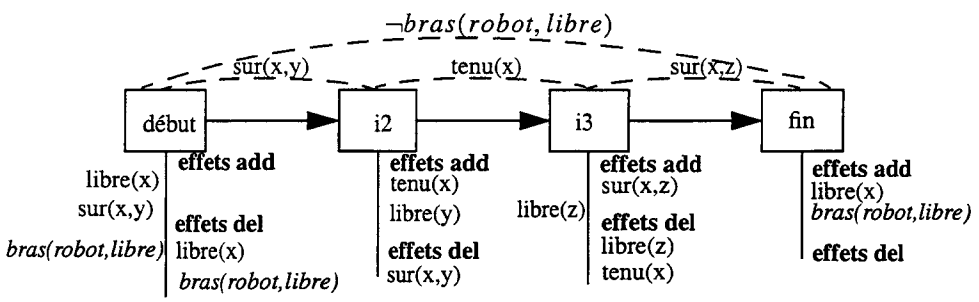


Figure 39: Description de l'action déplacer_x_y_z avec bras de robot

Il semble évident que l'action déplacer_x_y_z ne peut pas être insérée dans un plan dans le but d'établir le fait bras(robot,libre). Cet effet ne représente pas le résultat essentiel de l'action. C'est un effet secondaire qui complète la description des changements provoqués par l'action et qui pourra être utilisé une fois l'action insérée (pour réaliser un autre but) dans le plan.

Les effets primaires d'une action sont ses résultats principaux. Ils représentent les raisons pour lesquelles une action est appliquée.

Dans l'action déplacer_x_y_z les effets primaires peuvent être libre(y) dans i2 et sur(x,z) dans i3.

4.4.1.2 Effets secondaires

Ce sont toutes les propriétés du monde qui changent lors de l'application d'une action mais qui ne représentent pas ses résultats principaux.

Ils sont nécessaires à la cohérence de la description des changements de l'action.

Souvent les effets secondaires d'une action sont plus nombreux que les effets primaires. Dans l'action déplacer_x_y_z, les effets secondaires sont tous les effets qui n'ont pas été définis comme primaires.

4.4.2 Trois types de préconditions

Nous distinguons trois types de préconditions dans le but de réduire le nombre d'actions que l'on peut choisir en fonction de l'état du plan.

Dans le but de simplifier la génération des plans, nous avons choisi de ne pas utiliser de règles du domaine pour simplifier la représentation des actions (paragraphe 2.2.2). Ainsi, l'action "déplacer un cube c d'un cube x à un cube y" est représentée par deux opérateurs dans le cas où les cubes peuvent supporter un ou deux cubes.

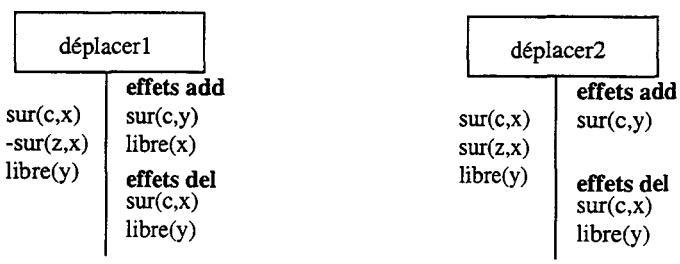


Figure 40: Représentation de déplacer_x_y_z dans le cas où un cube peut supporter un ou deux cubes

Supposons que sur(c,y) ait été choisi comme effet primaire de ces deux modèles d'action. Ces deux actions peuvent donc être choisies indifféremment pour établir le but sur(a,b). Supposons alors que déplacer2 soit choisie.

Cette action ne peut être appliquée que si toutes ses préconditions sont vérifiées. Alors, il semble logique de ne pas utiliser une seconde action déplacer pour établir les préconditions de la première: on ne va pas déplacer c sur x et z sur x pour pouvoir effectuer ensuite le déplacement de c depuis x vers y. Ou encore, dans le cas où seul c est sur x, on ne va pas déposer un second cube z sur x pour pouvoir appliquer l'action déplacer2.

Cette hypothèse nous empêchera alors de choisir déplacer2 si ces deux conditions ne sont pas déjà vérifiées; déplacer1 sera préférée.

Les préconditions sur(c,x) et sur(c,z) ont donc des particularités par rapport à la troisième précondition libre(y). En effet, celle-ci peut ne pas être vérifiée si un cube t est sur le cube y. Dans ce cas, l'utilisation d'une nouvelle action peut être nécessaire (et cohérente) pour établir le fait libre(y).

Afin de distinguer ces cas, nous particularisons les préconditions des opérateurs de tranformation.

4.4.2.1 Préconditions immuables

Ce sont des préconditions qui ne peuvent être établies par aucune action.

Ainsi, des préconditions caractérisant des données terrain (présence d'une rivière, d'une montagne, le fait qu'une ville possède un aéroport...) ne peuvent être établies par des actions.

Une précondition immuable est vérifiée dans la description du monde courant au moment où l'on insère l'action dans le plan, ou ne le sera jamais.

4.4.2.2 Hold-préconditions

Ce sont les préconditions qui doivent être établies dans le monde courant ou dans le plan pour que l'action concernée puisse être choisie pour établir un but.

Ainsi, sur(c,x) et sur(z,x) sont des hold-préconditions de l'action déplacer2; sur(c,x) et $\neg$sur(z, x) sont des hold-préconditions de l'action déplacer1.

4.4.2.3 Préconditions (classiques)

Ce sont les préconditions de l'opérateur qui ne sont pas particularisées, c'est-à-dire ni de type immuable, ni de type hold.

libre(x) est une précondition (classique) de déplacer1 et déplacer2.

La spécification des effets et des préconditions des actions sera utilisée à la fois pour limiter les choix des actions et pour déterminer l'ordre dans lequel seront traités les sous-buts non réalisés. Dans le paragraphe suivant nous décrivons la syntaxe de représentation des d-actions.

4.4.3 Remarque sur la caractérisation des effets et des préconditions

Les spécifications dont nous venons de parler sont fonction du sens que l'on veut donner à l'action et dépendent d'un grand nombre de paramètres. Ce choix est très souvent subjectif et très informel; certaines approches tentent d'automatiser la graduation des effets des opérateurs.

E. Fink et Q. Yang ([FIN 95]) montrent que la distinction entre les effets (classés en effets primaires et secondaires) et les préconditions (auxquelles on attribue des priorités de traitement), ainsi que le choix des effets primaires / secondaires et des différentes préconditions influencent l'efficacité du planificateur: la construction d'un plan peut être accélérée (de façon exponentielle) s'il est possible de distinguer des niveaux d'importance parmi les effets des actions utilisées. Cependant, si le choix des effets primaires n'est pas cohérent (si par exemple aucun effet primaire d'une action ne peut établir un but) la distinction effets primaires / secondaires peut conduire à une génération des plans nettement moins efficace, voire à l'incomplétude du planificateur. Pour éviter les choix subjectifs de l'utilisateur, ainsi que les risques d'échec suite à un mauvais choix, ils proposent dans [FIN 93] un algorithme permettant de les déterminer de façon automatique. Cet algorithme est basé sur un coût attribué à chaque opérateur (permettant de déterminer le coût total d'un plan) et sur l'augmentation maximale du coût total acceptée par l'utilisateur: l'algorithme tente de modifier un plan solution par retrait d'opérateur. Si suite au retrait d'un opérateur op du plan une précondition p ne

peut plus être établie, alors p est définie comme effet primaire de l'opérateur op. Cependant, si le choix des effets primaires et secondaires s'effectue automatiquement, ce sont désormais les coûts associés à chaque opérateur et l'augmentation précédement évoquée qui doivent être déterminés par l'utilisateur, et donc de façon subjective.

Nous avons également remarqué sur des jeux d'essais issu du monde des cubes, que le choix des effets primaires et secondaires (ainsi que la spécification des préconditions) dépend du problème à résoudre: si un premier choix conduit à l'amélioration de l'efficacité du planificateur pour un problème, elle peut conduire à un échec sur un autre problème.

Par ailleurs, il est apparu que l'utilisation conjuguée des spécifications des effets et des préconditions peut diminuer les risques de bouclage du type "empiler-dépiler-empiler- dépiler ...". Mais leur utilisation reste très limitée toujours à cause des critères subjectifs utilisés.

4.5 Description du plan initial

4.5.1 Situation initiale et situation finale

Un plan peut être vu comme une d-action permettant de passer d'une situation initiale dans laquelle les conditions d'application de la d-action sont vérifiées, à une situation finale représentant les buts à atteindre.

Un plan possède un début et une fin; la durée d'une d-action est définie par un opérateur instantané de début et un opérateur instantané de fin.

Dans un plan, les préconditions représentent un ensemble de sous-buts à atteindre pour rendre applicables les actions. ***La situation finale peut donc être confondue avec l'ensemble des préconditions d'un opérateur fictif représentant la fin de la d-action*** (ou du plan). Lorsque ces préconditions sont réalisées, la situation finale est atteinte.

Une précondition est vraie s'il existe un effet situé avant, permettant d'établir cette précondition, et protégé par un lien causal. Si le plan est assimilé à une d-action, celle-ci est donc applicable si ses préconditions sont établies dans la situation initiale. Nous représentons ***la situation initiale par l'ensemble des effets d'un opérateur fictif marquant le début du plan.***

Nous nommons ***begin*** l'opérateur de début de plan, et ***end*** l'opérateur de fin de plan. Begin est avant toutes les actions du plan, end est après toutes les actions. La figure 41 représente un plan initial dans lequel ne sont décrites que les situations initiale et finale.

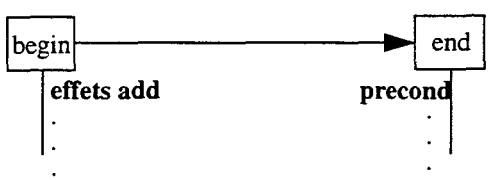


Figure 41: Description d'un plan initial

A l'instar d'une d-action, le plan initial peut contenir des buts intermédiaires et des contraintes sur les états.

4.5.2 Buts intermédiaires

Ils sont représentés par les préconditions d'opérateurs fictifs situés entre begin et end. Ils permettent de spécifier l'ordre dans lequel doivent être réalisés les buts (dans la figure 42, le but b1 doit être établi avant le but b2 et avant le but end), ou simplement spécifier que les trois buts doivent être atteints, peut importe l'ordre (buts b1, b2 et b3 dans la figure 43).

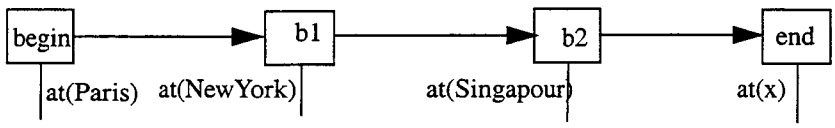


Figure 42: 3 buts à atteindre dans l'ordre b1<b2<end

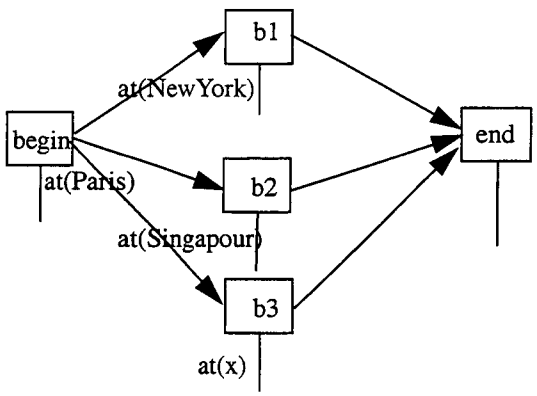


Figure 43: Trois buts à atteindre dans n'importe quel ordre

4.5.3 Des contraintes sur les états

Dans l'exemple du voyage cité en annexe B.3, et repris dans les deux exemples précédents, une contrainte est imposée sur le fait que la personne ne peut pas aller en X entre sa visite à New York et sa visite à Singapour, afin de ne pas froisser la susceptibilité de ses interlocuteurs. Cette contrainte interdit la réalisation du but at(x) entre b1 et b2, sans pour autant contraindre l'ordre dans lequel ces deux derniers sont réalisés. Le maintien est un moyen de traduire cette interdiction dans un plan traditionnel non linéaire à lien causaux. En effet, si on pose le maintien(-at(x), b1, b2), il sera impossible de réaliser at(x) entre b1 et b2 sans rendre le plan incohérent puisqu'au même moment la personne sera en x et n'y sera pas. Le plan de la figure 44 représente le problème du voyage.

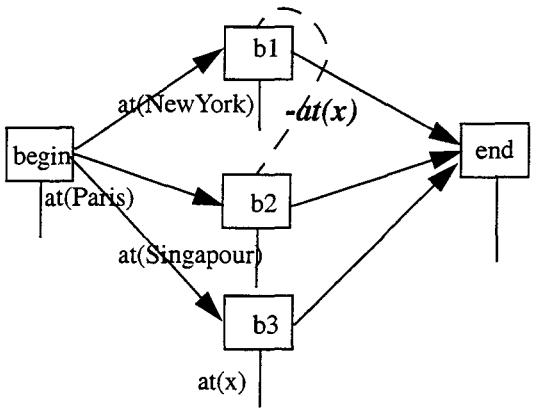


Figure 44: Trois buts b1, b2 et b3 à atteindre dans l'ordre:
$(b3 < b1 \wedge b3 < b2) \vee (b2 < b1 \wedge b3 < b1)$

Le paragraphe suivant est consacré à la syntaxe de description des d-actions.

4.6 Syntaxe de description des opérateurs

Un opérateur instantané modélise l'ensemble des changements discrets provoqués à un instant. Nous supposons que toutes les actions possèdent une durée. Dans ce sens, un opérateur instantané ne peut pas être considéré comme une action à part entière.

Seuls les opérateurs associés au début et à la fin du plan ont un sens, car ils représentent la situation initiale et la situation finale.

Deux concepts sont définis: l'opérateur instantané et la d-action. Le plan ne perçoit que les opérateurs instantanés, l'utilisateur voit essentiellement des d-actions, hormi begin et end.

4.6.1 Description des instants

Les changements instantanés sont décrits à l'aide d'opérateurs de type STRIPS. Nous décrivons ici les caractéristiques.

Un instant est associé à 4 champs et un identificateur ID:

- ***precond***
  l'ensemble des conditions du changement.
  La liste vide indique qu'il n'y a pas de préconditions.
  Soit p(x,y) une précondition;
  (p(x,y), h) est une hold-précondition et *(p(x,y), i)* est une précondition immuable.

- ***effets add***
  l'ensemble des effets ajoutés par l'opérateur de transformation. La syntaxe est identique aux préconditions.

p(x,y) est un effet ajouté primaire et *(p(x,y), s)* est un effet secondaire.

- ***effets del***
  l'ensemble des effets retirés par l'opérateur de transformation. La syntaxe est identique aux préconditions.

- ***contraintes***
  liste des contraintes de différenciation sur les variables de l'instant.
  dif(X,Y) signifie que les variables X et Y doivent être différentes.
  true indique qu'il n'y a pas de contrainte posée sur les variables de l'action.

La figure 45 présente la syntaxe de description d'un instant et la figure 46 correspond à la description de l'opérateur i2 de la figure 32.

```
( ID , instant,      precond P1 et P2 ...
                     effets  add E1 et E2 ...
                             del E3 et E4 ...
                     contraintes LISTE_CONTRAINTES)
```

Figure 45: Syntaxe de description d'un opérateur instantané

Les termes en majuscule correspondent à des variables.

```
( I2 , instant,      precond sur(X,Y)
                     effets  add tenu(X) et libre(Y)
                             del sur(X,Y)
                     contraintes ( dif(X,Y) )
```

Figure 46: Description de l'instant i2 de l'action déplacer_x_y_z

4.6.2 Description hiérarchique des d-actions

Une action peut être composée d'un ensemble de changements instantanés et de d-actions, elles-mêmes décrites par un ensemble d'instants et de d-actions: de telles actions sont couramment appelées macro-actions. Nous utilisons cette hiérarchie uniquement pour faciliter la description d'actions complexes; nous ne l'utilisons pas pour construire les plans comme ce qui est fait dans ABTWEAK ([YAN 90]), SIPE ([WIL 88]), ou O-Plan ([CUR 91]). En effet, avant d'être insérée dans le plan, toute d-action est expansée et transformée:

- en un ensemble d'opérateurs instantanés
- positionnés uniquement par la relation de précédence
- et en un ensemble de maintiens traduisant des persistances.

Une d-action possède un nom (NOM_ACTION) et des caractéristiques décrites dans les champs suivant:

- ***liste_variables***
  Nous définissons ici les *variables externes* visibles par les autres actions. Les variables externes nous permettent de faire appel à des actions déjà définies pour construire une autre action (macro-action) en conservant des liens entre les

variables de la macro-action et de ses composantes. (voir exemple figure 49, page 77).

- ***composantes***
  Description des instants contenus dans la d-action et les d-actions la composant.

  - Si la composante est un instant, elle est décrite directement dans l'action en suivant la syntaxe de description de l'instant.
  - Si la composante est une d-action, elle est définie ailleurs, et on lui attribue dans la d-action nommée NOM_ACTION un identificateur (ID).
    Elle est appelée en suivant la syntaxe: (ID, NOM, liste_variables LISTE). Cette liste de variables correspond aux variables externes de l'action identifiée par ID et permet les liens entre ses variables et celles de la d-action nommée NOM_ACTION.

- ***index***
  Définition des clés de recherche de l'action dans la bibliothèque de modèles d'actions.
  Les clés de recherche d'un instant sont ses effets primaires ajoutés.
  Soient C1, C2 et C3 les composantes de la d-action. Il est possible de définir les clés de la d-action comme sous_ensemble ou ensemble des index de C1, C2 et C3. Par exemple: index = index (C1) et index(C2) signifie que la d-action ne peut être recherchée que par les clés de C1 et C2.
  La liste vide est interdite.

- ***début***
  Identification de l'instant de début de la d-action. Il appartient à la liste des composantes:

  - soit c'est un instant défini dans la d-action: il est repéré par son identificateur.
  - soit c'est le début d'une des composantes de l'action: début(IDENTIFICATEUR).

- ***fin***
  Identification de l'instant de fin de la d-action. Il vérifie les mêmes propriétés que le début.

- ***contraintes_temporelles***
  Ce sont les relations temporelles symboliques entre composantes et instants qui définissent l'ordre dans lequel sont réalisés les changements. Par défaut, toutes les composantes sont situées entre l'instant de début et l'instant de fin définis dans les champs correspondant.
  Les relations temporelles entre les composantes d'une d-action sont:

  - la relation de précédence entre instants
  - toute relation temporelle d'Allen qui peut être exprimée uniquement à l'aide de la seule relation de précédence, c'est-à-dire toute relation appartenant aux 64 relations entre 2 d-actions issues des 6

primitives d'Allen ne faisant pas intervenir l'égalité entre instants (<, >, d, di, o, oi). Nous construisons et décrivons ce sous-ensemble dans la partie II de ce mémoire.
La liste vide indique qu'il n'y a aucune contraintes temporelles entre les composantes de l'action.

- ***contraintes_variables***
  Il est possible de poser des contraintes de différenciation sur les variables visibles depuis la d-action, c'est-à-dire les variables externes de la d-action et de ses composantes, ainsi que les variables apparaissant dans les instants.
  Les contraintes d'égalité sont posées naturellement en nommant identiquement les variables concernées.
  La contrainte "true" indique qu'il n'y a aucune contrainte entre les variables de la d-action, hormis celles définies dans les composantes.

- ***maintien***
  Les maintiens de propriété sont définis dans ce champ. Un maintien est toujours défini entre deux instants. On ne peut donc définir que les maintiens entre les instants accessibles depuis la d-action. La syntaxe d'un maintien est la suivante: *(propriété, identificateur_instant1, identificateur_instant2).*
  La liste vide indique qu'il n'y a pas de maintien.

La syntaxe de description d'une d-action est donnée dans la figure 49.

NOM_ACTION::: ***liste_variables*** VARIABLES EXTERNES
composantes COMP1 *&* COMP2 ...
index LISTE DES INDEX DE CERTAINES DES COMPOSANTES
contraintes_variables LISTE_CONTRAINTES
contraintes _temporelles CT1 *et* CT2 ...
debut DEBUT
fin FIN
maintien MAINT1 *et* MAINT2... .

Figure 47: Syntaxe de description d'une d-action

4.6.3 Exemples

Nous donnons ici deux façons de décrire l'action déplacer_x_y_z: l'une suppose que l'action est composée de 4 instants (figure 48), l'autre utilise une décomposition hiérarchique de l'action (figure 49) et fait appel aux actions prendre (figure 50) et poser (figure 51).

- Description par 4 instants

```
déplacer_x_y_z::: liste_variables [X, Y, Z]
                  composantes
                      ( DEBUT, instant,  precond      libre(X) et
                                                      (sur(X,Y), h)
                                         effets       add   [ ]
                                                      del   libre(X)
                                         contraintes (dif(X,Y))
                      ) &
                      ( I2, instant,     precond      [ ]
                                         effets       add   (tenu(X),s)
                                                           et libre(Y)
                                                      del   sur(X,Y)
                                         contraintes (dif(X,Y))
                      ) &
                      ( I3, instant,     precond      libre(Z)
                                         effets       add   sur(X,Z)
                                                      del   libre(Z)
                                                           et tenu(X)
                                         contraintes (dif(X,Z))
                      ) &
                      ( FIN, instant,    precond      [ ]
                                         effets       add   libre(X)
                                                      del   [ ]
                                         contraintes true
                      )
                  index                  index(I3) et index(I2)
                  contraintes_variables  true
                  contraintes _temporelles      I2 av I3
                  debut                  DEBUT
                  fin                    FIN
                  maintien               (sur(X,Y), DEBUT, I2)
                                         et (tenu(X), I2,I3)
                                         et (sur(X, Z), I3, FIN).
```

Figure 48: déplacer_x_y_z définie par 4 instants

- Description hiérarchique de déplacer_x_y_z

```
déplacer_x_y_z::: liste_variables [X, Y, Z]
                  composantes
                      ( A1, prendre_x_y, [X, Y]) &
                      ( A2, poser_x_z, [X, Z] )
                  index                  index(A1) et index(A2)
                  contraintes_variables  ( dif(Y,Z) )
                  contraintes _temporelles      A1 av A2
                  debut                  debut (A1)
                  fin                    fin (A2)
                  maintien               (tenu(X), fin(A1), debut(A2)).
```

Figure 49: Description de déplacer_x_y_z à l'aide de deux d-actions

- prendre_x_y

```
prendre_x_y_z:: :   liste_variables        [X, Y]
                    composantes
                        ( DEBUT, instant,   precond      libre(X) et
                                                         (sur(X,Y), h)
                                            effets       add   [ ]
                                                         del   libre(X)
                                            contraintes (dif(X,Y))
                        ) &
                        ( FIN, instant,     precond      [ ]
                                            effets       add   (tenu(X),s)
                                                               et libre(Y)
                                                         del   sur(X,Y)
                                            contraintes (dif(X,Y))
                        )
                    index                   index(FIN)
                    contraintes_variables   true
                    contraintes _temporelles [ ]
                    debut                   DEBUT
                    fin                     FIN
                    maintien                (sur(X,Y), DEBUT, FIN).
```

Figure 50: Description de prendre_x_y

- poser_x_z

```
poser_x_y_z::   :   liste_variables        [X, Z]
                    composantes
                        ( DEBUT, instant,   precond      libre(Z)
                                            effets       add   sur(X,Z)
                                                         del   libre(Z)
                                                               et tenu(X)
                                            contraintes (dif(X,Z))
                        ) &
                        ( FIN, instant,     precond      [ ]
                                            effets       add   libre(X)
                                                         del   [ ]
                                            contraintes true
                        )
                    index                   index(DEBUT)
                    contraintes_variables   true
                    contraintes _temporelles [ ]
                    debut                   DEBUT
                    fin                     FIN
                    maintien                (sur(X, Z), DEBUT, FIN).
```

Figure 51: Description de poser_x_y

D'autres exemples sont donnés dans les annexes B.1, B.2 et B.3. Ils nous ont permis de comparer les capacités de description de TCLP avec les planificateurs présentés dans le chapitre 3 (Descartes ([JOS 95]) ou Timelogic ([ALL 83b]), ZENO ([PEN 94]), IxTeT ([GHA 94]), et SIPE-2 ([WIL 90])). Nous résumons et commentons les résultats dans le tableau 4 de la conclusion page 82.

4.7 Conclusion

Une action peut être vue comme un plan partiellement ou totalement ordonné d'opérateurs de transformation instantanés (figure 52):

- elle possède un début et une fin
- les opérateurs instantanés représentent les instants intermédiaires où sont provoqués des changements. Ils sont positionnés entre eux par la relation < (ou >), ou n'ont aucune contrainte temporelle entre eux.
- des propriétés vraies sur des intervalles sont matérialisées par des maintiens entre les deux opérateurs instantanés qui représentent le début et la fin de cet intervalle.
- Lorsqu'une d-action est composée d'autres d-actions, celles-ci sont positionnées par les 64 relations temporelles d'Allen ne faisant pas intervenir l'égalité entre instants.

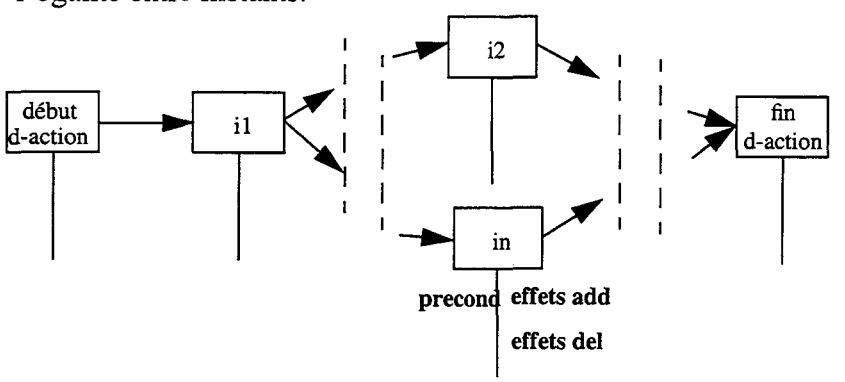


Figure 52: Représentation graphique d'une d-action

Conclusion

La représentation des actions que nous avons adoptée s'appuie sur la notion d'opérateurs de transformation de type STRIPS. Nous définissons une d-action comme un plan partiellement ordonné d'opérateurs instantanés. Cette représentation nous permet de prendre en compte la ***durée des actions*** pendant la construction des plans, et de décrire le ***déroulement*** des actions.

Suivre l'hypothèse de STRIPS nous permet de répondre au frame problem et d'assurer le déterminisme des d-actions. Nous supposons que la cohérence des d-actions est assurée par celui qui décrit les actions associées à un problème de planification.

La durée d'une d-action (intervalle associé à la d-action) est symbolisée par un couple d'opérateurs instantanés totalement ordonnés, représentant le début et la fin de l'action. La considération de la durée de l'action nous a conduits naturellement à considérer la constance d'une propriété du monde sur un intervalle. Nous avons ainsi défini le ***maintien d'une propriété*** entre deux instants, en étendant cette notion au fait que l'ordre temporel des deux instants peut être inconnu. Cette extension nous permet en effet d'exprimer des ***contraintes externes sur des propriétés du monde*** entre deux instants, sans connaître leur ordre (celui-ci sera spécifié lors de la synthèse du plan).

Si l'utilisation d'ensembles d'opérateurs de transformation instantanés pour décrire une d-action nous permet déjà d'introduire facilement la notion de durée dans les planificateurs traditionnels non linéaires, les concepts instant et précédence ne suffisent pas pour exprimer toutes les relations temporelles possibles entre deux d-actions. Or une d-action peut être décrite de façon hiérarchique, et donc être construite à partir d'autres d-actions. Ces dernières doivent alors être positionnées entre elles. Ces d-actions sont assimilables à des intervalles. Il peut donc être intéressant de décrire non pas des relations entre les instants composant ces d-actions, mais des relations entre intervalles (ie des relations de l'algèbre d'intervalles d'Allen). Nous montrons dans la partie suivante, comment, à l'aide de la seule relation de précédence, du maintien et des relations temporelles disjonctives supportées par les réparations des conflits dans les planificateurs traditionnels non linéaires à liens causaux, il est possible de décrire les 64 relations temporelles entre intervalles issues des 6 primitives d'Allen exprimables à l'aide de la seule relation de précédence (<, >, d, di, o, oi).

Avant d'expliquer notre approche, nous résumons dans le tableau 4, les caractéristiques de représentation des planificateurs que nous avons étudiés au chapitre 3.

	Descartes/ Timelogic	*ZENO*	*IxTeT*	*SIPE-2*	*TCLP*
représentation du temps	intervalle	instant	instant	?	instant
valeur numérique	oui / non	oui	oui	oui	non
relations temporelles	Allen	<, >, =	<, >, =	<, >, //	Allen sans l'égalité (voir partie suivante)
description hiérarchique	non / oui	non	oui	oui	oui
description du changement	fin de persistance: l'intervalle associé à un fait est l'intervalle maximal surlequel le fait est vrai; ailleurs il devient faux.	effet at(t, p(xo))	défini par le changement de la valeur d'un attribut (la valeur initiale doit être connue)	effet	effet d'un opérateur STRIPS
persistance de littéral négatif (-p)	impossible sans supposer qu'à l'extérieur de l'intervalle p est vrai	oui : $(\forall t)\neg at(t, p(xo))$	oui : en utilisant la valeur de l'attribut correspondant au littéral -p	?	oui: maintien(-p, t1, t2)

Tableau 4 : Comparaison de diverses représentations

PARTIE II

Description des relations temporelles entre d-actions

Introduction

Notre but est d'***autoriser des relations temporelles disjonctives entre d-actions qui n'appartiennent pas à l'algèbre d'instants***, tout en conservant une représentation basée sur l'instant et en utilisant la seule relation de précédence entre instants.Cette approche nous permet de travailler dans le cadre des planificateurs traditionnels non linéaires à liens causaux tout en considérant la durée des actions.

Nous établissons une passerelle entre un sous-ensemble des relations de l'algèbre d'Allen ne faisant pas intervenir l'égalité entre instants, et le formalisme temporel lié au fonctionnement de ces planificateurs.

Nous avons étudié la structure de ces plans, et avons identifié ***3 schémas temporels***. Un plan traditionnel non linéaire à liens causaux est une conjonction d'instanciations de ces 3 schémas.

Ces schémas font intervenir uniquement des instants, la seule relation de précédence et les réparations associées aux conflits qui existent dans de tels plans: le ***conflit maintien-maintien*** et le ***conflit maintien-effet***. Ces réparations sont des contraintes temporelles disjonctives entre instants qui n'appartiennent pas à l'algèbre d'instants, et qui interdisent certaines relations temporelles entre instants pour assurer la cohérence des plans. Ces interdictions sont directement associées à l'autorisation des relations complémentaires: en effet, si X et Y sont les deux seules solutions à un problème, le fait d'interdire X impose le choix de Y. Nous nous sommes basés sur ce principe pour imposer des relations temporelles entre d-actions à l'aide des interdictions supportées par les réparations des conflits.

Notre approche étant basée sur l'utilisation des techniques de réparations des conflits, nous étudions dans le chapitre 5 la structure des conflits des plans traditionnels non linéaires à liens causaux. Nous définissons les ***incompatibilités sémantiques***, les deux types de conflits que l'on peut détecter dans les plans et établissons la distinction entre ***conflit potentiel*** et ***conflit réel***.

Dans le chapitre 6, nous définissons les trois schémas temporels dont sont constitués les plans traditionnels non linéaires à liens causaux. ces schémas sont issus des conflits et de la relation de précédence entre instants. Nous contruisons alors une table de transition entre les 64 relations d'Allen issues des 6 primitives descriptibles à l'aide de la seule relation de précédence, et des conjonctions d'instances des 3 schémas temporels précédemment cités, compréhensibles des planificateurs traditionnels non linéaires à liens causaux.

Chapitre 5

Les conflits dans les plans traditionnels non linéaires à liens causaux

Nous étudions dans ce chapitre la structure des plans traditionnels non linéaires à liens causaux et plus particulièrement les conflits dans de tels plans. Nous définissons les ***incompatibilités sémantiques*** permettant de décrire les états incohérents du monde, ainsi que la notion de ***conflits*** détectés dans les plans lorsque deux états incompatibles sont susceptibles d'être vrais en même temps.

5.1 Les incompatibilités sémantiques

Les plans générés sont destinés à être exécutés (l'exécution peut être réelle ou simulée). Un plan est une séquence d'actions, dont l'application permet de transformer successivement le monde depuis la situation initiale jusque la situation finale. Les situations intermédiaires constituent des états du monde; ces états doivent donc être physiquement atteignables. Ainsi, si un plan conduit à une situation où Toto est en même temps à Lille et à Paris, le plan ne sera pas exécutable.

Soit A une action nécessitant la présence de Toto à Paris pendant toute son application. A se déroule sur l'intervalle IA.

Soit B une action qui produit l'effet "Toto est à Lille" en fin d'application. B se déroule sur l'intervalle IB.

Si ces deux actions sont mal positionnées l'une par rapport à l'autre, leur exécution peut conduire à un état incohérent. Ainsi, si aucune contrainte temporelle n'est posée, elles peuvent vérifier n'importe laquelle des primitives d'Allen; par exemple IA oi B (figure 53).

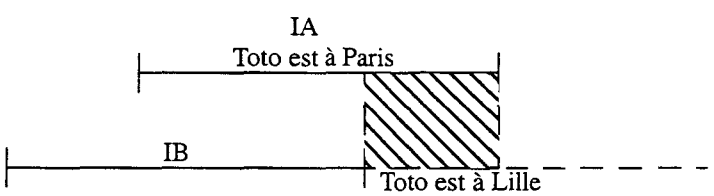


Figure 53: Exemple de situation incohérente

Dans l'intervalle commun hachuré dans la figure 53, Toto est à la fois à Paris et à Lille. Le plan associé n'est donc pas exécutable.

Le générateur de plans doit donc être capable d'interdire cette relation (ainsi que toutes celles desquelles il est possible de déduire que Toto est à la fois à Lille et à Paris). Pour cela, il doit connaître les états incohérents vis à vis du domaine. Les ***incompatibilités sémantiques*** sont un moyen pour l'utilisateur de les décrire.

5.1.1 Définition

Une incompatibilité sémantique entre deux propriétés du monde est une relation du domaine qui indique quelles sont les propriétés du monde qui ne peuvent être vraies simultanément. Elles permettent de réduire le nombre d'états possibles et donc le nombre de plans générés. Elles sont à la base de la détection des conflits.

Les incompatibilités sémantiques sont des règles du domaine interdisant des états du monde. Elles peuvent être comparées aux règles de cohérence définies par F. Garcia dans [GAR 93]. Cependant F. Garcia les utilisent pour le retour à la cohérence de la description du monde dans son processus de mise à jour. Si celle-ci n'est pas correcte suite à l'application d'une action (dont les effets ne contiennent que des faits de base), alors l'algorithme consiste à utiliser les règles de cohérence pour modifier cette description. Si plusieurs solutions sont possibles, alors est choisie la description du monde la plus proche de la précédente. La notion de proximité est définie dans sa thèse.

La notion d'incompatibilité sémantique que nous utilisons sert uniquement à augmenter le nombre d'états incohérents. Dans de nombreuses approches, seuls les états contenant une propriété et son contraire sont considérés lors de la détection des conflits. Or l'exemple de la figure 53 contient un état incohérent dans lequel Toto est à la fois à Lille et à Paris. Nous verrons plus loin dans ce paragraphe que l'utilisation des incompatibilités sémantiques accélèrent la détection des conflits dans les plans.

Une ***incompatibilité sémantique*** est la donnée de 2 propriétés (2 termes de la logique du premier ordre) et de contraintes sur les variables contenues dans les propriétés.

Nous notons p1(X)=(X1, .. Xi, .. Xn) et p2(Y)=(Y1, .. Yj, .. Yp) deux propriétés d'arité quelconque. Xi et Yj sont des variables.

Il est possible d'exprimer des relations entre les composantes de X et Y qui traduisent des contraintes sur les valeurs qu'elles peuvent prendre. Nous notons rij(X,Y) la relation entre Xi et Yj. Il existe deux types de contraintes:

- des ***contraintes d'égalité*** du type Xi = Yj qui signifient que lorsque les variables X et Y sont instanciées, les composantes Xi et Yj doivent être instanciées à la même valeur.
- des ***contraintes de différenciation*** du type $Xi \neq Yj$ qui signifient que lorsque X et Y sont instanciées, les composantes Xi et Yj doivent avoir des valeurs différentes.
- Lorsqu'aucune contrainte n'est posée entre deux composantes, aucune relation n'est précisée.

Si rij(X,Y) $\Leftrightarrow$ X=Y alors $\neg$rij(X, Y) $\Leftrightarrow X \neq Y$.

Si rij(X,Y) $\Leftrightarrow X \neq Y$ alors $\neg rij(X, Y) \Leftrightarrow X = Y$.

Nous notons R(X,Y) la conjonction des contraintes posées sur certaines des variables de X et de Y.

Une incompatibilité sémantique est de la forme: $\boxed{p1(X) \wedge p2(Y) \wedge R(X, Y) \rightarrow \perp}$.

Nous notons I(p(X), q(Y)) une relation d'incompatibilité entre p(X) et q(Y). Nous notons I l'ensemble des relations d'incompatibilité pour un problème de planification donné.

Par exemple, l'incompatibilité sémantique traduisant qu'un objet ne peut se trouver en deux endroits différents simultanément s'écrit:
$(en(X1, Y1) \wedge en(X2, Y2) \wedge Y1 \neq Y2 \wedge X1 = X2) \rightarrow \perp$ avec en(X1,Y1) pour décrire le fait que l'objet X1 se trouve en Y1.

Remarquons que l'incompatibilité
$p1(X1, ..., Xi, ..., Xn) \wedge \neg p1(Y1, ..., Yi, ..., Yn) \wedge (\forall i)(Xi=Yi) \rightarrow \perp$ est toujours vraie.

5.1.2 Incompatibilité possible et nécessaire

Les plans que nous construisons peuvent contenir des variables. En effet, lors de la construction du plan, certaines actions ne sont pas complètement définies. Par exemple, si l'action A modélise un déplacement, le lieu d'arrivée (en(toto,Lille)) peut être défini alors que le lieu de départ ne l'est pas encore (en(toto,?X)) (figure 54).

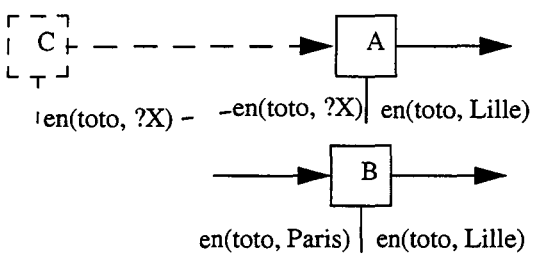


Figure 54: Deux actions partiellement définies et non ordonnées

Tant que X n'est pas connu, on ne peut pas savoir si le fait en(Toto, ?X) (qui doit être vrai entre C et A) est incohérent avec un fait du type en(Toto, Paris) (qui doit être vrai en B). En effet, si X prend la valeur Paris alors il n'y a pas d'état incohérent, alors que si X prend une valeur différente, il devient nécessaire de corriger le plan.

Dans le but de ne pas corriger inutilement le plan (cas où l'état incohérent prédit disparaît) et de ne pas parcourir l'ensemble du plan à chaque modification pour savoir quels sont les parties du plan incohérentes, nous distinguons deux types d'incompatibilité sémantique à la base de deux types de conflits:

- Les ***incompatibilités sémantiques possibles*** destinées à la détection d'éventuels états incohérents mémorisés pour la suite de la construction du plan.

- Les ***incompatibilités sémantiques nécessaires*** qui caractérisent un état nécessairement incohérent.

Soit l'incompatibilité définie par: $p1(X) \wedge p2(Y) \wedge R(X, Y) \rightarrow \perp$. Soit C l'ensemble des contraintes sur les variables du plan, issues de la construction du plan: C est une conjonction de contraintes de différenciation et d'égalité.

p1(X) et p2(Y) sont nécessairement incompatibles si:
$\forall(i, j)(C \rightarrow rij(X, Y))$

p1(X) et p2(Y) sont possiblement incompatibles si:
$\neg(\forall(i, j)(C \rightarrow rij(X, Y)))$

p1(X) et p2(Y) ne sont pas incompatibles si:
$\exists(io, jo)(C \rightarrow \neg riojo(X, Y))$

Nous allons maintenant définir menace et conflit à l'aide des incompatibilités sémantiques et des maintiens.

5.2 Définition des conflits

La notion de détection d'incohérence est étroitement liée à celle de vérité d'un fait dans un plan. Les ***conflits*** sont un moyen pour le planificateur de détecter les plans incohérents, et les ***réparations*** associées aux conflits sont un moyen de corriger les plans pour les rendre cohérents.

Nous travaillons dans le cas des planificateurs non linéaires: la planification non linéaire est un moyen de résoudre plusieurs buts simultanément (par opposition à l'hypothèse de linéarité définie dans [BAR 94]), et de réduire la taille de l'espace de recherche en conservant les disjonctions entre deux opérateurs instantanés (avant ou après) tant que rien ne permet de faire le choix ([MIN 91]). Deux approches majeures existent: l'une consulte un critère de vérité à chaque modification du plan pour détecter quels sont les faits vrais (nécessairement ou possiblement) et quel sont les faits qui ne sont pas encore établis. L'autre mémorise dans une structure les liens de dépendance entre les actions d'un plan. Nous présentons rapidement les deux méthodes pour en dégager les avantages et les inconvénients.

5.2.1 Les planificateurs de la famille TWEAK

La construction du planificateur TWEAK découle de la formalisation des travaux en planification non linéaire réalisée par D. Chapman ([CHA 87]).

Une proposition est nécessairement vraie dans un plan partiel si elle est vraie dans toutes les linéarisations du plan. Elle est possiblement vraie si elle est vraie dans au moins une linéarisation.

Le critère de vérité sert à déterminer la valeur de vérité d'une proposition dans un

plan partiellement ordonné sans devoir calculer toutes les linéarisations du plan (coût exponentiel en fonction du nombre d'actions dans le plan).

Le critère de vérité est le suivant: une proposition p est nécessairement vraie dans une situation s (p est vraie dans l'état du monde juste avant l'action s) si et seulement si les deux conditions suivantes sont vérifiées:

- il existe une situation t égale ou nécessairement avant s, dans laquelle p est nécessairement assertée (p est ajoutée par l'action t) ***et***
- pour chaque action C possiblement avant s et chaque proposition q qui co-désigne possiblement avec p, et qui est détruite par C, il existe une action W nécessairement entre C et S, qui asserte une proposition r co-désignant ([CHA 87]) avec p lorsque p et q co-désignent (figure 55).

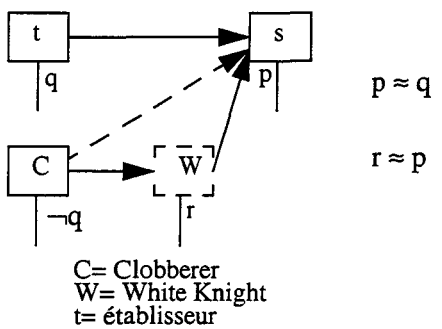


Figure 55: Représentation graphique du critère de vérité

Les actions C, t et s constituent un conflit. La réparation du conflit consiste:

- soit à placer le cloberrer C avant t ou après s (promotion, demotion)
- soit à définir un nouvel établisseur de p (W) entre C et s (technique du White Knight decloberring)

Ce critère est utilisé à chaque étape de construction d'un plan pour connaître la vérité d'une proposition. Si n est le nombre d'opérateurs dans le plan, alors cette vérification est en $O(n^4)$ [KNO ..].

La construction du plan est basée sur le choix et l'établissement d'une proposition non encore établie.

Cependant comme ce critère ne permet pas de mémoriser quelles sont les propositions établies, cette procédure peut conduire à des plans contenant des boucles répétitives. Ainsi TWEAK peut établir et détruire une proposition plusieurs fois. Un tel cas est représenté dans la figure 56 et a été mis en évidence dans [JAC 90]:

- les préconditions de An sont les buts à atteindre
- les effets de A0 sont les conditions initiales
- A1 et A2 sont les seules actions utilisables.

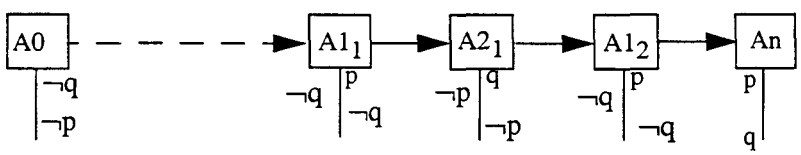


Figure 56: Bouclage dans l'espace de recherche TWEAK;
ordre d'insertion des actions dans le plan: $A1_1$ puis $A2_1$ puis $A1_2$

Ce critère offre un moyen de ne pas s'engager trop tôt sur le choix de l'établisseur d'une proposition. En effet, si le choix de t (figure 55) est contredit par la présence ultérieure de C alors il suffit de choisir un nouvel établisseur W pour p:

- si C peut être placée après s alors l'établisseur de p reste t et la contrainte temporelle s < C est ajoutée au plan.
- sinon, selon le critère de Chapman, le choix d'un autre établisseur s'avère nécessaire, et le clobberer C est placé avant le nouvel établisseur W (C < W).

Pour éviter des bouclages dans l'espace de recherche dus à l'utilisation du critère de vérité (qui peut conduire à l'examen de la même précondition plusieurs fois), ainsi que la complexité combinatoire du critère de vérité (appelé à chaque modification du plan), nous n'utilisons pas cette approche.

5.2.2 Les planificateurs à liens causaux

Le critère de vérité de TWEAK est un moyen de connaître la valeur de vérité d'une précondition dans un plan non linéaire: il repose sur l'analyse du plan à chacune de ses modifications. Les planificateurs à liens causaux constituent la deuxième grande famille des planificateurs traditionnels. Ici, le moyen d'établir la vérité d'un fait est de mémoriser son établissement.

Le premier planificateur qui a introduit ce principe est NONLIN ([TAT 77]), à travers la notion de ***lien de protection***. Dans NONLIN sont distingués les effets des actions qui réalisent des buts, des effets des actions qui ne sont pas utilisés dans le problème. Ceci lui permet, à la différence de NOAH ([SAC 75]), de définir des conditions minimales de réalisation des buts (ordonnancement minimal des actions): le lien de protection tiré entre l'effet et le but, protège la réalisation de ce but, et indique les effets utiles pour leur réalisation. Il empêche par ailleurs toute action de détruire le but protégé. Si une telle action est détectée dans le plan, alors elle doit être correctement positionnée par rapport au lien de protection.

Ce lien de protection est encore appelé ***lien causal*** car il permet de mémoriser les dépendances entre les actions d'un plan, la raison pour laquelle une action a été introduite dans le plan.

SNLP ([MCA 91]) formalise la notion de lien causal et le définit par:

un ***lien causal*** est un triplet < s, p,w > où p est un symbole de proposition, s et w des actions telles que p est une précondition de w et un effet ajouté de s. L'action s est appelée ***établisseur*** de p, l'effet ajouté p de s est le ***producteur*** du lien et la précondition de w est le ***consommateur*** du lien.

Un lien causal est noté: $s \xrightarrow{p} w$.

Les propriétés associées au lien causal sont décrites dans les 5 points suivants.

- **Il contraint l'établisseur à être avant le consommateur du lien: s < w.**

Le lien causal représente une relation entre le moment où le fait est établi par une action et le moment où il doit être vrai (pour être utilisé). Donc l'établisseur du fait doit être ***avant*** le consommateur (figure 57).

- **Il contraint le producteur et le consommateur du lien à codésigner.**

La définition du lien causal donnée dans SNLP peut être généralisée au cas où les préconditions et les effets sont des prédicats de la logique du premier ordre et contiennent donc des variables.

Ainsi, dans [PEN 92] le lien causal est défini par $s \xrightarrow{p(X),\, p(Y)} w$. Dans ce cas, les variables X et Y sont contraintes à codésigner: $X \approx Y$.

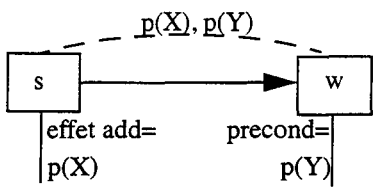


Figure 57: Le lien causal impose au producteur s d'être avant le consommateur w, la co-désignation de p(X) et p(Y)

- **Le lien causal représente un engagement sur le choix de l'établisseur**

Nous avons vu que les planificateurs de type TWEAK ne mémorisent pas l'établissement d'une précondition et peuvent changer l'établisseur lorsqu'un conflit est détecté par introduction d'un nouvel établisseur. Le choix d'un établisseur n'est donc pas définitif et peut être modifié sans retour-arrière.

L'utilisation du lien causal engendre le choix définitif de l'établisseur pour le reste de la construction du plan. Remettre en cause ce choix nécessite le retour-arrière depuis le point courant de la génération jusqu'au point où le choix de l'établisseur a été réalisé.

Cette approche nécessite des moyens supplémentaires pour guider le choix des établisseurs et éviter le retour-arrière, mais il a été montré que les planificateurs à liens causaux sont généralement plus efficaces que ceux utilisant un critère de vérité ([KAM 93]).

- **Le lien causal permet de connaître *instantanément* la valeur de vérité d'une précondition dans un plan.**

Puisque l'établissement d'une précondition est mémorisé au moment où il est réalisé par un lien causal entre l'établisseur et la précondition, connaître la valeur de la vérité de la précondition dans le plan est instantané.

En outre, si une précondition est vraie dans un plan traditionnel non linéaire à liens causaux, alors cette vérité est nécessaire puisque le lien causal protège l'établissement et

empêche toute menace (ou cloberrer selon D. Chapman) de détruire la précondition établie.

- **Le lien causal protège l'établissement**

La définition du lien causal et le sens qui lui est associé permettent de définir une menace et un conflit. Le lien causal peut être identifié à l'intervalle de temps minimal sur lequel le fait protégé doit être vrai. Alors toutes les actions susceptibles de détruire le fait protégé pendant l'intervalle représenté par le lien devront être déplacées pour agir avant ou après ce lien.

Nous reprenons ici la définition de la menace et du conflit utilisées dans SNLP ([MCA 91]).

Définition d'une menace (dans SNLP): l'opérateur instantané T est une menace pour le lien causal $E \xrightarrow{p} N$ (avec E et N opérateurs instantanés et p symbole de proposition) si T ajoute ou retire p.

Si T retire p, T est dite *menace négative*; si T ajoute p, T est dite *menace positive.*

Définition du conflit (dans SNLP): Un conflit est constitué d'un lien causal $E \xrightarrow{p} N$ et d'une menace T, telle que T puisse se situer entre E et N (figure 58).

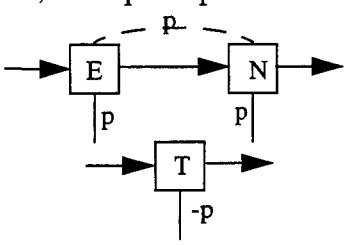


Figure 58: T menace pour le lien sur p entre E et N

Réparer un tel conflit consiste à imposer une des contraintes temporelles suivantes:
- T avant E ou
- N avant T.

La prise en compte des menaces positives conduit à un algorithme systématique: l'espace des plans parcouru est non redondant [KAM 93]. Cependant, cette approche sur-contraint les plans générés en obligeant le planificateur à linéariser les actions concernées par un "conflit positif". De plus, S. Kambhampati montre que cette approche n'est pas forcément la plus efficace. Nous ne tenons pas non plus compte des menaces positives.

Parce que le lien causal nous permet d'éviter les bouclages cités en 5.2.1, parce qu'il met en jeu des complexités moins importantes que l'utilisation d'un critère de vérité à la TWEAK, et parce qu'il permet de mémoriser les dépendances entre les actions d'un plan, nous utilisons cette technique pour établir la vérité d'un fait dans un plan.

Cependant, nous généralisons la notion de lien à celle de maintien (persistance d'un fait indépendamment des effets et des préconditions), ce qui nous permet d'exprimer des contraintes externes sur le plan, et redéfinissons les conflits à partir des incompatibilités sémantiques et des maintiens.

5.2.3 Maintien, incompatibilités sémantiques et conflits

5.2.3.1 Maintien et lien causal

Dans ce paragraphe, X et Y sont des vecteurs de variables: X = (X1, .. Xi, .. Xn) et Y = (Y1, .. Yj, .. Yp). les prédicats p(X), p(Y) et q(Y) désignent des faits.

La définition du maintien d'un fait lorsque ce fait est décrit par un prédicat de la logique du premier ordre est:

Un maintien sur un fait p(X) entre deux actions instantanées x et y est défini par:

$$maintien(p(X), x, y) \equiv (\exists X)(\forall t)(t < \min(x, y) \vee t \geq \max(x, y) \vee holds(p(X), t))$$

avec min(x,y) = x si x<y, max(x,y) = x si x > y et holds(p(X),t) signifiant que p(X) est vrai dans la description du monde à l'instant t.

Sa représentation graphique est donnée dans la figure 59.

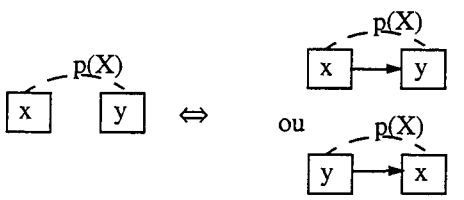


Figure 59: Le maintien de p(X) entre x et y est décrit par la ligne courbe

Il est alors possible de définir le traditionnel lien causal sur un fait p entre x et y à partir du maintien.

Un ***lien causal*** est défini par:

- p(X) appartient aux effets de x et
- x < y et
- maintien(p(X), x, y) et
- il existe p(Y) une précondition de y telle que $p(X) \approx p(Y)$.

Les incompatibilités sémantiques et le maintien étant définis, il est possible de définir deux types de conflits: les conflits maintien-effet et les conflits maintien-maintien.

5.2.3.2 Le conflit maintien-effet

Le conflit maintien-effet correspond à la situation où un fait p doit être vrai sur l'intervalle défini par 2 instants, alors qu' il est possible qu'un fait q, incompatible avec p, soit vrai à un instant appartenant à l'intervalle.

Un ***conflit maintien-effet*** est défini par:
Soit P un plan partiel P= (S, O) avec:
S l'ensemble des opérateurs instantanées du plan et
O l'ensemble des relations de précédence entre les opérateurs de S.
I est l'ensemble des incompatibilités du problème.
Soient x, y et z tels que $(x, y, z) \in S$.

conflit maintien-effet ((p(X), x, y), (q(Y), z))

$\Leftrightarrow$ maintien(p(X),x,y) $\wedge$
z produit q(Y) $\wedge$
incompatibles (p(X), q(Y)) $\in$ I $\wedge$
$O \cup \{x < z, z < y\} \vee O \cup \{y < z, z < x\}$

La figure 60 représente un conflit maintien-effet.

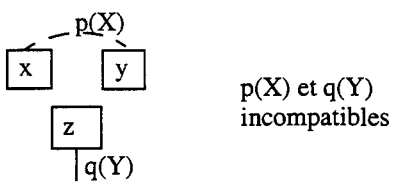


Figure 60: Conflit maintien-effet

Le conflit maintien-effet est potentiel si p(X) et q(Y) sont possiblement incompatibles. Il est réel si p(X) et q(Y) sont nécessairement incompatibles.

La réparation associée à un conflit maintien-effet consiste à placer z (la menace) avant ou après le maintien: $(z < x \wedge z < y) \vee (x < z \wedge y < z)$

L'application de la contrainte associée au premier membre de la disjonction conduit au plan de la figure 61.

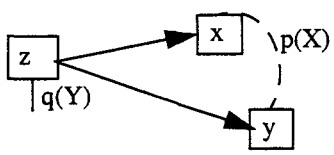


Figure 61: Première réparation du conflit de la figure 60

L'application de la contrainte associée au deuxième membre de la disjonction conduit au plan de la figure 62.

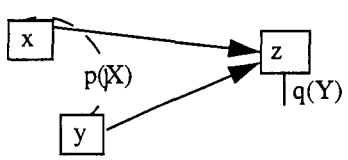


Figure 62: Deuxième réparation du conflit de la figure 60

Si le maintien fait partie d'un lien causal entre x et y, alors la réparation associée au conflit est simplifiée car x est avant y (figure 63): $z < x \vee y < z$

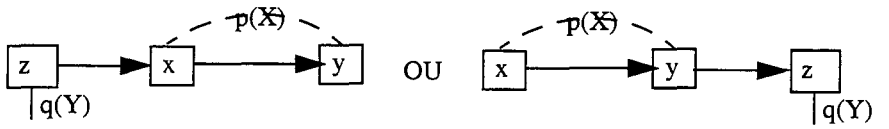


Figure 63: Les deux réparations associées au conflit maintien-effet lorsque le maintien fait partie d'un lien causal

5.2.3.3 Le conflit maintien-maintien

Soient les deux maintiens (p(X), x, y) et (q(Y), z, t) tels que p(X) et q(Y) sont incompatibles et les opérateurs instantanés x, y, z, t ne sont pas contraints temporellement (figure 64).

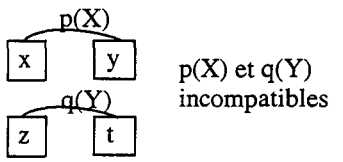


Figure 64: Un conflit maintien-maintien

Puisque p(X) et q(Y) sont incompatibles, ils ne peuvent être vrais en même temps: l'intervalle sur lequel p(X) est vrai (intervalle délimité par x et y) doit être disjoint de l'intervalle sur lequel q(Y) est vrai (intervalle délimité par z et t). Nous appelons une telle situation un conflit maintien-maintien et la définissons par:

Un ***conflit maintien-maintien*** est défini par:
Soit P un plan partiel P= (S, O) avec:
S l'ensemble des opérateurs instantanées du plan et
O l'ensemble des relations de précédence entre les opérateurs de S.
I est l'ensemble des incompatibilités du problème.
Soient x, y, z et t tels que $(x, y, z, t) \in S$.

conflit maintien-maintien ((p(X), x, y), (q(Y), z, t))

$\Leftrightarrow$ maintien(p(X),x,y) $\wedge$
maintien(p(X),x,y) $\wedge$
incompatibles (p(X), q(Y)) $\in I$ $\wedge$
$(O \cup \{x < z, z < y\} \vee \; O \cup \{y < z, z < x\} \vee$
$O \cup \{x < t, t < y\} \vee \; O \cup \{y < t, t < x\} \vee$
$O \cup \{z < x, x < t\} \vee \; O \cup \{t < x, x < z\} \vee$
$O \cup \{z < y, y < t\} \vee \; O \cup \{t < y, y < z\})$

Le conflit maintien-maintien est réel si p(X) et q(Y) sont nécessairement incompatibles. Il est potentiel si p(X) et q(Y) sont possiblement incompatibles.

La réparation associée à un conflit maintien-maintien consiste à placer le maintien (x,y) avant ou après le maintien (z,t):

$\boxed{(x < z \wedge x < t \wedge y < z \wedge y < t) \vee (z < x \wedge z < y \wedge t < x \wedge t < y)}$

L'application de la contrainte associée au premier membre de la disjonction conduit au plan de la figure 65.

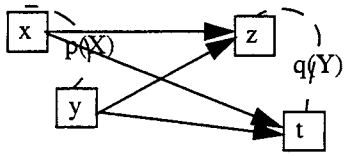


Figure 65: Première réparation du conflit de la figure 64

L'application de la contrainte associée au second membre de la disjonction conduit au plan de la figure 66.

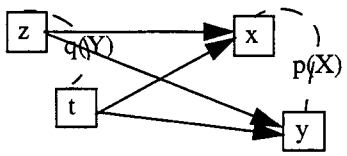


Figure 66: Seconde réparation du conflit de la figure 64

5.3 Conclusion

Nous avons défini les incompatibilités sémantiques qui permettent à l'utilisateur de décrire des relations binaires entre des propriétés du monde spécifiant des états incohérents.

Ces incompatibilités sont possibles si les contraintes sont potentiellement vérifiées: aucune contrainte n'est contredite, mais il existe des contraintes dont on ne sait pas si elles sont vérifiées ou non. Les incompatibilités sont nécessaires si toutes les contraintes sont nécessairement vérifiées.

Ces incompatibilités sont utilisées pour détecter les plans incohérents. Un plan est incohérent si, étant données les relations temporelles entre les actions, leur application peut conduire à un état incohérent (donc physiquement inatteignable).

Une incohérence dans un plan est détectée à l'aide de la structure des conflits. Il existe deux types de conflits:

- le conflit maintien-effet définissant une action produisant un fait incompatible avec un fait qui doit être vrai sur un intervalle de temps, et qui agit possiblement pendant cet intervalle. Le réparer consiste à placer l'instant où est produit le fait incompatible avant ou après l'intervalle.
- Le conflit maintien-maintien définissant deux intervalles sur lesquels sont respectivement vrais deux faits incompatibles se chevauchant possiblement. Réparer un tel conflit consiste à obliger les deux intervalles à être disjoints (l'un des intervalles est avant ou après l'autre).

Dans le chapitre suivant, nous expliquons comment, en utilisant la structure des conflits maintien-effet et maintien-maintien, ainsi que la seule relation de précédence entre instants, nous construisons une table de transition entre le formalisme d'Allen et celui lié aux planificateurs traditionnels non linéaires à liens causaux.

Chapitre 6

Traduction des relations d'Allen en conjonctions de schémas p1, p2 et p3

Nous identifions dans ce chapitre ***trois schémas temporels*** basé sur l'ordre total entre deux instants et sur les réparations associées aux conflits définis dans le chapitre 5. Nous remarquons que la structure temporelle d'un plan est une conjonction d'instanciations de ces schémas et établissons une passerelle entre le formalisme d'Allen et le formalisme temporel lié au fonctionnement des planificateurs traditionnels non linéaires à liens causaux. Cette traduction nous permet la prise en compte de relations temporelles disjonctives entre intervalles tout en conservant les principes des planificateurs traditionnels à liens causaux qui considèrent que les actions sont instantanées et qui ne gèrent que la relation de précédence entre actions.

6.1 Trois schémas temporels p1, p2 et p3

6.1.1 Un instant avant un instant

Dans un plan traditionnel deux instants peuvent être totalement ordonnés. L'établisseur d'un fait est avant le moment où le fait est utilisé.

La relation de précédence entre deux instants (ou opérateurs instantanés) x et y est notée: $\boxed{p1(x, y) \Leftrightarrow x < y}$.

6.1.2 Un instant avant ou après un intervalle

La réparation associée au conflit maintien-effet revient à interdire la présence de l'effet (appartenant à l'opérateur z) entre le début et la fin du maintien (entre les opérateurs x et y). Ce qui équivaut à placer l'effet avant le début du maintien ou après sa fin. La contrainte suivante est posée sur le plan:

$$\boxed{p2(z, x, y) \Leftrightarrow (z < x \wedge z < y) \vee (x < z \wedge y < z)}$$

Le couple d'instants (x,y) définit un intervalle XY et z est un opérateur instantané que l'on peut confondre avec un instant (intervalle de durée nulle).

p2(z, x, y) $\Leftrightarrow$ z < > XY.

6.1.3 Un intervalle avant ou après un intervalle

La réparation associée au conflit maintien-maintien consiste à interdire l'entrelacement des deux maintiens. Chaque maintien définit un intervalle. La réparation revient donc à interdire le chevauchement des deux intervalles définis par les couples d'opérateurs instantanés (x,y) et (z,t). La contrainte suivante est posée sur le plan:

$$\boxed{p3(x, y, z, t) \Leftrightarrow (x < z \wedge x < t \wedge y < z \wedge y < t) \vee (z < x \wedge z < y \wedge t < x \wedge t < y)}$$

Si XY est l'intervalle défini par le couple (x,y) et ZT celui défini par le couple (z,t) alors p3(x,y,z,t) $\Leftrightarrow$ XY < > ZT.

6.1.4 La structure temporelle d'un plan est une conjonction d'instanciations de p1, p2 et p3

Un plan est un ensemble d'opérateurs instantanés de type STRIPS. Un plan contient donc des effets et des préconditions en relations les uns avec les autres.

Les opérateurs instantanés constituent un ordre partiel strict.

Des relations de précédence peuvent exister entre des opérateurs instantanés si un opérateur produit un effet nécessaire à un autre opérateur (précondition). Dans ce cas, un maintien relie l'effet produit et la précondition.

Lorsque un maintien sur un fait existe entre deux opérateurs instantanés, cela signifie que le fait est vrai entre les 2 opérateurs. Il faut alors s'assurer de la cohérence du plan, en étudiant la présence d'une action venant produire un fait incohérent avec le fait précédent sur l'intervalle sur lequel il doit être vrai, ou d'un autre maintien sur un fait incohérent avec le précédent tel que les deux maintiens puissent s'entrelacer.

En conséquence, dans un plan, il existe différents types de relations entre les composants. Nous les décrivons ci-après et leur associons, si possible un schéma temporel:

- relation maintien - maintien représentée par p3
- relation maintien - précondition inexistante si la précondition n'est pas établie. Sinon, il peut y avoir une relation maintien - maintien associée à p3.
- relation maintien - effet associée à p2
- relation précondition - effet associée à p1 si l'effet établit la précondition.
- relation précondition - précondition inexistante si aucune des deux préconditions n'est établie.
- relation effet - effet inexistante car il n'y a pas d'égalité entre instants dans les plans traditionnels (cf paragraphe 6.7).

Un plan doit vérifier la cohérence de toutes les relations précédemment citées. Sa description est équivalente à la conjonction d'un ensemble de contraintes à vérifier, en l'occurence décrites par p1, p2 et p3. Remarquons que cette conjonction de schémas temporels est associée à des contraintes sur les variables (égalité et différenciation).

Ajouter une contrainte temporelle c'est ajouter un graphe p1, p2 ou p3 au graphe associé au plan. Le graphe résultant est la conjonction de la nouvelle contrainte temporelle et des contraintes temporelles du plan:

- ajouter une relation de précédence du type x<y équivaut à faire la conjonction

de p1(x,y) avec l'ensemble des modèles temporels du plan.
- détecter un conflit maintien-effet ou maintien-maintien équivaut à faire la conjonction de p2(z,x,y) ou p3(x,y,z,t) avec les modèles temporels du plan.

Nous allons utiliser ce principe et effectuer des conjonctions de schémas temporels pour décrire des relations entre d-actions associées à des couples d'instants représentant leur début et fin.

6.2 Principe et motivations

Nous utilisons la seule relation de précédence entre instants présente dans les plans traditionnels non linéaires à liens causaux. Nous cherchons à décrire des disjonctions temporelles entre d-actions, ce qui nous permet d'exprimer des relations temporelles imprécises.

Nous partons du principe que les plans sont des conjonctions de schémas temporels p1, p2 et p3 instanciés, et qu'il est possible d'utiliser les interdictions temporelles issues des conflits pour imposer les relations temporelles contraires sur le plan.

6.2.1 Une seule relation entre instants: la précédence

Seule la relation de précédence est autorisée dans les plans traditionnels. Soient X et Y deux d-actions. Ces deux d-actions sont équivalentes à deux intervalles, que nous appelons également X et Y. Ces deux intervalles sont représentés par deux couples d'instants totalement ordonnés représentant le début et la fin de chaque d-action. Pour une d-action quelconque A, nous notons a^- l'instant représentant le début de A et a^+ l'instant représentant la fin de A. Les opérateurs instantanés associés portent le même nom. x^- (respectivement y^-) est toujours avant x^+ (respectivement y^+), donc le modèle p1(x^-, x^+) (respectivement p1(y^-, y^+)) est toujours vrai.

Grâce à la relation de précédence et en considérant que la durée d'une d-action est représentée par (x^-, x^+), il est possible de décrire 6 des 13 primitives d'Allen. Ces primitives sont toutes celles que l'on peut exprimer à l'aide de conjonction de relations de précédence entre les débuts et fins des deux d-actions, c'est-à-dire toutes celles qui ne font pas intervenir l'égalité entre instants. Nous discutons le cas de l'égalité dans le paragraphe 6.7. Le tableau 5 reprend les 6 primitives d'Allen et leur traduction en terme de relations de précédence entre instants. Elles sont également décrites en annexe C.1.

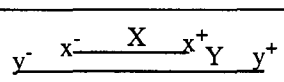
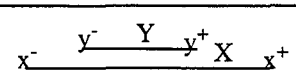
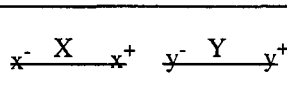
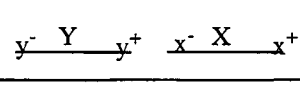
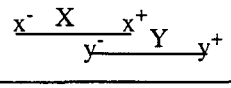
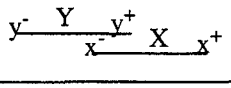
Intervalles	Représentation	Instants
X d Y		$p1(y^-, x^-) \wedge p1(x^+,y^+)$
X di Y		$p1(x^-, y^-) \wedge p1(y^+, x^+)$
X < Y		$p1(x^+, y^-)$
X > Y		$p1(y^+, x^-)$
X o Y		$p1(x^-,y^-) \wedge p1(y^-,x^+)$ $\wedge p1(x^+,y^+)$
X oi Y		$p1(y^-,x^-) \wedge p1(x^-,y^+)$ $\wedge p1(y^+<x^+)$

Tableau 5 : Représentation, à l'aide de la seule relation de précédence entre instants, des 6 primitives d'Allen sans l'égalité entre instants

En utilisant les 6 relations temporelles précédentes, un planificateur peut résoudre des problèmes où il est nécessaire d'imbriquer des actions.

Par exemple, la macro-action "clouer" est décrite par:

- tenir le clou (It) et
- frapper le clou *pendant un intervalle (If) inclus dans celui pendant lequel le clou est tenu.* (figure 67)

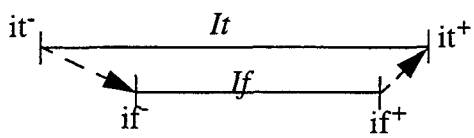


Figure 67: It di If $\Leftrightarrow$ $it^- < if^- \wedge if^+ < it^+$

De même, il est possible d'imposer que le soutien d'une troupe T1 par une troupe T2 (action A2) englobe complètement l'attaque réalisée par T1 (action A1): A1 d A2.

6.2.2 Des relations temporelles disjonctives pour exprimer l'imprécision

Lorsqu'une relation temporelle entre deux d-actions est parfaitement connue, elle est représentée exactement par une des primitives précédentes. Cependant, les relations entre les actions peuvent être ambiguës: par exemple, on voudrait pouvoir dire que des actions A et B se déroulent sur un intervalle de temps commun sans préciser comment est caractérisé cet intervalle.

Ainsi, au niveau stratégique de la simulation de combats, dire que deux actions A et B se déroulent en parallèle signifie que les intervalles associés à leur déroulement se chevauchent. La relation temporelle est précisée à un niveau de détail inférieur, par exemple à l'exécution. Cette contrainte est vérifiée dès que l'une des primitives d, di, o ou oi est vérifiée. Elle est décrite par la disjonction des primitives, qui est notée

A d di o oi B.

Dans le but d'exprimer n'importe quelle connaissance sur une relation temporelle entre deux d-actions, nous devons donc être capables de décrire les 64 disjonctions possibles induites par les 6 primitives précédentes.

L'utilisation de conjonctions de p1, p2 et p3 appliqués aux débuts et fins de 2 d-actions nous permet de décrire ces 64 relations.

6.2.3 Imposer une relation dans un plan = décrire l'interdiction de son contraire

Certaines relations ne sont pas directement exprimables dans un plan: ce sont toutes les relations n'appartenant pas à l'algèbre d'instants. Cependant, il semble qu'à l'aide de p2 ou p3 on sache exprimer des relations n'appartenant pas à cette algèbre. Par ailleurs, les relations qui découlent de p2 et p3 proviennent de l'interdiction exprimée par ces schémas. Par exemple, p2(z,x,y) interdit l'insertion de z entre x et y; ce qui équivaut à autoriser z à être avant x et avant y, ou z après x et après y. Nous allons nous inspirer de cette constatation pour exprimer des relations temporelles en interdisant leur contraire. En effet, si S est l'ensemble des valeurs possibles que peut prendre une variable v, restreindre cet ensemble à un sous-ensemble S' revient à interdire les valeurs de v appartenant au complémentaire de S' dans S.

Dans notre cas, S est l'ensemble des primitives du tableau 5: S= {<, >, d, di, o, oi}. Lorsqu'une primitive est interdite, toutes les autres sont autorisées. Ce qui équivaut à autoriser la disjonction des primitives autorisées. Ainsi, imposer A < > B revient à interdire A o oi d di B: $(A < B \vee A > B) \Leftrightarrow \neg(AoB \vee AoiB \vee AdB \vee AdiB)$ avec A et B deux d-actions.

Or dans l'algèbre d'instant, A o oi d di B $\Leftrightarrow$ $a^- < b^+ \wedge b^- < a^+$.

Donc $(A < B \vee A > B) \Leftrightarrow \neg (a^- < b^+ \wedge b^- < a^+)$

$\Leftrightarrow (b^+ < a^- \vee a^+ < b^-)$

Soit $(A < B \vee A > B) \Leftrightarrow p3(a^-,a^+,b^-,b^+)$.

Nous allons appliquer ce principe pour décrire des relations temporelles disjonctives entre deux d-actions dans le cadre des planificateurs traditionnels non linéaires à liens causaux.

6.3 Utilisation d'un seul schéma

Nous allons identifier toutes les relations temporelles disjonctives entre 2 intervalles qui correspondent à l'instanciation d'un seul schéma.

6.3.1 Principe

Nous désirons positionner une d-action X par rapport à une autre d-action Y.

$X= (x^-,x^+)$ et $Y= (y^-,y^+)$; $p1(x^-,x^+)$ et $p1(y^-,y^+)$ sont toujours vrais.

Nous allons construire toutes les instanciations possibles des 3 schémas précédents avec x^-, x^+, y^-, y^+.

6.3.2 Résultat

Schémas instanciés	X r Y
$p1(x^-,y^+)$	d o < oi di
$p1(y^+,x^-)$	>
$p1(y^-,x^+)$	d o oi di >
$p1(x^+,y^-)$	<
$p1(x^-,y^-)$	o < di
$p1(y^-,x^-)$	d oi >
$p1(x^+,y^+)$	d o <
$p1(y^+,x^+)$	oi di >
$p2(y^-,x^-,x^+)$	< > oi d
$p2(y^+,x^-,x^+)$	< > d o
$p2(x^-,y^-,y^+)$	< > o di
$p2(x^+,y^-,y^+)$	< > oi di
$p3(x^+, x^-, y^-, y^+)$	< >

Tableau 6 : Relations temporelles entre 2 d-actions X et Y;
$p1(x^-,x^+)$ et $p1(y^-,y^+)$ sont toujours vrais.

Le tableau 6 ne représente que les disjonctions temporelles entre 2 d-actions qui correspondent à des instanciations de schémas ***cohérentes***. Toutes les relations temporelles existant entre 2 d-actions seront construites à partir de ces relations simples.

Cette approche nous permet de décrire les relations temporelles entre intervalles qui n'appartiennent pas à l'algèbre d'instants identifiée par M.Ghallab et A.M. Alaoui ([GHA 89]), *la relation d'égalité entre instants étant retirée.*

En effet, certaines relations entre intervalles ne peuvent pas être représentées par des relations entre instants. Par exemple, la relation temporelle X < > Y, avec X et Y deux intervalles, est décrite par $(x^+ < y^- \vee y^+ < x^-) \wedge x^- < x^+ \wedge y^- < y^+$. Or la disjonction $(x^+ < y^- \vee y^+ < x^-)$ n'appartient pas à l'algèbre d'instants car elle fait intervenir plus de 2 instants. L'ensemble des relations temporelles entre intervalles qu'il est possible d'exprimer à l'aide de relations temporelles entre instants est décrits dans [VBE 90].

6.4 Utilisation de conjonctions de schémas

Réaliser des conjonctions de schémas (de relations entre instants) correspond à l'intersection de relations d'Allen.

6.4.1 Principe

Les primitives de l'annexe C.1 font intervenir des conjonctions de relations de précédence entre instants. De façon générale, une conjonction d'informations diminue l'incertitude sur l'information résultante.

Par ailleurs, chaque relation de précédence entre instants correspond à une relation temporelle, souvent disjonctive, entre intervalles. Par exemple, en lisant le tableau 6, nous pouvons dire que:

$x^- < y^-$ correspond à X o di < Y, (1)

$y^- < x^+$ correspond à X d di o oi > Y et (2)

$x^+ < y^+$ correspond à X d o < Y. (3)

Si on ne connaît que (1), la relation entre X et Y est imprécise puisqu'elle fait intervenir trois primitives possibles. Ajouter l'information (2), c'est-à-dire faire la conjonction de (1) et (2) diminue l'imprécision sur la relation entre X et Y car X o di < Y $\wedge$ X d di o oi > Y $\Rightarrow$ X o di Y. Cette déduction provient du fait que toute conjonction de primitives d'Allen est incohérente puisque les primitives sont mutuellement exclusives. L'ajout de la troisième information nous permet finalement de dire:

$x^- < y^- \wedge y^- < x^+ \wedge x^+ < y^+ \wedge (x^- < x^+ \wedge y^- < y^+)$

$\Leftrightarrow p1(x^-,y^-) \wedge p1(y^-,x^+) \wedge p1(x^+,y^+)$

$\Leftrightarrow$ X o Y

Cette relation est représentée dans la figure 68.

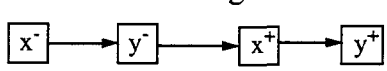


Figure 68: X o Y

En suivant ce principe, nous avons calculé toutes les conjonctions de schémas temporels instanciés par x^-, x^+, y^-, y^+ afin de déterminer toutes les relations temporelles que nous sommes capables d'exprimer dans un plan traditionnel non linéaire à liens causaux.

6.4.2 Résultat

Le tableau 7 décrit toutes les relations temporelles disjonctives ***cohérentes*** qu'il est possible de déduire des relations du tableau précédent.

Conjonctions de schémas	X r Y
$p1(x^-,y^-), p1(y^-,x^+), p1(x^+,y^+)$	o
$p1(y^-,x^-), p1(x^-,y^+), p1(y^+,x^+)$	oi
$p1(y^-,x^-), p1(x^+,y^+)$	d
$p1(x^-,y^-), p1(y^+,x^+)$	di
$p1(x^-,y^-), p1(x^+,y^+)$	< o
$p1(x^-,y^-), p1(y^-,x^+)$	di o
$p1(y^-,x^-), p1(x^-,y^+)$	d oi
$p1(y^-,x^-), p1(y^+,x^+)$	> oi
$p1(y^-,x^+), p1(x^+,y^+)$	d o
$p1(x^-,y^+), p1(y^+,x^+)$	di oi
$p1(x^-,y^+), p1(y^-,x^+)$	d di o oi
aucune contrainte entre X et Y	> < d di o oi
$p2(y^-,x^-,x^+), p2(y^+,x^-,x^+)$	< > d
$p2(x^-,y^-,y^+), p2(x^+,y^-,y^+)$	< > di
$p2(x^-,y^-,y^+), p2(y^+,x^-,x^+)$	< > o
$p2(y^-,x^-,x^+), p2(x^+,y^-,y^+)$	< > oi
$p1(y^-,x^-), p2(y^+,x^-,x^+)$	> d
$p1(y^-,x^+), p2(y^+,x^-,x^+)$	> d o
$p2(x^-,y^-,y^+), p1(y^+,x^+)$	> di
$p1(y^-,x^+), p2(x^-,y^-,y^+)$	> di o
$p2(x^-,y^-,y^+), p1(y^-,x^+), p2(y^+,x^-,x^+)$	> o
$p2(y^-,x^-,x^+), p1(x^+,y^+)$	< d
$p1(x^-,y^+), p2(y^-,x^-,x^+)$	< d oi
$p1(x^-,y^-), p2(x^+,y^-,y^+)$	< di
$p1(x^-,y^+), p2(x^+,y^-,y^+)$	< di oi
$p1(x^-,y^+), p2(y^-,x^-,x^+), p2(x^+,y^-,y^+)$	< oi

Tableau 7 : Les relations temporelles disjonctives entre intervalles correspondant aux conjonctions consistantes de p1, p2 et p3.

Ce tableau résume uniquement les conjonctions de schémas dont l'écriture et la gestion sont les plus simples. Nous décrivons les critères utilisés pour déterminer les conjonctions les plus simples dans le paragraphe suivant.

6.4.3 Choix de la représentation la plus simple: critères

Un plan traditionnel est un ensemble d'instants positionnés temporellement par la relation de précédence. Par défaut, deux instants sont non ordonnés, c'est-à-dire qu'aucune relation n'existe entre les instants. L'absence de relation entre instants correspond à la disjonction "avant ou après" qui est naturellement traitée par ces planificateurs. D'où les critères suivants:

- Si plusieurs conjonctions de schémas décrivent la même relation temporelle entre 2 d-actions, alors celle qui est choisie est celle qui contient le plus possible de schémas p1. En effet, comme nous travaillons dans le cadre des planificateurs traditionnels à liens causaux, insérer une relation de précédence (p1) dans un plan est une opération de modification de plan prédéfinie, qui ne demande pas d'amélioration du planificateur. De plus, moins les conjonctions feront intervenir de conflits (p2 ou p3), plus facile sera leur gestion, puisque le nombre de conflits n'augmentera pas.

- Si plusieurs conjonctions de schémas p1 satisfont le premier critère alors choisir la conjonction qui fait appel à un nombre minimal de schémas. Ce critère permet d'éviter les informations redondantes lors de l'étape de description des actions d'un problème de planification.

Nous présentons ici un exemple. La relation X > oi Y peut être décrite par 17 conjonctions de schémas différentes (nous donnons en annexe C.2 toutes les façons possibles de décrire les relations temporelles entre intervalles citées dans le tableau 7).

- Selon le premier critère, nous ne conservons que celles qui contiennent le moins de schémas p2 ou p3. Il reste alors:
  - $(p1(y^-,x^-) \wedge p1(y^+,x^+))$ ou
  - $(p1(y^-,x^-) \wedge p1(y^+,x^+) \wedge p1(y^-,x^+))$
- Selon le deuxième critère, nous choisissons de représenter X oi > Y par:
  $(p1(y^-,x^-) \wedge p1(y^+,x^+))$. En effet, $p1(y^-,x^+)$ est redondant car il peut être déduit des deux instances de schémas précédentes. Plus simple sera la description des relations, plus elle sera lisible. Le planificateur déduit alors toute l'information qu'il est capable de déduire à partir des relations temporelles données dans la description des actions, afin de connaître toutes les relations de précédence entre les instants d'un plan.

6.4.4 40 relations sur les 64 prévues...

Rappelons le principe de description des relations temporelles et les résultats à ce point:

- il existe 13 instances de schémas p1, p2 et p3 avec x^-, x^+, y^-, y^+
- un plan est une conjonction d'instances de schémas
- faire une conjonction de relations diminue l'incertitude sur la relation finale.
- le calcul de toutes les conjonctions cohérentes des 13 instances précédentes conduit à 40 des 64 relations temporelles disjonctives escomptées.

Nous expliquons ci-après pourquoi ces conjonctions ne contiennent pas toutes les

relations temporelles prévues.

Les conjonctions précédentes ne nous permettent cependant pas de décrire les 64 relations temporelles possibles entre 2 d-actions. En effet, l'étude des relations du tableau 6 montre qu'il est impossible de trouver des conjonctions menant aux 24 dernières relations (données dans le tableau 8). Deux groupes de relations constituent cet ensemble: les relations contenant la disjonction "d di" (colonne de droite) et celles contenant la disjonction "o oi" (colonne de gauche).

o oi	d di
< o oi	< d di
> o oi	> d di
o oi d	d di o
o oi di	d di oi
< > d o oi	> d di o
< > di o oi	< d di oi
< > o oi	< > d di
> d o oi	< > d di o
> di o oi	< > d di oi
< d o oi	> d di oi
< di o oi	< d di o

Tableau 8 : Les 24 relations disjonctives que l'on ne peut pas décrire par simple conjonction de schémas

Nous codons les 13 relations du tableau 6 par un vecteur binaire dans la base des 6 primitives. Ce vecteur est construit en indiquant 1 lorsqu'une des primitives fait partie de la disjonction à décrire, 0 sinon. Par exemple, la relation X < d o Y est représentée par le vecteur (1, 0, 1, 0, 1, 0) dans la base (d, di, o, oi, <, >). Le tableau 9 décrit la représentation des 13 instanciations de schémas précédemment citées.

Schémas instanciés	X r Y	d	di	o	oi	<	>
$p1(x^-,y^+)$	d o < oi di	1	1	1	1	1	0
$p1(y^+,x^-)$	>	0	0	0	0	0	1
$p1(y^-,x^+)$	d o oi di >	1	1	1	1	0	1
$p1(x^+,y^-)$	<	0	0	0	0	1	0
$p1(x^-,y^-)$	o < di	0	1	1	0	1	0
$p1(y^-,x^-)$	d oi >	1	0	0	1	0	1
$p1(x^+,y^+)$	d o <	1	0	1	0	1	0
$p1(y^+,x^+)$	oi di >	0	1	0	1	0	1
$p2(y^-,x^-,x^+)$	< > oi d	1	0	0	1	1	1
$p2(y^+,x^-,x^+)$	< > d o	1	0	1	0	1	1

Tableau 9 : Représentation binaire des relations issues d'un seul schéma instancié

Schémas instanciés	X r Y	d	di	o	oi	<	>
$p2(x^-,y^-,y^+)$	< > o di	0	1	1	0	1	1
$p2(x^+,y^-,y^+)$	< > oi di	0	1	0	1	1	1
$p3(x^+, x^-, y^-, y^+)$	< >	0	0	0	0	1	1

Tableau 9 : Représentation binaire des relations issues d'un seul schéma instancié

Faire des conjonctions de schémas, revient à faire un "et" binaire sur les coordonnées des vecteurs correspondants.

Nous cherchons à représenter toutes les relations temporelles disjonctives qui contiennent séparément d di et o oi. Il s'agit donc de trouver une conjonction de schémas simples autorisant cette description.

Prenons le cas de la disjonction X o oi Y (colonnes grisées dans le tableau 9). Il s'agit de trouver une conjonction de schémas instanciés pour laquelle seules les coordonnées correspondant aux primitives o et oi sont égales à 1. Il est donc nécessaire de trouver une conjonction d'instances telle qu'elles contiennent toutes o et oi. Les seules instances contenant o et oi sont $p1(x^-,y^+)$ et $p1(y^-,x^+)$ (lignes grisées dans le tableau 9). Par ailleurs, faire la conjonction de ces deux instances élimine les primitives < et > puisqu'elles ne sont pas vraies simultanément dans les deux instances. On obtient donc: $p1(x^-,y^+) \wedge p1(y^-,x^+) \Leftrightarrow X$ d di o oi Y.

Il reste encore à éliminer d di, tout en concervant o oi. Il s'agit donc de trouver des instances vérifiant ces deux caractéristiques. La lecture du tableau précédent montre qu'il n'existe aucune instance de ce type. Il est donc impossible de décrire la disjonction X o oi Y par ce principe. Il en est de même pour toutes les relations temporelles du tableau 8.

Il semble donc nécessaire de trouver un autre moyen que la conjonction de schémas instanciés par les débuts et fins des deux d-actions pour décrire les dernières relations.

6.5 Utilisation de la transitivité

La transitivité est un moyen de propager l'information temporelle afin d'obtenir sa parcimonie. Allen appelle cette opération "composition" de relations temporelles: $x\,R\,y \wedge y\,R'z \Leftrightarrow x\,R\,o\,R'\,z$. Cette opération permet donc d'obtenir de l'information supplémentaire sur la relation entre deux intervalles x et z à partir de la relation entre x et y et de la relation entre y et z. Nous utilisons cette caractéristique pour décrire de nouvelles disjonctions temporelles qui ne peuvent être décrite directement entre deux intervalles. Nous allons insérer des opérateurs fictifs F entre des d-actions tels que $x\,R\,F \wedge F\,R'y \Rightarrow x\,R''\,z$ avec $R' \neq R''$ et $R \neq R''$.

Cette méthode peut être comparée à la génération d'intervalles anonymes proposée par J. Révault dans [REV 96]. Il a construit un sous-ensemble de l'algèbre d'Allen à partir de la seule relation de meets (= m, et son inverse "is met by" = mi) et des intervalles. Cette structure lui permet de représenter un ensemble de relations par un

graphe dont les arcs sont étiquetés par la relation entre les intervalles identifiés par le sommet départ et le sommet arrivée de l'arc. L'observation de l'ensemble des relations temporelles entre intervalles qu'il peut décrire à l'aide de la seule relation meets l'a conduit à utiliser des intervalles supplémentaires autres que ceux qui interviennent dans les données. Ainsi, A s B (A débute B) exige l'introduction de trois intervalles anonymes (x1, x2, x3):

$$A\ s\ B \Leftrightarrow \begin{pmatrix} \exists x1 (A\ mi\ x1 \wedge x1\ m\ B) \\ \exists x2 \exists x3 (A\ m\ x2 \wedge x2\ m\ x3 \wedge x3\ mi\ B) \end{pmatrix}.$$

Nous appliquons le même principe en utilisant l'instant et la relation de précédence entre instants, dans le cadre des planificateurs traditionnels non linéaires à liens causaux.

6.5.1 Principe

6.5.1.1 Observation

Nous avons remarqué que, dans les plans, certaines d-actions peuvent vérifier des relations temporelles n'appartenant pas aux 40 décrites par simple conjonction de schémas. En effet, soient trois d-actions X, Y et Z, il est possible de déduire des nouvelles relations entre X et Z, connaissant la relation entre X et Y, et la relation entre Y et Z.

$X = (x^-, x^+)$, $Y = (y^-, y^+)$ et $Z = (z^-, z^+)$. Les schémas $p1(x^-, x^+)$, $p1(y^-, y^+)$ et $p1(z^-, z^+)$ sont toujours vrais. Nous ne les rappelons pas dans les formules qui suivent.

Soit le plan de la figure 69. Il contient les contraintes temporelles suivantes:
Une contrainte temporelle entre X et Y: X di Y. (4)
Une autre contrainte temporelle entre Y et Z: Y <> Z. (5)
Traduisons (4) et (5) en relations de précédence entre instants:

- (4) est équivalente à $x^- < y^- \wedge y^+ < x^+$
- (5) est équivalente à $y^+ < z^- \vee z^+ < y^-$.

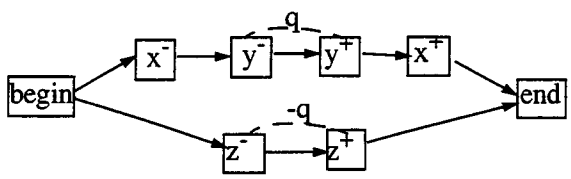


Figure 69: X di Y $\wedge$ Y <> Z

La structure temporelle de ce plan contient la conjonction de (4) et (5), qui peut également être décrite par la disjonction (6) $\vee$ (7) en distribuant le $\wedge$ par rapprrt au $\vee$:

$x^- < y^- \wedge y^+ < x^+ \wedge y^+ < z^- \wedge (x^- < x^+ \wedge y^- < y^+ \wedge z^- < z^+) \vee$ (6)

$x^- < y^- \wedge y^+ < x^+ \wedge z^+ < y^- \wedge (x^- < x^+ \wedge y^- < y^+ \wedge z^- < z^+)$ (7)

Le planificateur déduit toutes les relations de précédence entre instants. Pour cela, il applique la seule relation de transitivité entre instants: $i1 < i2 \wedge i2 < i3 \Rightarrow i1 < i3$, avec i1, i2 et i3 trois instants.

Ainsi, la formule (6) permet de déduire $x^- < z^-$, et la formule (7) permet de déduire $z^+ < x^+$. Or d'après le tableau 6:

- $x^- < z^- \Rightarrow X\ o < di\ Z$ et
- $z^+ < x^+ \Rightarrow X > oi\ di\ Z$.

Donc la disjonction (6) $\vee$ (7) $\Rightarrow$ $x^- < z^- \vee z^+ < x^+$

$\Rightarrow$ $X <> o\ oi\ di\ Z$.

En conclusion, $X\ di\ Y \wedge Y <> Z \Rightarrow X <> o\ oi\ di\ Z$. Il est donc possible en utilisant les règles de transitivité de déduire des relations n'appartenant pas aux 40 relations des tableaux 6 et 7.

6.5.1.2 Généralisation

Nous avons calculé les règles de transitivité issues des 6 primitives que nous considérons et à l'aide de la seule règle de transitivité issue de la relation de précédence entre instants. L'ensemble de ces règles est décrit dans la table de transitivité 10.

Z r' Y / X r Z	<	>	d	di	o	oi
<	$x^- < z^-$	?	$x^- < z^-$	$x^+ < z^-$	$x^+ < z^-$	$x^- < z^-$
>	?	$z^+ < x^-$	$z^- < x^-$	$z^+ < x^-$	$z^- < x^-$	$z^+ < x^-$
d	$x^+ < z^-$	$z^+ < x^-$	$z^- < x^- \wedge x^+ < z^+$	?	$x^- < z^-$	$z^- < x^-$
di	$x^- < z^-$	$z^+ < x^+$	$x^- < z^+ \wedge z^- < x^+$	$x^- < z^- \wedge z^+ < x^+$	$x^- < z^- \wedge z^- < x^+$	$x^- < z^+ \wedge z^+ < x^-$
o	$x^+ < z^-$	$z^+ < x^+$	$z^- < x^+ \wedge x^+ < z^+$	$x^- < z^-$	$x^- < z^- \wedge x^+ < z^+$	$x^- < z^+ \wedge z^- < x^+$
oi	$x^- < z^-$	$z^+ < x^-$	$z^- < x^- \wedge x^- < z^+$	$z^+ < x^+$	$x^- < z^+ \wedge z^- < x^+$	$z^- < x^- \wedge z^+ < x^+$

Tableau 10 : Table de transitivité (relations de précédence entre instants)

La traduction des relations du tableau 10 en relations temporelles entre intervalles est donné dans le tableau 11. Ce tableau est équivalent à la partie de la table de transitivité d'Allen concernant les primitives d, di, o, oi, < et > (annexe A.2) une fois retirées toutes les primitives faisant intervenir l'égalité entre instants.

Zr' Y / XrZ	<	>	d	di	o	oi
<	<	?	< o d	<	<	< o d
>	?	>	> oi d	>	> oi d	>
d	<	>	d	?	< o d	> oi d
di	< o di	> oi di	o oi d di	di	o di	oi di
o	<	> oi di	o d	< o di	< o	o oi d di
oi	< o di	>	oi d	> oi di	o oi d di	> oi

Tableau 11 : Table de transitivité (relations entre intervalles)

Soit la disjonction $A \bigvee_i \alpha_i \; C \quad \Leftrightarrow R$. Il est possible de décomposer cette disjonction en une disjonction de disjonctions, tel qu'il soit possible de déduire chacune des disjonctions à partir de règles de transitivité:

$R \Leftrightarrow (A\ \alpha_1 \vee \alpha_2 \vee \alpha_i\ C) \vee (A\ \alpha_k \vee \alpha_{k+1} \vee \alpha_m\ C) \vee \ldots \vee (A\ \alpha_p \vee \alpha_{p+1} \vee \alpha_n\ C)$

Par lecture du tableau de transitivité, il est possible de trouver des conjonctions faisant intervenir un opérateur supplémentaire menant à chacune des disjonctions:

$(Ar1X \wedge Xr3C) \vee (Ar1X \wedge Xr4C) \vee (Ar2X \wedge Xr3C) \vee (Ar2X \wedge Xr4C) \Leftrightarrow A\ r1 \vee r2\ X \wedge X\ r3 \vee r4\ C$

avec X action fictive, et:

$A\ r1 \vee r2\ X \wedge X\ r3 \vee r4\ C \Rightarrow$

$(A\ \alpha_1 \vee \alpha_2 \vee \alpha_i\ C) \vee (A\ \alpha_k \vee \alpha_{k+1} \vee \alpha_m\ C) \vee \ldots \vee (A\ \alpha_p \vee \alpha_{p+1} \vee \alpha_n\ C)$.

En appliquant ce principe, il est possible de décrire le reste des 64 relations temporelles.

6.5.1.3 Simplification

L'utilisation de la transitivité revient à introduire une action fictive F afin de pouvoir décrire la relation temporelle entre A et B à partir de la relation entre A et F, et de la relation entre F et B. Dans le paragraphe précédent, cette action avait une durée; elle peut être remplacée par un opérateur instantané fictif.

En effet, remplacer une d-action par un opérateur instantané revient à considérer que la durée de la d-action est nulle. Ce qui correspond à confondre le début et la fin de la d-action. Appliquons cette modification au schéma temporel p3.

Soient $F = (f^-, f^+)$, et $A = (a^-, a^+)$. $A <> F \Leftrightarrow p3(f^-, f^+, a^-, a^+)$. (figure 70)

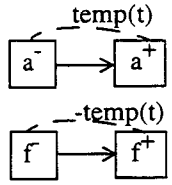


Figure 70: A <> F

Si F est remplacé par un opérateur instantané alors $f^- = f^+ = f$.
Dans ce cas, $p3(f^-,f^+, a^-, a^+) \Leftrightarrow p3(f, f, a^-,a^+) \Leftrightarrow f <> A$. (figure 71)

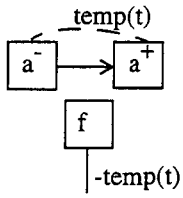


Figure 71: A <> f

Or la contrainte temporelle décrivant un instant (f) avant ou après un intervalle (A) est associée au schéma temporel $p2(f, a^-, a^+)$ (figure 71). Donc, dans le cas où un intervalle a une durée nulle, les schémas temporels p2 et p3 sont équivalents:
$p3(f,f,a^-,a^+) \Leftrightarrow p2(f,a^-,a^+)$. Chaque fois que cette simplification est possible, nous l'appliquons.

6.5.2 Résultat

Dans le tableau 12, $\alpha 1$ et $\alpha 2$ sont des opérateurs instantanés fictifs, c'est-à-dire qu'ils ne sont pas associés à un des changements du monde, mais servent uniquement à la description des relations temporelles. Les schémas $p1(x^-,x^+)$ et $p1(y^-,y^+)$ sont toujours vrais.

$p1(y^-,\alpha 2), p1(\alpha 2,y^+), p2(\alpha 2,x^-,x^+)$	<> d o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p2(\alpha 1,y^-,y^+)$	<> di o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(y^-,\alpha 2), p1(\alpha 2,y^+), p2(\alpha 1,y^-,y^+), p2(\alpha 2,x^-,x^+)$	<> o oi
$p1(x^-,\alpha 1), p1(y^-,x^+), p2(\alpha 1,y^-,x^+), p2(y^+,x^-,\alpha 1)$	> d di o
$p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(y^-,x^+), p2(\alpha 2,x^-,x^+)$	> d o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(y^-,x^+), p2(\alpha 1,y^-,y^+)$	> di o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(y^-,x^+), p2(\alpha 1,y^-,y^+), p2(\alpha 2,x^-,x^+)$	> o oi
$p1(y^-,\alpha 2), p2(\alpha 2,x^-,y^+), p2(x^+,y^-,\alpha 2)$	< d di oi
$p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(x^-,y^+), p2(\alpha 2,x^-,x^+)$	< d o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(x^-,y^+), p2(\alpha 1,y^-,y^+)$	< di o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(x^-,y^+), p2(\alpha 1,y^-,y^+), p2(\alpha 2,x^-,x^+)$	< o oi
$p1(x^-,\alpha 1), p1(\alpha 1,y^+), p1(y^-,\alpha 2), p1(\alpha 2,x^+), p2(\alpha 2,x^-,y^+), p2(\alpha 1,y^-,x^+)$	d di
$p1(x^-,\alpha 1), p1(\alpha 1,y^+), p1(y^-,x^+), p2(\alpha 1,y^-,x^+)$	d di o
$p1(y^-,\alpha 2), p1(\alpha 2,x^+), p1(x^-,y^+), p2(\alpha 2,x^-,y^+)$	d di oi
$p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(y^-,x^+), p1(x^-,y^+), p2(\alpha 2,x^-,x^+)$	d o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(x^-,y^+), p1(y^-,x^+), p2(\alpha 1,y^-,y^+)$	di o oi
$p1(x^-,\alpha 1), p1(\alpha 1,x^+), p1(y^-,\alpha 2), p1(\alpha 2,y^+), p1(x^-,y^+), p1(y^-,x^+), p2(\alpha 1,y^-,y^+), p2(\alpha 2,x^-,x^+)$	o oi
$p1(x^-,\alpha 2), p1(y^-,\alpha 1), p2(\alpha 2,y^-,x^+), p2(\alpha 1,x^-,y^+), p2(x^+,y^-,\alpha 1), p2(y^+,x^-,\alpha 2)$	<> d di
$p1(x^-,\alpha 2), p1(y^-,\alpha 1), p2(\alpha 2,y^-,x^+), p2(\alpha 1,x^-,y^+), p2(y^+,x^-,\alpha 1)$	<> d di o
$p1(x^-,\alpha 2), p1(y^-,\alpha 1), p2(\alpha 2,y^-,x^+), p2(\alpha 1,x^-,y^+), p2(x^+,y^-,\alpha 1)$	<> d di oi
$p1(x^-,\alpha 1), p1(y^-,\alpha 2), p1(\alpha 2,x^+), p2(\alpha 2,x^-,y^+), p2(\alpha 1,y^-,x^+), p2(y^+,x^-,\alpha 1)$	> d di

$p1(y^-,\alpha2), p1(\alpha2,x^+), p2(\alpha2,x^-,y^+)$	> d di oi
$p1(y^-,\alpha1), p1(x^-,\alpha2), p1(\alpha2,y^+), p2(\alpha2,y^-,x^+), p2(\alpha1,x^-,y^+), p2(x^+,y^-,\alpha1)$	< d di
$p1(x^-,\alpha2), p1(\alpha2,y^+), p2(\alpha2,y^-,x^+)$	< d di o

Tableau 12 : Utilisation de la transitivité pour décrire les 24 dernières relations entre 2 d-actions

Dans l'exemple développé dans le paragraphe 6.5.1.1, on sait déjà que

X di Y1 $\wedge$ Y1 < > Z $\Rightarrow$ X < > o oi di Z.

De la même façon, X < > Y2 $\wedge$ Y2 d Z $\Rightarrow$ X < > o oi d Z.

Donc X di Y1 $\wedge$ Y1 < > Z $\wedge$ X < > Y2 $\wedge$ Y2 d Z $\Rightarrow$ X < > o oi Z.

En utilisant la transitivité, nous sommes donc parvenus à décrire une disjonction temporelle entre deux intervalles contenant les deux primitives o et oi, sans contenir les primitives d et di. Il suffit alors d'appliquer le principe décrit dans le paragraphe 6.4.1 pour éliminer les primitives < et > de cette disjonction:

X < d di o oi Z $\wedge$ X > d di o oi Z $\Rightarrow$ X d di o oi Z.

Alors X d di o oi Z $\wedge$ X < > o oi Z $\Rightarrow$ X o oi Z.

En conclusion:

$p1(x^-,y1^-) \wedge p1(y1^+,x^+) \wedge p1(z^-,y2^-) \wedge p1(y2^+,z^+) \wedge p1(x^-,z^+) \wedge p1(z^-,x^+)$
$\wedge p3(y1^-, y1^+,z^-,z^+) \wedge p3(y2^-, y2^+,x^-,x^+) \Rightarrow$ X o oi Z . (figure 72)

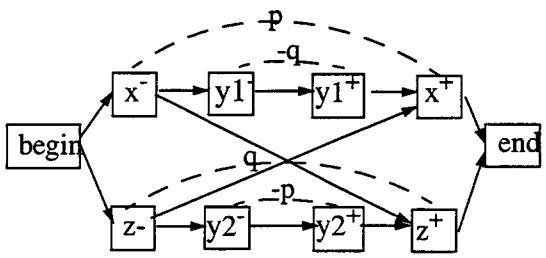


Figure 72: X o oi Z avec deux d-actions intermédiaires

Si onsuppose que les actions intermédiaires sont des opérateurs fictifs instantanés, alors $y1^- = y1^+ = y1$ et $y2^- = y2^+ = y2$, et

$p1(x^-,y1) \wedge p1(y1,x^+) \wedge p1(z^-,y2) \wedge p1(y2,z^+) \wedge p1(x^-,z^+) \wedge p1(z^-,x^+)$
$\wedge p2(y1, z^-, z^+) \wedge p2(y2, x^-, x^+) \Rightarrow$ X o oi Z . (figure 73)

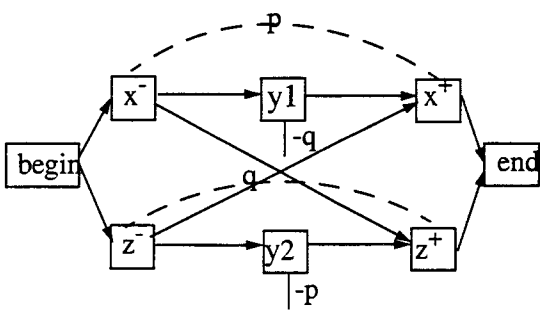


Figure 73: X o oi Z avec deux opérateurs fictifs intermédiaires

Nous avons expliqué comment utiliser la relation de précédence, les réparations des conflits maintien-effet et maintien-maintien, ainsi que la transitivité de la relation de précédence entre instants pour décrire des relations temporelles disjonctives entre deux

d-actions. Dans le paragraphe suivant nous décrivons l'implémentation de la table de transition.

6.6 Mise en oeuvre

6.6.1 Principe

Nous avons construit une table de transition entre un sous-ensemble des relations d'Allen et le formalisme des planificateurs traditionnels non linéaires à liens causaux.

L'implémentation de cette table consiste à insérer une étape de compilation du langage externe vers le langage des planificateurs traditionnels. Cette étape est lancée avant chaque phase de construction d'un plan en réponse à un problème. La figure 74 illustre les relations entre la description du problème, la table de transition et le planificateur traditionnel.

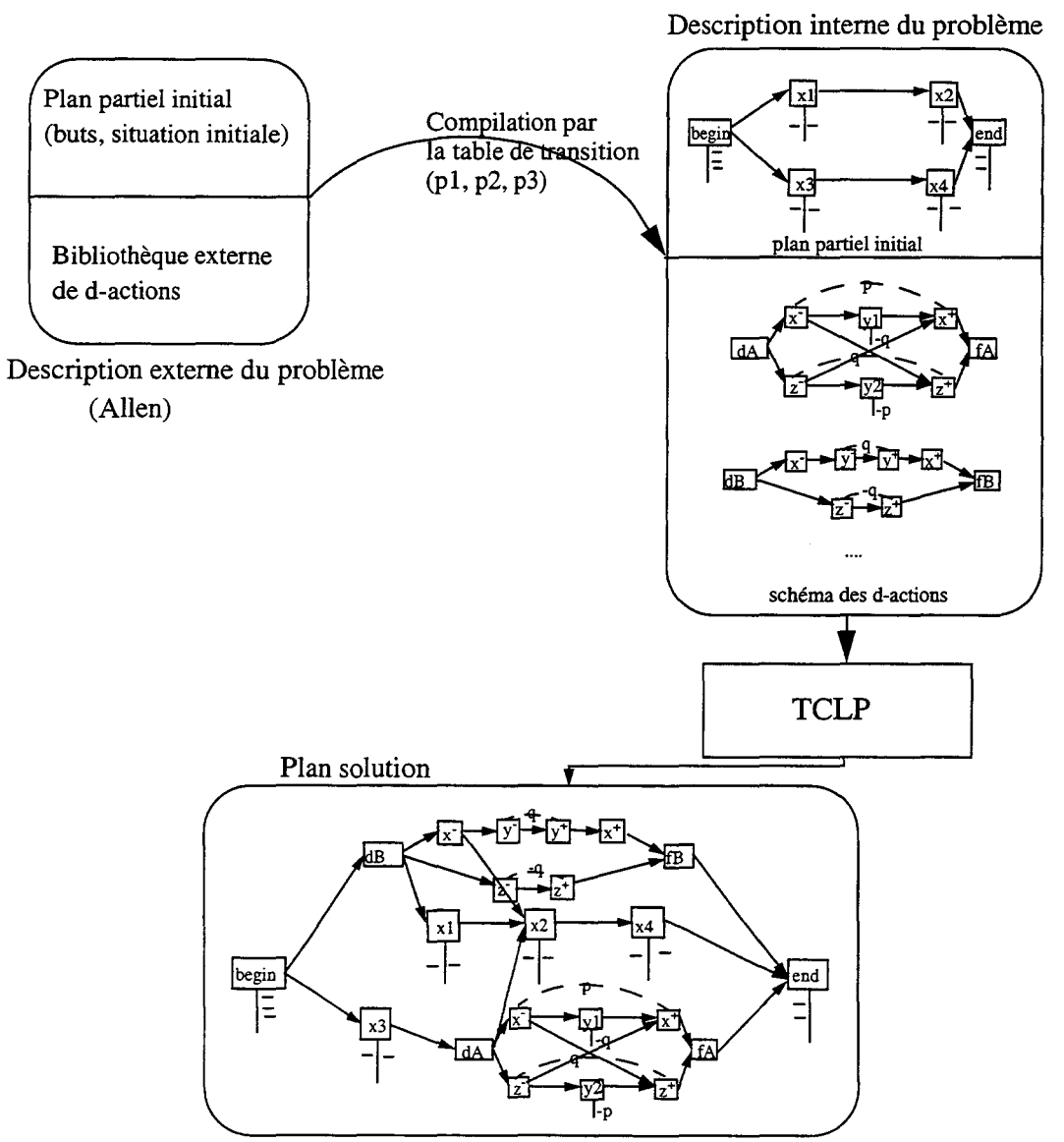


Figure 74: Insertion de la table de transition entre le langage externe et TCLP

A chaque relation temporelle lue dans la description des d-actions correspond une suite de mises à jour qui traduisent p1, p2 et p3.

Pour pouvoir utiliser les d-actions dans un planificateur à liens causaux, il suffit d'étendre la notion de lien causal à celle du maintien, et celle des conflits. Il est également nécessaire de modifier l'opération d'introduction d'une action dans un plan, car il s'agit d'introduire un plan partiel d'opérateurs instantanés, image d'une d-action dans le formalisme des plans traditionnels à liens causaux. Ces modifications sont décrites dans le chapitre 9, consacré à l'algorithme de planification de TCLP.

Cependant, une partie de la table peut être utilisée sans étendre le concept de lien causal. Dans ce cas, toutes les relations des tableaux 5, 6 et 7 sont utilisables, sauf les 7 dernières du tableau 7 car elles nécessitent le concept de maintien sans contraintes de précédence. Il faut cependant remarquer que p2 et p3 peuvent être implémentés à l'aide du lien causal seulement si des effets fictifs (producteurs de lien) et des préconditions fictives (consommateurs de lien) sont ajoutés à la description des opérateurs instantanés concernés par le lien causal.

Par exemple, p2(x^-, y^-, y^+) se traduit par:

- maintien(-p, y^-, y^+) et
- effet (p, x^-),

alors qu'il se traduit par:

- maintien(-p, y^-, y^+) et effet(-p, y^-) et precondition(-p, y^+)
- effet (p, x^-) si le concept de lien causal est utilisé.

Les modifications à réaliser sont simples et ne nécessitent pas de remise en cause des principes de base des planificateurs à liens causaux. Il s'agit de modifier les d-actions de telle sorte que la relation temporelle exprimée entre les d-actions soit transformée en ajout de relation de précédence entre instants, en ajout de maintien de fait temporel entre instants, en ajout d'effet temporel (ou effet fictif) et en ajout d'opérateurs fictifs.

- Le schéma p1(x, y) consiste à ajouter la relation de précédence entre l'opérateur instantané x et l'opérateur instantané y.
- Le schéma p2(z, x, y) consiste à ajouter un effet temporel de la forme -temp(T) et un maintien du fait temporel temp(T) entre les instants x et y.
- Le schéma p3(x,y,z,t) consiste à ajouter deux maintiens: maintien(-temp(T), x, y) et maintien (temp(T), z, t).
- L'utilisation de la transitivité consiste à créer un opérateur fictif instantané et à modifier la d-action concernée en insérant correctement l'opérateur fictif pour respecter p1, p2 et p3.

Nous décrivons dans le paragraphe suivant, les procédures de mise à jour de la table de transition. Les procédures concernant la détection des conflits basés sur le maintien sont données dans le chapitre 9.

6.6.2 Procédures de mise à jour

- **Définition des variables**

Une d-action A est définie par:
- A.***nom***: nom de la d-action
- A.***type***: type de l'action = d-action
- A.***code***: code identifiant de façon unique une d-action dans un plan.
- A.***comp***: liste des composantes de la d-action A (liste de d-actions)
- A.***op***: liste des opérateurs instantanés constituant la d-action A
- A.***début***: opérateur instantané de début de A
- A.***fin***: opérateur instantané de fin de A
- A.***maintiens***: liste des maintiens contenus dans la d-action A
- A.***contraintes***: liste des contraintes posées sur les variables de l'action
- A. ***index***: ensemble des effets primaires spécifiés comme index de recherche de de l'action

Les relations temporelles entre les composantes de la d-action A sont traduites dans les opérateurs instantanés (relation de précédence et effet) et dans les maintiens de la d-action.

Un opérateur instantané op est défini par:
- op.***nom***: nom de l'opérateur instantané
- op.***type***: type de l'opérateur = instant
- op.***code***: code identifiant de façon unique un instant dans un plan. Si op appartient à la d-action A, alors op.code = (A.code, code_op).
- op.***lprec***: liste des préconditions de l'opérateur op
- op.***add***: liste des effets ajoutés de l'opérateur op
- op.***del***: liste des effets retirés de l'opérateur op
- op.***d-action***: d-action à laquelle appartient l'opérateur op
- op.***lsucc***: liste des codes des opérateurs instantanés immédiatement après op.
- op.***lsucctot***: liste des codes des opérateurs instantanés après op.
- op.***contraintes***: liste des contraintes posées sur les variables de l'opérateur op.

- **Description des procédures de mise à jour**

Les algorithmes 1 à 5 correspondent aux modifications élémentaires nécessaires pour traduire p1, p2 et p3.

L'algorithme 6 correspond au chargement de la bibliothèque des d-actions pour initialiser le problème de planification.

L'algorithme 8 correspond à l'implémentation de la table de transition pour toutes les relations temporelles que nous acceptons.

Algorithme 1: p1(Comp1, Comp2)
Début

```
Si Comp1.type = instant & Comp2.type = instant alors
        Comp1.lsucc <- Comp1.lsucc + Comp2.code
Fsi
Si Comp1.type = instant & Comp2.type = d-action alors
        Deb2 <- Comp2.début
        Comp1.lsucc <- Comp1.lsucc + Deb2.code
Fsi
Si Comp1.type = d-action & Comp2.type = d-action alors
        Fin1 <- Comp1.fin
        Deb2 <- Comp2.début
        Fin1.lsucc <- Fin1.lsucc + Deb2.code
Fsi
```

Fin

Algorithme 2: p2(Instant3, Instant1, instant2,Fait)
Début

```
ajouter-maintien(Instant1, Instant2, temp(T) )
ajouter-effet(Instant3, -temp(T) )
Fait <- temp(T)
```

Fin

Algorithme 3: ajouter-maintien(Instant1, Instant2, Fait)
Début

```
A <- Instant1.d-action
A.maintiens <- A.maintiens + maintien (Fait, Instant1, Instant2)
```

Fin

Algorithme 4: ajouter-effet (Instant, Effet)
Début

```
Instant.add <- Instant.add + Effet
```

Fin

Algorithme 5: p3(Instant1, Instant2, Instant3, Instant4, Fait)
Début

```
ajouter-maintien(Instant1, Instant2, temp(T))
ajouter-maintien(Instant3, Instant4, -temp(T))
Fait <- temp(T)
```

Fin

Nous définissons ici tous les éléments nécessaires à description des algorithmes de transformations des relations temporelles au sein des actions définies dans la bibliothèque.

Soit ***Biblio*** l'ensemble des d-actions définies par l'utilisateur.
Soit A une d-action.
Soit Ci l'ensemble des composantes de la d-action sauf le début et la fin.
Soit D l'ensemble des relations temporelles disjonctives acceptées entre deux d-

actions. Ces relations sont décrites par l'utilisateur en utilisant la notation d'Allen (<, >, d, di o et oi); toutes les disjonctions sont données par ordre alphabétique dans la colonne de gauche de la table de transition en annexe C.3. A chacune des disjonctions est associée une procédure qui fait appel à p1, p2 et p3 (colonne de droite). Lorsque la disjonction temporelle nécessite l'utilisation de la transitivité, il est nécessaire de créer autant d'opérateurs fictifs que nécessaire, puis d'appliquer les procédures p1, p2 et p3 à l'ensemble des opérateurs instantanés.

Soit Rij la relation temporelle entre la composante Ci et la composante Cj; $Rij \in D$.

Soit A.***CT*** l'ensemble des contraintes temporelles entre les composantes de la d-action A. Une contrainte temporelle entre Ci et Cj est notée Ci Rij Cj.

Lors de l'initialisation du problème, toute d-action de la bibliothèque Biblio-externe est transformée en un ensemble d'opérateurs instantanés positionnés temporellement à l'aide de p1, p2 et p3. La traduction de la d-action est rangée dans Biblio.

Algorithme 6: initialisation-actions (Biblio-externe)
Début

```
Pour chaque A de Biblio
    A' <- maj-actions (A)      % transforme une d-action en un ensemble d'opérateurs instantanés,
                               % de maintiens entre opérateurs instantanés,
                               % met à jour les champs A.index, A.contraintes, A.début, A.fin
    maj-contraintes-temporelles(A)
    Biblio <- Biblio U {A'}
    Retourner Biblio
Finpour
Fin
```

Algorithme 7: maj-contraintes-temporelles (A)
Début

```
Pour chaque a de A.comp
    Si a.type = d-action alors
        maj-contraintes-temporelles (a)
    Fsi
Finpour

Pour chaque Ci Rij Cj de A.CT
    maj-contrainte-temporelle (Ci, Rij, Cj)
Finpour

Pour chaque a de A.comp
    p1(a.fin, A.fin)
    p1(A.début, a.début)
Finpour
Fin
```

L'algorithme 8 correspond au traitement des 64 relations temporelles possibles sous la forme "si R= ' ' alors faire...".

Chaque cas correspond exactement à l'application des algorithmes 1, 2 et 5 en fonction de l'utilisation de p1, p2 et p3 pour décrire les relations temporelles (voir

annexe C.3).

Algorithme 8: maj-contrainte-temporelle (A, R, B)
Début

Si R = 'A > B' **alors**
p1(B.fin, A.début)
Fsi

Si R = ' A > < B' **alors**
p3(A.début, A.fin, B.début, B.fin, Fait)
Fsi

Si R = ' A > < d B' **alors**
p2(B.début, A.début, A.fin, Fait)
ajouter-effet(B.fin, -Fait)
Fsi

...

Si R = ' A > d o oi B' **alors**
créer-opérateur-fictif(Y) % retourne un opérateur instantané Y de type no-op
p1(B.début, A.fin)
p1(B.début, Y)
p1(Y, B.fin)
p2(Y, A.début, A.fin, Fait)
Fsi

...

Fin

L'ensemble des plans partiels correspondant aux 64 relations temporelles possibles entre 2 d-actions est donné en annexe D. Ces relations ne font pas intervenir la relation d'égalité entre instants. Le paragraphe suivant est consacré à l'étude de cette relation.

6.7 Cas de l'égalité et de la simultanéité

Nous avons jusqu'à maintenant traité le problème des relations temporelles sans nous soucier du problème de l'absence d'égalité entre instants et plus généralement de la simultanéité. Dans ce paragraphe, nous expliquons pourquoi nous ne traitons pas l'égalité et essayons de comprendre en quoi cela peut être génant ou non.

Nous employons généralement le terme d'égalité pour relier des objets temporels (instants ou intervalles), et le terme de simultanéité pour relier des actions ou des événements.

6.7.1 Les besoins identifiés, les solutions apportées

Le simulateur de combats sur lequel nous nous sommes basés pour étudier l'adéquation de la planification à la synthèse de mission au niveau stratégique a soulevé

plusieurs besoins que nous rappelons ici:

- la ***construction efficace de plans*** d'actions (ie, ne faisant pas appel à des algorithmes dont la complexité est trop importante)
- la nécessité de construire des plans où ***plusieurs agents coopèrent*** pour réaliser des buts définissant la mission. Ainsi, la destruction d'un point cible peut nécessiter l'attaque de plusieurs troupes en parallèle. Dans certains cas (situation où l'ennemi est très présent), l'attaque réalisée par une troupe doit être soutenue par une autre troupe.
- la nécessité de décrire le ***déroulement*** de certaines actions dont les effets ne se manifestent que pendant leur application: l'action "soutenir" précédemment évoquée ne peut pas être décrite uniquement par les effets provoqués une fois l'action terminée, car le soutien n'est réalisé que pendant l'application de l'action.
- la traduction de ***contraintes*** représentant les doctrines militaires, du type "toute attaque réalisée dans le cas où l'ennemi est très présent ne peut être effectuée sans qu'un soutien n'ait été prévu". D'où la nécessité d'introduire dans le plan, en même temps que l'action "attaquer", l'action "soutenir" avec les contraintes temporelles adéquates.

Nous avons vu dans la partie I, que répondre à ces besoins passe nécessairement par la ***prise en compte de la durée des actions***, et la possibilité de décrire des situations où ***des actions s'entrelacent***.

Nous avons également identifié diverses solutions:

- les ***planificateurs traditionnels*** (TWEAK [CHA 87], SNLP [MCA 91], UCPOP [PEN 92]), connus pour leur efficacité mais qui, parce qu'il supposent que les actions sont instantanées, ne permettent pas de répondre aux besoins précédemment cités quant à la description des actions.
- les ***planificateurs temporels basés sur l'algèbre d'Allen*** (Timelogic [ALL 83b], TLP [TSA 86], Descartes[JOS 96]) sont directement adaptés pour traiter les problèmes d'interaction entre actions. Ils permettent de décrire le déroulement des actions et leur entrelacement. Cependant, ils sont confrontés à des algorithmes de gestion des relations temporelles très complexes, limitant leur utilisation. Cette complexité est notamment due au fait que ces planificateurs conservent toutes les disjonctions temporelles tant que rien ne leur permet de les simplifier. Par ailleurs, nous avons vu dans le chapitre 3 que la description des actions dans ce cadre est relativement complexe.
- les ***planificateurs temporels basés sur l'algèbre d'instants*** (IxTeT [GHA 94], ZENO [PEN 94]) qui répondent également au problème de la prise en compte de la durée des actions et de leur entrelacement, et qui utilisent des algorithmes dont la complexité est diminuée grâce à une restriction des relations temporelles autorisées (algèbre d'instants ou algèbre d'intervalles restreintes) ([GHA 89], [VIL 86]). Cette restriction ne permet pas d'exprimer toutes les relations temporelles possibles entre deux intervalles (ou deux d-actions). En outre les planificateurs que nous avons étudiés ne conservent pas les disjonctions temporelles, mais font des choix issus de leurs principes de construction des plans.

Afin de bénéficier de l'efficacité des planificateurs traditionnels ainsi que des capacités d'expression des deux autres approches, nous avons construit un modèle d'actions (les d-actions) qui permet d'étendre leur capacité de description sans modifier leurs principes de base.

Afin de construire des plans contenant des actions s'entrelaçant, nous avons également construit une table de transition entre des relations d'Allen et le formalisme temporel de ces planificateurs (relation de précédence entre instants, réparations disjonctives des conflits).

Cependant, l'intégration de notre approche dans les planificateurs traditionnels nécessite l'utilisation de la seule relation de précédence entre instants. C'est pourquoi nous ne pouvons pas décrire des situations telles que " A et B débutent au même instant" ou "A débute dès que B s'achève".

En nous basant sur l'idée que la simultanéité n'est pas pratiquement réalisable et qu'elle n'est pas requise dans notre application, nous supposons que l'absence d'égalité n'est pas une gène dans la construction des plans. Nous développons ce point dans le paragraphe suivant.

6.7.2 Utilité de l'égalité

L'égalité permet de synchroniser des événements ou des actions:

- ***deux actions sont dites consécutives*** lorsque la deuxième action débute dès que la première se termine (figure 75).

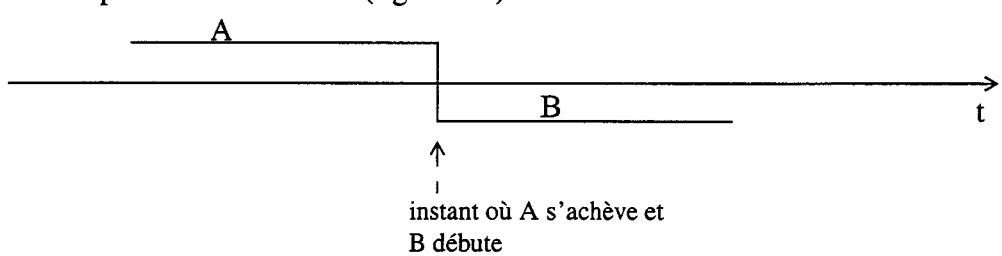


Figure 75: A et B sont deux actions consécutives

La succession de deux actions implique certains liens entre les actions:

- A et B ne peuvent pas se chevaucher car leur chevauchement est incohérent (utilisation d'une même ressource non partageable).
- B a besoin d'un fait produit par A (le coureur démarre dès que le juge donne le signal de départ)
- l'exécution simultanée de A et B conduit à un état incohérent car l'une produit un effet qui est détruit par l'autre.
- des contraintes temporelles sur la durée d'exécution du plan nécessitent que A et B soient consécutives plutôt que séquentielles uniquement.

- ***deux actions sont simultanées*** si elles débutent au même instant et s'achèvent au même instant (figure 76).

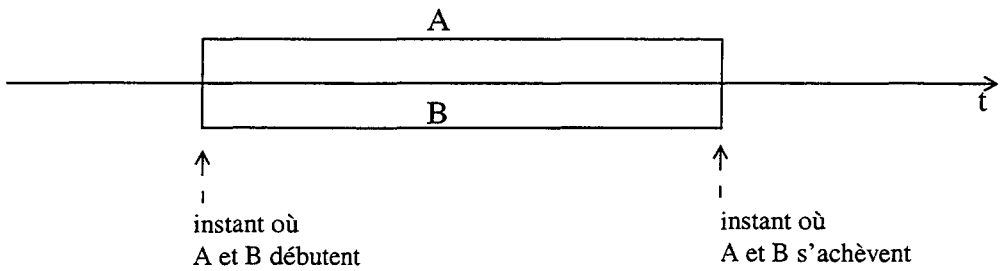


Figure 76: A et B sont deux actions simultanées

Deux actions peuvent être exécutées simultanément pour diverses raisons:

- deux actions sont simultanées pour réaliser un but commun ("enfoncer un clou" est l'application simultanée de "tenir le clou" et "frapper le clou avec un marteau").
- deux actions sont simultanées pour construire un comportement général différent du comportement individuel associé à chaque action (une attaque seule comparée à une attaque soutenue).
- les deux actions sont simultanées parce que des contraintes temporelles sur la durée d'exécution du plan nécessitent la réalisation simultanée de ces deux actions. Ces deux actions sont dans ce cas complètement indépendantes.

Cependant, est-il possible de synchroniser parfaitement les événements entre eux?

6.7.3 Prise en compte de la simultanéité

De façon générale, il est admis que la simultanéité n'est pas réalisable en pratique et, dans les exemples précédents la simultanéité est une approximation. Par exemple, dans le cas où A et B sont consécutives parce que B attend le résultat de A, "A débute dès que B s'achève" signifie que B débute ***juste après*** la fin de A. De même, dire que deux instants sont égaux signifie qu'il existe un intervalle de temps négligeable entre les deux instants, vis à vis des durées considérées dans le problème traité.

Ainsi, 1 seconde peut être négligeable devant des durées de l'ordre de l'heure. Elle devient non négligeable si les autres durées sont également de l'ordre de la seconde. Dans l'exemple du coureur, on associe très souvent le signal du départ et le démarrage du coureur au même instant si on se place à l'échelle de la perception humaine. A plus fine échelle, il existe quelques centièmes de secondes entre les deux événements.

De même, synchroniser deux actions (par les instants de début et/ou fin) signifie qu'il existe un intervalle de temps entre les instants à synchroniser qui est négligeable devant la durée des actions: si les actions ont une durée de l'ordre de l'heure, la synchronisation pourra se faire à quelques secondes près alors que si la durée des actions est de l'ordre de la minute, la synchronisation ne devra pas, par exemple, dépasser la seconde.

Il semble donc que l'égalité admise au niveau des relations symboliques entre objets temporels symboliques (instants ou intervalles) se traduise en des contraintes temporelles numériques visant à définir une durée maximale ε exprimant l'intervalle de

temps maximal accepté entre deux événements e1 et e2, dits simultanés: $|e1 - e2| < \varepsilon$.

Nous avons par ailleurs étudié la prise en compte de la simultanéité au sein de différents planificateurs définis en vue de pouvoir exécuter des plans mettant en jeu plusieurs agents, dont les actions peuvent vérifier les relations temporelles citées dans le paragraphe 6.7.2. Cette étude montre que:

- l'égalité est souvent perçue comme un outil algorithmique de construction de plans
- généralement parallélisme remplace simultanéité
- la considération de l'égalité "théorique" peut conduire à des difficultés quant à la définition de la correction des plans
- l'égalité permet cependant d'exprimer des contraintes supplémentaires entre des actions.

6.7.3.1 L'égalité dans les planificateurs temporels (IxTeT, Descartes et Timelogic)

Dans IxTeT ([LAR 94]), comme dans Timelogic ([ALL 83b]) ou Descartes ([JOS 95]), lors de la construction du plan, l'égalité entre instants (ou entre intervalles) sert à unifier des instants appartenant à des actions différentes. Nous nous inspirons des exemples décrits dans l'annexe B.4 pour expliquer ce point de vue.

- **IxTeT**

Soit le plan décrit dans la figure 77, suivant notre formalisme. Les actions sont décrites dans le formalisme d'IxTeT (figure 78).

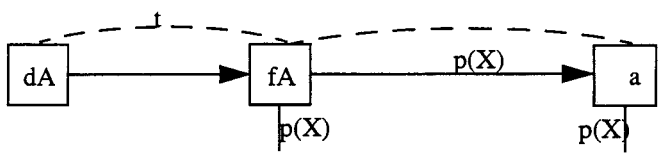


Figure 77: Description d'un plan partiel

```
Task A {
    arg (X) (dA, fA)
    hold(t:1, (dA,fA) )
    event( p(X): (0.1), fA)
            }

Task a {
    arg (X) (a)
    hold (p(X):1, (?T,a))
....
            }
```

Figure 78: Description des actions

Nous expliquons ici comment a été construit ce plan en suivant l'algorithme

d'IxTeT. On suppose que le choix des défauts et des résolvantes est guidé par leur coût associé. La précondition p(X) n'est pas expliquée:

Explication de hold (p(X):1, (?T, a))
Choix de l'establisher = event (p(X) : (_, 1), fA)
Introduction de la tâche A dans le plan
=> p(X) = p(Z), 1=1, ***fA = ?T => fA < a***
et hold(p(X):1, (fA, a))
Propagation des contraintes temporelles: dA < a.

L'égalité a donc été utilisée pour réaliser l'unification entre fA et ?T. En aucun cas l'égalité n'impose que les actions A et a soient consécutives.

Il en est de même lorsqu'une action doit être synchronisée sur un événement prévu.

Soit un événement défini par event (e: (val1, val2), t0). Soit l'action B définie dans la figure 79. Cette action ne peut être appliquée que si la condition hold(e: val2, (?T1, dB)) est vraie (figure 80). Cette action pourra donc être appliquée après l'événement e, c'est-à-dire après t0, pourvu qu'au moment où elle est appliquée, e ait toujours la valeur val2.

```
Task B{
    arg (dB, fB)
    hold(e: val2 , (?T1, dB) )
    ...
    }
```

Figure 79: Définition de l'action B

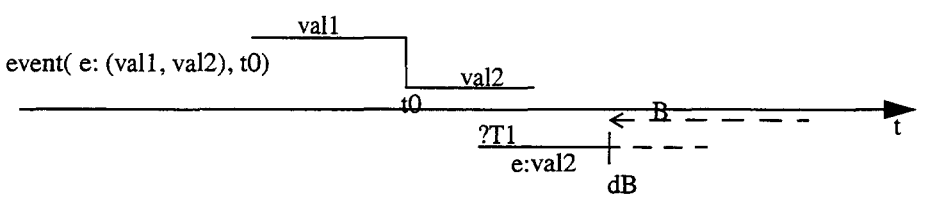


Figure 80: Synchronisation d'un événement et d'une action

IxTeT identifie alors t0, l'instant auquel e prend la valeur val2, avec ?T1, l'instant à partir duquel e prend la valeur val2, pour que l'action B puisse s'appliquer: t0 = ?T1 (figure 81).

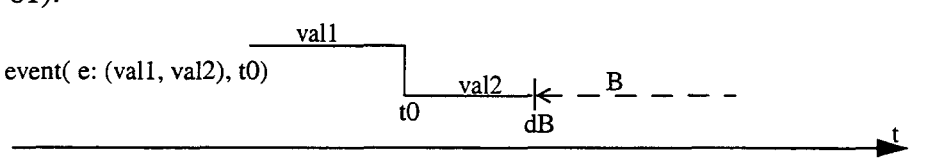


Figure 81: e prend la valeur val2 de t0 à dB (au moins): B peut être appliquée

Encore une fois l'égalité entre instants sert pour la construction du plan, mais ne permet pas d'imposer la stricte simultanéité entre un événement et le déclenchement d'une action. Rien n'empêche le planificateur d'insérer une action entre t0 et dB, pourvu que cette action ne modifie pas la valeur de e, et que sa durée ne soit pas supérieure à celle de l'intervalle délimité par t0 et dB.

- **Timelogic / Descartes**

Le même raisonnement peut être tenu dans le cadre des planificateurs Timelogic et descartes, tous deux basés sur l'algèbre d'intervalle d'Allen.

Nous rappelons le cas étudié dans la figure 82 dans notre formalisme.

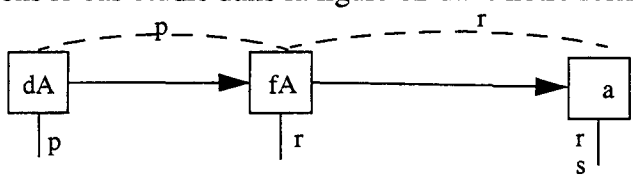


Figure 82: Plan partiel

Une action X est décrite dans le formalisme de Timelogic / Descartes par:

- un intervalle IX
- des préconditions PX
- des effets EX
- des contraintes temporelles (Allen) et des contraintes sur les variables
- un switch SX servant à la construction des plans.

Toutes les propositions (effets et préconditions) sont qualifiées temporellement:
< p , -, -, Ip > signifie que la propriété p est vraie sur l'intervalle Ip.

La description des actions dans le formalisme Descartes est donnée dans la figure 83:

A = < IA, PA, EA, KA, SA >	***a = < Ia, Pa, Ea, Ka, Sa >***
PA = []	Pa = < r, -, -, ir > ,
EA = < p, -, -, Ip >	< s, -, -, is>
< r,-,-, Ir >	Ea = []
KA = { Ip = IA , Ir mi IA }	KA = { is m Ia, ir m Ia }

Figure 83: Description des actions

La précondition r de l'action a n'est pas expliquée:

Etablissement de r:
Choix de A (seul établisseur possible de r)=>
< r, -, -, Ir > = < r, -, -, ir > => ***Ir = ir*** => ***IA < Ia***

Lors de l'unification des propositions pour l'établissement de r, les intervalles sont également unifiés en utilisant la contrainte d'égalité entre les intervalles.

6.7.3.2 Recherche de parallélisme entre actions dans des plans partiellement ou totalement ordonnés

Certains systèmes de planification sont constitués de deux modules: un module de construction des plans qui utilise des techniques de planification traditionnelle pour construire les plans, et un module de recherche de parallélisme dans les plans construits.

Ainsi, dans le système SPEEDY ([BAS 95]), les plans sont construits en utilisant des techniques de planification linéaire, puis optimisés par recherche de parallélisme ([REG 91]). Dans [REG 91], le parallélisme est défini par "deux actions A et B sont exécutables en parallèle si elles sont exécutables dans n'importe quel ordre", c'est-à-dire

si l'intersection des intervalles associés est quelconque, non nulle: A d di f fi o oi s si = B. L'exécution en parallèle de deux actions ne signifie pas que les instants de début (et de fin) sont identiques. Cette définition du parallélisme est comparable à celle utilisée dans CARNEADE: deux actions sont exécutées en parallèle si l'intersection des intervalles associés est non nulle. Comme la simultanéité n'est pas envisagée dans ce système, A et B sont en parallèle si A d di o oi B. P. Régnier et B. Fade définissent alors des conditions supplémentaires (s'ajoutant à la définition des conflits dans les plans) pour rendre les plans exécutables en parallèle: deux actions sont exécutables en parallèle si leurs effets et conditions d'applications sont mutuellement compatibles et s'il n'existe pas d'actions entre A et B non exécutables en parallèle.

Suivant le même principe, M.M Veloso, M.A Perez et J.G. Carbonel ont développé un algorithme de parallélisation à partir de plans non linéaires ([VEL 90]). Ils définissent comme précédemment des conditions d'exécution en parallèle supplémentaires aux critères de correction des plans. Ces conditions sont essentiellement inspirée de l'utilisation de plusieurs ressources par plusieurs agents dans un plan.

En fait, ces contraintes supplémentaires proviennent du fait qu'accepter l'exécution d'actions en parallèle nécessite la prise en compte du problème des actions concurrentes interagissant.

6.7.3.3 Extension des planificateurs traditionnels pour considérer les actions concurrentes

Ces approches sont basées sur la même idée que la nôtre: étendre les capacités du modèle STRIPS pour décrire des propriétés supplémentaires, tout en conservant les principes de construction des planificateurs traditionnels.

Dans [HOR 94], A. Horz étudie la structure des planificateurs traditionnels et temporels. Il met en évidence la nécessité d'affiner la notion d'interaction entre les actions lorsque celles-ci peuvent s'exécuter simultanément.

Il définit la ***simultanéité de deux actions*** comme un cas spécifique de la concurrence: les intervalles associés aux deux actions sont égaux.

Il définit l'***indépendance de deux actions*** par: deux actions sont indépendantes si quel que soit l'ordre dans lequel elles sont exécutées, le résultat obtenu est identique.

Des actions concurrentes peuvent s'exécuter simultanément; l'exécution simultanée des deux actions doit être cohérente, c'est-à-dire conduire à un monde cohérent. Alors la définition des conflits utilisée dans les planificateurs traditionnels n'est plus suffisante pour assurer la cohérence des plans vis à vis de l'exécution.

En effet, soient deux actions A et B de type STRIPS: A produit p et B détruit p.

- Si p est utilisé dans le plan (producteur d'un lien causal dans le cas des planificateurs à liens causaux), alors le planificateur détecte un conflit, et impose une contrainte qui empêche l'exécution simultanée des deux actions (figure 84).

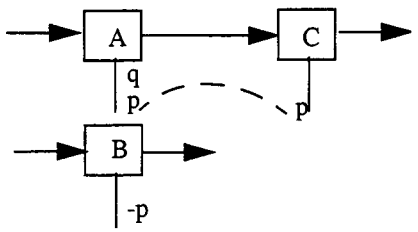


Figure 84: Plan partiel avec conflit:
B ne peut pas être exécutée en parallèle avec A

- Si p n'est pas utilisé dans le plan, alors le planificateur traditionnel ne détecte pas de conflit (figure 85). Ces deux actions ne cependant pas exécutables en parallèle car selon l'ordre d'application, le résultat obtenu n'est pas identique. De plus, dans le cas où les actions sont exécutées simultanément, quel est la structure du monde obtenu: p est-il vrai dans le monde résultant où non?

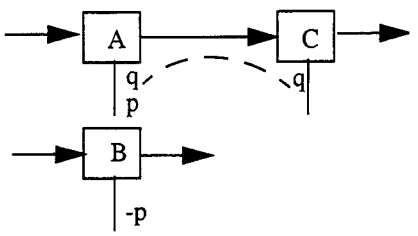


Figure 85: Plan partiel sans conflit (p n'est pas utilisé dans le plan)

Ce problème provient du fait que la simultanéité est considérée comme réalisable. Or si la situation est analysée à petite échelle de temps, il existe toujours un intervalle de temps (même très petit) entre la fin d'une action et la fin de l'autre. Dans ce cas, on peut admettre que les actions de la figure 85 sont exécutables en parallèle, avec les fins des deux actions non ordonnées (avant ou après). Et, dans le cas où p n'est pas utilisé dans le plan, peu importe si p est vrai ou non à la fin de l'exécution des deux actions.

C.A. Knoblock est aussi confronté à ce problème dans [KNO 94], et suppose également que la simultanéité est réalisable: il cherche à optimiser le temps d'exécution des plans en exécutant des actions (de type STRIPS) en parallèle. Il décrit plusieurs critères plus ou moins contraignants, permettant de déterminer les actions exécutables en parallèle dans un plan partiellement ordonné: actions indépendantes, actions indépendantes relativement à un but, sous-plans indépendants relativement à un but, actions interagissant. Cependant, le travail effectué par C.A. Knoblock est destiné à gérer les conflits de ressource entre actions (qui, étant basées sur le type STRIPS, ne contiennent pas la description d'utilisation de ressource), et non pas à construire des plans où l'interaction de plusieurs actions est nécessaire pour le succès du plan.

Dans ce paragraphe, nous avons mis en évidence les difficultés théoriques rencontrées lorsque la simultanéité est considérée comme réalisable, alors que dans la pratique elle ne l'est pas.

Nous ne rencontrons pas les problèmes évoqués car nous ne considérons pas l'égalité entre instants, et ce pour pouvoir intégrer notre approche dans les planificateurs traditionnels qui ne considèrent que la relation de précédence entre instants. Alors, les conflits classiques sont suffisants pour déterminer la cohérence des plans, même vis à vis

de l'exécution.

Cependant, l'égalité dans les relations symboliques peut être vue comme une durée imperceptible dans le cas de relations numériques, et sert à synchroniser des événements de façon plus précise que la relation de précédence. Il semble donc nécessaire de différencier la relation de précédence entre instants du concept de proximité temporelle lié à l'égalité. Cette distinction fait l'objet des travaux réalisés sur une représentation granulaire du temps.

6.7.3.4 Comment remédier à l'absence de concept de simultanéité?

Le but d'une représentation granulaire du temps est de représenter le temps sous plusieurs niveaux de détails ([EUZ 93]). Chaque niveau est appelé granularité. Cette approche conserve la représentation symbolique du temps et simplifie la représentation numérique. J. Euzénat propose dans [EUZ 93] des opérateurs de conversion des représentations entre deux granularités.

Nous avons vu que, dans CARNEADE, deux actions X et Y sont parallèles si elles se déroulent sur un intervalle de temps commun. Nous traduisons actuellement cette contrainte par la disjonction X d di o oi Y. Cependant, cette relation peut conduire à un intervalle commun très réduit détruisant la notion de parallélisme (figure 86, cas b).

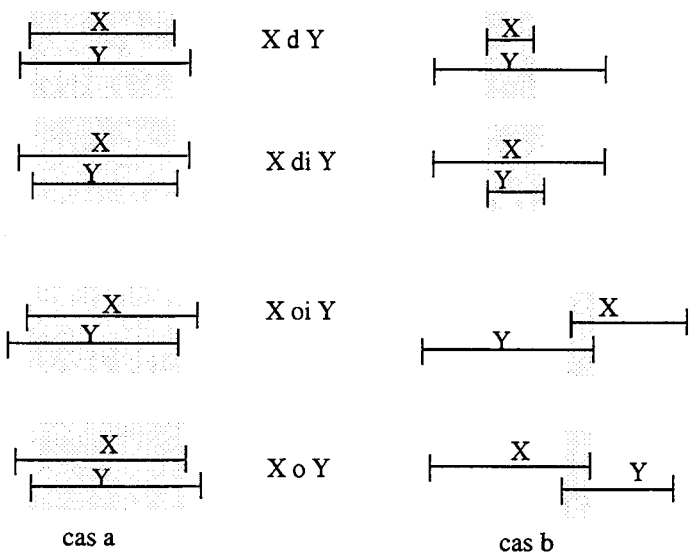


Figure 86: X d di o oi Y; en grisé, l'intervalle commun au deux actions

Ce problème peut être rapproché de la notion de granularité: à un niveau grossier de la représentation (ou au niveau stratégique de la simulation de combats) les actions se déroulent en même temps, à un niveau de détail plus fin, on distingue un intervalle de temps faible entre les instants de début et fin des actions.

Nous ne sommes pas capables actuellement de distinguer un intervalle de temps faible d'un intervalle de temps élevé. On ne sait donc pas distinguer les cas a et b de la figure 86. Cependant, on pourrait imaginer un processus traduisant des relations grossières en relations plus fines en utilisant uniquement la relation de précédence et en définissant le début et la fin de chaque d-action comme des intervalles de durée négligeable. Cette représentation traduit l'imprécision du début et de la fin des actions correpondant à l'incapacité de respecter une simultanéité stricte.

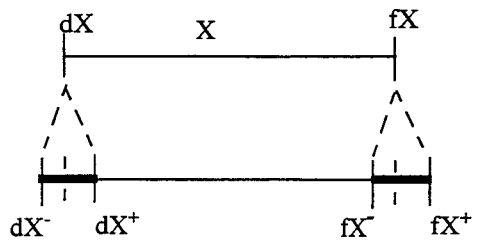


Figure 87: Le début de l'action X se situe entre dX^- et dX^+;
la fin de l'action X se situe entre fX^- et fX^+

L'utilisateur perçoit dX et fX comme des instants. Le plan les perçoit comme des intervalles particuliers, de durée imperceptible.

Les intervalles dX et fX pour toute d-action X ont un type: le type ***début-fin***.

Tout intervalle de type début-fin peut s'entrelacer avec tout autre intervalle de type début-fin.

Un intervalle de type début-fin ne peut pas s'entrelacer avec un intervalle d'un autre type (ie, les intervalles représentant la durée des d-actions).

En utilisant cette technique il est alors possible de distinguer les cas a et b de la figure 86. La relation X d di o oi Y (telle que l'intervalle commun au deux actions soit non négligeable, cas a) est représentée dans la figure 88. La description externe de cette relation serait X = Y.

Le début d'une d-action X est notée $dX= X^- = (X^-_-, X^-_+)$; la fin de la d-action est notée $fX= X^+ = (X^+_-, X^+_+)$.

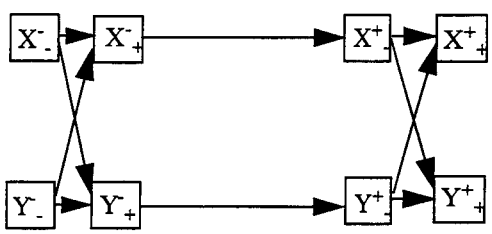


Figure 88: X d di o oi Y ≡ dX d di o oi dY ∧ fX d di o oi fY

Il est possible de décrire les 13 relations d'Allen en décrivant les relations entre X^-, X^+ et Y^- et Y^+ désormais vus comme des intervalles.

X r Y	Relations entre intervalles de début et fin
X < Y	fX < dY
X m Y	fX d di o oi dY
X o Y	dX<dY, dY<fX, fX<fY
X s Y	dX d di o oi dY, fX < fY
X d Y	dY < dX, fX< fY

Tableau 13 : Traduction des 13 primitives d'Allen en conjonctions de relations entre intervalles de début et fin des actions.

X r Y	Relations entre intervalles de début et fin
X f Y	dY<dX, fX d di o oi fY
X oi Y	dY<dX, dX< fY, fY<fX
X mi Y	fY d di o oi dX
X > Y	fY < dX
X si Y	dX d di o oi dY, fY < fX
X = Y	dX d di o oi dY, fX d di o oi fY
X fi Y	dX < dY, fX d di o oi fY
X di Y	dX < dY, fY< fX

Tableau 13 : Traduction des 13 primitives d'Allen en conjonctions de relations entre intervalles de début et fin des actions.

Ce point reste à étudier, notamment par rapport à la complexité de la propagation des relations et à l'expression de l'ensemble des relations d'Allen à l'aide de notre table de transition.

6.8 Conclusion

Nous avons construit dans ce chapitre une table de correspondance entre un ensemble de relations entre intervalles issues de l'algèbre d'Allen et le formalisme temporel lié au fonctionnement des planificateurs traditionnels non linéaires à liens causaux. Cet ensemble de relation correspond à toutes les disjonctions d'Allen ne faisant pas intervenir l'égalité entre instants.

Cette table nous permet de construire des plans où les actions s'entrelacent tout en conservant les mécanismes de construction de ces planificateurs, et d'exprimer des relations temporelles disjonctives telles que X < > Y.

L'égalité entre instants n'étant pas physiquement réalisable, nous considérons que nous sommes capables de décrire toutes les relations temporelles possibles entre deux intervalles.

Conclusion

Afin de conserver des principes de construction de plans efficaces, nous nous sommes attachés à travailler dans le cadre des planificateurs traditionnels non linéaires à liens causaux.

Nous avons alors établi une passerelle entre l'algèbre d'Allen (sans l'égalité entre instants) et le formalisme temporel lié à ce type de planificateur. Pour cela, nous avons identifiés trois schémas temporels de base qui constituent le squelette temporel des plans traditionnels non linéaires à liens causaux. Ces schémas utilisent la seule relation de précédence entre instants et les réparations disjonctives associées à deux types de conflits: le conflit maintien-maintien et le conflit maintien-effet. Nous avons défini ces conflits à partir de la notion d'incompatibilité sémantique et du concept de maintien.

Dans cette partie, nous nous sommes uniquement préoccupés de la description et de la traduction des relations temporelles entre deux d-actions. Cette traduction étant en partie basée sur les conflits, leur nombre augmente dans les plans: en plus des conflits liés à la sémantique du problème, apparaissent tous les conflits issus des contraintes temporelles entre les d-actions. Nous consacrons la troisième partie du document à la construction de plans avec des d-actions et plus particulièrement à la gestion des conflits.

PARTIE III

Gestion des relations temporelles dans TCLP, un Planificateur Traditionnel non linéaire à Liens Causaux

Introduction

Dans cette partie, nous expliquons comment, dans TCLP, nous gérons des relations temporelles entre d-actions à l'aide des techniques de résolutions de conflits; TCLP est le planificateur que nous avons construit et implémenté. Ce planificateur respecte les caractéristiques des planificateurs traditionnels non linéaires, utilise le maintien (abstraction du lien causal) et intègre la table de transition entre les relations d'Allen et les concepts d'instants, de précédence et de maintien.

Nous rappelons que nous avons étendu le pouvoir d'expression des planificateurs traditionnels afin de pouvoir tenir compte de la durée des actions. Nous avons nommé d-action une action possédant une durée (partie I).

Nous avons également construit une passerelle entre le formalisme d'Allen et le formalisme temporel lié au fonctionnement des planificateurs traditionnels non linéaires à liens causaux (partie II), ceci afin de:

- construire des plans contenant des actions s'entrelaçant
- positionner temporellement deux d-actions à l'aide de n'importe laquelle des 64 relations temporelles disjonctives de l'algèbre d'intervalles ne faisant pas intervenir l'égalité.

Pour cela, nous avons utilisé la seule relation de précédence et les techniques de réparation des conflits. Ainsi, certaines disjonctions temporelles ne sont pas décrites par un ensemble de relation de précédences, mais sont traduites en conflits qui sont traités à l'instar des conflits apparaissant lors de la construction des plans.

Une instance de plan (ou scénario pour P. Van Beek dans [VBE 92]) vérifie non seulement les relations de précédences posées lors de l'élaboration du plan, mais aussi les contraintes temporelles issues des conflits.

Par exemple, le plan de la figure 89 donne lieu à 4 scénari possibles:

- t3 < t4 < t1 < t2 ou
- t3 < t1 < t4 < t2 ou
- t3 < t1 < t2 < t4 ou
- t1 < t2 < t3 < t4

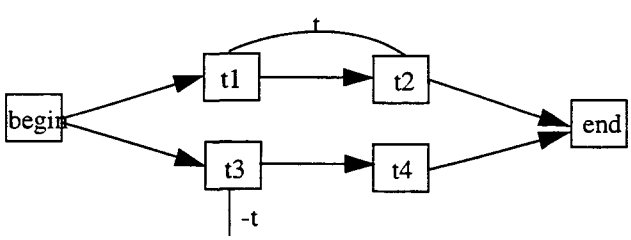


Figure 89: Plan partiel avec conflit maintien-effet

Donc si t1 est le début d'une d-action A, si t2 est sa fin, et si t3 est le début d'une d-action B, et t4 sa fin, ce plan représente la disjonction temporelle A < > d oi B qui n'appartient pas à l'algèbre d'instants. Lorsque l'utilisateur impose cette contrainte entre

A et B, elle est traduite par le conflit maintien-effet du plan de la figure 89.

Notre approche conduit donc à une augmentation du nombre de conflits dans les plans. Cette augmentation est non seulement due au fait que certaines relations temporelles sont traduites en conflits, mais aussi parce que les actions possédant une durée, il est désormais possible de décrire des effets maintenus sur un intervalle, et des conditions d'application vraies sur un intervalle; ces deux caractéristiques étant traduites par des maintiens.

Pour faire face à l'augmentation du nombre des conflits, mais aussi afin de conserver les disjonctions temporelles et diminuer le taux de retour-arrière, il devient nécessaire d'utiliser des stratégies de résolution de conflits particulières.

Dans le chapitre 7, nous décrivons tous les types de conflits que nous traitons dans dans TCLP.

Dans le chapitre 8, nous expliquons comment nous gérons les conflits et comment nous sommes capables de conserver les disjonctions temporelles en retardant les réparations associées aux conflits détectés dans les plans.

Dans le chapitre 9, nous présentons la mise en oeuvre de TCLP (Temporal Causal Link Planner), le planificateur que nous avons construit.

Chapitre 7

Les différents types de conflits dans TCLP

Dans ce chapitre, nous décrivons tous les types de conflits que nous rencontrons dans des plans constitués d'un ordre partiel strict sur un ensemble d'opérateurs et d'un ensemble de maintiens entre opérateurs. Ces types de conflits sont tous issus de la structure générale des conflits maintien-maintien et maintien-effet; ils permettent de simplifier les réparations associées.Nous rappelons la définition des deux conflits dans le paragraphe suivant.

7.1 Définition générale des conflits: rappel

Nous avons identifié deux types de conflits dans les plans traditionnels non linéaires à liens causaux, définis à partir du maintien.

Un ***conflit maintien-effet*** est défini par:
Soit P un plan partiel P= (S, O) avec:
S l'ensemble des opérateurs instantanées du plan et
O l'ensemble des relations de précédence entre les opérateurs de S.
I est l'ensemble des incompatibilités du problème.
Soient x, y et z tels que $(x, y, z) \in S$.

conflit maintien-effet ((p(X), x, y), (q(Y), z))

$\Leftrightarrow$ maintien(p(X),x,y) $\wedge$
z produit q(Y) $\wedge$
incompatibles (p(X), q(Y)) $\in$ I $\wedge$
$O \cup \{x < z, z < y\} \vee O \cup \{y < z, z < x\}$

Un ***conflit maintien-maintien*** est défini par:
Soit P un plan partiel P= (S, O) avec:
S l'ensemble des opérateurs instantanées du plan et
O l'ensemble des relations de précédence entre les opérateurs de S.
I est l'ensemble des incompatibilités du problème.
Soient x, y, z et t tels que $(x, y, z, t) \in S$.

conflit maintien-maintien ((p(X), x, y), (q(Y), z, t))

$\Leftrightarrow$ maintien(p(X),x,y) $\wedge$
maintien(p(X),x,y) $\wedge$
incompatibles (p(X), q(Y)) $\in I$ $\wedge$
$(O \cup \{x < z, z < y\} \vee \quad O \cup \{y < z, z < x\} \vee$
$O \cup \{x < t, t < y\} \vee \quad O \cup \{y < t, t < x\} \vee$
$O \cup \{z < x, x < t\} \vee \quad O \cup \{t < x, x < z\} \vee$
$O \cup \{z < y, y < t\} \vee \quad O \cup \{t < y, y < z\})$

Par ailleurs, J.Hertzberg et A. Horz ont formalisés le concept de conflit dans [HER 89] et ont définis 4 types de conflits: le conflit droit, le conflit gauche, le conflit linéaire et le conflit parallèle. Ces 4 types de conflits sont schématisés dans la figure .

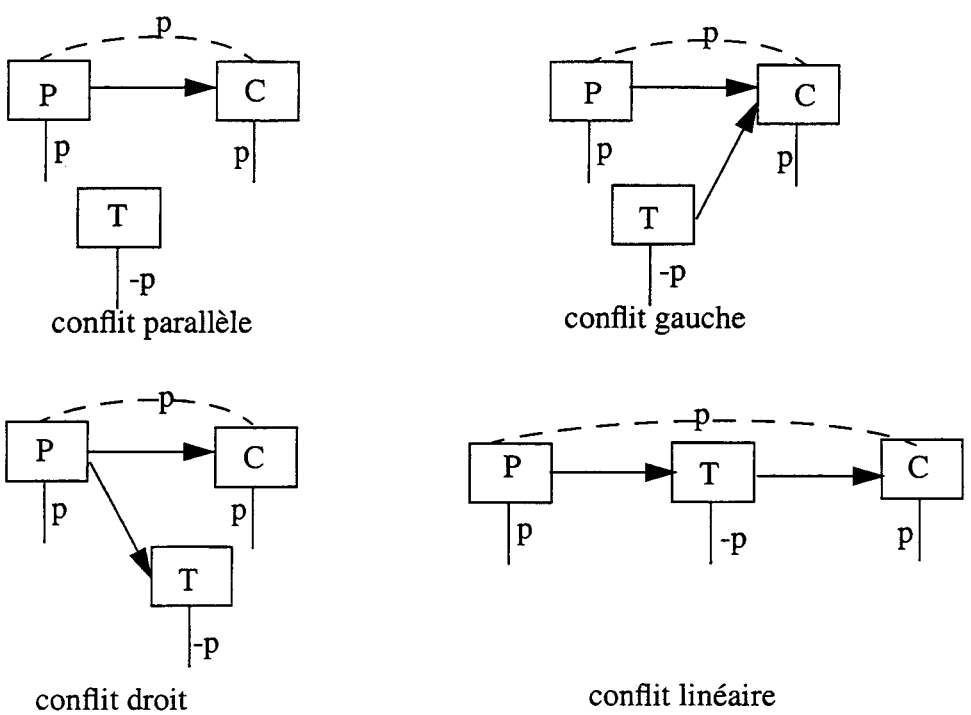


Figure 90: Les 4 types de conflits de [HER 89]

Afin de simplifier les réparations, et donc de diminuer le nombre de points de choix lorsque ces réparations sont multiples, nous avons identifiés tous les types de

conflits maintien-effet et maintien-maintien issus des deux définitions précédentes. Ces types de conflits suivent la même distinction que les conflits de type droit et de type gauche définis par J. Herzberg et A. Horz. Nous ne considérons pas le conflit linéaire, que nous interdisons dans TCLP: dès qu'une telle situation est identifiée, le plan partiel est abandonné.

7.2 Les différents types de conflits maintien-maintien et maintien-effet

Les différents types de conflits identifiés dans ce paragraphe sont construits en utilisant la définition générale du conflit maintien-effet et du conflit maintien-maintien, et en ajoutant les informations supplémentaires que l'on possède sur la position temporelle des opérateurs contenus dans le conflit.

7.2.1 Définitions

Les 7 premiers types de conflits concernent des maintiens entre deux opérateurs totalement ordonnés. Ils apparaissent par exemple lorsque le maintien est issu d'un lien causal. Nous appelons ***maintien ordonné*** un maintien entre deux opérateurs totalement ordonnés.

Dans les différents cas ci-dessous, un maintien est toujours tiré entre un opérateur de type P et un opérateur de type U. L'effet incompatible appartient à l'opérateur T. Nous ne décrivons pas cet effet dans les schémas ci-dessous, et nous n'étiquettons pas les maintiens par les faits maintenus. Nous supposons donc que lorsque le schéma fait intervenir deux maintiens, ces maintiens sont incompatibles.

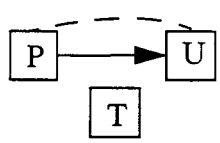
Nous décrivons chaque type de conflit en suivant la structure de représentation suivante:

- ***nom du type de conflit*** : information temporelle
  Réparation associée

(schéma associé)

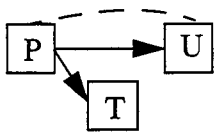
Dans les six points suivants, sont décrits les types de conflits concernant un maintien ordonné. Un maintien (p, x, y) est ordonné s'il existe une relation de précédence entre x et y.

- ***Conflit maintien-effet ordonné***: $P < U$
  Réparation $\Leftrightarrow T < P \vee U < T$

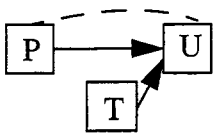


- ***Conflit maintien-effet ordonné droit***: $P < U \wedge P < T$

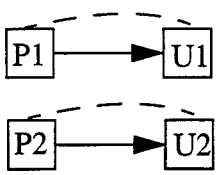
Réparation $\Leftrightarrow U < T$



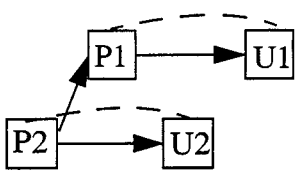
- ***Conflit maintien-effet ordonné gauche***: $P < U \wedge T < U$
  Réparation $\Leftrightarrow T < P$



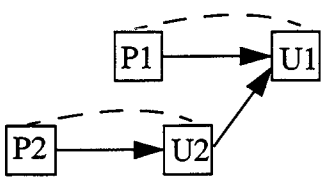
- ***Conflit maintien ordonné - maintien ordonné***: $P1 < U1 \wedge P2 < U2$
  Réparation: $\Leftrightarrow U2 < P1 \vee U1 < P2$



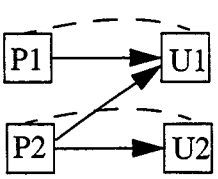
- ***Conflit maintien ordonné - maintien ordonné droit***:
  $P1 < U1 \wedge P2 < U2 \wedge P2 < P1$



ou
$P1 < U1 \wedge P2 < U2 \wedge U2 < U1$

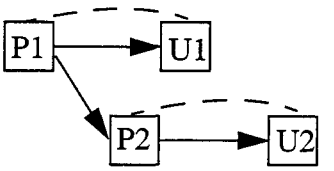


ou
$P1 < U1 \wedge P2 < U2 \wedge P2 < U1$

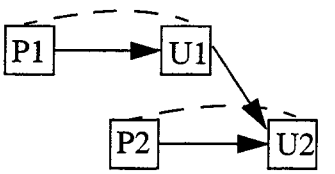


Pour les conflits de type maintien ordonné-maintien ordonné droit la réparation est $U2 < P1$

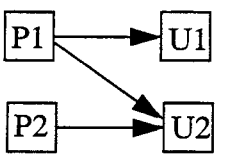
- ***Conflit maintien ordonné - maintien ordonné gauche***:
  $P1 < U1 \wedge P2 < U2 \wedge P1 < P2$



ou
$P1 < U1 \wedge P2 < U2 \wedge U1 < U2$



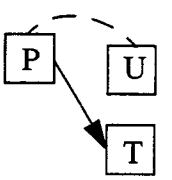
ou
$P1 < U1 \wedge P2 < U2 \wedge P1 < U2$



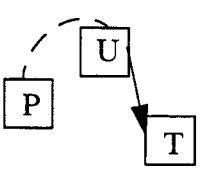
Pour les conflits de type maintien ordonné-maintien ordonné gauche la réparation est $U1 < P2$

Les types de conflits décrits ci-après concernent les conflits associés à des maintiens quelconques, c'est à dire non ordonnés.

- ***Conflit maintien-effet droit***:
  $P < T$

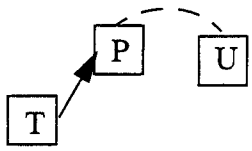


Réparation $\Leftrightarrow U < T$
ou
$U < T$

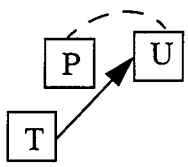


Réparation $\Leftrightarrow P < T$

- ***Conflit maintien-effet gauche***:
  T < P

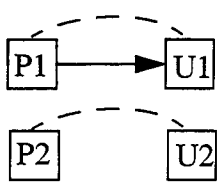


  Réparation ⇔ T < U
  ou
  T < U

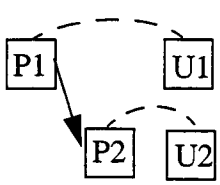


  Réparation ⇔ T < P

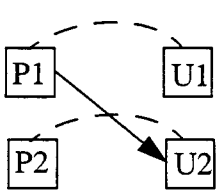
- ***Conflit maintien ordonné -maintien***: P1 < U1
  Réparation ⇔ (P2 < P1 ∧ U2 < P1) ∨ (U1 < P2 ∧ U1 < U2)



- ***Conflit maintien-maintien droit***:
  P1 < P2
  Réparation ⇔ U1 < P2 ∧ U1 < U2 ∧ P1 < U2

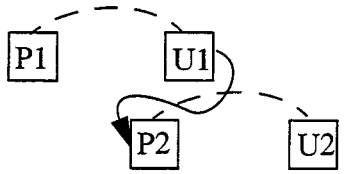


  ou
  P1<U2
  Réparation ⇔ P1 < P2 ∧ U1 < U2 ∧ U1 < P2

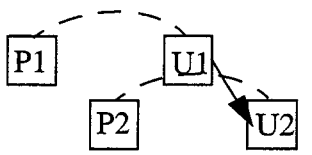


  ou
  U1 < P2

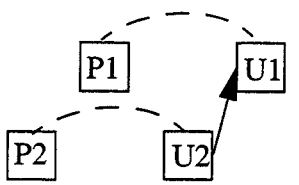
Réparation $\Leftrightarrow U1 < U2 \wedge P1 < P2 \wedge P1 < U2$



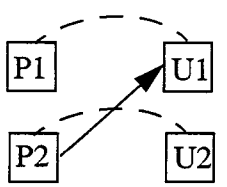
ou
$U1 < U2$
Réparation $\Leftrightarrow U1 < P2 \wedge P1 < P2 \wedge P1 < U2$



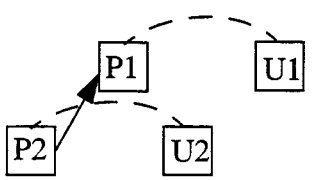
- ***Conflit maintien-maintien gauche***:
  $U2 < U1$
  Réparation $\Leftrightarrow U2 < P1 \wedge P2 < P1 \wedge P2 < U1$



ou
$P2<U1$
Réparation $\Leftrightarrow P2 < P1 \wedge U2 < P1 \wedge U2 < U1$

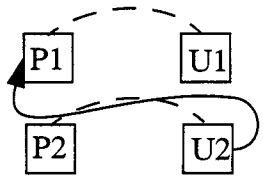


ou
$P2<P1$
Réparation $\Leftrightarrow P2 < U1 \wedge U2 < P1 \wedge U2 < U1$



ou
$U2<P1$

Réparation $\Leftrightarrow U2 < U1 \wedge P2 < P1 \wedge P2 < U1$



Chacun des conflits décrits précédemment est détecté en évaluant si dans le plan, les caractéristiques données dans la définition générale des conflits sont vérifiées. Lors du calcul des réparations possibles, la forme particulière du conflit est étudiée, et permet de réduire le nombre de réparations possibles.

Seuls les conflits maintien-effet, maintien-maintien, maintien ordonné-maintien ordonné et maintien ordonné-maintien (4 cas sur 13) requièrent des réparations disjonctives. Tous les autres types de conflits sont associés à une seule réparation possible. L'utilisation de ces différents types de conflits conduit à diminuer les points de choix vis à vis des réparations associées à un conflit. Cette approche diminue donc le taux de retour arrière.

Nous verrons par ailleurs (paragraphe 8.1.3) que cette technique simplifie l'évaluation des réparations globales possibles associées à un ensemble de conflits.

7.2.2 Exemple d'utilisation

Pour mettre en évidence l'utilisation des types de conflits, nous reprenons ici un exemple développé dans [POL 97] concernant le monde des tuiles (The Tileworld Problem).

Le problème consiste à boucher des trous avec des tuiles. Un trou est à un endroit, la tuile à un second endroit et le robot à un troisième endroit. Trois actions sont possibles: aller de X à Y pour le robot (go(X,Y)), prendre une tuile (pickup (T)) et boucher un trou(fill (H)) (figure 91).

Nous employons les minuscules pour décrire des variables instanciées, des majuscules pour décrire des variables non instanciées. Ces dernières peuvent cependant être liées à d'autres variables (non instanciées elles-mêmes).

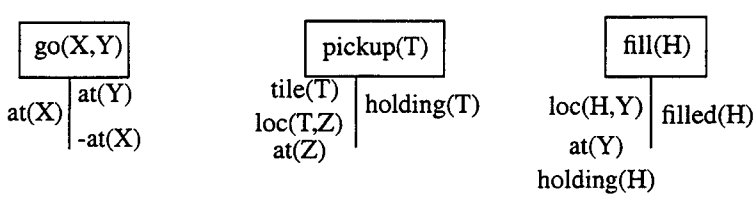


Figure 91: Les 3 actions du problème des tuiles

Le problème initial est décrit dans la figure 92:

Il y a un trou h1 en x1: loc(h1,x1).
t1 est une tuile: tile(t1).
Il y a une tuile t1 en x2:loc(t1,x2).
Le robot est en x3: at(x3).
Il faut remplir le trou h1: filled(h1).

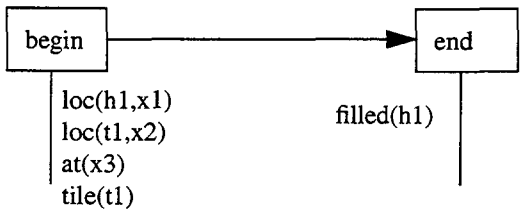


Figure 92: Description du problème

Soit le plan partiel de la figure 93.

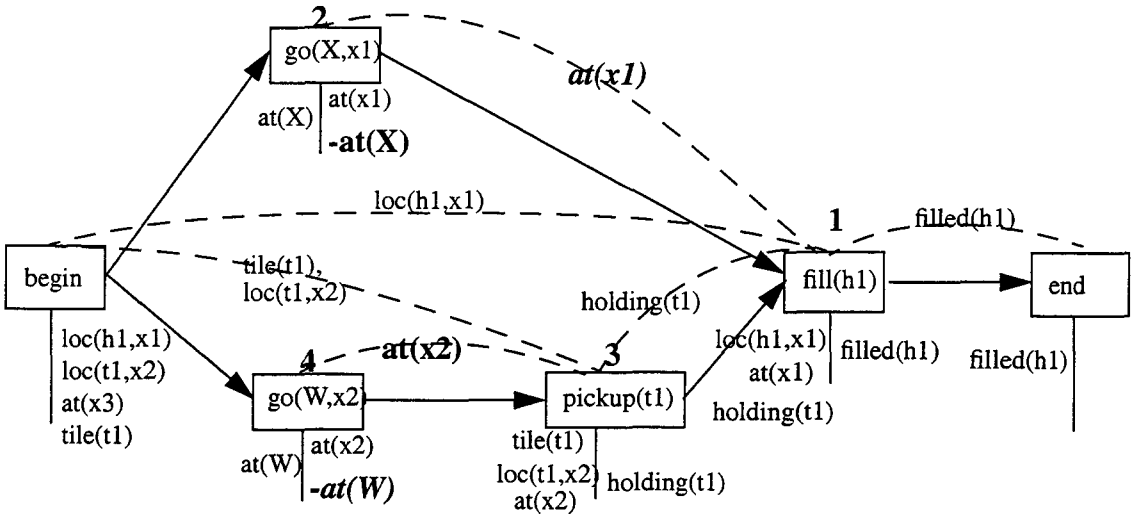


Figure 93: Plan partiel avec deux conflits maintien-effet:
maintien(at(x1),2,1)- effet(-at(W),4) et
maintien(at(x2),4,3)- effet(-at(X),2)

Selon M.E. Pollack et al. dans le plan partiel de la figure 93, suite à l'introduction de l'action 4 pour établir la condition at(x2) de l'action 2, sont apparus deux conflits:

- C1: conflit maintien-effet entre le maintien(at(x2), 4, 3) et l'effet -at(X) de l'action 2.
- C2: conflit maintien-effet entre le maintien(at(x1), 2, 1) et l'effet -at(W) de l'action 4

- **Utilisation des définitions générales des conflits.**

Les réparations associées aux conflits sont alors:

- pour C1: $(2 < 4 \wedge 2 < 3) \vee (4 < 2 \wedge 3 < 2) \Leftrightarrow r1(C1) \vee r2(C1)$
- pour C2: $(4 < 2 \wedge 4 < 1) \vee (2 < 4 \wedge 1 < 4) \Leftrightarrow r1(C2) \vee r2(C2)$.

Supposons que le planificateur choisisse de réparer C1 puis C2. Pour réparer C1, le planificateur prend les réparations dans l'ordre de description. Nous décrivons l'arbre des choix correspondants dans la figure 94.

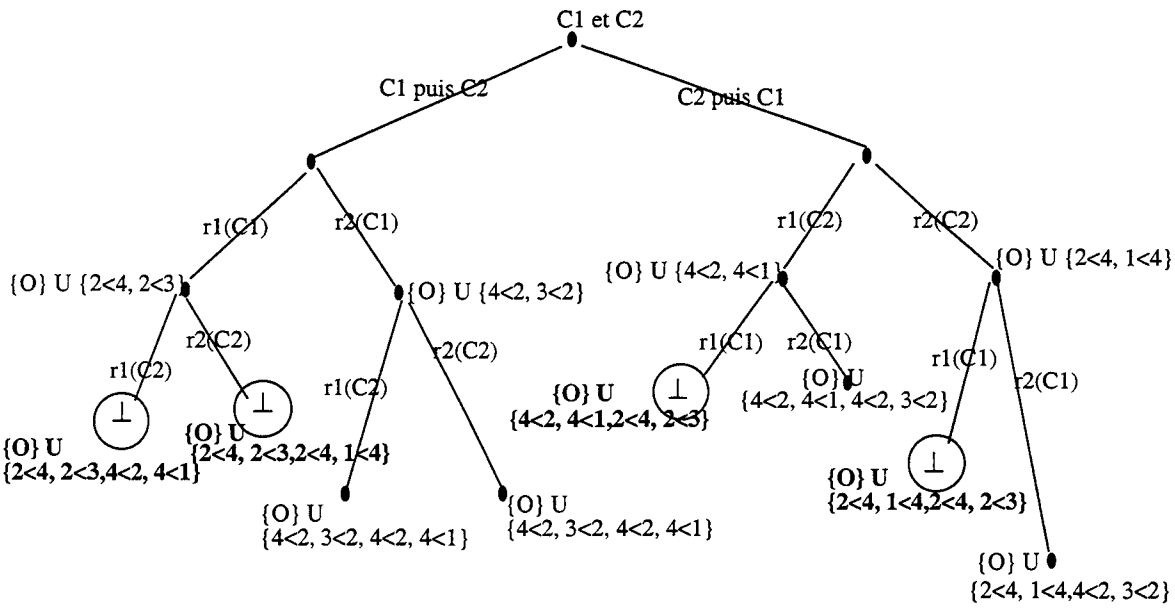


Figure 94: Ensemble des choix pour réparer C1 et C2

La réparation des conflits C1 et C2 conduit à 8 solutions potentielles, dont 4 sont incohérentes (repérées en gras dans la figure 94). Le nombre de choix est important vis à vis du nombre de conflits à réparer. De plus, chaque conflit possédant deux réparations possibles, il est impossible de guider le choix des réparations en fonction de leur nombre: on pourrait imaginer une stratégie cherchant à réparer les conflits dans l'ordre croissant du nombre de leurs réparations. Cette stratégie correspond à prendre les décisions les plus contraignantes avant les autres.

- **Utilisation des types de conflits.**

D'après la définition des conflits que nous avons donnée précédemment, C1 est un conflit maintien-effet ordonné (d'après le plan on sait que 4<3) et C2 est un conflit maintien- effet ordonné gauche (2<1 et 4<1). Dans ce cas, les réparations deviennent:

- pour C1: $2 < 4 \vee 3 < 2 \Leftrightarrow r1(C1) \vee r2(C1)$
- pour C2: $4 < 2 \Leftrightarrow r1(C2)$.

L'arbre de décision se réduit alors à celui de la figure 95.

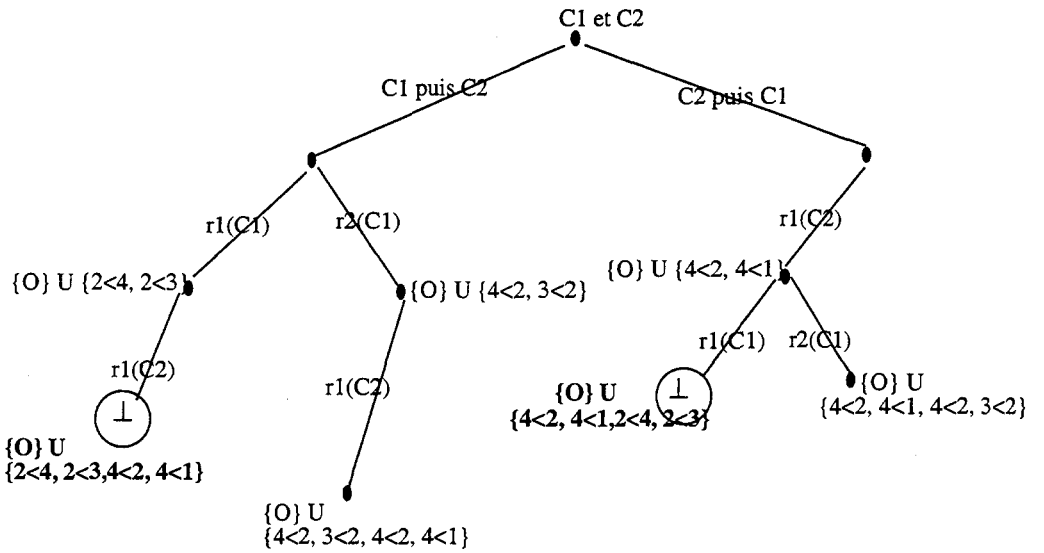


Figure 95: Arbre de décision avec utilisation des différents types de défauts

Lorsque les types de conflits sont utilisés, le nombre de solutions potentielles est réduit à 4. Par ailleurs, si le conflit C1 possède encore deux réparations possibles, le conflit C2 ne peut lui être réparé que par une seule réparation. Si cette réparation n'est pas cohérente avec le plan, alors quelque soit l'ordre dans lequel sont appliquées les diverses réparations, le plan est une impasse. Il est donc conseillé d'essayer de réparer C2 avant C1, afin de savoir immédiatement si C2 est réparable. Diminuer le nombre de réparations nous offre donc un moyen de définir des heuristiques pour guider les choix.

7.3 Conclusion

Nous avons présenté dans ce chapitre divers types de conflits issus de la structure générale des conflits maintien-effet et maintien-maintien nous permettant de réduire le nombre de réparations individuelles possibles. Ainsi, dans l'exemple du paragraphe 7.2.2, l'utilisation de la structure générale des conflits conduit à 8 solutions potentielles pour réparer deux conflits, alors que l'utilisation des types de conflits conduit seulement à 4 réparations possibles. Dans le chapitre suivant, nous présentons les stratégies de résolution utilisées et les comparons à celles que nous avons rencontrées dans la littérature.

Chapitre 8

Gestion des conflits dans TCLP - Gestion des relations temporelles

Ce chapitre est consacré à la résolution des conflits et aux relations entre les diverses stratégies utilisées et la gestion des relations temporelles.

Avant de présenter nos choix vis à vis de la réparation des conflits, nous résumons les stratégies proposées dans divers systèmes de planification.

8.1 Les stratégies rencontrées dans la littérature

8.1.1 Réparation incrémentale immédiate

8.1.1.1 Principe

Cette stratégie consiste à réparer un conflit dès son apparition dans un plan partiel.

Dans l'exemple du paragraphe 7.2.2, repris dans la figure 96, les conflits sont traités dès qu'ils apparaissent. Il en est de même dans UCPOP ([PEN 92]), ou SNLP ([MCA 91]).

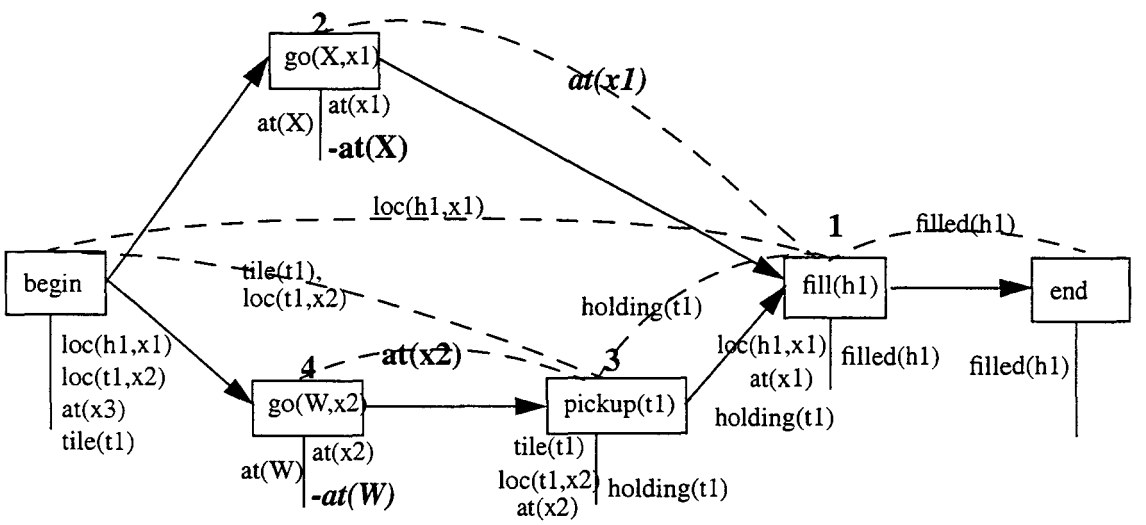


Figure 96: Plan partiel avec deux conflits maintien-effet: maintien(at(x1),2,1)- effet(-at(W),4) etmaintien(at(x2),4,3)- effet(-at(X),2)

8.1.1.2 Avantages et inconvénients de l'approche

Dans [WEL 94], D.S. Weld estime qu'il est préférable de réparer chaque conflit dès son apparition.

En effet, soit a est le nombre d'actions dans un plan, alors il peut y avoir $o(a^2)$ liens (ou maintiens) dans le plan. Et donc $o(a^3)$ conflits. Balayer de façon globale le plan pour savoir s'il y a des conflits se fait donc en $O(a^3)$. Or si la détection est faite de façon incrémentale, son coût peut être limitée. Ainsi, le coût associé au balayage de toutes les actions du plan chaque fois qu'un lien est créé est en o(a). De même, le coût associé au balayage de tous les liens du plan chaque fois qu'une nouvelle action est insérée est en $o(a^2)$.

Cette technique peut également être avantageuse par rapport à des approches où les conflits ne sont pas traités dès leur apparition. En effet, cette approche permet la construction d'une nouvelle étape sur la base d'un plan partiel correct, où toutes les informations nécessaires sont connues (car tous les choix ont été faits pour réparer tous les conflits). Dans le cas où les conflits ne sont pas réparés immédiatement, si un conflit n'a pas été réparé, rien n'interdit d'insérer la menace entre le producteur et le consommateur d'un lien par exemple.

L'étude de l'exemple précédent nous montre cependant que cette approche peut conduire à un nombre de points de choix important et donc à un taux de retour arrière excessif.

Un moyen de diminuer le taux de retour-arrière consiste à diminuer l'engagement lorsqu'un choix apparaît. Cette technique est apparue avec la planification non linéaire ([SAC 75]), où un moindre engagement vis à vis de l'ordre d'application des actions est respecté, et a été élargie dans TWEAK, où l'établisseur d'une condition peut être modifié (technique du White Knight).

Une première approche consiste à étudier la structure des conflits afin de dégager des stratégies de réparation en fonction du type des conflits détectés (paragraphe 8.1.2). Une seconde approche consiste à étudier globalement la structure d'un ensemble de conflits et de dégager des stratégies de réparation en fonction des relations qui existent entre les conflits (paragraphe 8.1.3).

8.1.2 Réparation incrémentale retardée

Comme nous l'avons vu dans le chapitre et le paragraphe précédents, choisir a priori des contraintes temporelles pour réparer des conflits peut conduire à un taux de retour-arrière important. Un premier moyen de réduire ce taux consiste à réduire le nombre de réparations possibles. Il est également possible de retarder la réparation des conflits en fonction de leur structure. Dans [PEO 93], M.A. Peot et D.E. Smith ont étudié diverses stratégies applicables à des planificateurs de type SNLP, dont nous nous sommes inspirés pour également retarder la réparation des conflits dans notre contexte de planification. Ces stratégies permettent de retarder les réparations de chaque conflit en fonction de critères prédéfinis, et proposent une réparation incrémentale des conflits une fois qu'il devient nécessaire de les réparer afin de poursuivre la construction du plan.

Avant de décrire diverses stratégies, nous rappelons la définition de la séparation telle qu'elle est utilisée dans [PEO 93]. M.A. Peot et D.E. Smith définissent des conflits séparables et des conflits non séparables. Des conflits séparables sont réparables en posant une contrainte visant à séparer les variables concernées par le conflit. Ainsi, les deux conflits identifiés dans le plan de la figure 96, sont des conflits séparables. Nous rappelons la définition de ces conflits:

- C1: conflit maintien-effet entre le maintien(at(x2), 4, 3) et l'effet -at(X) de l'action 2
- C2: conflit maintien-effet entre le maintien(at(x1), 2, 1) et l'effet -at(W) de l'action 4.

Dans chacun des conflits intervient une variable dont on ne connaît pas la valeur. Il est alors possible que X soit différent de x2, et que W soit différent de x1. Dans ce cas, les conflits n'ont plus lieu d'être caractérisés comme tels.

Lorsque les variables sont autorisées dans les plans (dans les effets, préconditions, maintiens), une troisième réparation de conflit est utilisée: la séparation des variables ([CHA 87]). Cette technique est formalisée dans [PEO 93]: elle consiste à poser des contraintes sur les variables de telle sorte que le conflit disparaisse. Par exemple, dans le cas des conflits précédents, la séparation consiste à imposer $X \neq x2$ et $W \neq x1$.

Cependant, cette technique peut conduire à des impasses plus fréquemment que si la séparation des variables est réalisée par le planificateur lors de la construction du plan.

Soit le plan partiel de la figure 97. Ce plan contient un conflit séparable entre le maintien ordonné (q(X1), 1, end) et l'effet -q(X2) de l'action 2. La séquence des réparationsqui ont été effectuées sont les suivantes:

- R1: établissement de q(X1) par l'action 1
- R2: établissement de p(X2) par l'action 2

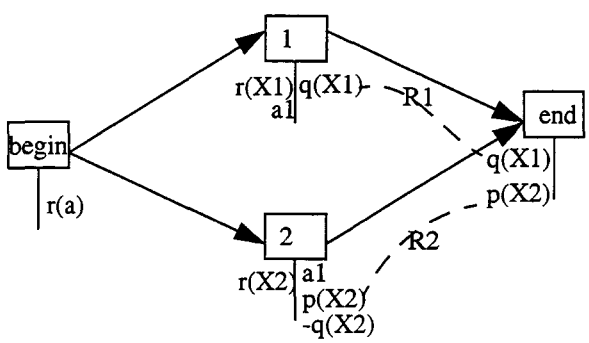


Figure 97: Exemple de plan partiel: 2 réparations R1 et R2 ont été effectuées

Le conflit entre le maintien (q(X1), 1, end) et l'effet -q(X2) de l'action 2 est détecté, il doit être réparé.

Supposons que la technique de séparation soit appliquée.

- R3: ajout de la contrainte $X1 \neq X2$.
- R4: établissement de a1 par l'action 2 => ajout de la contrainte 2 < 1 (figure 98)

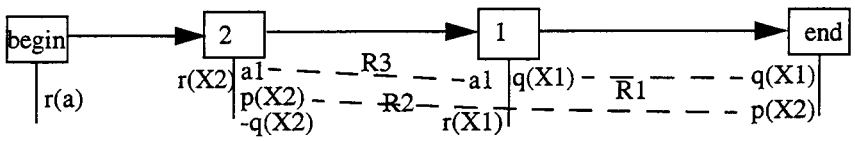


Figure 98: Exemple de plan partiel: R1, R2 et R3 ont été effectuées

- R5: établissement de r(X1) par l'action begin => ajout de la contrainte X1 = a. Donc X2 ne peut pas être égal à a.

- R6: tentative d'établissement de r(X2). Cet établissement est impossible car la seule solution consiste à établir r(X2) grâce à l'effet r(a) de l'action begin. Or X doit être différent de a. D'où le *retour sur les points de choix précédents.*

Ce retour arrière aurait pu être évité si les variables X1 et X2 n'avaient été contraintes à être différentes; dans ce cas, r(X2) aurait pu être établi grâce à l'effet r(a) de l'action begin. En effet, lors de la réparation R4, les actions 2 et 1 sont ordonnées. Le conflit identifié en R2 disparaît. La contrainte posée sur les variables X1 et X2 n'est donc plus nécessaire pour la suite du plan. La séparation conduit donc ici à une impasse non justifiée.

Les points suivants concernent les stratégies identifiées par M.A. Peot et D.E. Smith.

- **DSep: seuls les conflits non séparables sont réparés**

Cette stratégie consiste à examiner le type des conflits pour savoir s'ils sont séparables ou non, et à ne réparer que les conflits non séparables.

- **DUnf: seuls les conflits ne possédant qu'une seule réparation sont réparés**

Cette stratégie consiste à examiner les types de réparations et leur nombre pour savoir quels sont les conflits à réparer à la prochaine étape: seuls les conflits possédant une unique réparation possible sont réparés.

Il s'agit d'évaluer le nombre de réparation associées à un conflit dans le plan courant. Soit NR(C) le nombre de réparations associées au conflit C:

- Si NR(C) = 0 alors impasse: retour-arrière
- Si NR(C) = 1 alors appliquer réparation
- Si NR(C) $\geq$ 2 alors poursuivre la construction du plan sans réparer le conflit.

Nous appelons ***réparations individuelles*** les réparations associées à chaque conflit.

Chaque conflit est examiné individuellement: s'il vérifie le troisième critère, alors il n'est pas réparé. Cependant deux conflits peuvent avoir, de façon individuelle,

plusieurs réparations possibles, alors que la conjonction de réparations individuelles (pour réparer plusieurs conflits) est incohérente. Cette stratégie peut donc conduire à des impasses qu'il aurait été possible de découvrir par une stratégie de type réparation incrémentale immédiate.

Cette stratégie n'offre pas d'avantages par rapport à la précédente si elle n'est pas couplée à un système d'évaluation globale de l'ensemble des réparations afin de détecter les incohérences intrinsèques à l'ensemble des réparations ainsi que les incohérences dues à l'application successive de réparations individuelles sur le plan.

- **DRes: ignorer les conflits jusqu'à ce qu'il soit impossible de les réparer**

Cette stratégie est coûteuse et peu efficace. En effet, non seulement elle nécessite, à l'instar de DUnf, l'évaluation des réparations de chaque conflit pour connaître le nombre de réparations possibles, mais elle conduit forcément à un retour-arrière lorsqu'il devient impossible de réparer un conflit. Le plan partiel courant est alors simplement éliminé.

- **DEnd: retarder la réparation des conflits jusqu'à ce que toutes les préconditions du plan aient été établies**

Cette stratégie présente l'avantage de ne consacrer aucun temps à l'évaluation des conflits pendant la construction du plan.

Elle peut être très efficace dans les problèmes où les buts et les sous-buts sont indépendants, un peu moins lorsque les buts interagissent mais peuvent être trivialement mis en séquence, et est totalement inefficace lorsque les buts peuvent être difficilement mis en séquence, c'est à dire lorsque les diverses branches du plan partiel interagissent fortement. A. Barret et D.S. Weld ont défini une hiérarchie d'interactions entre sous-buts permettant de caractériser le problème en terme de sérialisation ([BAR 93]). Il pourrait être intéressant d'utiliser cette caractérisation afin de déterminer l'intérêt de l'utilisation de la stratégie DEnd en fonction d'un problème donné.

Les stratégies précédentes présentent deux inconvénients:

- elles ne permettent pas de savoir si un ensemble de conflits, dont les réparations individuelles sont cohérentes une à une avec le plan, est réparable globalement.
- elles effectuent un choix a priori lorsque plusieurs des réparations associées à un conflit sont cohérentes une à une par rapport au plan (sauf DUnf et DEnd).

Dans le paragraphe suivant, nous décrivons des stratégies visant à répondre à ces deux problèmes.

8.1.3 Stratégies de résolution globale

8.1.3.1 Principe général

Ces stratégies sont basées sur deux points:
- très souvent, dans un plan, plusieurs conflits apparaissent simultanément suite à l'application d'une réparation
- chaque conflit peut être réparé de plusieurs façons différentes.

Il apparaît alors, qu'en présence d'un grand nombre de conflits:
- l'ordre dans lequel ils sont réparés influence considérablement l'efficacité de la construction du plan
- le choix d'une réparation pour réparer un conflit est lié au choix des réparations des autres conflits.

Deux types d'approches tentent de remédier à ces problèmes:
- Q. Yang ([YAN 90], [YAN 91]) a étudié la structure globale d'un ensemble de conflits afin de dégager des relations entre les réparations individuelles, et de construire, si elle existe, une réparation globale permettant de réparer l'ensemble des conflits détectés. L'analyse les relations entre les réparations individuelles lui permet de construire des heuristiques sur le choix des réparations individuelles associées à chaque conflit et de construire une réparation globale en utilisant ces heuristiques.
- S. Kambhampati et X. Yang ([KAM 96a], [KAM 96b]) proposent de retarder le choix des réparations individuelles lorsque celles-ci sont disjonctives (par exemple dans le cas où promotion et demotion sont possibles). Il propose en outre une mise à jour incrémentale des réparations individuelles au fur et à mesure de l'ajout de nouvelles contraintes issues de la construction du plan.

Nous détaillons ces deux approches dans le paragraphe suivant.

8.1.3.2 Deux types de stratégies globales

- **Réparation globale**

Q.Yang compare la réparation d'un ensemble de conflits à la résolution d'un ensemble de contraintes (CSP) et utilise les algorithmes de propagation directement issus de ces théories.

Soit D un ensemble de conflits à réparer dans un plan P: D = {C1, ...,Ci, ... Cn}; O est l'ensemble des relations de précédence entre les opérateurs de P.

Chaque conflit est associé à une variable du CSP. Le domaine de définition Di de chaque variable correspond aux différentes réparations associées à chaque conflit (dans notre cas, promotion ou demotion): Di = (ri1, ri2). Nous utilisons ri pour parler indifféremment de ri1 ou de ri2. Trouver une réparation pour chaque conflit, qui soit cohérente avec les autres réparations et O, consiste à trouver une valeur pour chaque variable Ci de telle sorte que l'ensemble des valeurs vérifie l'ensemble des contraintes

posées.

Q. Yang définit des relations de subsomption et d'incohérence entre les réparations individuelles:

- S(ri, rj): ri subsume rj si et seulement si $\{ri\} \cup O \supset rj$
- I(ri, rj): ri est incohérent avec rj si et seulement si $\{ri, rj\} \cup O \supset false$

Ces relations lui permettent alors de simplifier son problème en définissant des règles de mise à jour des réparations associées aux conflits, et des règles lui permettant de supprimer des conflits du système de contraintes. Ces règles sont détaillées dans [YAN 90]. Elles traduisent l'idée que si toutes les réparations d'un conflit subsument une réparation d'un autre conflit, alors cette dernière peut être retirée des réparations associées au conflit. Ou encore, si le nombre de réparations associées à un conflit est réduit à 0 (par détection d'incohérence) alors le système de contraintes n'a pas de solution; ce qui revient à dire que l'ensemble des conflits ne peut être réparé.

Les règles sont appliquées jusqu'à stabilisation du problème de contraintes. Alors, si chaque variable peut prendre au moins une valeur, c'est que l'ensemble de conflits correspondant possède une réparation globale.

Il est alors possible de guider le choix des réparations individuelles restantes en fonction de leur nombre et/ou en fonction du nombre de conflits réparés lorsqu'une réparation individuelle est appliquée. Il est par exemple recommandé de réparer en premier les conflits qui ne peuvent être réparés que par une seule réparation avant de réparer ceux en possédant plusieurs. Et si plusieurs conflits possèdent le même nombre de réparations alors réparer d'abord ceux dont une réparation subsume le maximum d'autres réparations.

- **Réparations individuelles disjonctives**

Récemment S. Kambhampati a formalisé les techniques de réparations disjonctives que ce soit par rapport aux conflits, ou par rapport aux établisseurs d'une condition ([KAM 92], [KAM 96a], [KAM 96b]). Nous nous intéressons ici uniquement aux techniques développées dans le but de dégager des stratégies de résolution des conflits, afin de retarder les choix non justifiés.

Dans [KAM 95], S. Kambhampati et. al. considèrent la réparation de conflit (promotion, demotion, separation et confrontation ([WEL 94])) comme une étape quelconque dans la construction d'un plan. Le choix d'une réparation peut donc être guidé par des heuristiques, et l'application de la réparation peut être retardée à l'instar du choix de la prochaine précondition à établir ou du choix de l'établisseur d'une précondition. Cette remarque provient du fait que dans SNLP, la réparation des conflits est une étape figée appliquée immédiatement après chaque nouvel établissement, si celui-ci conduit à la création de nouveaux conflits.

Dans [KAM 96a], S. Kambhampati souligne que réparer un conflit en choisissant la promotion ou la demotion de la menace par rapport au but protégé va à l'encontre du principe de moindre engagement revendiqué par les planificateurs non linéaires. En effet, lorsque promotion et demotion sont toutes deux envisageables lors de la réparation d'un conflit, le planificateur effectue un choix a priori: deux plans partiels distincts sont

construits, et l'un des deux est choisi pour la suite de l'élaboration du plan. Il propose alors de réduire le facteur de branchement en définissant une réparation disjonctive qu'il applique lorsque tous les membres de la disjonction sont envisageables, et qu'il modifie incrémentalement pour éliminer de la disjonction les réparations incohérentes avec la nouvelle structure du plan.

La technique qu'il présente dans [KAM 96a] se limite aux contraintes d'ordre posées par la promotion et la demotion lors de la réparation d'un conflit. Ce cadre de travail correspond au nôtre puisque nous n'utilisons pas la séparation.

Reprenons l'exemple du paragraphe 7.2.2 (figure 99).

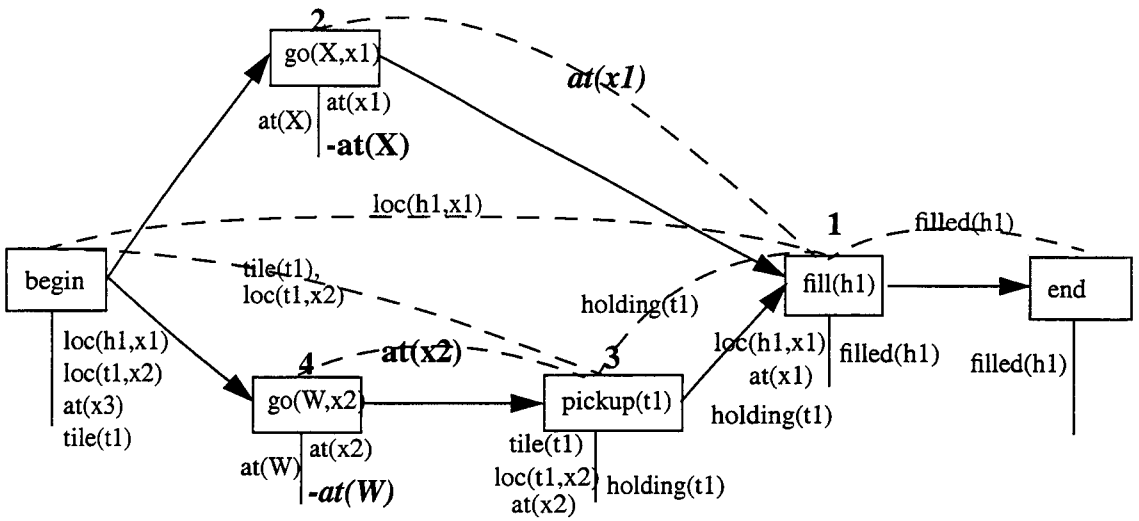


Figure 99: Plan partiel avec deux conflits maintien-effet:
maintien(at(x1),2,1)- effet(-at(W),4) et
maintien(at(x2),4,3)- effet(-at(X),2)

Les conflits identifiés dans ce plan et leurs réparations associées sont décrites dans le tableau 14. Les réparations sont construites en suivant les hypothèses de S. Kambhampati: il n'y a pas de simplification des réparations en fonction des contraintes temporelles présentes dans le plan.

Conflit	Réparation
C1 entre le maintien(at(x2), 4, 3) et l'effet -at(X) de l'action 2	$2 < 4 \vee 3 < 2$
C2 entre le maintien(at(x1), 2, 1) et l'effet -at(W) de l'action 4	$4 < 2 \vee 1 < 4$

Tableau 14 : Conflits du plan de la figure 99 et réparations associées

Si le conflit C1 est réparé en imposant 2<4 alors C2 n'est pas réparable quel que soit la réparation utilisée. Or ce choix a été réalisé parce que le planificateur ne sait pas gérer la disjonction introduite par la réparation. Cependant, les deux réparations sont cohérentes avec le plan.

S. Kambhampati propose donc de poser la contrainte temporelle disjonctive plutôt

que de construire deux plans distincts où chacune des contraintes appartenant à la disjonction est posée.

Soit O l'ensemble des relations temporelles du plan:
O = {begin< 1, begin < 2, begin < 3, begin < 4, 1 < end, 2<end, 3<end, 4 < end, begin < end, 4 < 3, 4 < 1, 3< 1 }

Etape 1: réparation de C1
O = {begin< 1, begin < 2, begin < 3, begin < 4, 1 < end, 2<end, 3<end, 4 < end, begin < end, 4 < 3, 4 < 1, 3< 1 } $\boldsymbol{U \{2 < 4 \vee 3 < 2\}}$

Etape 2: réparation de C2
O = {begin< 1, begin < 2, begin < 3, begin < 4, 1 < end, 2<end, 3<end, 4 < end, begin < end, 4 < 3, 4 < 1, 3< 1 } U {2 < 4 ∨ 3 < 2 } $\boldsymbol{U \{4 < 2 \vee 1 < 4\}}$.

Cet ensemble de contraintes temporelles est-il cohérent? Répondre à cette question nécessite la propagation des information.

L'inconvénient de cette approche est donc le coût de la gestion de relations temporelles disjonctives par rapport à la gestion de contraintes temporelles simples. Le gain obtenu au niveau de la taille de l'espace de recherche est perdu lors de la gestion de ces disjonctions. Un moyen d'annuler ce coût est de ne pas vérifier la consistance de l'ensemble des relations disjonctives. Cela revient à ne pas détecter les conflits tant que le plan n'est pas terminé. Cette stratégie est comparable à la stratégie DEnd décrite par Peot et Smith dans [PEO 93]: ceux-ci ont montré qu'elle n'est pas efficace.

S. Kambhampati propose alors un algorithme de propagation incrémental utilisé chaque fois qu'une nouvelle relation temporelle est ajoutée. Cet algorithme consiste à simplifier une contrainte temporelle disjonctive en une contrainte temporelle simple (représentant alors l'unique façon de réparer un conflit) par détection d'incohérence et à simplifier l'ensemble des contraintes temporelles par détection de redondance (l'application d'une contrainte temporelle pour réparer un conflit entraîne la réparation d'autres conflits).

Cette approche lui permet d'avoir une vision globale du problème de gestion des conflits tout en travaillant de façon incrémentale au niveau des réparations individuelles. Elle nécessite peu de modification quant à l'extension d'un planificateur classique pour tenir compte de réparations disjonctives. L'algorithme correspondant est donné en détails dans [KAM 96a]. Une extension a été réalisée sur UCPOP. Cependant, S. Kambhampati remarque que l'utilisation des techniques de CSP améliorerait sensiblement l'efficacité de la propagation.

Dans le paragraphe suivant, nous présentons la résolution des conflits dans TCLP.

8.2 Les stratégies utilisées dans TCLP

Nous traduisons la plupart des relations temporelles entre d-actions à l'aide de la structure des conflits maintien-maintien ou maintien-effet. Le nombre de conflits détectés dans les plans augmente donc de façon sensible. Il s'agit alors d'améliorer la stratégie de résolution des conflits afin de faire face à cette augmentation. Nous

présentons dans ce paragraphe, les choix que nous avons effectués en matière de résolution de conflits.

Nous distinguons ***conflits sémantiques*** et ***conflits temporels*** afin de distinguer également des stratégies propres à la résolution de chacun des types de conflits. Certaines stratégies sont cependant appliquées aux deux types de conflits.

Nous parlons de ***conflit sémantique*** lorsque le conflit détecté traduit une incohérence vis à vis du domaine. Cette incohérence est décrite par l'intermédiaire des incompatibilités sémantiques lors de la description du problème. Par exemple, la non-ubiquité interdit des plans où une personne pourrait se trouver à deux endroits différents au même moment. Cette incohérence peut être traduite par l'incompatibilité: $en(Personne, X) \wedge en(Personne, Y) \wedge X \neq Y \rightarrow \perp$; en(Personne, X) signifie que la personne Personne est en X.

Nous parlons de ***conflit temporel*** lorsque le conflit détecté traduit une relation temporelle d'Allen. Par exemple, la relation A < > B avec $A=(a^-,a^+)$ et $B=(b^-, b^+)$ deux d-actions, est traduite par le conflit maintien ordonné-maintien ordonné entre le $maintien(temp(X), a^-,a^+)$ et le $maintien(-temp(X), b^-, b^+)$. Ce type de conflit est toujours réel.

8.2.1 Les conflits sémantiques: conflit réel et conflit potentiel

Nous avons implémenté la technique de séparation des conflits dans une version antérieure de TCLP, et avons testé le planificateur sur divers exemples. Ces tests nous ont poussés à retirer cette réparation des techniques de résolution des conflits car la séparation des variables contraint artificiellement les plans et conduit à des échecs plus fréquemment que dans les cas où la séparation n'est pas utilisée.

Néanmoins, nous distinguons conflit potentiel et conflit réel comme sont distingués les conflits séparables des conflits non séparables.

Un conflit potentiel est susceptible de disparaître suite à une instanciation particulière des variables; il n'est donc pas ajouté à la liste des conflits à réparer, mais est conservé dans la liste des conflits potentiels et est vérifié à chaque fois qu'une nouvelle instanciation est réalisée:

- si le conflit devient réel, il est ajouté à la liste des conflits à traiter
- si l'instanciation fait disparaître le conflit alors il est retiré de la liste des conflits
- si l'instanciation n'a pas d'influence sur le conflit potentiel alors il est conservé dans la liste des conflits potentiels

La distinction entre conflit potentiel et conflit réel nous autorise à retarder la réparation des conflits jusqu'au moment où ils existent réellement dans le plan, mais n'est pas utilisée comme un moyen de contraindre les variables du plan par la technique de séparation.

Nous résumons ici les critères que nous utilisons pour définir conflits potentiels et conflits réels dans un plan (figure 100).

Soit le conflit entre les faits p(X) et q(Y). Ce conflit peut être de type maintien-

effet ou maintien-maintien. X et Y sont des vecteurs de variables.

Les faits p(X) et q(Y) sont incompatibles: incompatibles(p(X), q(Y)).

X_σ et Y_σ sont les instanciations de X et Y obtenues par la substitution σ. Cette substitution peut être partielle: certaines variables du vecteur de variables peuvent ne pas être instanciées.

Si les actions concernées ne vérifient pas les positions temporelles définies dans le conflit
alors le conflit potentiel disparaît.

Si $p(X_\sigma)$et $p(Y_\sigma)$ sont nécessairement compatibles
alors le conflit potentiel disparaît.

Sinon
Si $p(X_\sigma)$et $p(Y_\sigma)$ sont nécessairement incompatibles **alors** le conflit devient réel et est ajouté à la liste des conflits réels à traiter.
Si $p(X_\sigma)$et $p(Y_\sigma)$ sont possiblement incompatibles **alors** le conflit reste potentiel.

Figure 100: Critères de définition des conflits réels et potentiels

Appliquons cette stratégie à l'exemple de la figure 101.

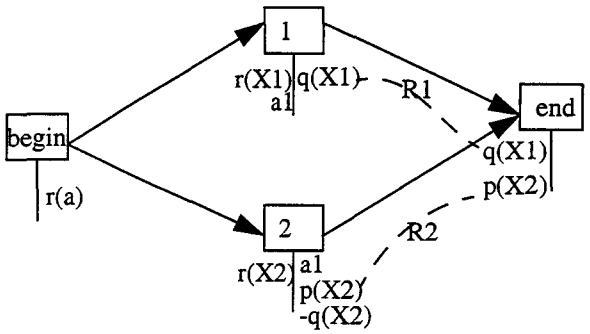


Figure 101: Exemple de plan partiel

Quel que soit le fait F, F est incompatible avec son contraire, noté -F: incompatibles (F, -F).

- S est l'ensemble des actions (instantanées) du plan.
- O est l'ensemble des relations de précédence entre les actions.
- L est l'ensemble des contraintes entre variables (contraintes d'unification (X=Y) ou de différenciation ($X \neq Y$)).
- A est l'agenda des défauts à traiter; usp(p,a) signifie que la précondition p appartenant à l'action a n'est pas établie.
- Confpot est l'agenda des conflits potentiels à évaluer; $Confpot = \varnothing$.

La construction du plan est la suivante:

- Les réparations R1 et R2 ont été appliquées.
  S = {begin, 1, 2, end}.
  O = { begin < 1, begin < 2, 1 < end, 2 < end, begin < end }.
  L= { q(X) = q(X1), p(Y) = p(X2) }
  A = { usp(a1), 1), usp(r(X1), 1), usp(r(X2), 2) }
  Le conflit maintien-effet entre le maintien (q(X1), 1, end) et l'effet (-q(X2)) de l'action 2 est détecté. Nous le notons C1 = {maintien (q(X1), 1, end), effet (-

q(X2)) }
Confpot = {C1}.
Le conflit est potentiel et de type maintien-effet ordonné gauche car:

- 1 < end (maintien ordonné)
- 2 < end (gauche)
  Donc, d'après la structure du conflit, la seule réparation possible est 2 < 1; elle est cohérente car O U {2 < 1} -> true
- X1 et X2 ne sont pas liés: L U { X1 = X2} -> true et
  L U { X1 ≠ X2 } -> true.

Le planificateur ne répare donc pas ce conflit. Il le conserve dans la liste des conflits potentiels.

- Application de R3: réparation de usp(a1) => ajout de 2 < 1
  O = { begin < 1, begin < 2, 2 < 1, 1 < end, 2 < end, begin < end}
  Evaluation de C1 dans Conf_pot: O U { 1 < 2} -> false donc le conflit potentiel a disparu. A = {usp(r(X1), 1), usp(r(X2), 2) } et Confpot = ∅.

L'utilisation du concept de conflit potentiel nous permet donc de ne pas réparer les conflits susceptibles de disparaître.

Les stratégies que nous présentons dans le paragraphe suivant ne s'appliquent qu'aux conflits réels.

8.2.2 Les conflits sémantiques réels et les conflits temporels

8.2.2.1 Utilisation de l'approche de Q. yang

Soit un plan partiel et A l'agenda des conflits réels.
Soit d_0 le conflit de A traité à l'étape i.

point de choix 1: d_0 peut être réparé par 2 réparations différentes r_{01} et r_{02}.

Le choix est porté sur r_{01}. L'application de la réparation r_{01} crée de nouveaux défauts d_1, ..d_j,...d_p s'ajoutant aux défauts de A. Nous supposons ici que tous les défauts sont des conflits. Chaque conflit d_j possède deux réparations r_{j1} et r_{j2}.

Nous considérons que la stratégie utilisée par le planificateur impose la réparation des conflits avant tout autre type de défaut et que les défauts sont traités dans l'ordre FIFO de détection.

point de choix 2: choix et application de la réparation r_{11} du conflit d_1.
La réparation d'un conflit ne peut pas entraîner la création de nouveau défaut.

Le cycle continue: réparation du conflit suivant, choix de la réparation et application.

Soit le point de choix j: l'application de toutes les réparations r_{jk} proposées donnent lieu à une impasse. Il est donc nécessaire de revenir à l'étape précédente. Il est alors possible d'imaginer une situation où toutes les autres réparations des conflits d_j à d_2

conduisent à des impasses. Dans ce cas, le planificateur aura effectué les choix 2^{j-2} choix possibles de d_2 à d_j sans succès, pour finalement revenir en arrière sur le premier choix effectué.

La figure 102 représente l'arbre de décision correspondant à l'ensemble des choix que nous venons de décrire.

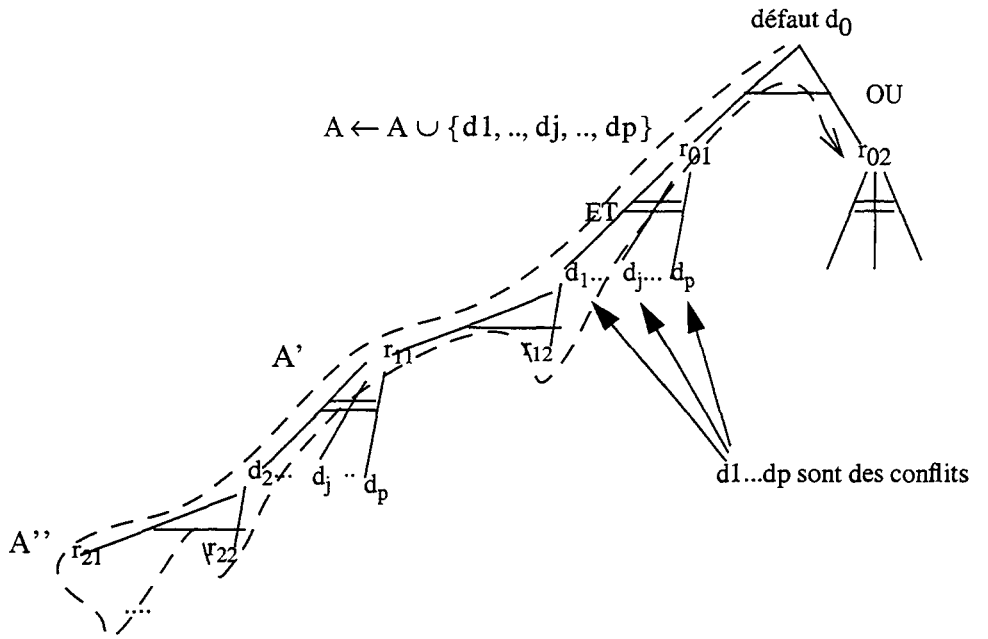


Figure 102: Arbre de décision lié à la réparation incrémentale de conflits

Afin d'éviter de telles situations, Q.Yang propose un moyen d'évaluer les relations entre les réparations associées à des conflits. Ce qui lui permet par exemple de détecter plus tôt de telles incohérences.

Nous allons nous inspirer de cette approche pour traiter globalement un ensemble de conflits. Nous appelons ***réparation globale***, un ensemble de réparations individuelles correctement choisies pour réparer un ensemble de conflits.

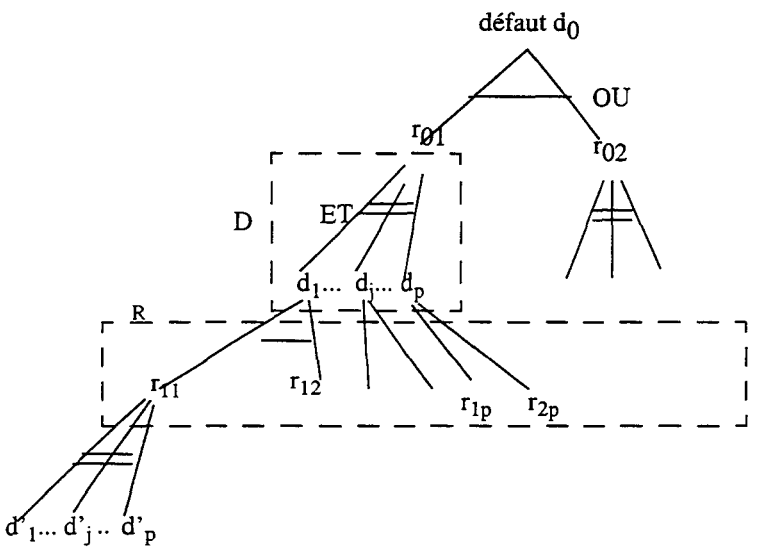


Figure 103: Regroupement des conflits et réparations individuelles

Plutôt que de considérer chaque conflit individuellement, nous considérons le défaut D constitué de la conjonction des p conflits d_1, ..., d_j, ..., d_p (figure 103):
$D = (d_1, ..., d_j, ..., d_p)$.

Nous déterminons alors un ensemble R de réparations globales Rk pour D:

$R = \bigvee R_k$ pour k=1 à 2^p .

Chaque R_k est construite de la façon suivante:

Soit r_j la disjonction de réparations individuelles associées au conflit dj: $r_j \Leftrightarrow r_{j1} \vee r_{j2}$. Nous notons $r_j = \{r_{j1}, r_{j2}\}$

Soit $E = r_1 \times ... \times r_j \times ... \times r_p$ le produit cartésien des r_j. Un élément de E correspond à une conjonction de réparations individuelles en choisissant pour chaque conflit d_j une réparation parmi celles appartenant à r_j. Nous notons R_k une telle conjonction. Au maximum, il y a 2^p conjonctions de réparations individuelles possibles.

La figure 104 représente les R_k associées à l'ensemble des conflits de D.

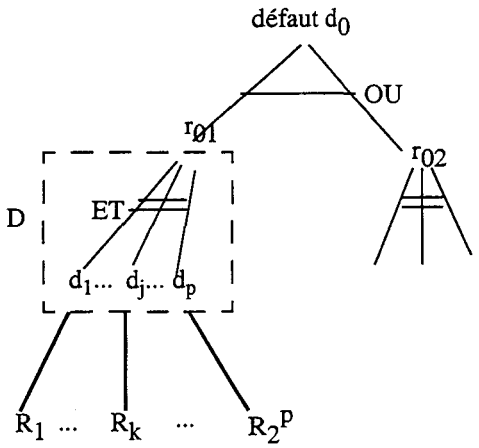


Figure 104: Construction des réparations globales associées à l'ensemble des conflits de D

Dans ce cas, le seul point de backtrack après application de R_1, R_k ou $R_2{}^p$ est r_{01}, alors que dans le premier cas, il est nécessaire de parcourir tous les points de choix liés aux réparations individuelles r_{ij}.

Cette approche nous permet donc de diminuer les points de backtrack lorsqu'il s'agit de choisir une conjonction de réparations pour réparer un ensemble de conflits présents simultanément dans un plan.

Cependant le calcul des réparations globales peut s'avérer aussi coûteux que le backtrack dû à l'approche incrémentale. Il semble donc nécessaire d'être capable de calculer un minimum de réparations globales.

Par ailleurs, nous cherchons à conserver les disjonctions temporelles aussi longtemps que possible, tant que rien n'oblige le planificateur à faire un choix. Or, lorsque plusieurs réparations globales sont possibles, si le planificateur suit la stratégie précédemment décrite, alors il effectue un choix, ce qui revient à choisir un ensemble de relations temporelles positionnant de façon certaines les actions du plan. Il en est de même lorsqu'un relation temporelle est traduite par un conflit: classiquement le

planificateur choisit une des réparations possibles afin de réparer le conflit, ce qui revient à ne pas conserver la disjonction temporelle.

Par exemple, la disjonction temporelle A < > di o B est traduite par la conjonction de schémas temporels suivante: $p2(a^-, b^-, b^+) \wedge p1(a^-, a^+) \wedge p1(b^-, b^+)$. Le schéma plan correspondant est donné dans la figure 105.

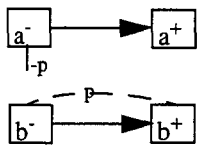


Figure 105: A < > di o B

Le conflit C1 exprimé par $p2(a^-, b^-, b^+)$ peut être réparé de deux façons différentes:

- $r_{11} = b^+ < a^-$ ou
- $r_{12} = a^- < b^-$

Les deux réparations sont cohérentes avec le plan; rien n'oblige le planificateur à faire un choix.

Soit la disjonction temporelle B < > C, qui est traduite par la conjonction de schémas temporels suivante: $p3(c^-, c^+, b^-, b^+) \wedge p1(c^-, c^+) \wedge p1(b^-, b^+)$; (figure 106).

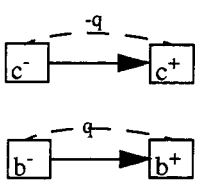


Figure 106: B < > C

Le conflit C2 exprimé par $p3(c^-, c^+, b^-, b^+)$ peut être réparé de deux façons différentes:

- $r_{21} = b^+ < c^-$ ou
- $r_{22} = c^+ < b^-$

Les deux réparations sont cohérentes avec le plan; rien n'oblige le planificateur à faire un choix.

Le plan contenant les actions A, B et C respectant les deux contraintes temporelles citées ci-dessus est représenté dans la figure 107.

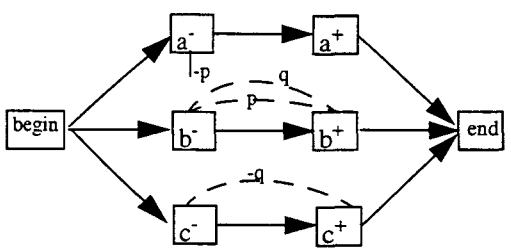


Figure 107: A < > di o B et B < > C

Le plan de la figure 107 contient deux conflits possédant chacun deux réparations individuelles possibles. Il est alors possible de construire 4 réparations globales résumées

dans le tableau 15.

Construction des réparations globales	conjonction de précédence entre instants	relations temporelles entre d-actions	relations déductibles entre A et C
$R_1 \Leftrightarrow r_{11} \wedge r_{21}$	$b^+ < a^- \wedge b^+ < c^-$	$A > B \wedge B < C$	A ? C
$R_2 \Leftrightarrow r_{11} \wedge r_{22}$	$b^+ < a^- \wedge c^+ < b^-$	$A > B \wedge B > C$	A > C
$R_3 \Leftrightarrow r_{12} \wedge r_{21}$	$a^- < b^- \wedge b^+ < c^-$	$A < di\ o\ B \wedge B < C$	$A < o\ di\ C$
$R_4 \Leftrightarrow r_{12} \wedge r_{22}$	$a^- < b^- \wedge c^+ < b^-$	$A < di\ o\ B \wedge B > C$	A ? C

Tableau 15 : Construction des 4 réparations globales possibles

Les 4 réparations globales sont possibles et rien n'oblige le planificateur à faire un choix. Or la stratégie présentée précédemment conduit forcément à l'application d'une réparation globale.

Nous nous sommes alors inspirés de la stratégie DUnf décrite dans le paragraphe 8.1.2 pour conserver les réparations globales disjonctives le plus longtemps possible.

8.2.2.2 Retarder le choix d'une réparation globale parmi plusieurs possibles

La stratégie DUnf stipule que si un conflit peut être réparé de plusieurs façons différentes, alors mieux vaut ne pas le réparer plutôt que de choisir a priori une des réparations. Nous allons l'appliquer à un conflit global constitué d'une conjonction de conflits. Nous la complétons par un test de consistance de réparation globale qui nous permet d'écarter les réparations inconsistantes, et nous limitons l'évaluation des réparations à deux réparations globales possibles. ce qui nous permet de ne pas calculer les 2^p réparations globales possibles associées aux p conflits.

Cette stratégie est appliquée de façon incrémentale: elle consiste à construire une réparation globale en ajoutant une à une les réparations individuelles des conflits. Si l'ajout d'une nouvelle réparation individuelle conduit à l'inconsistance de la réparation globale courante avec les contraintes du plan, alors cette réparation globale est abandonnée. Dès que deux réparations globales sont consistantes alors le calcul est arrêté: l'existence de deux réparations globales possibles suffit pour qu'il y ait choix.

Soit ***O*** l'ensemble des relations de précédence du plan.

Soit ***Aattente*** l'ensemble des conflits réels à réparer: Aattente = { C1,...Ci, ..., Cp}.

Chaque conflit peut être réparé par deux réparations au plus (promotion, demotion). Nous notons r_i l'ensemble des réparations individuelles associées au conflit Ci: $r_i \Leftrightarrow r_{i1} \vee r_{i2}$.

Certains conflits ne possèdent qu'une seule réparation possible (la réparation dépend du type de conflit). Tous les conflits de ce type sont retirés de Aattente et sont mis dans ***A_ind***.

A_ind = { Ci / $r_i = r_{ij}$, j = 1 ou 2 } = {C1, ... Cno }

A_mult est l'ensemble des conflits possédant plusieurs réparations.

A_mult = { Ci / $r_i = r_{i1} \vee r_{i2}$ } = { Cno+1, ... Cp }

Rind est la réparation globale de l'ensemble des conflits de A_ind: Rind $\Leftrightarrow \bigwedge r_i$ pour i= 1 à no . Nous confondons Rind avec l'ensemble $\{r_1, ..., r_{no}\}$.

Chaque réparation globale contient au moins Rind, et pour chaque conflit Cno+1 à Cp, une réparation individuelle est choisie.

L'évaluation d'une réparation globale se compose des étapes suivantes:

- Si Rind et O sont incohérents alors il n'y a pas de réparation globale possible.
- Sinon
  - appliquer Rind: O <- $\{O\} \cup \{r_1, ..., r_{no}\}$
  - evaluer_solution_globale ($\varnothing$, $\varnothing$, p, {rno+1, ..., rp}): retourne vrai et retourne dans RES les réparations globales cohérentes s'il existe au moins une réparation globale cohérente (au maximum RES contient 2 réparations globales cohérentes). NG est le nombre de réparations globales cohérentes
    - si NG = 0 alors il n'y a pas de réparation globale cohérente
    - si NG =1 alors il existe une seule réparation globale cohérente dans RES: appliquer cette réparation et poursuivre la construction du plan
    - si NG $\geq$ 2 (ici NG =2) alors ne pas choisir de réparation globale, poursuivre la construction du plan

La procédure d'évaluation des solutions globales correspond à l'algorithme 9.

rj est la réparation individuelle courante choisie. Au départ elle est égale à $\varnothing$.
$r_{(j+1)1}$ est la première réparation individuelle suivante à r_j.
$r_{(j+1)2}$ est la seconde réparation individuelle suivante à r_j.
L est la réparation globale courante. Au départ elle est égale à $\varnothing$.
p est le nombre de couple de réparations individuelles (= nombre de conflits)

Algorithme 9: évaluer-solution-globale (r_j, L, p, LR)

```
Début

  Si j = p alors          % la fin de la liste des réparations individuelles est atteinte

     Si {L} U{O} U {rj} -> true alors
        NG <- NG +1
        RES <- RES +{ {L} U {rj} }      % calcul du résultat total
        retourner (vrai, NG, RES)
     Sinon retourner faux
     Fsi

  Sinon

     Si {L} U {O} U{rj} -> true alors      % la réparation constituée de r(no+1) à rj est cohérente

        Si evaluer-solution-globale(r(j+1)1, {L} U {rj}, p, LR) alors

           Si NG =1 alors          % 1 solution existe déjà => y-en-a-t-il une 2e?
              evaluer-solution-globale( r(j+1)2, {L} U {rj}, p, LR)
```

retourner vrai
Sinon retourner evaluer-solution-globale($r_{(j+1)2}$, {L} U {r_j}, p, LR)
Fsi

Fsi

Sinon retourner faux % la réparation constituée de r_{no+1} à r_j n'est pas cohérente
Fsi

Fsi
Retourner RES

Fin

Nous utilisons l'algorithme 9 dans le paragraphe suivant lors de la construction et l'application d'une réparation globale (algorithme 13).

8.2.3 Algorithme général de détection des conflits et construction des réparations

Nous décrivons dans ce paragraphe les procédures de détection des conflits et de construction des réparations globales.

Dans TCLP sont gérés trois agendas:
- ***A-attente*** un agenda temporaire où sont stockés les conflits réels détectés
- ***Areel*** l'agenda qui contient les défauts à traiter (et donc les conflits réels) avec leurs réparations.
- ***Confpot*** l'agenda des conflits potentiels.
- ***I*** l'ensemble des incompatibilités sémantiques.

Nous avons défini deux types généraux de conflits qui nous servent à détecter les conflits dans les plans. Deux procédures de détection des conflits sont donc utilisées: ***détection-conflit-maintien-effet*** et ***détection-conflit-maintien-maintien***.

Les divers types de conflit que nous avons définis à partir des deux structures générales précédentes nous servent à construire des réparations individuelles les plus simples possibles.

La procédure ***incompatible-nec(F1, F2)*** retourne vrai s'il existe une relation d'incompatibilité entre P1 et P2 (incompatibles(P1, P2) $\in$ I) et si P1 et P2 sont nécessairement incompatibles; elle retourne faux sinon.

La procédure ***incompatible-pot(F1, F2)*** retourne vrai s'il existe une relation d'incompatibilité entre P1 et P2 (incompatibles(P1, P2) $\in$ I) et si P1 et P2 sont possiblement incompatibles; elle retourne faux sinon.

S est l'ensemble des actions instantanées du plan.

O est l'ensemble des relations de précédence entre les actions contenues dans S.

M est l'ensemble des maintiens du plan: un maintien est décrit par ***maintien(FAIT, s1, s2)*** avec s1 et s2 appartenant à S.

C est l'ensemble des contraintes sur les variables du plan.

Un plan ***P*** est défini par ***(S, O, M, C)***

Une action possède des effets. Si e est un effet de l'action s alors on note: ***effet(e, s)***.

Les procédures de détection des conflits maintien-effet et maintien-maintien (algorithmes 10 et 11) rangent dans A-attente les conflits réels et dans Confpot les conflits potentiels. Elles sont appelées en fonction des modifications réalisées sur le plan.

Algorithme 10: détection-conflit-maintien-effet(maintien(F, s1, s2), effet(E, s))

Début

Si incompatible-nec (F, E) **alors**
  Si O U {s1<s, s<s2) **ou** O U {s2<s, s<s1} **alors**
    C <- conflit-maintien-effet ((F,s1,s2), (E,s))
    A-attente <- A-attente + {C}
  Fsi
Sinon
  Si incompatible-pot(F,E) **alors**
    Si O U {s1<s, s<s2) **ou** O U {s2<s, s<s1} **alors**
      C <- conflit-maintien-effet ((F,s1,s2), (E,s))
      Confpot <- Confpot + {C}
    Fsi
  Fsi
Fsi
  Retourner A-attente
  Retourner Confpot
Fin

Algorithme 11: détection-conflit-maintien-maintien(maintien(F1, s1, s2), maintien(F2, s3, s4))

Début

Si ({O} ∪ {s3 < s1, s1 < s4} **ou** {O} ∪ {s4 < s1, s1 < s3} **ou** {O} ∪ {s3 < s2, s2 < s4} **ou** {O} ∪ {s4 < s2, s2 < s3} **ou** {O} ∪ {s1 < s3, s3 < s2} **ou** {O} ∪ {s2 < s3, s3 < s1} **ou** {O} ∪ {s1 < s4, s4 < s2} **ou** {O} ∪ {s2 < s4, s4 < s1}) **alors**

  Si incompatible-nec (F1, F2) **alors**
    C <- conflit-maintien-maintien ((F1,s1,s2), (F2, s3, s4))
    A-attente <- A-attente + {C}
  Sinon
    Si incompatible-pot(F1,F2) **alors**
      C <- conflit-maintien-maintien ((F1,s1,s2), (F2, s3, s4))
      Confpot <- Confpot + {C}
    Fsi
  Fsi
Fsi
  Retourner A-attente
  Retourner Confpot
Fin

L' à chaque conflit sa réparation en fonction du type de conflit. Si le conflit n'est pas interdit alors ***construction-réparation-individuelle*** (algorithme 12) associe à chaque conflit ses réparations individuelles et construit le conflit global D qu'il ajoute à Areel (l'agenda des défauts réels). D = ({Ci} + {ri}). Cette procédure est appelée après la

détection des conflits réels (algorithmes 10 et 11) dans la boucle principale de l'algorithme de planification (algorithmes 17 et 18)

Algorithme 12: construction-réparation-individuelle(A-attente)
Début

Pourchaque C de A-attente
Si C = conflit-maintien-effet((F, s1, s2), (E,s)) alors

Si (O U {s<s1} ⊃ false **et** O U {s2<s} ⊃ false) **ou**
(O U {s<s2} ⊃ false **et** O U {s1<s} ⊃ false) **alors**
Fail (conflit irréparable)
Sinon
Cas O U {s2 < s1} ⊃ false :
Cas O U {s<s1} ⊃ false : ri <- s2 < s
O U { s2<s} ⊃ false : ri <- s < s1
defaut : ri <- (s2 < s ∨ s<s1)
Fincas
O U {s1 < s2} ⊃ false :
Cas O U { s<s2} ⊃ false : ri <- s1 < s
O U { s1<s} ⊃ false : ri <- s < s2
defaut : ri <- (s1 < s ∨ s<s2)
Fincas
defaut :
Cas O U { s<s1} ⊃ false : ri <- s2 < s
O U { s<s2 } ⊃ false : ri <- s1 < s
O U { s1<s} ⊃ false : ri <- s < s2
O U { s2<s} ⊃ false : ri <- s < s1
Fincas
Fincas
R <- R + {ri}
D <- D U {C}
Areel <- Areel U {(D, R)}
Fsi

Sinon
Si C = conflit-maintien-effet((F1, s1, s2), (F2, s3, s4)) **alors**

Si (O U {s3<s1} ⊃ false **et** O U {s2<s3} ⊃ false) **ou**
(O U {s4<s1} ⊃ false **et** O U {s2<s4} ⊃ false) **ou**
(O U {s1<s3} ⊃ false **et** O U {s4<s1} ⊃ false) **ou**
(O U {s2<s3} ⊃ false **et** O U {s4<s2} ⊃ false) **ou**
(O U {s3<s2} ⊃ false **et** O U {s1<s3} ⊃ false) **ou**
(O U {s4<s2} ⊃ false **et** O U {s1<s4} ⊃ false) **ou**
(O U {s1<s4} ⊃ false **et** O U {s3<s1} ⊃ false) **ou**
(O U {s2<s4} ⊃ false **et** O U {s3<s2} ⊃ false) **alors**

Fail (conflit irréparable)
Sinon
Cas O U {s2<s1} ⊃ false **et** O U {s4<s3} ⊃ false :
Cas (O U {s1<s3} ⊃ false **ou** O U {s2<s4} ⊃ false
ou O U {s2<s3} ⊃ false) : ri <- {s2 < s3}

(O U {s3<s1} ⊃ false **ou** O U {s4<s2} ⊃ false **ou**
O U {s4<s1} ⊃ false) : ri <- {s4 < s1}
Fincas

O U {s2<s1} ⊃ false **et** O U {s3<s4} ⊃ false :
cas (O U {s1<s3} ⊃ false **ou** O U {s4<s1} ⊃ false
ou O U {s2<s3} ⊃ false) : ri <- {s3 < s1}

(O U {s3<s1} ⊃ false **ou** O U {s3<s2} ⊃ false **ou**
O U {s4<s1} ⊃ false) : ri <- {s2<s4}
Fincas
O U {s1<s2} ⊃ false **et** O U {s4<s3} ⊃ false :
Cas (O U {s2<s3} ⊃ false **ou** O U {s1<s3} ⊃ false
ou O U {s1<s4} ⊃ false) : ri <- {s4 < s2}

(O U {s3<s2} ⊃ false **ou** O U {s3<s1} ⊃ false **ou**
O U {s4<s1} ⊃ false) : ri <- {s1<s3}
Fincas
O U {s1<s2} ⊃ false **et** O U {s3<s4} ⊃ false :
Cas (O U {s2<s4} ⊃ false **ou** O U {s1<s4} ⊃ false
ou O U {s1<s3} ⊃ false) : ri <- {s3 < s2}

(O U {s4<s2} ⊃ false **ou** O U {s3<s2} ⊃ false **ou**
O U {s3<s1} ⊃ false) : ri <- {s1<s4}
Fincas
O U {s2<s1} ⊃ false : ri <- { (s3<s1, s4<s1) ∨ (s2<s3, s2<s4) }
O U {s1<s2} ⊃ false : ri <- { (s3<s2, s4<s2) ∨ (s1<s3, s1<s4) }
O U {s4<s3} ⊃ false : ri <- { (s1<s4, s2<s4) ∨ (s3<s1, s3<s2) }
O U {s3<s4} ⊃ false : ri <- { (s1<s3, s2<s3) ∨ (s4<s1, s4<s2) }
O U {s3<s1} ⊃ false : ri <- {s2<s3, s2<s4, s1<s4}
O U {s4<s1} ⊃ false : ri <- {s1<s3, s2<s4, s2<s3}
O U {s3<s2} ⊃ false : ri <- {s2<s4, s1<s3, s1<s4}
O U {s4<s2} ⊃ false : ri <- {s2<s3, s1<s2, s1<s4}
O U {s2<s4} ⊃ false : ri <- {s4<s1, s3<s1, s3<s2}
O U {s2<s3} ⊃ false : ri <- {s3<s1, s4<s1, s4<s2}
O U {s1<s3} ⊃ false : ri <- {s3<s2, s4<s1, s4<s2}
O U {s1<s4} ⊃ false : ri <- {s4<s2, s3<s1, s3<s2}

Fincas
R <- R + {ri}
D <- D U {C}
Areel <- Areel U {(D, R)}
Fsi
Fsi
Finpour
Aattente <-∅
Retourner Areel
Fin

L'algorithme 13 traduit la stratégie décrite dans le paragraphe 8.2.2.2.

Séparer-individuelle-multiple sépare la liste des réparations individuelles simples (Lind) de la liste des réparations individuelles disjonctives (Lmult).

Taille(L) donne le nombre d'éléments dans L.

Algorithme 13: Appliquer-réparation-globale ((D, R))

```
% D est un conflit global, R contient l'ensemble des réparations associées
Début

   (Lind, Lmult) <- Séparer-individuelle-multiple (R)

   Si O U { Lind } ⊃ false alors
         Fail
   Sinon O <- O U {Lind}
         p <- Taille {Lmult}
         (V, NG, RES) <- evaluer-solution-globale(∅ , ∅ , p, Lmult)
         Si V alors
               Si NG = 1 alors     O <- O U {RES}
                                   Areel <- Areel / {(D, R)}
                                   Retourner P = (S, O, M, C)
                                   Retourner A = (Areel, Confpot)
               Fsi
         Sinon Fail
         Fsi
   Fsi
Fin
```

Les algorithmes que nous venons de décrire sont utilisés de façon spécifique après certaines modifications de plan.

Par exemple, lors de l'établissement d'une précondition par un effet d'une action située dans le plan avant la précondition, il y a ajout d'un maintien entre l'effet et la précondition. Il est alors nécessaire de détecter tous les conflits possibles entre ce maintien et les maintiens du plan et tous les conflits entre ce maintien et les effets du plan.

Lorsque cet établissement nécessite l'insertion d'une nouvelle action, il faut en plus détecter tous les conflits entre les effets de cette action et les maintiens du plan, tous les conflits entre les maintiens de cette action et les effets du plan, et tous les conflits entre les maintiens de cette action et les maintiens du plan.

Notons ici que le fait de considérer la durée des actions conduit à l'augmentation du nombre de maintiens, car naturellement certaines préconditions de l'action instantanée équivalente (STRIPS) sont traduites en conditions d'application qui doivent être vérifiées non seulement pour débuter l'action mais aussi pendant tout son déroulement. De même certains effets de l'opérateur instantané peuvent correspondre à des effets produits pendant le déroulement de l'action et maintenus jusque sa fin. Dans ces deux cas, la persistance de la précondition ou de l'effet est traduite par un maintien qui peut donc être en conflit avec d'autres effets ou maintiens du plan.

La stratégie utilisée dans l'algorithme 13 peut être comparée à la stratégie Dmin proposée par D.E. Smith et M.A. Peot dans [SMI 94]. Cette stratégie correspond à une combinaison des stratégies DSep et DUnf. Elle se déroule en trois étapes, décrites ci-dessous:

1 -Réparer tous les conflits ne possédant qu'une seule réparation.

2 -Séparer tous les conflits possédant plusieurs réparations en l'ensemble des conflits séparables et en l'ensemble des conflits non séparables. Ce dernier ensemble correspond à A-mult dans notre stratégie. Les conflits de cet ensemble sont réparables par promotion ou demotion.

3 -Rechercher une résolution de l'ensemble des conflits non séparables. Si cette résolution existe alors ne pas l'appliquer et conserver les contraintes disjonctives.

Il semble que le dernier point sous-entende qu'il existe plusieurs résolutions possibles de l'ensemble des conflits non séparables. Sinon, pourquoi ne pas appliquer la résolution globale trouvée, si celle-ci est unique. Notre stratégie réalise cette distinction, et conserve l'ensemble des conflits non résolus seulement s'il existe plusieurs façons de les réparer.

La stratégie que nous développons dans l'algorithme 13 conduit à une diminution du taux de retour-arrière vis à vis de la résolution des conflits. Cependant l'étape de la stratégie concernant l'évaluation d'une solution globale est coûteuse (NP-Complet) puisqu'elle correspond à la recherche d'un scénario consistant dans un ensemble de relations temporelles disjonctives.

Cette contrepartie semble néanmoins atténuée par le fait que le nombre de réparations disjonctives est très souvent faible devant le nombre de conflits à réparer. En effet, les réparations disjonctives associées à des conflits sont simplifiées par le jeu de contraintes temporelles issues du plan. C'est pourquoi nous avons défini différents types de conflits conduisant à la simplification des réparations associées dès leur construction (algorithme 12).

Nous pouvons nous demander ici si, étant donné A-mult, l'ensemble des conflits réparables par promotion et démotion, la connaissance de l'existence d'une réparation globale est immédiate.

Le problème se résume au suivant:

Soit un ensemble I de triplets et de quadruplets d'instants (i1, i2, i3) et (j1, j2, j3, j4) tels que: si $l \neq m$ alors $il \neq im$ et $jl \neq jm$ et

$((i1 < i2 \wedge i1 < i3) \vee (i2 < i1 \wedge i3 < i1)) \wedge$

$((j1 < j3 \wedge j1 < j4 \wedge j2 < j3 \wedge j2 < j4) \vee (j3 < j1 \wedge j4 < j1 \wedge j3 < j2 \wedge j4 < j2))$.

Alors, existe-t-il toujours deux ordres totaux différents issus de I vérifiant les contraintes posées?

Si oui, alors il n'est plus nécessaire d'évaluer les réparations globales possibles de l'ensemble des conflits de A-mult. Il suffit d'abstraire ce sous-ensemble de l'ensemble des conflits réels à réparer, de réparer, si c'est possible, les conflits possédant des réparations individuelles uniques (R-ind) et de conserver A-mult puisque cet ensemble de conflit peut être réparé de plusieurs façons différentes. Cette approche nécessite une seconde vérification des réparations de A-mult pour vérifier que certaines réparations disjonctives ne se sont pas simplifiées suite à l'application des réparations de R-ind.

8.3 Conclusion

Nous avons construit une table de transition entre un sous-ensemble de l'algèbre d'Allen (toutes les disjonctions de primitives d'Allen ne nécessitant pas l'égalité entre instants) et le formalisme temporel des planificateurs traditionnels non linéaires à liens causaux.

44 de ces relations sont traduites à l'aide de la structure des conflits maintien-effet

ou maintien-maintien. Notre approche augmente donc de façon sensible le nombre de conflits, et la gestion des relations temporelles est fortement liée à celle des conflits.

Or les 20 relations temporelles restantes, qui sont exprimées à l'aide de la seule relation de précédence, bénéficient du fait que si une relation représente une disjonction de primitives d'Allen, alors cette disjonction est naturellement préservée dans les plans tant qu'aucune contrainte supplémentaire n'impose de modification. Par exemple, la relation temporelle disjonctive A > oi B avec A et B 2 d-actions est représentée par la conjonction $pl(b^-, a^-) \wedge pl(b^+, a^+)$ (figure 108). Si aucune des deux contraintes $pl(b^+, a^-)$ ou $pl(a^-, b^+)$ n'est ajoutée au plan alors la disjonction A > oi B est conservée.

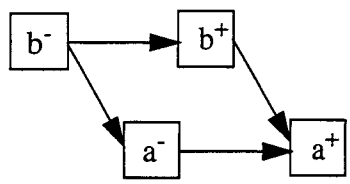


Figure 108: A > oi B

Cependant, lorsqu'une disjonction temporelle est décrite par un conflit, l'utilisation d'une stratégie classique de résolution des conflits (par exemple réparation incrémentale immédiate ou globale immédiate) conduit à éliminer les disjonctions le plus tôt possible. Cette solution va à l'encontre de la gestion de relations temporelles traditionnellement basée sur la propagation des relations par clôture transitive.

Afin d'améliorer l'efficacité de la gestion des conflits et afin de conserver les disjonctions temporelles, nous avons modifié la stratégie de résolution des conflits de diverses manières que nous rappelons ci-après:

- Définition de différents types de conflits

Les différents types de conflits que nous avons définis nous permettent de simplifier les réparations individuelles possibles. Ceci conduit à une simplification de la réparation globale associée à un ensemble de conflits et à diminuer les points de choix.

- Traitement global d'un ensemble de conflit

L'approche consistant à considérer un ensemble de conflits plutôt que considérer incrémentalement chaque conflit nous permet d'avoir une vision globale de l'ensemble des relations temporelles traduites par cet ensemble de conflits. Ce qui nous permet de déterminer des dépendances entre les relations et de détecter les incohérences plus rapidement que lors d'une gestion incrémentale.

- Moindre engagement dans le choix d'une réparation globale

Retarder le choix d'une réparation globale parmi plusieurs réparations globales possibles revient à ne pas opter pour un ensemble solution de relations temporelles lorsque plusieurs solutions existent. Cette stratégie nous permet donc de conserver les disjonctions temporelles.

Chapitre 9

Mise en oeuvre

Ce chapitre vise à expliquer la démarche que nous avons suivie pour, d'une part, identifier les problèmes à résoudre, et d'autre part, contraindre la structure du planificateur que nous avons construit et implémenté.

Dans le premier paragraphe nous décrivons en quoi l'application de laquelle sont issus ces travaux nécessite des techniques de planification, et comment ses caractéristiques ont influencé le choix du planificateur réactif IPEM ([AMB 87] , [AMB 88]).

Dans le second paragraphe, nous présentons les caractéristiques d'IPEM et les résultats des tests qui nous ont conduits à la conception de TCLP.

Le troisième paragraphe est consacré à l'algorithme général de planification de TCLP.

Nous terminons en expliquant comment il est possible de réintégrer les caractéristiques réactives d'IPEM dans TCLP, celui-ci étant basé sur celui-là.

9.1 Problématique de CARNEADE

9.1.1 Structure actuelle de CARNEADE

Nous rappelons que CARNEADE ([MAR 92]) est un système de simulation de combats dont les buts sont:

- ***l'entraînement*** des officiers face à de nouvelles situations et des élèves-officiers pour les familiariser avec le maximum de situations différentes,
- ***l'étude*** de nouveaux systèmes de force (évaluation quantitative et qualitative de nouvelles hypothèses, comparaison objective de solutions potentielles)
- ***l'aide à la décision opérationnelle*** (en temps de paix grâce à la simulation de plans de manoeuvre et en temps de crise, grâce à l'évaluation des hypothèses de manoeuvre en fonction de la situation militaire courante).

Ce système de simulation est constitué de deux parties majeures à l'origine de sa spécificité:

- une ***partie tactique*** utilisée pour la simulation du combat, c'est-à-dire la modélisation des mouvements des troupes, la modélisation précise des données terrain...

- une ***partie stratégique*** où sont simulées les prises de décision. Les décisions sont normalement prises à haut niveau dans la hiérarchie militaire: l'état major identifie une situation de crise devant laquelle il faut réagir, transmet au niveau hiérarchique inférieur une politique d'action et des buts généraux à atteindre. A ce niveau est élaboré un premier plan peu détaillé, qui traduit les ordres en actions et en buts plus précis. Ce plan constitue lui-même un ensemble d'ordre pour le niveau inférieur. Plus le plan est détaillé, c'est-à-dire plus les ordres sont précis, moins les décisions à prendre sont importantes. Au plus bas niveau de la hiérarchie, la réception des ordres se traduit en leur exécution.

Ce simulateur de combats est donc capable de reproduire non seulement les actions, mais aussi les décisions. Nous nous sommes intéressés, dans le cadre de ce travail, uniquement à la partie stratégique de la simulation de combats, et plus précisément à la prise de décision.

D'après ce que nous venons de décrire concernant la prise de décision, il semble cohérent de dire que la prise de décision résulte ici en une suite d'actions pour réaliser un ensemble d'objectifs. C'est pourquoi nous nous sommes intéressés au techniques de planification pour simuler la prise de décision. Actuellement, les plans sont construits à partir d'un système expert qui traduit les actions à appliquer en fonction de la situation identifiée.

9.1.2 Les besoins

Nous avons vu dans l'introduction de ce mémoire (paragraphe 1.2) en quoi l'utilisation de la planification est avantageuse par rapport à celle d'un système expert.

Se pose également le problème de la construction de plan dans un contexte réactif: dans le cas de la simulation de combats, les troupes amies coopèrent et évoluent face à des troupes ennemies. C'est pourquoi, il est nécessaire d'envisager les actions ennemies afin de construire des plans en fonction de leurs effets.

Actuellement, dans CARNEADE, cet aspect est traité par la construction de plans conditionnels dont les phases les plus éloignées de la situation courante sont peu détaillées. Les conditions posées correspondent à des situations futures prévues. Le niveau de détails des branches conditionnelles est fonction de la probabilité avec laquelle la situation envisagée peut se produire et de sa position temporelle par rapport à la situation courante. Dans le vocabulaire militaire, un plan d'action est assimilé à un ***mode d'action***. Celui de la figure 109 est constitué de trois ***phases***; les phases constituent les grandes étapes d'une mission à haut niveau; elles sont peut détaillées. Chaque phase est divisée en ***temps*** consécutifs. Pendant un temps d'un mode d'action sont exécutées des ***actions*** en parallèle.

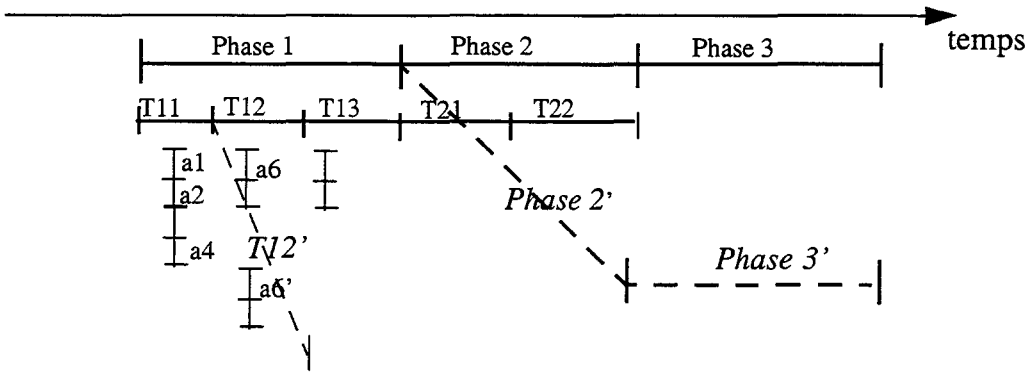


Figure 109: Structure d'un mode d'action

Dans le mode d'action précédent deux branches conditionnelles ont été développées: l'une, au niveau des phases, traduit le fait que si le but de la phase 1 n'a pas été atteint, alors opter pour la phase 2'. L'autre se situe au niveau du temps T12; elle est plus détaillée, car si on se place au début du mode d'action (début de la phase 1), il faut être prêt à réagir rapidement si le temps T11 n'aboutit pas comme prévu.

La stratégie de construction actuelle des plans peut se décrire de la façon suivante:

- Planification complète des grandes phases (étape 1 dans la figure 110)
- Impossibilité de tout connaître à l'avance =>
    - Décomposition des phases *les plus proches* en temps
    - Détail des temps *les plus proches* en actions (étape 2 dans la figure 110)
- Exécution des actions = ordres (étape 3 dans la figure 110)
- Adaptation = Elaboration plus détaillée de la suite du plan en fonction des retours d'exécution (retour 4-a dans la figure 110) et modification du plan en cours d'exécution si nécessaire (retour 4-b dans la figure 110).

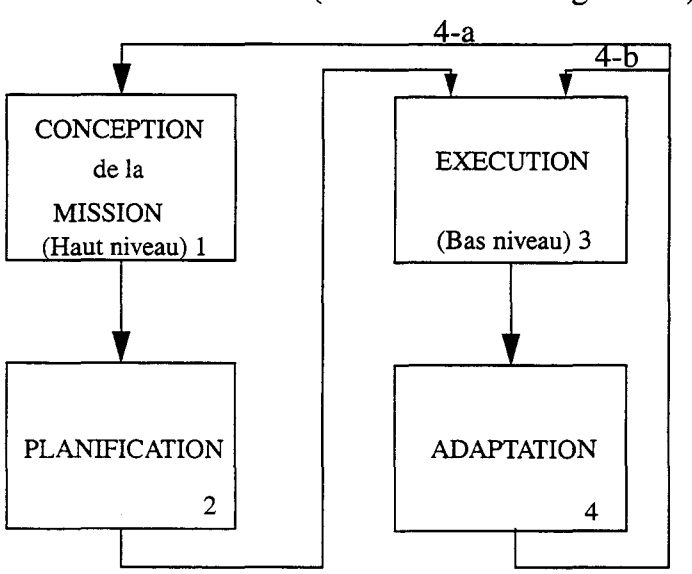


Figure 110: Les étapes d'élaboration et d'exécution d'un mode d'action

Les caractéristiques de l'application et la stratégie de construction des plans nous ont conduits à:

- utiliser la planification au lieu du système expert dans deux buts précis:
    - être capable de construire un plan en fonction d'une situation pour toute nouvelle situation et
    - ne pas avoir affaire à l'être humain dans la boucle de simulation ailleurs que lors de l'étape de modélisation du problème (modélisation de nouvelles actions possibles par exemple)
- utiliser un planificateur réactif capable:
    - de modifier son plan en fonction de changements du monde courant sans que ce changement ait été envisagé dans une branche conditionnelle
    - d'entrelacer les étapes de planification et d'exécution lorsque les grandes phases de la mission ont été élaborées

C'est pourquoi nous avons pensé que le planificateur IPEM serait un moyen de répondre à ces problèmes. Nous l'avons alors implémenté en Prolog et avons réalisé des tests avec l'aide de personnel militaire. Ces tests nous ont alors conduits à modifier certaines caractéristiques d'IPEM et à construire TCLP.

Nous décrivons rapidement les caractéristiques d'IPEM dans le paragraphe suivant.

9.2 Les tests avec IPEM

9.2.1 Caractéristiques d'IPEM

IPEM ([AMB 87] , [AMB 88]) est un planificateur capable d'entrelacer les étapes de planification et d'exécution de façon à exécuter le plus tôt possible des parties de plans générés, et à modifier le plus rapidement possible les plans en fonction des retours d'exécution (échec ou effet miraculeux). L'entrelacement des deux étapes est réalisé grâce à une structure de contrôle commune à la construction et à l'exécution des plans. En effet, J. A. Ambros-Ingerson a étendu le principe "défaut-réparation" traditionnellement utilisé pour la construction des plans, à leur exécution. L'algorithme de construction et d'exécution des plans est basé sur l'algorithme 14.

Ai est l'agenda des défauts contenus dans le plan partiel Pi. Nous supposons que toutes les caractéristiques du plan sont décrites dans Pi.

B est l'ensemble des buts à atteindre.

I est la situation initiale.

A0 contient les buts de B non établis.

Nous appelons ***étape*** de planification, l'ensemble des actions suivantes:

- choix d'un défaut dans l'agenda
- réparation du défaut => mise à jour du plan
- détection des nouveaux défauts => mise à jour de l'agenda

Algorithme 14: ipem(Ai, Pi)
Début

Tantque Ai $\neq \varnothing$ et B $\not\subset$ I **faire**

Choisir un défaut d dans Ai
Evaluer les réparations possibles rij de ce défaut dans Pi
Choisir une réparation rij
Appliquer rij: Pi+1 <- rij (Pi)
Evaluer les nouveaux défauts créés par l'application de rij (= { d' })
Ai+1 <- A - d + {d'}
ipem (Ai+1, Pi+1)

Fintantque
Retourner Pi
Fin

Nous décrivons les caractéristiques de ce planificateur en distinguant les défauts et réparations liés la construction des plans des défauts et réparations liés à l'exécution.

9.2.1.1 Description des actions et du problème initial

Les actions sont supposées instantanées. Elles sont représentées par des opérateurs de type STRIPS, étendus de façon à accepter des variables. De cette façon les actions utilisables peuvent être modélisées de manière générique et sont instanciées en fonction des liens avec les autres actions du plan.

La situation initiale est représentée par les effets d'une action fictive appelée begin et située avant toutes les actions du plan.

La situation finale est décrite par les préconditions d'une action fictive appelée end, et située après toutes les actions du plan.

Le plan de la figure 111 représente la structure d'un plan initial.

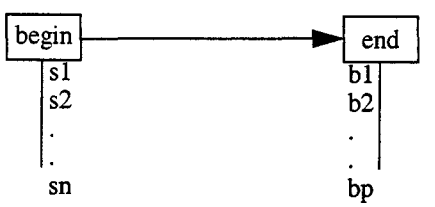


Figure 111: Description du plan initial dans IPEM

9.2.1.2 Elaboration d'un plan

IPEM est un planificateur traditionnel non linéaire à liens causaux.
Trois types de défauts sont utilisés pour construire les plans.

- **Les préconditions non supportées (usp)**

Elles représentent les buts et sous-buts non établis dans le plan courant. Ce défaut peut être réparé de 3 façons possibles.

- ***reduction-prior***: établissement de la précondition d'une action A par un effet d'une action B située avant A dans le plan, telle que l'effet choisi soit susceptible de co-désigner avec la précondition.
  Cette réparation consiste:

  - à ajouter un lien entre la précondition et l'effet, et
  - à unifier les variables contenues dans ces deux faits.
- ***reduction-parallel***: établissement de la précondition d'une action A par un effet d'une action B en parallèle avec A dans le plan (c'est-à-dire, A et B ne sont pas ordonnées), telle que l'effet choisi soit susceptible de co-désigner avec la précondition.
  Cette réparation consiste:
  - à ajouter une relation de précédence entre B et A,
  - à ajouter un lien entre la précondition et l'effet, et
  - à unifier les variables contenues dans ces deux faits.
- ***reduction-new***: établissement de la précondition d'une action A par un effet d'une nouvelle action B choisie dans la bibliothèque d'opérateurs, telle que l'effet choisi soit susceptible de co-désigner avec la précondition.
  Cette réparation consiste:
  - à ajouter la nouvelle action dans le plan,
  - à ajouter une relation de précédence entre B et A, et entre begin et B,
  - à ajouter un lien entre la précondition et l'effet,
  - à unifier les variables contenues dans ces deux faits et
  - à ajouter toutes les contraintes accompagnant B

- **Les conflits non réparés (urc)**

Un conflit est défini par la présence d'un lien entre un effet F d'une action P et une précondition F d'une action N, et une action T détruisant F telle que T est possiblement entre P et N (figure 112).

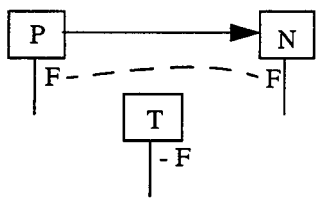


Figure 112: Représentation graphique d'un conflit dans IPEM

La réparation consiste à placer T<P ou N <T.

Le conflit de la figure 112 est appelé conflit parallèle. Si dans le plan, T est nécessairement entre P et N, c'est-à-dire que pour toutes les linéarisations du plan, P<T<N, alors le conflit est linéaire. Il ne peut pas être réparé, le plan est éliminé.

- **Les actions non expansées (uea)**

Dans IPEM, une action peut être ***primitive*** ou "***expansable***". Cette distinction correspond à la distinction entre action primitive et macro-action dans d'autres planificateurs. Une action primitive est directement exécutable, alors qu'une macro-action doit être expansée jusqu'à ce qu'elle soit décrite uniquement par des actions primitives. L'originalité dans IPEM, c'est l'assimilation de ce concept à celui d'un défaut: on dira que le plan contient un défaut (uea) tout le temps qu'une action

expansable n'aura pas été expansée.

La réparation associée correspond à l'expansion de l'action, c'est-à-dire au remplacement de l'action non expansée par son schéma d'expansion. Ce schéma d' expansion peut lui-même contenir des actions expansables.

9.2.1.3 Exécution d'un plan

La représentation des actions est étendue de façon à pouvoir gérer leur exécution: une action primitive est associée à une procédure d'exécution implémentant l'action dans le domaine, et à un time-out indiquant à partir de quel moment il n'est plus nécessaire d'attendre les effets d'une action dont l'exécution a été lancée.

La sémantique des effets de l'action begin a également été étendue. A l'initialisation du problème, les effets de begin représentent la ***situation initiale***. Pendant la construction du plan et son exécution, les effets de begin représentent la ***situation courante*** telle que la perçoit le planificateur (modélisation du monde courant). Ainsi, la prise en compte d'événements extérieurs imprévus, ou la prise en compte des effets de l'exécution d'une action est réalisée par l'intermédiaire des effets de Begin: par exemple, si un fait n'est plus vrai dans le monde courant, alors il est supprimé des effets de begin.

L'exécution d'un plan est gérée par 5 défauts.

- **Lien non supporté (usr)**

Si un effet de begin n'est plus vrai (événement extérieur) ou si un effet attendu d'une action n'a pas pu être réalisé, alors si l'effet produisait un lien pour une précondition du plan, ce lien ne peut plus être supporté.

La réparation consiste à retirer le lien consommé par la précondition. Celle-ci n'est donc plus supportée. Il sera alors nécessaire de recréer un lien pour la supporter.

Ce défaut permet de tenir compte des événements extérieurs durant la construction des plans.

- **Action non exécutée (unexeca)**

Une action A est exécutable si:

- A est une action primitive
- toutes ses préconditions sont établies et vraies dans la situation courante
- A est directement après une action exécutée (begin est toujours exécutée)
- A n'est pas en conflit avec d'autres actions du plan
- il n'existe pas d'action B avant A, telle que A produise un effet nécessaire à l'application de B (précondition de B; B est en cours d'exécution)
- il n'existe pas d'action B avant A telle que A et B produisent des

effets contradictoires (B est en cours d'exécution).

La réparation consiste à exécuter l'action exécutable: à ajouter ses effets à begin au fur et à mesure qu'ils sont réalisés.

Lorsque la fin de l'exécution est atteinte (cf time-out) il est alors possible de détecter à travers les effets de begin les effets réalisés et ceux qui ne l'ont pas été. En effet, certains effets décrits dans le modèle d'action peuvent ne pas être réalisés lors de l'exécution.

- **Lien non étendu (uer)**

Soit p une précondition de l'action C.

Soit R le lien entre p et un effet de l'action P. P est avant C.

Soit P' une action située avant P, et possédant un effet susceptible de co-désigner avec p.

Alors, s'il n'existe pas d'action P" entre P' et P, telle que P" détruise p, le lien R est un lien non étendu.

La réparation consiste à détruire le lien R entre P et C, et à le remplacer par le lien R' entre P' et P (figure 113).

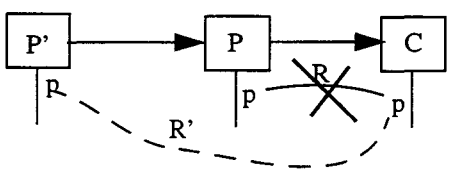


Figure 113: R est un lien non étendu

Ce défaut sert à détecter des effets miraculeux dans la situation courante: une action a été insérée dans le plan dans le but de réaliser un effet, et cet effet est miraculeusement vrai dans le monde courant.

- **Action redondante (reda)**

Une action est dite redondante lorsqu'elle ne produit plus d'effet utilisé dans le plan. C'est-à-dire lorsqu'aucun lien n'est produit par les effets de cette action.

La réparation consiste alors à retirer du plan l'action et les liens qu'elle consommait, puisqu'elle devient inutile.

- **Action terminée (toa)**

L'exécution d'une action est dite terminée lorsque le time-out est atteint.

La réparation consiste à retirer l'action du plan, avec tous les liens qu'elle produisait encore (effets non réalisés dans begin) et les liens qu'elle consommait.

9.2.1.4 Relation défaut - réparation et relation réparation - défaut.

Chaque défaut possède une ou plusieurs réparations possibles parfaitement déterminées.

Chaque réparation possible crée de nouveaux défauts qui sont également parfaitement identifiés. La relation entre réparation et défaut permet de limiter la recherche de nouveaux défauts dans le plan, que ce soit vis à vis du type de défaut à rechercher ou vis à vis de la partie de plan à parcourir pour les détecter. Nous résumons les deux types de relations dans le tableau 16.

Défauts détectés	Réparations associées possibles	Nouveaux défauts à détecter
usp (précondition non supportée)	reduction prior	• urc (nouveau lien, effets du plan) • unexeca
	reduction paral	• urc(nouveau lien, effets du plan)
	reduction new	• usp • urc(nouveau lien, effets du plan) + • urc(nouveaux effets, liens du plan) • uea
urc (conflit non résolu)	promotion	• unexeca • uer
	demotion	
uea (action non expansée)	expansion	• uea • usp • urc
usr (lien non supporté)	Retirer lien	• usp • reda
unexeca (action non exécutée)	Exécuter action	• toa • unexeca
uer (lien non étendu)	Etendre lien	• reda
reda (action redondante)	Retirer action	• usp • uer
toa (action timed-out)	Retirer action	• usr

Tableau 16 : Relations défauts-réparations-défauts

L'instanciation des défauts, et donc la limitation de la recherche, dépend de l'instanciation du défaut réparé et de la réparation qui vient d'être appliquée.

Nous avons résumé les caractéristiques d'IPEM; nous allons maintenant nous pencher sur les choix que nous avons faits en ce qui concerne l'implémentation de ce planificateur.

9.2.2 Les problèmes posés par l'implémentation

Une première étape dans le travail a consisté à implémenter une maquette d'IPEM en prolog, afin de le tester sur des exemples issus de CARNEADE et de dégager les avantages et les inconvénients de son utilisation.

Cette maquette reprenait tous les aspects d'IPEM que nous venons de décrire hormis la planification hiérarchique traduite par les actions expansables et le time-out de l'exécution des actions.

Nous avons choisi de l'implémenter en prolog pour les raisons qui suivent:

- obtenir rapidement une maquette utilisable pour réaliser des tests
- bénéficier de l'adéquation de prolog à la planification:
    - utiliser les mécanismes d'unification prolog pour réaliser l'unification des variables du plan à travers les liens, et propager naturellement ces unifications
    - utiliser le mécanisme de backtrack chronologique de prolog pour effectuer les recherches

Les choix d'implémentation ont été guidés par la partie réactive d'IPEM. En effet, nous cherchions à tester les capacités d'IPEM à modifier le plan en cours d'élaboration en fonction des résultats des exécutions des actions et des événements extérieurs imprévus (ici des actions ennemies ou des informations sur des données terrains).

Les points difficiles concernaient la gestion des contraintes d'unification et la gestion des relations de précédence. En effet, l'unification de deux variables correspond soit à la création d'un lien par le planificateur, soit à la propagation des unifications à travers une séquence de liens. Par exemple, dans la figure 114, les liens sont:

- R1 (p(X), p(a)): X= a
- R2 (q(Y), q(X)): X=Y
- R3 (r(Z), r(Y)): Z = Y

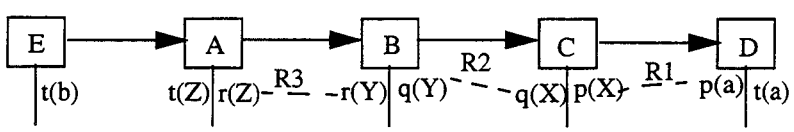


Figure 114: Exemple de plan avec ensemble de liens

Alors deux options existent pour gérer ces liens:

1 - soit la propagation est faite une fois pour toutes dès qu'une nouvelle unification est réalisée. Dans ce cas le mécanisme est le suivant:

- R1 => X=a
- R2 => Y=X, X=a => Y=a
- R3 => Z=Y, Y=X, X=a => Z=X, Z=a

Ce mécanisme permet de connaître instantanément les valeurs des variables.

Ainsi, si le planificateur tente de construire un quatrième lien entre t(Z) et t(b), il sait immédiatement si c'est possible: ici, Z=b et Z=a est incohérent. Donc le planificateur tente un autre lien.

2 - soit l'unification des variables et la propagation sont réalisées ponctuellement à chaque fois que le planificateur tente d'ajouter un

nouveau lien, mais sont défaites une fois que les contrôles ont été effectués. Ainsi, dans le plan de la figure 114, le planificateur connaît les 3 unifications, mais indépendamment les unes des autres. S'il tente d'ajouter le quatrième lien entre t(Z) et t(b), il doit d'abord parcourir l'ensemble des liens concernant la variable Z pour savoir quelle est sa valeur, puis ajouter Z=b et vérifier si les deux valeurs obtenues sont cohérentes.

La première approche est plus efficace lors de la planification car la propagation des unifications est faite incrémentalement, et la vérification de la consistance d'un ensemble d'unification est immédiate puisque toute l'information nécessaire a été calculée. Cependant, lors d'une étape d'exécution cette approche est plus lourde que la deuxième. En effet, lors d'une étape d'exécution, il s'agit très souvent de retirer des liens ou d'en ajouter et de retirer des actions du plan, sans utiliser les techniques de backtrack liées à prolog. Ce sont des modifications spécifiques et volontaires qui ne sont pas liées à des points de choix. Alors, l'utilisation de la première approche nécessite la mise à jour de toutes les variables concernées par le retrait d'un lien. Par exemple, si on veut retirer le lien R1, alors il est nécessaire de retirer les unifications entre Z et a, et Y et a, en plus de l'unification entre X et a. L'utilisation de la deuxième approche nous permet de retirer uniquement le lien R1, puisque les unifications ne sont faites que ponctuellement. Seules les unifications "directes" issues des liens construits par le planificateur sont mémorisées.

Nous avons choisi la deuxième approche afin de faciliter les étapes d'exécution des plans et nous nous reposons sur la rapidité du mécanisme d'unification de prolog pour effectuer la propagation.

Le même problème se pose en ce qui concerne les relations de précédence.

1 - L'approche classique consiste à propager toute l'information temporelle de façon à détecter les incohérences, mais aussi pour bénéficier immédiatement de toute l'information nécessaire à l'ajout d'une nouvelle relation de précédence. Si nous reprenons l'exemple de la figure 114, l'ajout des différents liens implique les relations de précédence suivantes:

- R1: C < D
- R2: B < C
- R3: A < B

L'approche classique calcule toutes les relations temporelles entre toutes les actions: O = (A< B, A< C, A < D, B<C, B<D, C<D). Alors lors de l'ajout d'une nouvelle relation de précédence X < Y , l'évaluation de la cohérence du nouvel ensemble de relations est immédiate: si $O \cup (X < Y) \supset false$ alors il est impossible d'ajouter la relation X<Y.

2 - Une seconde approche consiste à ne mémoriser que les relations de précédence immédiate, c'est-à-dire-celles issues de la construction du plan. Dans ce cas, lors de l'ajout d'une nouvelle relation, il est nécessaire de propager les relations temporelles à partir de Y, pour savoir si X est après Y dans le plan. Si la réponse est non, alors X<Y peut être ajouté, sinon le plan est incohérent.

Toujours dans le but de faciliter les étapes d'exécution des plans, nous avons choisi cette deuxième approche.

En conclusion, si la construction d'un planificateur classique nécessite des approches où les informations sont calculées complètement et de façon incrémentale, l'intégration d'un module d'exécution intégré au sein d'un planificateur conduit à opter pour des approches ne mémorisant que l'information minimum nécessaire complétée ponctuellement et temporairement à chaque besoin.

Dans le paragraphe suivant, nous présentons des résultats de quelques tests que nous avons effectués en collaboration avec du personnel militaire.

9.2.3 Les résultats des tests et les modifications réalisées

Nous avons proposé à différentes personnes de l'état major militaire d'utiliser le planificateur IPEM. Nous leur avons présenté la problématique qui nous intéressait, et leur avons démandé de construire des jeux d'essais issus du domaine militaire. Ces jeux d'essais consistaient en la définition d'un problème (description des buts, description de la situation initiale et description des actions possibles), et en la construction et exécution des plans à l'aide de notre implémentation d'IPEM.

Nous présentons ici un exemple de jeux d'essais, et les conclusions générales que nous avons tirés des tests.

9.2.3.1 Exemple de jeux d'essais

La construction d'un jeu d'essai nécessite une étape de définition du vocabulaire utilisé pour décrire les actions et les situations.

Les lettres majuscules correspondent à des variables, les minuscules à des constantes.

L'exemple suivant est issu du jeu d'essai baptisé "attaque-double" décrit complètement dans l'annexe E.

- **Vocabulaire utilisé**

Nous expliquons dans la figure 115 le sens associé à chaque prédicat.

subordonne(X, SUP) : X est un subordonné de SUP;
X et SUP sont des troupes militaires
pos(U, X): l'unité U se trouve au lieu identifié X
libre(U): l'unité (ou toute autre ressource) est libre
vide(E, X, Y): E n'est pas entre le lieu X et le lieu Y
entre(E, X, Y): E est entre X et Y.
passage(E, X, Y): il y a un passage au sein de E (unité) entre le lieu X et le lieu Y
zone_engagement(X, Y): il y a une zone d'engagement entre le lieu X et le lieu Y

Figure 115: Sémantique des prédicats

- **Situation initiale**

Dans la situation initiale sont décrites toutes les connaissances sur le terrain, les troupes amies et ennemies (figure 116).

```
subordonne(a1,ami).
subordonne(a2,ami).
subordonne(e1,eni).
subordonne(e2,eni).
pos(a1,p1).
pos(a2,p1).
libre(a1).
libre(a2).
vide(e2,p1,ld).
vide(e1,p1,ld).
entre(e2,ld,li).
entre(e1,li,lo).
```

Figure 116: Description de la situation initiale du jeu 1

- **Situation finale**

Dans la situation finale sont décrits uniquement les buts à atteindre (figure 117).

```
vide(e2,ld,li).
vide(e1,li,lo).
pos(a1,lo).
```

Figure 117: Description des buts du jeu 1

- **Les actions possibles**

Nous décrivons dans la figure 118 quelques actions utilisées pour ce problème. La totalité des actions est donnée dans l'annexe E.

```
deplacer_ligne_directe ::si     subordonne(A,ami)
                                et subordonne(E,eni)
                                et (vide(E,Q,P),h)
                                et pos(A,Q)
                        alors   pos(A,P)
                                et (-pos(A,Q),s)
                        contrainte (diff(E,A), diff(P,Q) ).

detruire ::     si              subordonne(A,ami)
                                et subordonne(B,ami)
                                et entre(E,P,Q)
                                et pos(A,P)
                                et pos(B, P)
                                et soutien(A, B)
                        alors   (pos(A,Q),s)
                                et (-pos(A,P),s)
                                et vide(E,P,Q)
                        contrainte (diff(E, A), diff(P, Q) ).

soutenir::              si      subordonne(A,ami)
                                et      subordonne(B,ami)
                        et      pos(A,P)
                        et      pos(B,P)
                        alors   soutien(A,B)
                        contrainte diff(B,A).
```

Figure 118: Description de diverses actions du jeu "attaque-double"

Nous avons réalisé des tests dans deux buts précis:

- pour savoir si les caractéristiques d'IPEM correspondaient aux besoins de planification et d'exécution dégagés dans CARNEADE.
- pour savoir si le formalisme de description des problèmes utilisé dans IPEM correspondait aux besoins issus de CARNEADE.

Le paragraphe suivant est consacré aux résultats des premiers types de tests. Le paragraphe 9.2.3.3 reprend les conclusions que nous avons tirées de la deuxième série de tests. Nous décrivons dans ces deux paragraphes les extensions que nous avons proposées et implémentées dans TCLP.

9.2.3.2 Résultats vis à vis du planificateur

- **Les caractéristiques réactives du planificateur**

La partie exécution des plans d'IPEM est satisfaisante. Elle permet en outre d'analyser la suite des situations intermédiaires atteintes avant la situation finale, et donc de vérifier à nouveau la correction des plans générés.

Afin de tester la partie réactive d'IPEM, nous avons instauré un système d'interruption du planificateur permettant à l'utilisateur (en l'occurence un militaire) de modifier la description du monde courant et de voir évoluer le plan en fonction de ces

nouvelles informations. Cette interruption "manuelle" peut être améliorée par un système de processus en parallèle, l'un simulant les modifications du monde extérieur, l'autre représentant la détection et la prise en compte de ces changements par le planificateur.

Nous ne nous attardons pas sur cet aspect des tests, car globalement le comportement d'IPEM nous a convenu. Nous pouvons simplement remarquer, que les corrections réalisées sur le plan par IPEM sont des corrections locales. En effet, le mécanisme défaut-réparation utilisé pour l'exécution des plans est un concept "local". Il n'englobe pas de critères globaux d'évaluation des plans (utilité, préférences...). IPEM est donc incapable d'abandonner complètement le plan courant, si une meilleure solution se présente. Ce cas peut cependant survenir par le mécanisme de relation défauts/ réparations, et réparations/défauts, si toutes les actions planifiées deviennent une à une redondante.

- **Influence de l'exécution des plans sur la planification**

L'entrelacement de la planification et de l'exécution des plans peut soit conduire à des impasses irrémédiables, soit faciliter la construction des plans. Nous nous basons sur un exemple général pour expliquer ce point de vue.

Soient deux actions A1 et A2 toutes deux exécutables dans une même situation courante. A1 et A2 ont un effet principal identique. Le planificateur peut choisir l'une ou l'autre pour réaliser cet effet.

- ***cas d'impasse***:
  Imaginons qu'il ait choisi A1. A1 devient exécutable dans le plan; IPEM l'exécute, et poursuit la construction du plan. Or il se peut que le choix de A1 ait des conséquences sur le reste du plan (l'exécution de A1 peut détruire des faits qui auraient pu être nécessaires à une autre action dans le plan, alors que A2 ne les aurait pas détruits). Or il est facile d'imaginer une situation où il n'existe pas d'actions applicables permettant de rétablir les faits détruits. Il semble alors que la seule solution consiste à revenir sur l'insertion de A1 dans le plan. Or cette action a été exécutée, il est donc impossible de revenir sur son exécution. Si aucune autre solution n'est possible IPEM ne trouve pas la solution contenant l'action A2.
- ***Cas de simplification***:
  Ce cas apparaît lorsque l'exécution d'une action élimine des points de choix (puisque l'on considère que l'exécution d'une action n'est pas réversible) qui auraient pu être l'origine d'impasse ou de boucle infinie.

Les deux cas que nous venons de décrire montrent non seulement l'influence de l'exécution des plans sur leur génération, mais soulèvent également le problème l'influence du choix des défauts: dans le premier cas, si le défaut "unexec A1" n'avait

pas été choisi, le planificateur aurait pu revenir sur l'utilisation de A1 et trouver une solution avec A2.

Plus généralement, nous expliquons l'influence du choix des défauts sur la construction des plans dans le point suivant.

- **Stratégies de choix des défauts**

Nous distinguons deux types de stratégies: celles basées sur l'ordre dans lequel sont détectés les défauts et celles basées sur le type de défauts.

La stratégie que nous utilisons cherche à réparer les défauts dont les réparations modifient le moins possible la structure du plan en ce qui concerne l'ajout de nouveaux éléments. Afin de modifier à volonté les stratégies de choix dépendant du type de défaut, nous avons associé des priorités à chaque type de défaut. Ces priorités peuvent être modifiées avant chaque simulation (ie, planification + exécution) et restent alors figées pendant toute la simulation.

Nous avons vu qu'il existe deux types de défauts: ceux qui concernent la synthèse des plans, et ceux qui concernent leur exécution. Selon l'ordre de priorité des défauts d'exécution et de planification, l'entrelacement planification/exécution peut être modifié.

Ainsi, si tous les défauts de planification sont prioritaires devant les défauts d'exécution alors l'exécution du plan n'est lancée que lorsque le plan est complètement achevé.

Si les défauts d'exécution sont prioritaires devant les défauts de planification alors dès qu'une action planifiée est exécutable, elle est exécutée. Dans ce cas, le planificateur réagit plus rapidement aux changements du monde. En effet, dans le premier cas, il ne peut réagir que lorsque le plan est terminé, au risque de devoir abandonné le plan si celui-ci n'est plus valide.

Ce sont ces deux types de stratégies qui nous ont permis de mettre en évidence les avantages et les inconvénients de l'entrelacement de la planification et de l'exécution évoqués dans le point précédent.

En ce qui concerne les défauts de planification (usp et urc dans notre implémentation), nous pouvons implémenté la stratégie utilisée dans SNLP en donnant une priorité supérieure à la réparation des conflits (urc), ou la stratégie appelée DEnd dans [PEO 93], en donnant une priorité supérieure au traitement des préconditions non établies (usp).

La stratégie classique que nous avons utilisée correspond au classement des défauts suivant, par ordre de priorité décroissante:

- action non exécutée,
- lien non supporté,
- lien non étendu,
- conflit non résolu,
- précondition non supportée.

Cette stratégie consiste à exécuter les actions dès qu'elles sont exécutables, à réagir aux événements extérieurs le plus rapidement possible, à corriger le plan dès qu'il

devient incorrect et à poursuivre la construction du plan dans tous les autres cas.

La stratégie de choix du type de réparation est figée, cablée dans le programme, en associant dans un ordre prédéfini les différents types de réparation d'un défaut à ce défaut.

Par exemple si le défaut est une précondition non supportée, il peut être réparé par trois réparations (reduction prior, reduction parallel, reduction new). Le planificateur tente alors de réparer ce défaut en modifiant le moins possible le plan (ie en choisissant d'abord reduction prior), et se tourne vers des modifications plus importante seulement s'il ne peut réparer le défaut avec la première solution.

Nous avons complété ces stratégies par des stratégies jouant sur le backtrack du planificateur et donc sur sa complétude. Ainsi, lors du lancement de la construction d'un plan pour un jeu-d'essai (plan(jeu-d'essai, TYPE, ***STRAT***)) il est possible de choisir une stratégie particulière s'ajoutant aux stratégies précédentes ou les remplaçant.

- Si ***STRAT*** = ***tete*** (FIFO) alors à chaque étape de planification, le défaut le plus anciennement détecté est choisi et est réparé. Si la réparation échoue, alors le retour-arrière se fait jusqu'à l'étape précédente. A ce point, si le défaut qui a été choisi peut être réparé par une autre réparation, le planificateur reprend la planification avec cette nouvelle réparation. Sinon, le planificateur revient au choix précédent...Cette stratégie conduit à construire des plans en "largeur", si la largeur d'un plan correspond au nombre de branches parallèles dans le plan. En effet, si deux actions a1 et a2 ont été successivement introduites dans le plan pour établir deux buts b1 et b2, alors les préconditions de la première action sont traitées (avec introduction de a3), puis celles de la deuxième, puis celle de a3... (voir figure 119). Cette stratégie annule toute notion de priorité associée aux types de défaut.

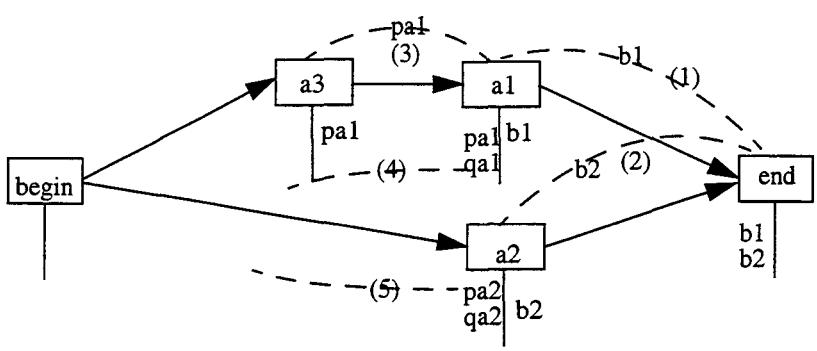


Figure 119: Construction d'un plan suivant la stratégie tete;
(n) indique l'ordre d'application des réparations

- Si ***STRAT*** = ***queue*** (LIFO) alors le dernier défaut détecté à l'étape courante est traité à l'étape suivante. Cette stratégie annule toute notion de priorité associée aux types de défaut, et conduit à construire des plans en "longueur", par opposition à "largeur". Le plan de la figure 120 correspond à la construction du plan précédent suivant la stratégie queue.

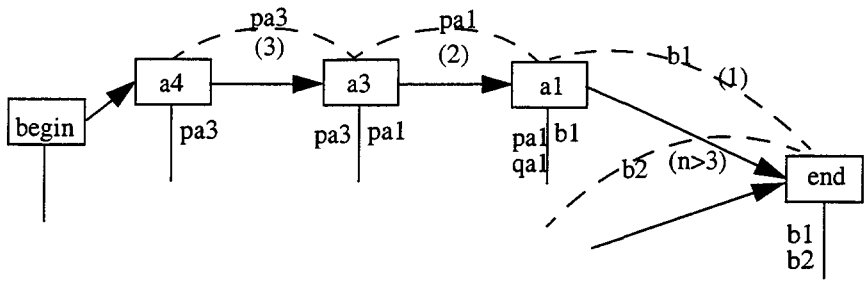


Figure 120: Construction d'un plan suivant la stratégie queue; (n) indique l'ordre d'application des réparations

- Si ***STRAT = tous*** alors les défauts sont choisis
  1 - dans l'ordre décroissant des priorités

  2 - si plusieurs défauts ont la même priorité alors ils sont choisis dans l'ordre chronologique de détection

  3 - si la réparation d'un défaut à l'étape courante échoue et si le défaut n'est pas d'un type spécial (voir paragraphe suivant), alors le défaut est remis dans l'agenda et un autre défaut est essayé en suivant la même stratégie. Si aucun des défauts de l'agenda courant ne peut être réparé, alors le planificateur revient à l'étape précédente. Si le défaut est d'un type spécial, et si la réparation échoue alors le retour arrière est immédiat.

- Si ***STRAT = max*** alors
  1 - un défaut de priorité maximale (dans l'agenda) est choisi et est traité

  2 - si plusieurs défauts ont une priorité maximale alors ils sont choisis dans l'ordre chronologique de leur détection.

  3 - si le défaut choisi ne peut pas être réparé, et s'il n'est pas d'un type spécial, alors il est remis dans l'agenda et s'il existe un autre défaut de priorité maximale, ce défaut est traité. Si aucun des défauts de priorité maximale ne peut être traité alors le planificateur revient à l'étape précédente. Si le défaut est d'un type spécial, et si la réparation échoue alors le retour arrière est immédiat.

- Si ***STRAT = perm*** alors
  1 - le défaut le plus ancien est choisi

  2 - si la réparation de ce défaut échoue, et s'il n'est pas d'un type spécial, alors il est replacé dans l'agenda et le défaut suivant le plus ancien est choisi. Si aucun des défauts de l'agenda courant ne peut être réparé alors le planificateur revient à l'étape précédente. Si le défaut est d'un type spécial, et si la réparation échoue alors le retour arrière est immédiat.

Le planificateur est complet lorsque la stratégie ***tous*** ou la stratégie ***perm*** est

utilisée car tous les défauts de l'agenda sont choisis avant de revenir en arrière.

La stratégie ***max*** consiste à dire que si les défauts les plus difficiles à réparer (en fonction de leur type) ne peuvent pas être réparés alors ce n'est pas la peine d'essayer de réparer les autres défauts.

Les stratégies ***tete*** et ***queue*** restreignent considérablement le backtrack vis à vis du choix des défauts dans l'agenda. Elles deviennent rapidement obsolètes lorsque la construction du plan nécessite de nombreuses tentatives avant de trouver la bonne solution. Elles conduisent à un échec ou à une réussite rapide du planificateur.

Le paragraphe suivant est consacré aux conclusions que nous avons tirées des tests vis à vis de la représentation du domaine et des actions.

9.2.3.3 Résultats vis à vis de la représentation du domaine et des actions

- **Les buts**

La traduction d'un ordre verbal (issu de la mission à remplir) en un ensemble de buts à atteindre n'a pas posé de problèmes majeurs. La description de la situation finale est partielle, elle ne contient que les buts à atteindre.

Il semble cependant nécessaire de hiérarchiser les buts à réaliser de façon à pouvoir guider la construction du plan en fonction des buts, et à reproduire le concept d'effet majeur utilisé lors de la construction de plans d'actions dans le cadre d'une mission. En ce qui concerne la construction des plans, cette hiérarchie pourrait être facilement réalisée en associant des priorités à chaque but et en obligeant le planificateur à traiter les buts les plus prioritaires en premier. Cette extension doit cependant tenir compte de l'exécution des plans. En effet, si l'exécution d'un plan échoue partiellement et ne permet pas la réalisation de l'effet majeur, alors l'échec peut avoir des conséquences sur la totalité du plan: il peut alors être nécessaire d'abandonner tout le plan. Si cet échec concerne un but "secondaire", alors il est peut être possible de modifier localement le plan et de poursuivre la construction en tenant compte de cet échec.

- **La situation initiale**

La description de la situation initiale a été plus délicate. Ce problème est généralisable à tout problème de description du monde lorsqu'il est demandé de décrire toutes les connaissances que l'on possède sur ce monde. Plusieurs problèmes se posent:

- choisir un modèle de représentation: dans notre cas nous utilisons la logique des prédicats du premier ordre
- limiter la description du monde en fonction du problème à traiter: définition du vocabulaire utilisé pour décrire les situations et les actions. Cette étape permet également de structurer le langage en fonction des liens entre les caractéristiques du monde à décrire. Par exemple, un terrain peut être constitué de lieux identifiés, de différentes liaisons entre ces lieux, de regroupement de lieux...
- évaluer la suffisance des données étant donné le grand nombre de données à

représenter.
- évaluer la cohérence des données

Afin de limiter le nombre de données que nous avions à représenter, et parce que les préconditions des actions décrites ne contiennent pas de préconditions négatives, nous suivons l'hypothèse du monde clos.

Cependant, s'il devient nécessaire d'étendre la représentation des actions pour différencier des préconditions connues vraies, connues fausses et inconnues, alors il devient nécessaire de décrire dans les effets de l'action begin, non seulement ce qui est vrai mais également ce qui est faux.

L'évaluation de la cohérence et de la suffisance des données a reposé sur le bon sens et l'expertise des personnes chargées de construire les jeux d'essais. Elles peuvent être en outre évaluées en faisant tourner le planificateur sur les jeux d'essais construits. Cependant, cette approche nécessite la construction à la main des plans et le suivi pas à pas de l'élaboration du plan effectuée par le planificateur. Cette technique devient rapidement laborieuse lorsque la synthèse des plans est complexe.

- **Plan partiel initial: contraintes sur la forme du plan**

La possiblité de pouvoir décrire un plan initial plus détaillé que la donnée des buts et de la sitiation initiale afin de restreindre les solutions possibles a également été envisagée. Ainsi, lors des tests, ont été émis les besoins de pouvoir décrire:
- des états obligatoires intermédiaires de façon à donner un squelette initial au plan. Cette approche revient à scinder les buts à atteindre et à les positionner temporellement. Les états obligatoires peuvent également être imposés sur des intervalles de temps.
- des actions obligatoires
- des états interdits

Le premier point se transcrit facilement par la description d'un plan partiel contenant plusieurs situations "finales" représentées par des opérateurs fictifs ne produisant et ne détruisant rien. Les préconditions d'un tel opérateur traduisent la nécessité d'établir les buts correspondant. Les effets (identiques aux préconditions) indiquent qu'il est possible d'utiliser les buts établis pour le reste de la synthèse des plans. Ainsi, dans le plan de la figure 121, le but final est décrit par les préconditions de end, les états intermédiaires imposés sont décrits par les opérateurs end 1 et end 2. Ce qui signifie qu'il est obligatoire de réaliser end 1 et end 2 avant end, sans imposer d'ordre entre end 1 et end 2. Pour indiquer que b1 doit être vrai jusque la fin du plan (end), un maintien est posé sur b1 entre end 1 et end.

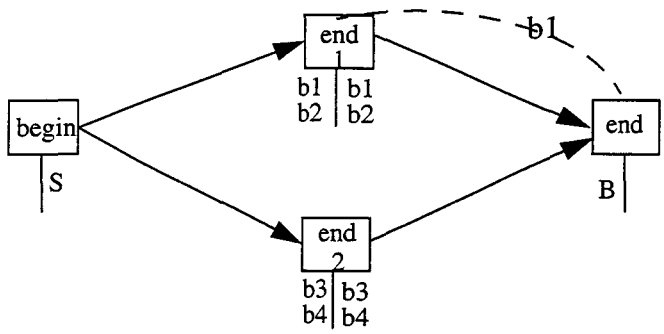


Figure 121: Plan partiel initial avec buts intermédiaires

Le deuxième point est plus délicat car il impose l'utilisation des effets de l'action obligatoire. Nous ne l'avons pas implémenté.

Le troisième point peut être partiellement implémenté. En effet, nous avons vu dans le chapitre 4 que le maintien est un moyen d'interdire des états entre deux instants. Cependant, les états qui peuvent être interdits sont limités à des faits uniques: il est impossible d'interdire une conjonction de faits. Cette restriction est due à la complexité de la détermination de la vérité d'un fait dans un plan partiellement ordonné. Il est uniquement possible de savoir quels sont les faits vrais et faux juste après l'application d'une action, ou entre deux actions si elles sont totalement ordonnées et si aucune autre action ne peut être appliquée entre les deux. Dans le cas où elles sont indépendantes et non ordonnées, la succession des situations dépend de l'ordre dans lequel elles sont appliquées.

Afin de fournir au planificateur le moyen de détecter la cohérence des plans, à l'instar de la cohérence des situations, il est nécessaire de définir des incompatibilités sémantiques. La construction de la liste des incompatibilités permet en outre de revenir sur la cohérence du vocabulaire utilisé pour décrire le problème.

- **Les incompatibilités sémantiques**

Nous avons vu dans le chapitre 5 l'utilisation et le rôle des incompatibilités sémantiques.

Il est apparu nécessaire de définir ces incompatibilités sémantiques lors de tests où des plans partiels auraient pu être corrigés plus tôt si les incompatibilités sémantiques avaient été plus précises que la seule règle du domaine interdisant un fait et son contraire d'être vrais dans la même situation.

Soit le plan de la figure 122.

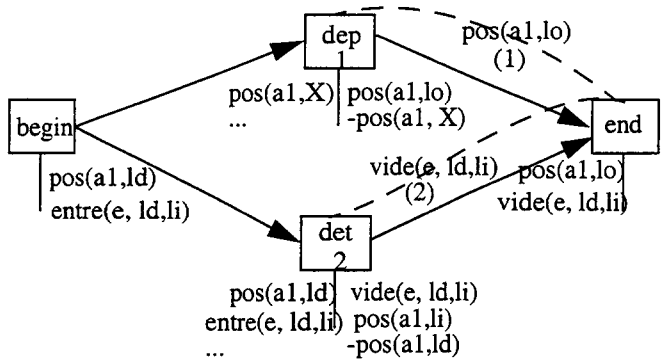


Figure 122: Plan partiel incorrect
dep = deplacer_ligne_directe;
det= detruire

- Cas 1: seul un fait et son contraire vrais dans la même situation est interdit. Dans ce cas le plan de la figure 122 ne contient pas de conflits. Or il est incohérent de dire que pos(a1, lo) et pos(a1, li) peuvent être vrai en même temps.

- Cas 2: la règle suivante est rajoutée: pos(A, X) et pos(A, Y) ne peuvent être vrais en même temps si X est différent de Y. Dans ce cas, un conflit est détecté entre le maintien pos(a1, lo) et pos(a1, li). L'action 2 est placée avant l'action1 dès cette étape de construction du plan.

- **Les actions**

Très rapidement la description des actions à l'aide du modèle STRIPS s'est avérée insuffisante. Le besoin d'exprimer la durée des actions est apparu de plusieurs manières:

- ***afin de coller à la réalité***: pourquoi planifier avec des actions instantanées si l'on sait pertinemment qu'elles se dérouleront sur un intervalle de temps non nul lors de l'exécution?
- ***afin de décrire l'entrelacement de plusieurs actions***. Lorsque plusieurs agents interviennent dans les plans, pourquoi les obliger à agir les uns après les autres si leurs actions sont indépendantes?
- ***afin de décrire l'utilisation de ressource pendant l'application d'une action*** et la libération de ressource à la fin de l'action. Le déplacement d'une troupe interdit l'utilisation de cette troupe pour certaines autres actions simultanément.
- ***afin de décrire plus précisément le déroulement des actions***: pourquoi imposer la description des actions uniquement par des conditions d'application qui doivent être vraies avant l'application des actions et par des effets qui ne seront vrais qu'après leur application.
- ***afin de décrire des actions dont les effets ne se manifestent que pendant leur application***. Certaines actions n'ont pas le sens qu'on veut leur donner si elles sont décrites à l'aide du modèle STRIPS. Ainsi, l'action soutenir décrite dans la figure 118 traduit le fait que le soutien est réalisé une fois l'action terminée; ce qui semble incohérent.

C'est pourquoi nous avons construit une représentation de l'action autorisant la prise en compte de leur durée (les d-actions) et la description des propriétés précédemment évoquées. Les caractéristiques des d-actions ont été définies dans le chapitre 4 de ce mémoire. IPEM fournit un moyen d'entrelacer l'exécution des actions grâce au time-out. cependant, il n'est pas possible de répondre aux problèmes précédemment évoqués ci-dessus à l'aide du time-out, notamment parce que dans les plans, les actions sont instantanées et positionnées uniquement par la relation de précédence. Alors l'entrelacement des actions ne peut être imposé. Cette technique représente uniquement un moyen de ne pas restreindre l'exécution des actions une à une et donc d'accélérer l'exécution des plans.

Les résultats des tests que nous venons de présenter nous ont conduits à construire TCLP, en mettant l'accent sur l'extension du modèle d'action et des relations temporelles. C'est pourquoi, nous n'avons pas repris les caractéristiques réactives d'IPEM dans TCLP. Néanmoins nous avons tenu à conserver un formalisme de représentation des plans et des mécanismes de construction des plans identiques à ceux d'IPEM afin de pouvoir réintégrer le plus facilement possible ses caractéristiques réactives; en l'occurence le formalisme des plans traditionnels non linéaires à liens causaux.

Le paragraphe suivant résume les caractéristiques de TCLP qui ont été implémentées, et décrit les algorithmes de planification utilisés.

9.3 Le passage à TCLP

9.3.1 Les caractéristiques de TCLP

9.3.1.1 Structure générale du planificateur

Nous avons conservé la structure d'IPEM en ce qui concerne les mécanismes de détection et de réparation des défauts liés à la construction des plans.

9.3.1.2 Les défauts et les réparations

Nous avons étendu le nombre de défauts de façon à spécialiser les réparations associées.

Ainsi, il est possible de distinguer trois types de préconditions non supportées:
- les ***préconditions immuables*** ne pouvant être réparées que par reduction-prior et faisant l'objet d'un traitement spécial dans les stratégies de choix des défauts (appelé type ***spécial*** dans le paragraphe précédent)
- les ***hold-préconditions*** pouvant être réparées par reduction prior et reduction parallel
- les ***préconditions*** (classiques) pouvant être réparées par reduction prior, parallel ou new.

Nous avons également distingué ***23 types de conflit*** basés sur le maintien, et

conduisant à la simplification des réparations associées. Ces types de conflits et ces réparations sont décrites dans le chapitre 7.

Nous avons inclus la ***distinction entre conflit potentiel et conflit réel***, afin de ne pas réparer a priori des conflits susceptibles de disparaître.

Nous avons ajouté la notion de ***conflit global***, permettant de traiter globalement un ensemble de conflits et de diminuer le backtrack.

Nous avons également modifié la stratégie de réparation des conflits de façon à diminuer l'engagement en ne choisissant pas une réparation a priori lorsque plusieurs réparations sont possibles.

Nous avons ***modifié la réparation reduction new*** de la façon suivante:

- recherche d'une action par les effets primaires appartenant à l'index décrit dans l'action (action.index)
- insertion d'une d-action = insertion dans le plan d'un ensemble d'opérateurs STRIPS, d'un ensemble de relations de précédence et d'un ensemble de maintiens au lieu d'un seul opérateur STRIPS
- détection des conflits entre le nouveau maintien (issu du lien causal) et les effets du plan, entre les nouveaux effets de la d-action et les maintiens du plan, mais également détection des conflits entre les maintiens de la d-action et les maintiens du plan, ainsi que les maintiens de la d-action et les effets du plan.

9.3.1.3 La représentation du domaine

TCLP accepte des plans initiaux contenant des états obligatoires et des interdictions sur des faits uniques.

TCLP construit des plans constitués de d-actions. Le plan solution fourni est un plan d'opérateurs instantanés (voir annexe E).

Nous avons implémenté la ***table de transition*** qui permet de traduire la description des d-actions depuis un langage évolué (description hiérarchique, relations d'Allen, opérateur de description) vers une structure compréhensible des planificateurs traditionnels non linéaires à liens causaux.

Nous donnons en annexe E deux exemples de problèmes modélisés en suivant le formalisme de représentation de TCLP.

9.3.2 Les algorithmes de planification

- **Définition des variables**

Nous reprenons les mêmes notations que celles utilisées dans le chapitre 8. Nous les indexons par le numéro de l'étape courante. Nous rappelons ici leur signification.

- ***i*** le numéro de l'étape courante
- ***Pi*** le plan courant = (Si, Oi, Mi, Ci)
- ***Oi*** = l'ensemble des relations de précédence entre les opérateurs instantanés du plan courant Pi
- ***Ci*** = l'ensemble des contraintes sur les variables du plan Pi

- ***Si*** = l'ensemble des opérateurs instantanés du plan Pi
- ***Mi*** = l'ensemble des maintiens du plan Pi
- ***Ai*** = agenda des défauts. Ai = (Areeli, Confpoti) avec ***Areeli*** l'agenda des défauts réels, ***Confpoti*** l'agenda des conflits potentiels.
- ***Biblio***: l'ensemble des d-actions traduites suivant le formalisme TCLP
- ***Monde***: plan partiel initial, décrit sous la forme d'une d-action
- ***Biblio-externe***: l'ensemble des d-actions définies pour le problème
- ***I*** = l'ensemble des incompatibilités du domaine

- Un défaut ***D*** est défini par:
  - ***D.nom***: le nom du défaut (usp, imm-usp, hold-usp, conflit-maintien-maintien, ..., conflit-global)
  - ***D. ptél, ..., D.ptén***: des propriétés permettant de définir l'instance du défaut
  - ***D.réparations***: la liste des types de réparations possibles partiellement instanciées en fonction du défaut (red-prior, red-paral, red-new, (<))

- **Algorithme de planification**

Les procédures en italique ne sont pas développées. Leur rôle est expliqué à la suite de l'algorithme 18.

Algorithme 15: tclp (Monde, Biblio-externe, R, Strat)
Début

(P0, A0, Biblio) <- initialisation-plan(Biblio_externe, Monde) % algorithme 16
(S, O, M, C) <- plan(P0, A0, Biblio, Strat) % algorithme 17
retourner (S, O, M, C)
Fin

Algorithme 16: initialisation-plan(Biblio-externe, Monde)
Début

Biblio <- initialisation-action (Biblio-externe)
(P0, A0) <- *initialisation* (Monde) % algorithme 6
% transforme la description externe du plan partiel initial
% dans le formalisme TCLP; détecte tous les défauts à réparer.
% Met à jour de P0= (S0, O0, M0, C0) et A0
Retourner (P0, A0, Biblio)
Fin

Algorithme 17: plan(Pi, Ai, Biblio, Strat)
Début

Tantque Areeli $\neq \varnothing$ **faire**

Di <- *Choisir-selon-stratégie-1-défaut*(Areeli, Strat)
Areeli <- Areeli - Di
Si Di.nom = conflit-global **alors**
(Ai+1, Pi+1) <- appliquer-réparation-globale (Di) % algorithme 13

```
        Sinon
            rij <- choisir-réparation (Di. réparation)   % le choix dépend d'une stratégie fixée
            (Ai+1, Pi+1) <- appliquer-réparation (Di, rij) % algorithme 18

        Fsi
        plan (Ai+1, Pi+1)
    Fintantque
Fin
```

Algorithme 18: appliquer-réparation (D, r)

```
Début

    Si r = red-prior (precond, N) alors
        (V, P, effet-et) <- choisir-établisseur-new (precond, N)
        Si V alors
            Oi+1 <- Oi
            Si+1 <- Si
            Mi+1 <- Mi U maintien(precond, P, N)
            Ci+1 <- Ci U (precond = effet-et)
            (Conf-reel, Conf-poti') <- parcourir-conflit-potentiel (Conf-poti)

            % détection des nouveaux conflits
            % retourne les nouveaux conflits-reels CRi+1
            % et les nouveaux conflits potentiels CPi+1
            pour tout op de Si+1
                pour tout effet de op.add
                    detecter-conflit-maintien-effet (maintien(precond, P, N), (effet, op)
                    % algorithme 10
                Finpour
            Finpour
            A-attentei <- CRi+1
            Confpoti+1 <- Confpoti' + CPi+1
            (Areeli+1) <- Construction-réparation-individuelle(A-attentei) % algorithme 12
            Retourner Ai+1 = (Areeli+1, Confpoti+1)
            Retourner Pi+1 = (Si, Oi+1, Mi+1, Ci+1)
        Sinon fail
        Fsi

    Sinon
        Si r = red-paral (precond, N) alors
            (V, P, effet-et) <- choisir-établisseur-paral (precond, N)
            Si V alors
                Oi+1 <- Oi U (P < N)
                Si+1 <- Si
                Mi+1 <- Mi U {maintien(precond, P, N)}
                Ci+1 <- Ci U {(precond = effet-et)}
                (Conf-reel, Conf-poti') <- parcourir-conflit-potentiel (Conf-poti)

                % détection des nouveaux conflits
                % retourne les nouveaux conflits-reels CRi+1
                % et les nouveaux conflits potentiels CPi+1
                pour tout op de Si+1
                    pour tout effet de op.add
                        detecter-conflit-maintien-effet (maintien(precond, P, N), (effet, op)
                    Finpour
                Finpour
```

```
                A-attentei <- CRi+1
                Confpoti+1 <- Confpoti' + CPi+1
                (Areeli+1) <- Construction-réparation-individuelle(A-attentei)
                Retourner Ai+1 = (Areeli+1, Confpoti+1)
                Retourner Pi+1 = (Si, Oi+1, Mi+1, Ci+1)
            Sinon fail
            Fsi
        Fsi

Sinon
        Si r = red-new (precond, N) alors
            (V, P, effet-et) <- choisir-établisseur-new (precond, N)
            Si V alors
                pour tout Op de P.op
                    Si effet – et ∈ Op.add alors retourner Op Fsi
                Finpour
                Oi+1 <- Oi U (Op < N, begin < P. debut, P.fin < end)
                Si+1 <- Si U {P.op}
                Mi+1 <- Mi U {maintien(precond, Op, N), P. maintiens}
                Ci+1 <- Ci U {(precond = effet-et), P. contraintes}

                % detection des nouvelles preconditions non supportees
                pour tout Op de P.op
                    pour toute precond de Op.precond
                        Si maintien(precond, S, Op) ∉ Mi+1 alors
                            Si type(precond) = immuable alors
                                A <- Ai + (imm-usp(precond, Op),
                                        (red-prior(precond, Op))
                            sinon
                                Si type(precond) = hold alors
                                A <- Ai + (hold-usp(precond, Op),
                                        (red-prior(precond, Op),
                                        red-paral(precond, Op)))
                                Fsi
                            sinon A <- Ai + (usp(precond, Op)
                                        (red-prior(precond, Op),
                                        red-paral (precond, Op),
                                        red-new(precond, Op)))
                            Fsi
                        Fsi
                    Finpour
                Finpour

                (Conf-reel, Conf-poti') <- parcourir-conflit-potentiel (Conf-poti)

                % détection des nouveaux conflits
                % retourne les nouveaux conflits-reels CRi+1
                % et les nouveaux conflits potentiels CPi+1
                pour tout op de Si+1
                    pour tout effet de op.add
                        detecter-conflit-maintien-effet (maintien(precond, P, N), (effet, op)
                        pour tout maintien de P. maintien
                            detecter-conflit-maintien-effet(maintien, (effet, op))
                        Finpour
                    Finpour
                Finpour
```

```
                    pour tout maintien de Mi+1
                        pour tout maintien de P. maintien
                            detecter-conflit-maintien-maintien
                                (maintien,maintien(precond,P,N)) % algorithme 11
                        Finpour
                    Finpour
                    A-attentei <- CRi+1
                    Confpoti+1 <- Confpoti' + CPi+1
                    (Areeli+1) <- A + Construction-réparation-individuelle(A-attentei)
                    Retourner Ai+1 = (Areeli+1, Confpoti+1)
                    Retourner Pi+1 = (Si, Oi+1, Mi+1, Ci+1)
                Sinon fail
                Fsi
            Fsi
        Fsi
    Fin
```

La procédure ***choisir-selon-stratégie-1-défaut*** choisit selon la stratégie Strat le premier défaut répondant à cette stratégie.

La procédure ***choisir-etablisseur-prior*** recherche parmi les actions instantanées du plan une action située avant la précondition à établir, et possédant un effet susceptible de co-désigner avec la précondition. Elle retourne un triplet (vrai, effet, action) si elle trouve un établisseur potentiel, (faux, _, _) sinon.

La procédure ***choisir-etablisseur-paral*** recherche parmi les actions instantanées du plan une action non ordonnée avec l'action contenant la précondition à établir, et possédant un effet susceptible de co-désigner avec la précondition. Elle retourne un triplet (vrai, effet, action) si elle trouve un établisseur potentiel, (faux, _, _) sinon.

La procédure ***choisir-etablisseur-new*** recherche parmi les d-actions de Biblio une d-action possédant un effet primaire susceptible de co-désigner avec la précondition. Elle retourne un triplet (vrai, effet, d-action) si elle trouve un établisseur potentiel, (faux, _, _) sinon.

La procédure ***parcourir-conflit-potentiel*** parcourt l'ensemble des conflits potentiels mémorisés depuis l'étape précédente pour déterminer les nouveaux conflits réels et les conflits restant potentiels.

9.4 Conclusion

Nous avons consacré ce chapitre à la mise en oeuvre de notre approche. Nous avons rappelé la démarche que nous avons suivie durant ce travail de recherche, afin de mieux comprendre comment nous nous sommes positionnés, et quelles étaient nos intentions. Nous avons décrit les algorithmes de planification de TCLP.

Conclusion

Nous avons consacré les deux premiers chapitres de cette partie à l'étude des conflits dans TCLP. La gestion des conflits est un point crucial dans la construction des plans: c'est de la bonne détection des conflits que découle la correction des plans, et c'est de la stratégie de réparation des conflits que dépend en partie importante l'efficacité des algorithmes utilisés. Or notre approche consiste à traduire des relations temporelles disjonctives entre d-actions en conjonctions de conflits maintien-effet et maintien-maintien. Nous nous sommes donc penchés sur les stratégies de résolution des conflits, et avons expliqué dans cette partie comment nous réduisons l'engagement et le retour-arrière vis à vis du choix des réparations des conflits, et comment nous conservons les relations disjonctives.

Le dernier chapitre de cette partie concerne la description des algorithmes de planification utilisant les algorithmes de détection et de réparations des conflits décrits dans le chapitre 8. Nous y résumons également la démarche que nous avons suivie tout au long du travail. Elle peut être résumée et généralisée de la façon suivante:

- existence d'un problème et d'une solution partiellement satisfaisante
- étude des lacunes de l'approche
- étude des solutions existantes susceptibles de mieux répondre au problème: choix de la meilleure solution a priori
- évaluation "concrète" de la solution choisie: comment adapter cette solution pour mieux répondre aux nouveaux problèmes soulevés?
- extension de la nouvelle solution
- évaluation de la nouvelle approche
- comparaison à d'autres approches susceptibles de couvrir les nouveaux besoins; comparaison par rapport à la toute première solution, actuellement utilisée.

L'avant dernière étape a été partiellement réalisée sur des exemples simples. Pour mieux évaluer, d'un point de vue pratique, l'approche que nous avons proposée, il serait nécessaire de réaliser des tests issus de l'application d'origine, CARNEADE, comme ce qui a été fait avec IPEM.

La dernière étape nécessite la comparaison théorique de notre approche avec d'autres théories existantes telles que celles utilisées dans le planificateur temporel IxTeT, ou dans Descartes. Nous l'avons réalisée en ce qui concerne l'expressivité de notre approche; elle peut être également réalisée sur le plan de la complexité du traitement des informations.

Chapitre 10

Conclusion

10.1 Contributions

Confrontés aux besoins issus d'une application (le simulateur de combats CARNEADE), nous avons identifié la planification réactive comme une solution potentielle aux problèmes soulevés.

Les caractéristiques de l'application nous ont en outre amenés à choisir le planificateur IPEM pour ses capacités à entrelacer les étapes de planification et d'exécution des plans, tout en utilisant les techniques des planificateurs traditionnels non linéaires à liens causaux reconnus pour leur efficacité.

Nous avons ***implémenté et testé ce planificateur***. Cette étape a alors soulevé des problèmes de représentation du domaine (actions, règles du domaine, relations temporelles). Nous nous sommes alors concentrés sur les problèmes de construction des plans dans le cadre des planificateurs traditionnels non linéaires à liens causaux lorsque la modélisation des actions par des opérateurs de type STRIPS n'est plus suffisante pour décrire le domaine.

En effet, l'inadéquation du modèle STRIPS pour représenter des actions dont les effets se manisfestent uniquement pendant leur application et pour décrire des plans dont certaines actions s'entrelacent, nous a conduits à considérer la durée des actions. Nous avons alors construit les ***d-actions***. Une d-action est un ensemble partiellement ordonné d'opérateurs de type STRIPS situés entre un instant de début et un instant fin.

Nous avons également défini la notion de ***maintien*** pour décrire la persistance d'un fait sur un intervalle. Ce maintien nous autorise la description de conditions d'application d'action vraies sur un intervalle, d'effets maintenus sur un intervalle, et de contraintes sur des états entre deux instants (interdiction par exemple) dans le cadre des planificateurs traditionnels à liens causaux.

Dans le but de conserver les caractéristiques d'IPEM, nous avons réalisé ces extensions en respectant la structure des planificateurs traditionnels non linéaires à liens causaux.

Afin d'exprimer les relations temporelles entre des d-actions s'entrelaçant, nous nous sommes posés le problème de la description de ces relations.

Les d-actions possèdent une durée associable à un intervalle et décrite par un couple totalement ordonné d'instants. Traditionnellement, les relations temporelles entre deux intervalles sont décrites à l'aide de l'algèbre d'Allen. Cependant, si l'expressivité

de cette algèbre est considérable (il est effet possible de décrire toutes les relations possibles entre deux intervalles), la gestion de ces relations est trop complexe pour pouvoir être considérée comme efficace. De plus, nous désirons conserver les caractéristiques des planificateurs précédemment cités. Nous avons donc établi une ***table de transition*** entre un sous-ensemble des relations d'Allen (toutes les relations ne faisant pas intervenir l'égalité entre instants) et le formalisme temporel lié à la structure des planificateurs traditionnels non linéaires à liens causaux. Cette table traduit chacune des 64 relations temporelles disjonctives d'Allen issues des primitives <, >, d, di, o et oi, en conjonction de relations de précédence entre instants, conflits maintien-effet et conflits maintien-maintien. Les réparations disjonctives associées à ces deux derniers conflits traduisent les relations disjonctives entre d-actions.

La complexité de la gestion d'un ensemble de relations temporelles est donc reportée sur celle d'un ensemble de conflits. La troisième partie du travail a donc été consacrée à l'amélioration des ***stratégies de gestion des conflits***.

Dans le but de diminuer les points de choix vis à vis des réparations associées aux conflits, nous avons tout d'abord spécifié les conflits afin de simplifier leur réparation, puis utilisé une approche globale de résolution des conflits plutôt qu'incrémentale. Afin de conserver les disjonctions temporelles (représentées par des réparations disjonctives) nous employons le moindre engagement sur ces réparations disjonctives: ne pas faire de choix tant qu'il n'est pas obligatoire. Nous pensons en outre, que le nombre de réparations disjonctives est moindre devant le nombre de conflits générés. Ce qui nous permet de conserver une complexité acceptable vis à vis de celle liée à l'algèbre d'Allen.

Nous avons implémenté cette approche et construit ***le planificateur TCLP***, en utilisant des techniques de planification traditionnelle non linéaire et à liens causaux et en insérant la table de transition entre la description du domaine réalisée dans un langage de haut niveau et le formalisme de ces planificateurs.

Cette approche nous a permis de répondre aux besoins soulevés par la première série de tests réalisée avec IPEM sur des jeux d'essais issus du simulateur CARNEADE. Elle peut cependant être complétée suivant diverses directions.

10.2 Travaux futurs

La suite immédiate de ces travaux consisterait à évaluer de façon pratique et théorique notre approche. De façon pratique en réalisant une nouvelle série de tests issus de CARNEADE, de façon théorique, en essayant de formaliser le problème de la gestion des relations temporelles dans TCLP, et en essayant de traduire notre problème dans le formalisme de l'algèbre d'Allen et de planificateurs utilisant l'algèbre d'instants.

Une première extension possible consisterait à étudier la description de l'égalité à l'aide d'une représentation granulaire du temps, comme celle que nous avons évoquée dans le chapitre 6. En effet, si l'égalité peut être jugée irréalisable dans la pratique, elle permet quand même de différencier des événements consécutifs proches d'événements en séquence, et donc de synchroniser des événements de façon plus précise qu'avec la

seule relation de précédence.

Une autre voie pourrait être consacrée à la réintégration des caractéristiques réactives d'IPEM dans TCLP, puisque le second a été construit sur les bases du premier. Cette extension doit cependant tenir compte des aspects d'implémentation liés à la partie réactive, que nous avons soulevés dans le chapitre 9.

Afin de faire face à des problèmes de ressource émanant classiquement des problèmes réels (par opposition à jouet), il serait intéressant de pouvoir attribuer une durée numérique aux d-actions et au plan. Le problème consisterait alors à étudier le couplage entre notre approche et la gestion de contraintes numériques.

Durant la phase d'implémentation et de tests, nous avons régulièrement été confrontés à deux problèmes se recoupant: comment imposer au planificateur des contraintes traduisant des règles de comportement obligatoires issues du domaine? Comment traduire l'impossibilité pour le planificateur de faire et défaire x fois le même but (problème du bouclage à l'infini)?

Ces deux problèmes nécessitent l'élicitation de règles du domaine se traduisant soit sur la structure des actions, soit sur la structure des plans. Il semble en outre que le contrôle de la génération des plans soit un point crucial quant à l'utilisation des techniques de planification dans le cadre d'applications réelles, nécessitant des algorithmes efficaces.

Références

[ALL 83a] J.F. Allen. "Maintaining knowledge about Temporal Intervals". *Communications of the ACM*, vol. 26 num 11, pp832-843, 1983.

[ALL 83b] J.F. Allen & J.A. Koomen. "Planning using a Temporal World Model", In *Proceedings of the Eight International Joint Conference on Artificial Intelligence*, 1983, pp741-747.

[ALL 84] J.F. Allen. " Towards a general Theory of Action and Time", *Artificial Intelligence* 23, pp 123-154, 1984.

[ALL 91] J.F. Allen. "Temporal reasoning and planning". In J.Allen, H. Kautz, R. Pelavin and J. Tenenberg, editors, *Reasoning about Plans, chapter 1*, Morgan Kaufmann, San Mateo, 1991.

[AMB 87] J. A. Ambros Ingerson. "IPEM: Integrated Planning, Execution and Monitoring", *Master of Philosophy*, Department of Computer Science, Unversity of Essex, 1987.

[AMB 88] J. A. Ambros Ingerson & S. Steel. "Integrating Planning, Execution and Monitoring", *In proceedings of AAAI'88*, pp 83-88, 1988.

[BAR 93] A. Barrett & D.S. Weld. "Characterizing Subgoals Interactions for Planning", *In Proceedings of IJCAI'93*, pp 1388-1393, 1993.

[BAR 94] A. Barrett & D. S. Weld. "Partial-order planning: evaluating possible efficiency gains", *Artificial Intelligence* 67 (1), May 1994.

[BAS 95] C. Bastié & P. Régnier. "Planning and execution in a dynamic environment: the supervision of execution in SPEEDY", *In proceedings of the 14th UK Workshop on planing and Scheduling*, Colchester (UK), Novembre 1995.

[CHA 87] D. Chapman. "Planning for conjunctive goals", *Artificial Intelligence* 32, pp333-377, 1987.

[CUR 91] K. Currie & A. Tate. "O-Plan: the open planning architecture", *Artificial Intelligence* 52, pp 49-86, 1991

[DEA 87] T. Dean & D. McDermott. "Temporal Data Base Management", *Artificial Intelligence* 32, pp1-55, 1987.

[EUZ 93] J. Euzénat. "Représentation granulaire du temps", *Revue d'Intelligence*

Artificielle, volume 7 (3), pp 329-361, 1993.

[FIK 88] R.E. Fikes & N.J. Nilsson. "STRIPS: a new approach to the application of theorem proving to problem solving". *Artificial Intelligence* 35, pp165-195, 1988.

[FIN 93] E. Fink & Q. Yang. "Characterizing and Automatically finding primary effects in Planning". *In Proceedings of IJCAI'93*, pp 1374-1379.

[FIN 95] E. Fink & Q. Yang. "Planning with primary effects: Experiments and Analysis", *In proceedings of IJCAI'95*, pp 1606-1611, 1995.

[GAR 93] F. Garcia. Révision des croyances et du raisonnement pour la planification", *Thèse de l'Ecole Nationale Supérieure de l'Aéronautique et de l'Espace*, Février 1993.

[GEO 87] M.P. Georgeff & Amy L. Lansky. "Reactive Reasoning and Planning", *In Proceedings of AAAI'87*, pp 677-682, Seattle, WA, 1987.

[GHA 89] M.Ghallab & A.M. Alaoui. "Relations temporelles symboliques: représentations et algorithmes", *Revue d'Intelligence Artificielle*, vol 3, pp67-115.

[GHA 94] M. Ghallab & H. Laruelle. "Representation and Control in IxTeT, a temporal planner". *In Proceedings of AIPS* 1994.

[GIN 88] M. Ginsberg & D.E. Smith. "Reasoning about Action II: The qualification problem", *Artificial Intelligence 35*, pp 311-342, 1988.

[HER 89] J. Hertzberg & A. Horz. "Towards a theory of Conflict Detection and resolution in Nonlinear Plans", *In proceedings of IJCAI'89*, pp937-942, 1989.

[HOR 94] A. Horz. "Relating Classical and Temporal Planning", *In Current Trends in AI Planning*, Eds C. Backstrom & E. Sandewall, IOS Press, pp 145-157,1994.

[ING 92] F. Ingrand, M.P. Georgeff & A.S Rao. "An architecture for Real-Time Reasoning and System Control", *IEEE Expert* , pp34-44, Dec 1992.

[JAC 90] E. Jacopin & J.F. Puget. "From TWEAK to PWEAK: Reducing the nonlinear planning framework", *Rapport technique LAFORIA 90/29*, Université de Paris, 1990.

[JOS 95] D. Joslin. "Passive and Active Decision Postponement in Plan Generation", *PhDissertation*, University of Pittsburg, Sept 1995.

[JOS 96] D. Joslin & M.E. Pollack. "Passive and Active Decision Postponement in

Plan Generation", In proceedings of EWSP'95, *New Directions in AI Planning*, Eds M. Ghallab & A. Milani, pp 37-48.

[KAM 92] S. Kambhampati. "Characterizing Multi-contributor Causal Structures for Planning", *In proceedings of AIPS'92*, 1992.

[KAM 93] S. Kambhampati. "On the utility of systematicity: understanding the tradeoffs between redundancy and commitment in partial-order planning", *In proceedings of Spring Symposium AAAI*, 1993.

[KAM 95] S.Kambhampati, C. Knoblock & Q. Yang. "Planning as refinement search: a unified framework for evaluating design tradeoffs in partial order planning", *Artificial Intelligence Special issue on Planning and Scheduling*, vol 76, 1995.

[KAM 96a] S. Kambhampati. "Using Disjunctive orderings instead of Conflict Resolution in Partial order Planning", *Technical Report of Arizona State University*, ASU CSE TR 96-002, Janvier 1996.

[KAM 96b] S. Kambhampati & X. Yang. "On the role of disjunctive representations and constraint propagation in refinement planning", *In proceedings of KR'96*, http://rakaposhi.eas.asu.edu/yocham.html.

[KNO 94] C.A. Knoblock. "Generating Parallel Execution Plans with a Partial-Order Planner", *In Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*, Morgan Kaufamnn, San Mateo, 1994.

[KNO 96] C. Knoblock et al. "AI planning systems in the real world" , *IEEE Expert,* pp4-12, Dec 1996.

[KNO ..] C. Knoblock & Q. Yang. "Relating the performance of partial-order planning algorithms to domain Features", *Sigart Bulletin* vol 6 num 1.

[LAR 94] H. Laruelle, "Planification temporelle et exécution de tâches en robotique", *Thèse, LAAS*, Toulouse, 1994.

[LIF 86] V. Lifschitz. " On the sematics of STRIPS". *In Readings in Planning*, Eds by J.Allen, J.Hendler & A. Tate, Morgan Kaufmann, pp 523-531, 1996.

[MAR 92] F. Martelle & E. Lemoine. "Modélisation de la prise de décision dans un jeu de guerre", *Journées Sciences et Défense 92*, Paris, 1992.

[MCA 91] D. McAllester & D. Rosenblitt. "Systematic Nonlinear Planning", *In Proceedings of AAAI 1991*, pp634-639

[MCC 69] J. McCarthy & P.J.Hayes. " Some philosophical Problems from the standpoint of Artificial Intelligence", *Machine Intelligence,* vol 4, pp463-502.

[MCD 82] D. McDermott. "A temporal logic for reasoning about Processes and Plans". *Cognitive Science* 6, pp101-155, 1982.

[MIN 91] S. Minton, J. Bresina & M. Drummond. "Commitment Strategies in Planning: a comparative analysis", *In the proceedings of IJCAI'91*, pp259-264, 1991.

[PEN 92] J.S. Penberthy & D.S. Weld. "UCPOP: A sound, complete, partial order planner for ADL". *In proceedings of Knowledge Representation and Reasoning (KR 92),* 1992.

[PEN 94] J.S. Penberthy & D.S. Weld. "Temporal planning with continuous change", *In proceedings of AAAI*, 1994.

[PEO 93] M.A. Peot & D. E. Smith. "Threat-removal Strategies for Partial Order Planning", *In proceedings of AAAI'93*, pp492-499, 1993.

[POL 97] M.E. Pollack, D. Joslin & M. Paolucci. "Flaw Selection Strategies for Partial-Order Planning", *In Journal of Artificial Intelligence Research (JAIR) 6*, pp 223-262, Juin 1997.

[REG 91] P.Régnier & B. Fade. "Complete determination of parallel actions and temporal optimization in linear plans of actions". *In proceedings of EWSP'91*, Sankt Augustin, Germany.

[REV 96] J. Révault. "Une modélisation par le graphe de la relation meet pour traiter des contraintes temporelles exprimées à l'aide d'intervalles", *Thèse de L'université de Nantes*, Novembre 1996.

[RUT 93] E. Rutten & J. Hertzberg. "Temporal Planner = Nonlinear planner + Time Map manager". *AI Communication*, vol 6 num 1, Mai 1993.

[SAC 75] E.D. Sacerdoti. "The Nonlinear Nature of Plans", *In proceedings of IJCAI'75*, pp206-214, 1975.

[SAL 93] M.Salisbury & H. Tallis. "Automated Planning and Replanning for Battlefield Simulation", *MITRE Corporation*, 1993.

[SHO 87] Y. Shoham. "Temporal Logics in AI: Semantical and Ontological Considerations", *Artificial Intelligence 33*, pp89-104, 1987.

[SMI 93] D.E. Smith & M.A. Peot. "Postponing Threats in Partial Order planning", *In proccedings of AAAI'93*, pp500-506, 1993.

[SMI 94] D.E. Smith & M.A. Peot. "A Note on the Dmin Strategy", 2 pages, Janvier 1994.

[TSA 86] E.P.K. Tsang. "Plan generation in temporal frame", *In proceedings of*

ECAI'86, pp 479-493, 1986

[TSA 87] E.P.K. Tsang. "Time Structures for AI", *In proceedings of IJCAI'87*, Eds J. McDermott, MILAN, pp456-461, 1987

[TAT 77] A. Tate. "Generating Project Networks", *In proceedings of IJCAI'77*, pp888-893, 1977.

[VBE 90] P. Van Beek & R. Cohen. "Exact and approximate reasoning about Temporal Relations", Computational Intelligence num 6, pp132-144, 1990.

[VBE 92] P.Van Beek. "Reasoning about qualitative Temporal Information", *Artificial Intelligence vol 58*, pp 297-326, 1992.

[VEL 90] M.M. Veloso, M.A. Perez & J.G. Carbonell. "Nonlinear planning with parallel resource allocation". In *WS. Innovative Approaches to Planning, Scheduling and Control*, pp207-212, San Diego, CA, USA, Nov. 1990.

[VER 83] S.A. Vere. "Planning in time: windows and durations for activities and goals". *IEEE Transactions on pattern analysis and machine intelligence*, Mai 1983.

[VIL 82] M. Vilain. "A system for reasoning about time", *In Proceedings of AAAI'82*, Pittsburg, 1982.

[VIL 86] M.Vilain & H. Kautz. "Constraint propagation Algorithms for temporal reasoning", *In the Proceedings of AAAI'86*, Philadelphia (US), 1986.

[WEL 94] D.S. Weld. "An introduction to least commitment planning", *In Artificial Intelligence Magazine*, Winter 1994.

[WIL 88] D. Wilkins. "Practical Planning: Extending the classical AI planning paradigm", Eds Morgan Kaufmann, 1988

[WIL 90] D.E. Wilkins. "Can AI planners solve practical problems?", *Computational Intelligence* num 6(4), pp 232-246, 1990.

[WIL 94] D.E Wilkins & R.V. Desimone. "Applying an AI Planner to Military Operations Planning", *Intelligent Scheduling*, Monte Zweben and Mark Fox Editirs, Morgan Kaufman, San Mateo, CA 1994.

[WOZ 95] M. Wozniak & P. Taillibert. "Reactive planning for battlefield simulation", *In proceedings of the 14th UK Workshop on planing and Scheduling*, Colchester (UK), Novembre 1995.

[WOZ 97] M.Wozniak, P. Taillibert & P. Vanheeghe. "TCLP: a temporal causal link planner with disjunctive temporal relations", *In the proceedings of the 16th Workshop of the UK planning and scheduling SIG*, Durham (UK) ,

Décembre 1997.

[WOZ 98] M.Wozniak, P. Taillibert & P. Vanheeghe. "Disjonctions Temporelles dans les planificateurs à liens causaux". *In the proceedings of RFIA'98*, Clermont-Ferrand, Janvier 1998.

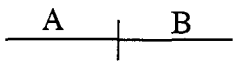
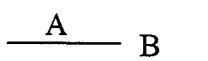
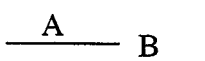
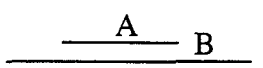
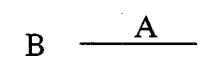
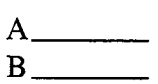
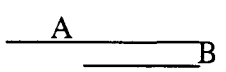
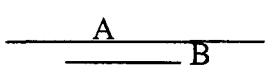
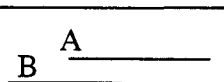
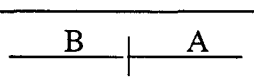
[YAN 90] Q. Yang. "An algebraic Approach to conflict resolution in planning", *In the Proccedings of AAAI'90.*

[YAN 91] Q. Yang. "A Theory of conflict resolution in planning", *In Artificial Intelligence 58,* pp 361-392, 1991.

ANNEXES

Annexe A: Algèbre d'Allen

A.1: Primitives temporelles

Relation	Nom	Abréviation
	A est avant B	A < B
	A rencontre B	A m B
	A chevauche B	A o B
	A débute B	A s B
	A est pendant B	A d B
	A termine B	A f B
	A est égal à B	A = B
	A est terminé par B	A fi B
	A contient B	A di B
	A est débuté par B	A si B
	A est chevauché par B	A oi B
	A est rencontré par B	A mi B
	A est après B	A > B

A.2: Table de transitivité (sans l'égalité entre intervalles)

A r1 B ∧ B r2 C → A r3 C ; dur = (d, s, f); con = (di, si, fi)

	<	>	d	di	o	oi	m	mi	s	si	f	fi
<	<	no info	< o m d s	<	<	< o m d s	<	< o m d s	<	<	< o m d s	s
>	no info	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	>	>
d	<	>	d	no info	< o m d s	> oi mi d f	<	>	d	> oi mi d f	d	< o m d s
di	<o m di fi	> oi di mi si	o oi dur con =	di	o di fi	oi di si	o di fi	oi di si	di fi o	di	di si oi	di
o	<	> oi di mi si	o d s	< o m di fi	< o m	o oi dur con =	<	oi di si	o	di fi o	d s o	< o m
oi	< o m di fi	>	oi d f	> oi mi di si	o oi dur con =	> oi mi	o di fi	>	oi d f	oi > mi	oi	oi di si
m	<	> oi mi di si	o d s	<	<	o d s	<	f fi =	m	m	d s o	<
mi	< o m di fi	>	oi d f	>	oi d f	>	s si =	>	d f oi	>	mi	mi
s	<	>	d	< o m di fi	< o m	oi d f	<	mi	s	s si =	d	< m o
si	< o m di fi	>	oi d f	di	o di fi	oi	o di fi	mi	s si =	si	oi	di
f	<	>	d	> oi mi di si	o d s	> oi mi	m	>	d	> oi mi	f	f fi =
fi	<	> oi mi di si	o d s	di	o	oi di si	m	si oi di	o	di	f fi =	fi

Annexe B: 4 éléments de comparaison

Cette annexe résume 4 exemples qui nous servent lors de la comparaison:

- de la capacité d'expression des modèles d'action utilisés dans différents planificateurs
- de la gestion des contraintes temporelles sous-jacente.

B.1:

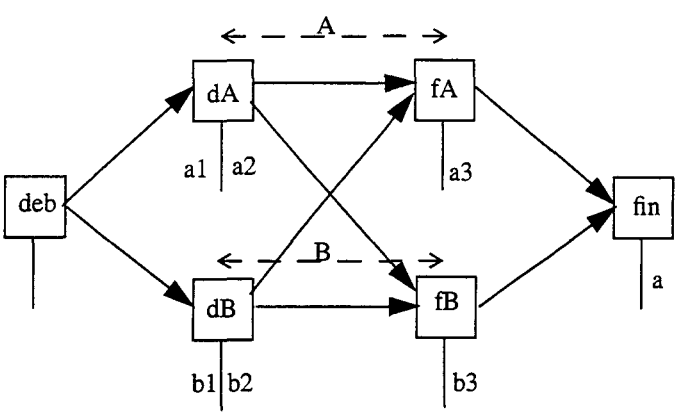
Exemple de macro action composée de deux d-actions A et B

Soit une macro-action quelconque composée de deux d-actions A et B telles que A et B se déroulent sur un intervalle de temps commun.

Cette contrainte se traduit par:

- A d di o oi B dans l'algèbre d'Allen (si les d-actions sont confondues avec les intervalles correspondants)
- dA<fB $\wedge$ dB<fA dans l'algèbre d'instants si (dA,fA) et (dB,fB) sont les couples d'instants représentant le début et la fin de chaque intervalle.

Nous donnons ici la représentation graphique TCLP d'une telle macro-action.



Il est possible de décrire de façon hiérarchique cette action, en décrivant séparément a, b puis la macro-action.

```
macro_action::      liste_variables      []
                    composantes
                          ( DEBUT, instant,    precond      []
                                               effets       add    [ ]
                                                            del    []
                                               contraintes  true
                          ) &
                          (A, a, []
                          ) &
                          ( B, b, []
                          ) &
                          ( FIN, instant,      precond      [ ]
                                               effets       add    a
                                                            del    [ ]
                                               contraintes  true
                          )
                    index                index(FIN)
                    contraintes_variables      true
                    contraintes _temporelles        A d di o oi B
                    debut                DEBUT
                    fin                  FIN
                    maintien             [].
```

avec les descriptions de a et b:

```
a          ::       liste_variables      []
                    composantes
                          ( DEBUT, instant,    precond      a1
                                               effets       add    a2
                                                            del    []
                                               contraintes  true
                          ) &
                          ( FIN, instant,      precond      [ ]
                                               effets       add    a3
                                                            del    [ ]
                                               contraintes  true
                          )
                    index                index(FIN)
                    contraintes_variables      true
                    contraintes _temporelles        true
                    debut                DEBUT
                    fin                  FIN
                    maintien             [].
```

```
b       ::      liste_variables     []
                composantes
                        ( DEBUT, instant,   precond     b1
                                            effets      add     b2
                                                        del     []
                                            contraintes true
                        ) &
                        ( FIN, instant,     precond     [ ]
                                            effets      add     b3
                                                        del     [ ]
                                            contraintes true
                        )
                index           index(FIN)
                contraintes_variables       true
                contraintes _temporelles        true
                debut           DEBUT
                fin             FIN
                maintien        [].
```

B.2:

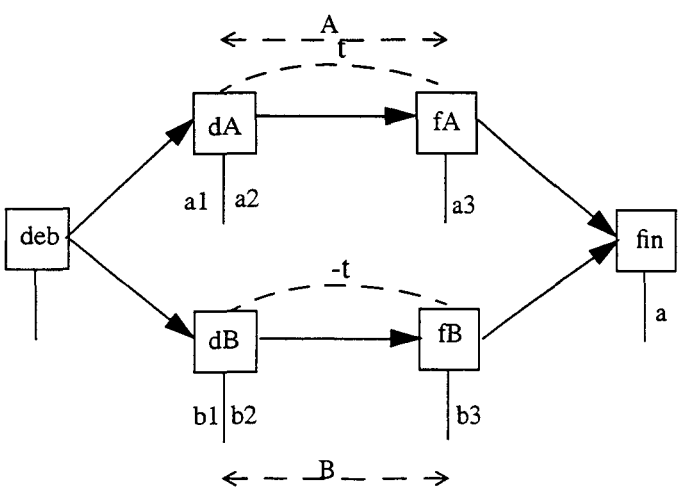
Appeler les secours

Appeler les secours nécessite d'appeler les pompiers (action A) et d'appeler la police (action B). Il est impossible de réaliser les deux en même temps car une seule peut le faire. Cependant l'ordre des appels n'est pas spécifié à l'avance et dépend de l'incident identifié. S'il s'agit d'un accident de la route il est nécessaire d'appeler d'abord la police. S'il s'agit d'un incendie, il est plus judicieux d'appeler les pompiers en premier. L'ordre est déterminé par le planificateur. Cette contrainte est traduite par:

- A <> B dans l'algèbre d'Allen.

Elle ne peut être traduite dans l'algèbre d'instants car elle fait intervenir une disjonction de relations entre 4 instants différents: $dA < fA \wedge dB < fB \wedge (fB < dA \vee fA < dB)$

La description graphique TCLP est la suivante:



La description de cette action est donnée ci-dessous; les composantes de cette actions sont les mêmes que pour le cas 1 (actions a et b). Seule la contrainte temporelle entre a et b varie.

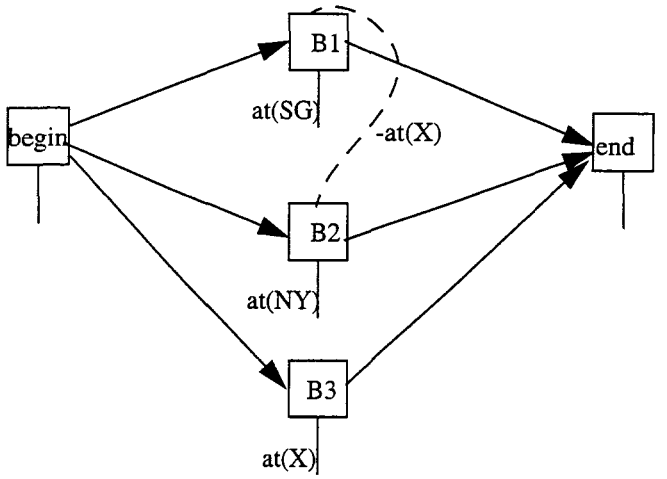
```
macro_action::      liste_variables      []
                    composantes
                          ( DEBUT, instant,    precond      []
                                               effets       add   [ ]
                                                            del   []
                                               contraintes true
                          ) &
                          (A, a, []
                          ) &
                          ( B, b, []
                          ) &
                          ( FIN, instant,      precond      [ ]
                                               effets       add   a
                                                            del   [ ]
                                               contraintes true
                          )
                    index                index(FIN)
                    contraintes_variables      true
                    contraintes _temporelles        (A < > B)
                    debut                DEBUT
                    fin                  FIN
                    maintien             [].
```

B.3:
Contraintes externes

- **Le voyage**

Je planifie un voyage, dois aller en X et pose des contraintes supplémentaires sur le moment où je dois y aller. Je suis à Paris. Je dois aller à NY et à SG; je dois aller en X, mais il est impossible d'y aller pendant le trajet entre les villes NY et SG. Il n'y a pas de

contrainte sur l'ordre dans lequel je visite les villes.



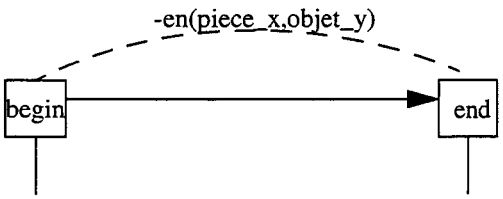
La traduction TCLP de cette contrainte est un maintien entre B1 et B2 empêchant at(X) d'être réalisé entre les deux.

Je peux aller en X avant de me rendre dans les deux villes ou aller en X après m'y être rendue. Je ne peux pas y aller entre les deux. La description de ce plan initial suit la même syntaxe que toute autre action. Elle est la suivante:

```
voyage  ::      liste_variables     []
                composantes
                        ( BEGIN, instant,   precond     libre(X) et
                                                        (sur(X,Y), h)
                                            effets      add     [ ]
                                                        del     libre(X)
                                            contraintes (dif(X,Y)) ) &
                        ( B1, instant,      precond     at(sg)
                                            effets      add     []
                                                        del     []
                                            contraintes true ) &
                        ( B2, instant,      precond     at(ny)
                                            effets      add     []
                                                        del     []
                                            contraintes true ) &
                        ( B3, instant,      precond     at(x)
                                            effets      add     []
                                                        del     []
                                            contraintes true ) &
                        ( END, instant,     precond     [ ]
                                            effets      add     []
                                                        del     [ ]
                                            contraintes true )
                index           [ ]
                contraintes_variables       true
                contraintes _temporelles            BEGIN av END
                debut           BEGIN
                fin             END
                maintien        (-at(x), B1, B2) .
```

- **Interdiction d'obtenir un état particulier**

Un sinistre a lieu dans une entreprise et interdit l'utilisation d'une pièce pour déposer des objets dangereux: il est interdit d'obtenir l'effet en(piece_x, objet_y) avec type(objet_y, dangereux). Toutes les actions instanciées de cette façon sont interdites dans le plan.



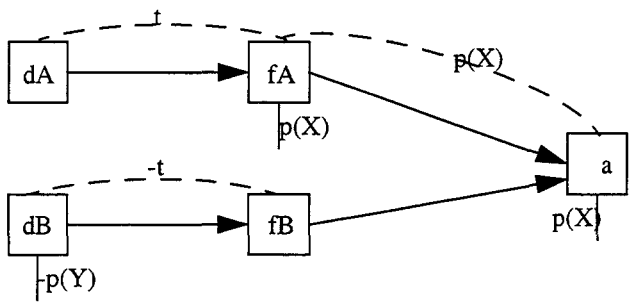
La description TCLP est la suivante:

```
voyage ::        liste_variables      []
                 composantes
                       ( BEGIN, instant,   precond      []
                                           effets       add    _
                                                        del    []
                                           contraintes true
                       ) &
                       ( END, instant,     precond      _
                                           effets       add    []
                                                        del    [ ]
                                           contraintes true
                       )
                 index                 []
                 contraintes_variables       true
                 contraintes _temporelles          BEGIN av END
                 debut                 BEGIN
                 fin                   END
                 maintien              (-en(piece_x,objet_y), BEGIN, END) .
```

B.4:
Résolution incrémentale d'un ensemble de conflits dans un plan partiel

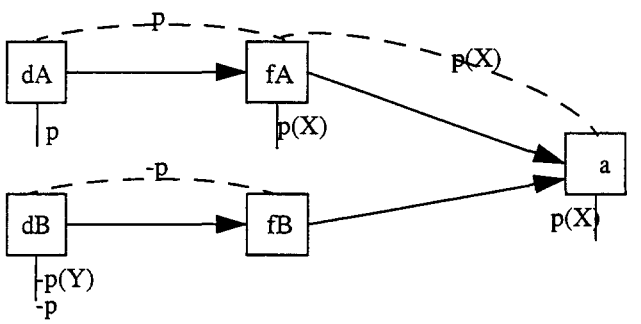
Nous décrivons ici plusieurs plans partiels TCLP contenant un ou plusieurs conflits réels ou potentiels. Ils sont destinés à dégager les principes de résolution des conflits dans les divers planificateurs considérés.

- **Plan partiel avec conflit réel entre deux maintiens avec deux réparations possibles et un conflit potentiel à une seule réparation possible.**

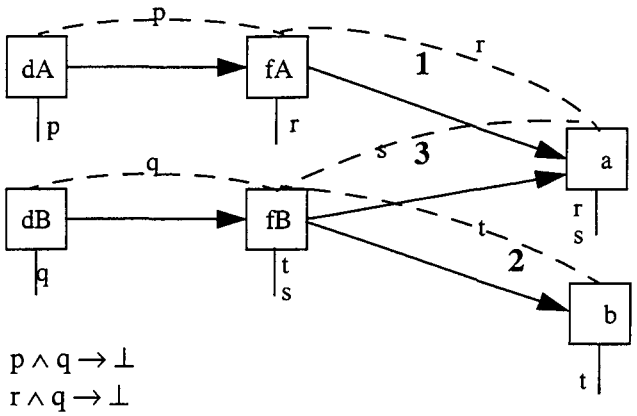


Tant que $X \neq Y$ pas de conflit entre le maintien (pX), fA, a) et l'effet -p(Y) de l'action dB.
Conflit entre les maintiens (t,dA,fA) et (-t,dB,fB); deux solutions possibles: fB < dA ou fA < dB.

- **Mêmes conflits mais différence entre un effet établi en un instant, et un effet établi en un instant et maintenu sur un intervalle de temps.**



- **Même structure, mais uniquement des conflits réels avec deux réparations possibles**



- Les contraintes liées au début et à la fin des actions sont:
  dA < fA
  dB < fB
- Etablissement de r par A (1) => fA < a
- Etablissement de t par B (2) => fB < b
  et création de conflits:
  - conflit 1 entre A et B avec $p \wedge q \rightarrow \perp$ => fB < dA ∨ fA < dB
  - conflit 2 entre [fA,a] et B avec $r \wedge q \rightarrow \perp$ => fB < fA ∨ a < dB
- Etablissement de s par B (3) => fB < a

Le dernier établissement impose une unique solution pour résoudre l'ensemble des conflits présents dans le plan partiel: B < A < a.

Annexe C: Disjonctions d'Allen sans égalité

C.1: Les 6 primitives ne faisant pas intervenir l'égalité entre instants

Intervalles	Instants
X d Y	$y^- < x^- \wedge x^+ < y^+$
X di Y	$x^- < y^- \wedge y^+ < x^+$
X < Y	$x^+ < y^-$
X > Y	$y^+ < x^-$
X o Y	$x^- < y^- \wedge y^- < x^+ \wedge x^+ < y^+$
X oi Y	$y^- < x^- \wedge x^- < y^+ \wedge y^+ < x^+$

C.2: Liste des conjonctions d'instances de schémas cohérentes

Ce tableau représentent toutes les conjonctions d'instances de schémas temporels cohérentes.

Les ',' séparant les schémas instanciés correspondent à la conjonction.

> di oi	(p2(x-,y-,y+), p2(x+,y-,y+))
< > o	(p2(y+,x-,x+), p2(x-,y-,y+))
< > oi	(p2(y-,x-,x+), p2(x+,y-,y+))
< > d	(p2(y-,x-,x+), p2(y+,x-,x+))
> di	(p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(y+,x+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(y-,x+), p1(y+,x+), p2(x-,y-,y+), p2(x+,y-,y+))
> oi	(p1(y+,x+), p2(y-,x-,x+)) ***ou*** (p1(y-,x-), p2(x+,y-,y+)) ***ou*** (p1(y-,x-), p1(y+,x+)) ***ou*** (p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x-), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x-), p1(y+,x+), p2(y-,x-,x+)), (p1(y-,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y+,x+), p2(y-,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(y+,x+)) ***ou*** (p1(y-,x-), p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y+,x+), p2(y-,x-,x+), p1(x+,y-)), (p1(y-,x+), p1(y-,x-), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(y-,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+))

Tableau 17 :

< o	(p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y-), p2(y+,x-,x+)) ***ou*** (p1(x-,y-), p1(x+,y+)) ***ou*** (p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y-), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y-), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y-), p1(x+,y+), p2(y+,x-,x+)), (p1(x-,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(x+,y+)) ***ou*** (p1(x-,y-), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)), (p1(x-,y+), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+))
< d	(p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+))
> d	(p1(y-,x-), p2(y+,x-,x+)) ***ou*** (p1(y-,x-), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p2(y-,x-,x+), p2(y+,x-,x+))
d	(p1(y-,x-), p1(x+,y+)) ***ou*** (p1(y-,x-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(y-,x-), p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(y-,x+), p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(x+,y+)) ***ou*** (p1(x-,y+), p1(y-,x-), p2(y+,x-,x+)) ***ou*** (p1(y-,x-), p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x-), p2(y-,x-,x+), p2(y+,x-,x+)), (p1(x-,y+), p1(y-,x-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x-), p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p2(y+,x-,x+)), (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(x+,y+)) ***ou*** (p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x-), p1(x+,y+), p2(y-,x-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+), p2(y-,x-,x+)), (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(x+,y+), p2(y-,x-,x+))
< di	(p1(x-,y-), p2(x+,y-,y+)) ***ou*** (p1(x-,y-), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p2(x-,y-,y+), p2(x+,y-,y+))
di	(p1(x-,y-), p1(y+,x+)) ***ou*** (p1(x-,y-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y-), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(y+,x+)) ***ou*** (p1(x-,y-), p1(y+,x+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y+,x+), p2(x-,y-,y+), p2(x+,y-,y+)), (p1(x-,y+), p1(x-,y-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(x-,y-), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y+,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(y+,x+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(y+,x+), p2(x-,y-,y+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p2(x-,y-,y+), p2(x+,y-,y+)), (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(y+,x+), p2(x-,y-,y+))
> di o	(p1(y-,x+), p2(x-,y-,y+))
> d o	(p1(y-,x+), p2(y+,x-,x+))

Tableau 17 :

> o	(p1(y-,x+), p2(y+,x-,x+), p2(x-,y-,y+))
d o	(p1(y-,x+), p1(x+,y+)) ***ou*** (p1(y-,x+), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+), p2(y+,x-,x+))
o	(p1(y-,x+), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p2(y+,x-,x+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(x+,y+)) ***ou*** (p1(y-,x+), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou***(p1(y-,x+), p1(x-,y-), p2(y+,x-,x+), p2(x-,y-,y+)), (p1(y-,x+), p1(x-,y-), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(x+,y+), p2(y+,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p2(y+,x-,x+)), (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(x+,y+)) ***ou*** (p1(y-,x+), p1(x-,y-), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x+,y+), p2(y+,x-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p2(y+,x-,x+), p2(x-,y-,y+)), (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(x+,y+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p1(x+,y+), p2(y+,x-,x+))
di o	(p1(y-,x+), p1(x-,y-)) ***ou*** (p1(y-,x+), p1(x-,y-), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(x-,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(x-,y-), p2(x-,y-,y+))
< di oi	(p1(x-,y+), p2(x+,y-,y+))
< d oi	(p1(x-,y+), p2(y-,x-,x+))
< oi	(p1(x-,y+), p2(y-,x-,x+), p2(x+,y-,y+))
di oi	(p1(x-,y+), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y+,x+), p2(x+,y-,y+))
oi	(p1(x-,y+), p1(y+,x+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x-), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x-), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x-), p2(y-,x-,x+), p2(x+,y-,y+)), (p1(x-,y+), p1(y-,x-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y+,x+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(y-,x-,x+)), (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(y+,x+)) ***ou*** (p1(x-,y+), p1(y-,x-), p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y+,x+), p2(y-,x-,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p2(y-,x-,x+), p2(x+,y-,y+)), (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(x+,y-,y+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p1(y+,x+), p2(y-,x-,x+))
d oi	(p1(x-,y+), p1(y-,x-)) ***ou*** (p1(x-,y+), p1(y-,x-), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p2(y-,x-,x+)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-)) ***ou*** (p1(x-,y+), p1(y-,x+), p1(y-,x-), p2(y-,x-,x+))
d di o oi	(p1(x-,y+), p1(y-,x+))

Tableau 17 :

C.3: Traduction des disjonctions d'Allen sans l'égalité en conjonctions de p1, p2 et p3

On suppose que $p1(x^-,x^+)$ et $p1(y^-,y^+)$ sont toujours vrais.

Les opérateurs instantanés $\alpha 1$ et $\alpha 2$ sont insérés entre les débuts et fins des actions X et Y pour bénéficier de la transitivité des relations temporelles.

Les relations temporelles sont classées par ordre alphabétique des abréviations

utilisées. C'est-à-dire: > (ap), < (av), d, di, o, oi.

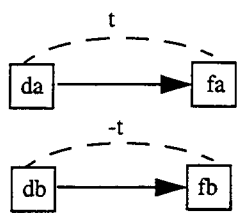
X r Y	Conjonctions de schémas
>	$p1(y^+,x^-)$
<>	$p3(x^+, x^-, y^-, y^+)$
<> d	$p2(y^-,x^-,x^+)$, $p2(y^+,x^-,x^+)$
<> d di	$p1(x^-,\alpha2)$, $p1(y^-,\alpha1)$, $p2(\alpha2,y^-,x^+)$, $p2(\alpha1,x^-,y^+)$, $p2(x^+,y^-,\alpha1)$, $p2(y^+,x^-,\alpha2)$
<> d di o	$p1(x^-,\alpha2)$, $p1(y^-,\alpha1)$, $p2(\alpha2,y^-,x^+)$, $p2(\alpha1,x^-,y^+)$, $p2(y^+,x^-,\alpha1)$
<> d di o oi	aucune contrainte entre X et Y
<> d di oi	$p1(x^-,\alpha2)$, $p1(y^-,\alpha1)$, $p2(\alpha2,y^-,x^+)$, $p2(\alpha1,x^-,y^+)$, $p2(x^+,y^-,\alpha1)$
<> d o	$p2(y^+,x^-,x^+)$
<> d o oi	$p1(y^-,\alpha2)$, $p1(\alpha2,y^+)$, $p2(\alpha2,x^-,x^+)$
<> oi d	$p2(y^-,x^-,x^+)$
<> di	$p2(x^-,y^-,y^+)$, $p2(x^+,y^-,y^+)$
<> di o	$p2(x^-,y^-,y^+)$
<> di o oi	$p1(x^-,\alpha1)$, $p1(\alpha1,x^+)$, $p2(\alpha1,y^-,y^+)$
<> oi di	$p2(x^+,y^-,y^+)$
<> o	$p2(x^-,y^-,y^+)$, $p2(y^+,x^-,x^+)$
<> o oi	$p1(x^-,\alpha1)$, $p1(\alpha1,x^+)$, $p1(y^-,\alpha2)$, $p1(\alpha2,y^+)$, $p2(\alpha1,y^-,y^+)$, $p2(\alpha2,x^-,x^+)$
<> oi	$p2(y^-,x^-,x^+)$, $p2(x^+,y^-,y^+)$
> d	$p1(y^-,x^-)$, $p2(y^+,x^-,x^+)$
> d di	$p1(x^-,\alpha1)$, $p1(y^-,\alpha2)$, $p1(\alpha2,x^+)$, $p2(\alpha2,x^-,y^+)$, $p2(\alpha1,y^-,x^+)$, $p2(y^+,x^-,\alpha1)$
> d di o	$p1(x^-,\alpha1)$, $p1(y^-,x^+)$, $p2(\alpha1,y^-,x^+)$, $p2(y^+,x^-,\alpha1)$
> d di o oi	$p1(y^-,x^+)$
> d di oi	$p1(y^-,\alpha2)$, $p1(\alpha2,x^+)$, $p2(\alpha2,x^-,y^+)$
> d o	$p1(y^-,x^+)$, $p2(y^+,x^-,x^+)$
> d o oi	$p1(y^-,\alpha2)$, $p1(\alpha2,y^+)$, $p1(y^-,x^+)$, $p2(\alpha2,x^-,x^+)$
> d oi	$p1(y^-,x^-)$
> di	$p2(x^-,y^-,y^+)$, $p1(y^+,x^+)$
> di o	$p1(y^-,x^+)$, $p2(x^-,y^-,y^+)$
> di oi	$p1(y^+,x^+)$
> di o oi	$p1(x^-,\alpha1)$, $p1(\alpha1,x^+)$, $p1(y^-,x^+)$, $p2(\alpha1,y^-,y^+)$
> o oi	$p1(x^-,\alpha1)$, $p1(\alpha1,x^+)$, $p1(y^-,\alpha2)$, $p1(\alpha2,y^+)$, $p1(y^-,x^+)$, $p2(\alpha1,y^-,y^+)$, $p2(\alpha2,x^-,x^+)$
> o	$p2(x^-,y^-,y^+)$, $p1(y^-,x^+)$, $p2(y^+,x^-,x^+)$
> oi	$p1(y^-,x^-)$, $p1(y^+,x^+)$
<	$p1(x^+,y^-)$
< d	$p2(y^-,x^-,x^+)$, $p1(x^+,y^+)$
< d di	$p1(y^-,\alpha1)$, $p1(x^-,\alpha2)$, $p1(\alpha2,y^+)$, $p2(\alpha2,y^-,x^+)$, $p2(\alpha1,x^-,y^+)$, $p2(x^+,y^-,\alpha1)$

X r Y	Conjonctions de schémas
< d di o	$p1(x^-,\alpha2), p1(\alpha2,y^+), p2(\alpha2,y^-,x^+)$
< d di o oi	$p1(x^-,y^+)$
< d di oi	$p1(y^-,\alpha2), p2(\alpha2,x^-,y^+), p2(x^+,y^-,\alpha2)$
< d o	$p1(x^+,y^+)$
< d o oi	$p1(y^-,\alpha2), p1(\alpha2,y^+), p1(x^-,y^+), p2(\alpha2,x^-,x^+)$
< d oi	$p1(x^-,y^+), p2(y^-,x^-,x^+)$
< di	$p1(x^-,y^-), p2(x^+,y^-,y^+)$
< di o	$p1(x^-,y^-)$
< di o oi	$p1(x^-,\alpha1), p1(\alpha1,x^+), p1(x^-,y^+), p2(\alpha1,y^-,y^+)$
< di oi	$p1(x^-,y^+), p2(x^+,y^-,y^+)$
< o	$p1(x^-,y^-), p1(x^+,y^+)$
< o oi	$p1(x^-,\alpha1), p1(\alpha1,x^+), p1(y^-,\alpha2), p1(\alpha2,y^+), p1(x^-,y^+), p2(\alpha1,y^-,y^+), p2(\alpha2,x^-,x^+)$
< oi	$p1(x^-,y^+), p2(y^-,x^-,x^+), p2(x^+,y^-,y^+)$
d	$p1(y^-,x^-), p1(x^+,y^+)$
d di	$p1(x^-,\alpha1), p1(\alpha1,y^+), p1(y^-,\alpha2), p1(\alpha2,x^+), p2(\alpha2,x^-,y^+), p2(\alpha1,y^-,x^+)$
d di o	$p1(x^-,\alpha1), p1(\alpha1,y^+), p1(y^-,x^+), p2(\alpha1,y^-,x^+)$
d di o oi	$p1(x^-,y^+), p1(y^-,x^+)$
d di oi	$p1(y^-,\alpha2), p1(\alpha2,x^+), p1(x^-,y^+), p2(\alpha2,x^-,y^+)$
d o	$p1(y^-,x^+), p1(x^+,y^+)$
d o oi	$p1(y^-,\alpha2), p1(\alpha2,y^+), p1(y^-,x^+), p1(x^-,y^+), p2(\alpha2,x^-,x^+)$
d oi	$p1(y^-,x^-), p1(x^-,y^+)$
di	$p1(x^-,y^-), p1(y^+,x^+)$
di o	$p1(x^-,y^-), p1(y^-,x^+)$
di o oi	$p1(x^-,\alpha1), p1(\alpha1,x^+), p1(x^-,y^+), p1(y^-,x^+), p2(\alpha1,y^-,y^+)$
di oi	$p1(x^-,y^+), p1(y^+,x^+)$
o	$p1(x^-,y^-), p1(y^-,x^+), p1(x^+,y^+)$
o oi	$p1(x^-,\alpha1), p1(\alpha1,x^+), p1(y^-,\alpha2), p1(\alpha2,y^+), p1(x^-,y^+), p1(y^-,x^+), p2(\alpha1,y^-,y^+), p2(\alpha2,x^-,x^+)$
oi	$p1(y^-,x^-), p1(x^-,y^+), p1(y^+,x^+)$

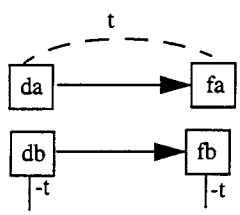
Annexe D: Représentation des disjonctions d'Allen dans les plans traditionnels non linéaires à liens causaux

- A > B

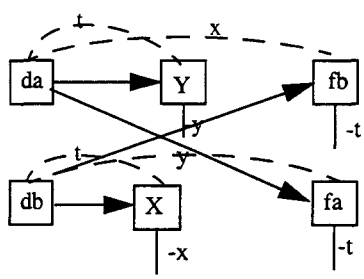
- A < > B



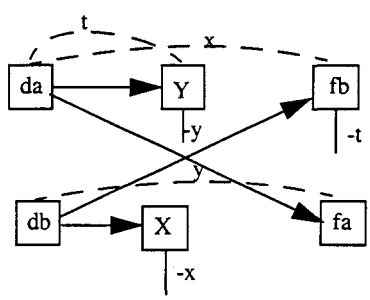
- A < > d B



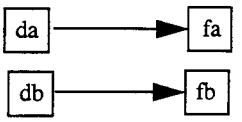
- A < > d di B



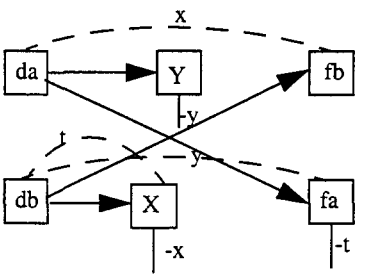
- A < > d di o B



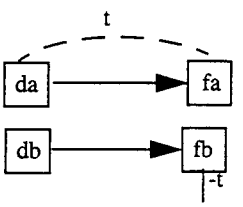
- A < > d di o oi B



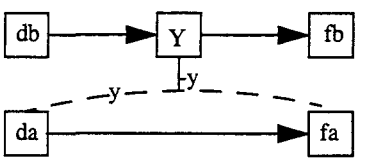
- A < > d di oi B



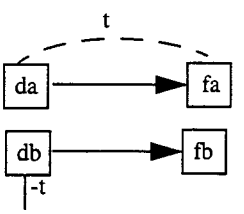
- A < > d o B



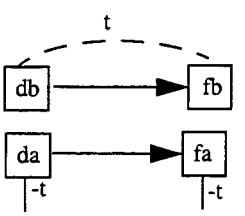
- A < > d o oi B



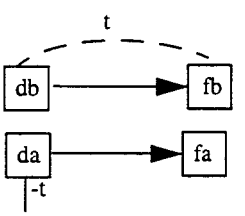
- A < > oi d B



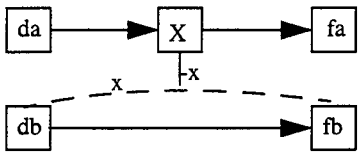
- A < > di B



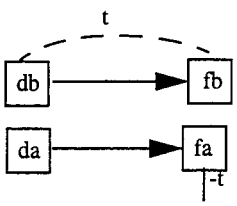
- A < > di o B



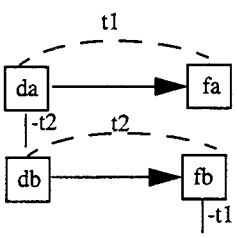
- A < > di o oi B



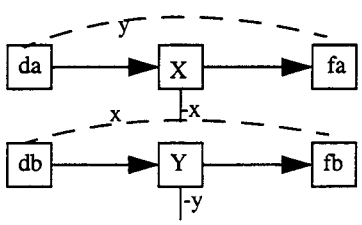
- A < > oi di B



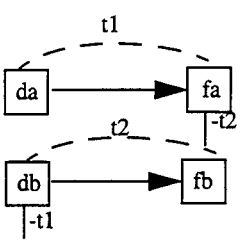
- A < > o B



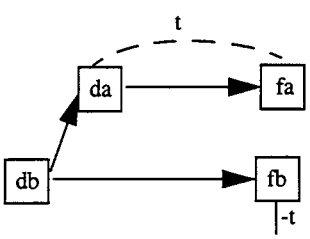
- A < > o oi B



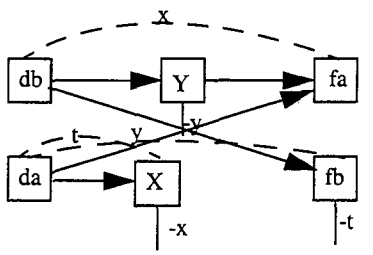
- A < > oi B



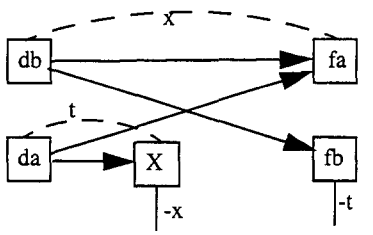
- A > d B



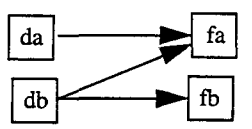
- A > d di B



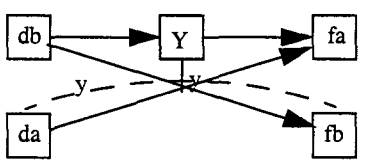
- A > d di o B



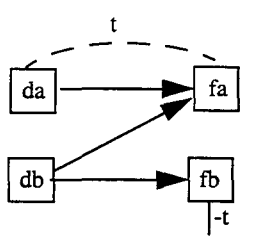
- A > d di o oi B



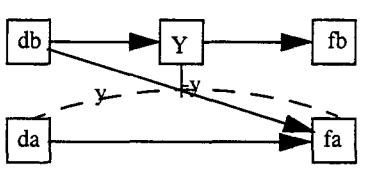
- A > d di oi B



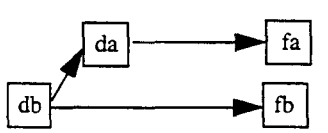
- A > d o B



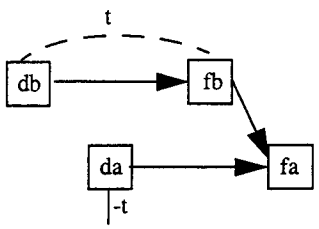
- A > d o oi B



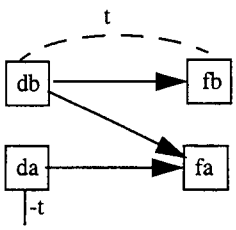
- A > d oi B



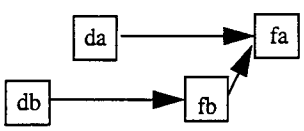
- A > di B



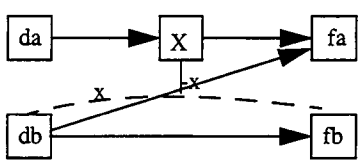
- A > di o B



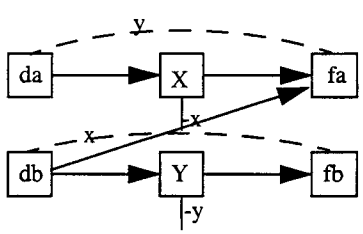
- A > di oi B



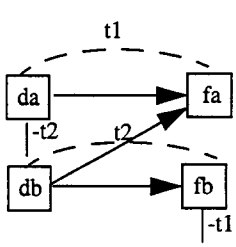
- A > di o oi B



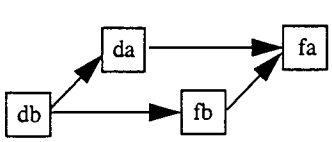
- A > o oi B



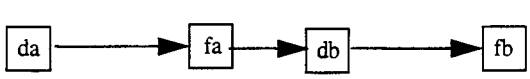
- A > o B



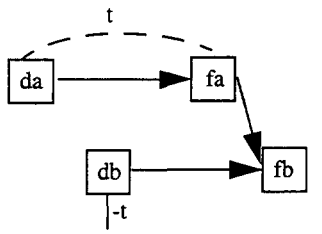
- A > oi B



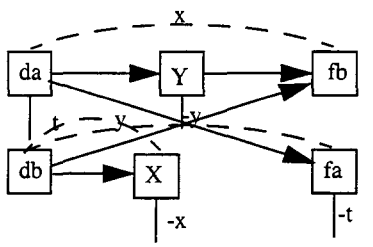
- A < B



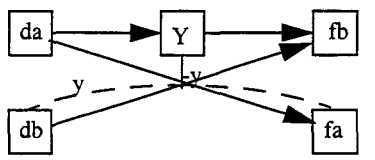
- A < d B



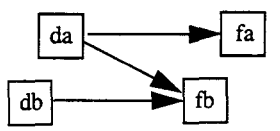
- A < d di B



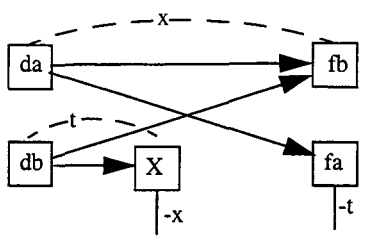
- A < d di o B



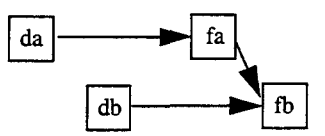
- A < d di o oi B



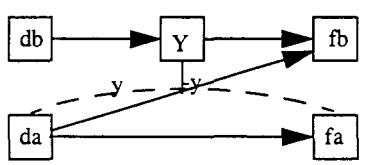
- A < d di oi B



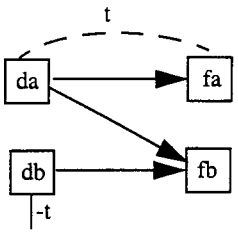
- A < d o B



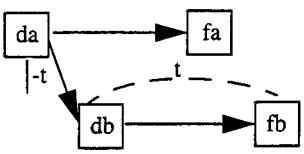
- A < d o oi B



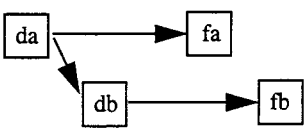
- A < d oi B



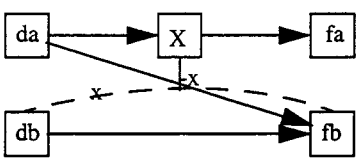
- A < di B



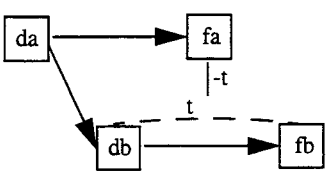
- A < di o B



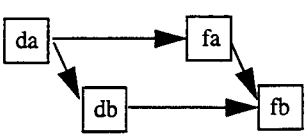
- A < di o oi B



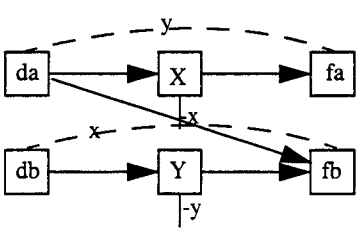
- A < di oi B



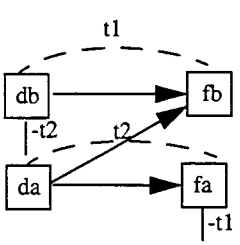
- A < o B



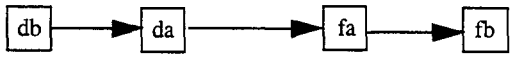
- A < o oi B



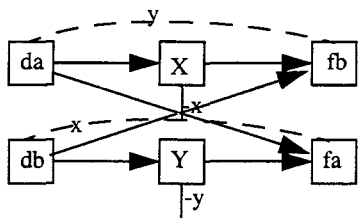
- A < oi B



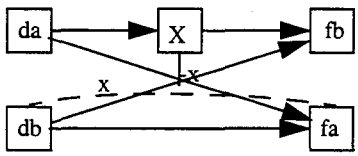
- A d B



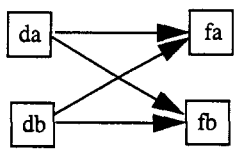
- A d di B



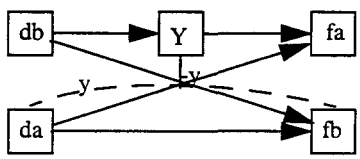
- A d di o B



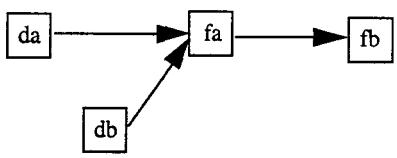
- A d di o oi B



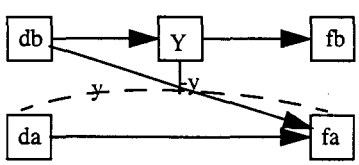
- A d di oi B



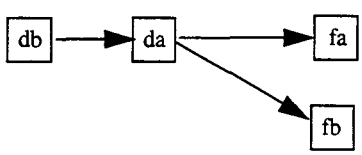
- A d o B



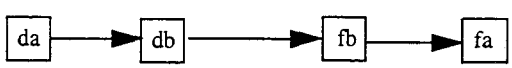
- A d o oi B



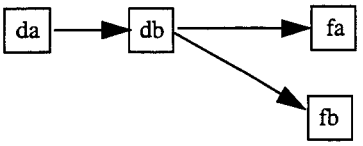
- A d oi B



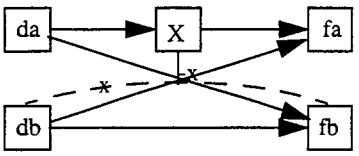
- A di B



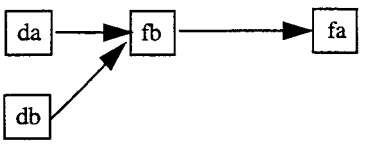
- A di o B



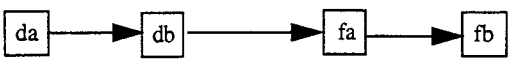
- A di o oi B



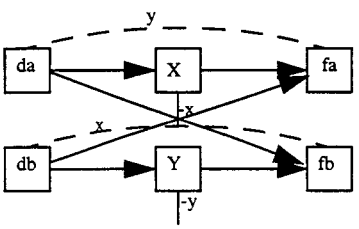
- A di oi B



- A o B



- A o oi B



- A oi B

Annexe E: Exemple de jeux d'essais

E.1: Attaque double (représentation à l'aide d'opérateurs instantanés)

- **Définition de la situation initiale**

```
subordonne(a1,ami).
subordonne(a2,ami).
subordonne(e1,eni).
subordonne(e2,eni).
pos(a1,p1).
pos(a2,p1).
libre(a1).
libre(a2).
vide(e2,p1,ld).
vide(e1,p1,ld).
entre(e2,ld,li).
entre(e1,li,lo).
```

- **Définition des buts**

```
vide(e2,ld,li).
vide(e1,li,lo).
pos(a1,lo).
```

- **Description des actions**

```
deplacer_ligne_directe ::si   subordonne(A,ami)
                              et subordonne(E,eni)
                              et (vide(E,Q,P),h)
                              et pos(A,Q)
                       alors  pos(A,P)
                              et (-pos(A,Q),s)
                       contrainte (diff(E,A), diff(P,Q) ).

detruire ::    si             subordonne(A,ami)
                              et subordonne(B,ami)
                              et entre(E,P,Q)
                              et pos(A,P)
                              et pos(B, P)
                              et soutien(A, B)
                       alors  (pos(A,Q),s)
                              et (-pos(A,P),s)
```

```
                    et vide(E,P,Q)
                contrainte (diff(E, A), diff(P, Q) ).

soutenir::      si      subordonne(A,ami)
                        et      subordonne(B,ami)
                et      pos(A,P)
                et      pos(B,P)
                alors   soutien(A,B)
                contrainte diff(B,A).

deplacer_ds_passage ::si subordonne(A,ami)
                        et subordonne(E,eni)
                        et pos(A,Q)
                        et (passage(E,Q,P),h)
                        et entre(E,Q,P)
                alors   pos(A,P)
                        et (-pos(A,Q),s)
                contrainte
                (when((nonvar(E),nonvar(A)),\+(E=A)),
                when((nonvar(P),nonvar(Q)),\+(P=Q))).

deborder ::     si      subordonne(A,ami)
                        et subordonne(E,eni)
                        et pos(A,Q)
                        et (entre(E,Q,P),h)
                alors   pos(A,P)
                        et (-pos(A,Q),s)
                contrainte
                (when((nonvar(E),nonvar(A)),\+(E=A)),
                when((nonvar(P),nonvar(Q)),\+(P=Q))).

percer ::       si      subordonne(A,ami)
                        et subordonne(E,eni)
                        et entre(E,Q,P)
                        et pos(A,Q)
                alors   passage(E,Q,P)
                contrainte
                (when((nonvar(E),nonvar(A)),\+(E=A)),
                when((nonvar(P),nonvar(Q)),\+(P=Q))).

couper_soutien::si      subordonne(A,ami)
                        et subordonne(E,eni)
                        et subordonne(Y,eni)
                        et pos(A,P)
                        et (soutien(E,Y),h)
                        et entre(E,P,Q)
                alors   -soutien(E,Y)
                contrainte
                (when((nonvar(E),nonvar(A)),\+(E=A)),
                when((nonvar(P),nonvar(Q)),\+(P=Q)),
                when((nonvar(Y),nonvar(E)),\+(Y=E))).

controler::     si      vide(E,Q,P)
                        et pos(A,P)
                        et subordonne(A,ami)
                alors   controle(ami,P)
```

contrainte true.

E.2: Le problème de la valise (actions instantanées décrites dans le formalisme TCLP)

- **Description des actions**

```
oter_objet ::liste_variables []
            composantes
            (A, instant,precond at(OBJ, valise)
                                et at(valise, X)
                        effet   add   at(OBJ, X)
                                      et -at(OBJ, valise)
                                del   at(OBJ, valise)
                                contraintes
            (when( nonvar(OBJ), \+(OBJ=valise)),
            when(nonvar(X), \+(X=valise)) )

                        )
            index  index(A)
            contraintes_variables true
            contraintes_temporelles []
            debut  A
            fin    A
            maintien [].

deplacer ::liste_variables [X,Y]
            composantes
            (A, instant,precond at(valise, X)
                        effet   add   at(valise, Y)
                                      et -at(valise, X)
                                del   at(valise, X)
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        )
            index  index(A)
            contraintes_variables true
            contraintes_temporelles []
            debut  A
            fin    A
            maintien [].

mettre_objet ::liste_variables []
            composantes
            (A, instant,precond at(OBJ,X)
                                et at(valise, X)
                        effet   add   at(OBJ, valise)
                                      et -at(OBJ, X)
                                del   at(OBJ, X)
                                contraintes
                    when((nonvar(X)), \+(X=valise))
                        )
```

```
index  index(A)
contraintes_variables true
contraintes_temporelles []
debut  A
fin    A
maintien [].
```

- **description des mondes**

```
init :: liste_variables []
                composantes
                        (BEGIN, begin, liste_variables []) &
                        (END, end, liste_variables [])
                index  []
                contraintes_variables true
                contraintes_temporelles (BEGIN av END)
                debut  BEGIN
                fin    END
                maintien [].

begin ::liste_variables []
                composantes
                        (A1, instant,precond []
                                effet  add    at(dico,home)
                                              et at(valise,home)
                                              et at(cheque, valise)
                                       del    []
                                       contraintes true)
                index  []
                contraintes_variables true
                contraintes_temporelles []
                debut  A1
                fin    A1
                maintien [].

end :: liste_variables []
                composantes
                        (A1, instant,precond at(cheque,bank)
                                       et at(dico,work)
                                       et at(valise, home)
                                       effet  add    []
                                              del    []
                                       contraintes true)
                index  []
                contraintes_variables true
                contraintes_temporelles []
                debut  A1
                fin    A1
                maintien [].
```

- **incompatibilités**

```
incompatibilite(at(X,T),at(Y,U)) :: (diff(T,U),egal(X,Y)).
```

E.3: Alkoarm (Formalisme TCLP): issu de [ALL 83b]

Version 1: 1 seul bras de robot

- **actions**

```
move ::liste_variables [X,F,T]
        composantes
                (A,pickup,liste_variables[X,F]) &
                (B,putdown, liste_variables[X,T])
        index  index(B) et index(A)
        contraintes_variables
                        when((nonvar(F),nonvar(T)), \+(T=F))
        contraintes_temporelles A av B
        debut  debut(A)
        fin    fin(B)
        maintien (-arm_is_free,debut(A),fin(B))
                        et(holding(X),fin(A),debut(B)).
pickup ::liste_variables [X,Y]
        composantes
                (A, instant,precond clear(X)
                                        et on(X,Y)
                                        et arm_is_free
                                effet  add    -clear(X)
                                              et -arm_is_free
                                       del    arm_is_free
                                              et clear(X)
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        ) &
                (B, instant,precond []
                                effet  add
                                              (clear(Y),s)
                                              et holding(X)
                                              et -on(X,Y)
                                       del    clear(X) et on(X,Y)
                                              et -holding(X)
                                              et -clear(Y)
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        )
        index  index(B)
        contraintes_variables true
        contraintes_temporelles A av B
        debut  A
        fin    B
        maintien (on(X,Y),A,B) et (-clear(X),A,B).

putdown ::liste_variables [X,Y]
        composantes
                (A, instant,precond holding(X) et clear(Y)
                                effet  add    -clear(Y)
                                       del    clear(Y)
                                contraintes
```

```
                when((nonvar(X),nonvar(Y)), \+(X=Y))
                ) &
        (B, instant,precond []
                        effet  add    on(X,Y)
                                      et (clear(X),s)
                                      et (-holding(X),s)
                                      et (arm_is_free,s)
                               del    -on(X,Y)
                                      et -clear(X)
                                      et holding(X)
                                      et -arm_is_free
                        contraintes true
                )
    index  index(A) et index(B)
    contraintes_variables true
    contraintes_temporelles A av B
    debut  A
    fin    B
    maintien (holding(X),A,B) et (on(X,Y),A,B) et (-clear(Y),A,B).
```

- **description des mondes**

```
init ::liste_variables []
        composantes
                (BEGIN, begin, liste_variables []) &
                (END, end, liste_variables [])
        index  []
        contraintes_variables true
        contraintes_temporelles (BEGIN av END)
        debut  BEGIN
        fin    END
        maintien [].

begin ::liste_variables []
        composantes
                (A1, instant,precond []
                                effet  add    on(a1,t1) et
                                              on(a2,t2) et
                                              clear(a1) et
                                              on(a3,t3) et
                                              clear(a3) et
                                              clear(a2) et
                                              arm_is_free
                                       del    []
                                contraintes true)
        index  []
        contraintes_variables true
        contraintes_temporelles []
        debut  A1
        fin    A1
        maintien [].

end ::liste_variables []
        composantes
                (A1, instant,precond on(a1,a2) et on(a2,a3)
```

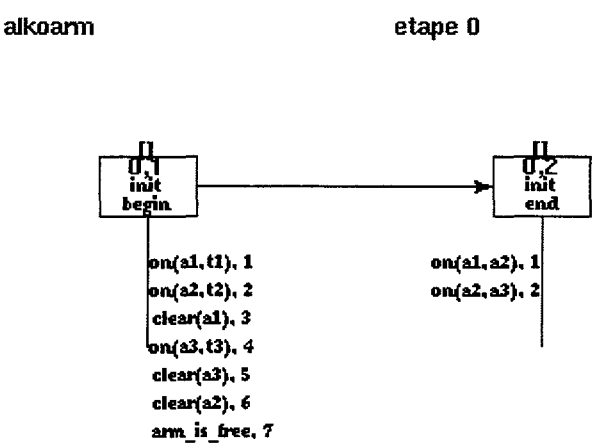
effet add []
del []
contraintes true)
index []
contraintes_variables true
contraintes_temporelles []
debut A1
fin A1
maintien [].

- **incompatibilités**

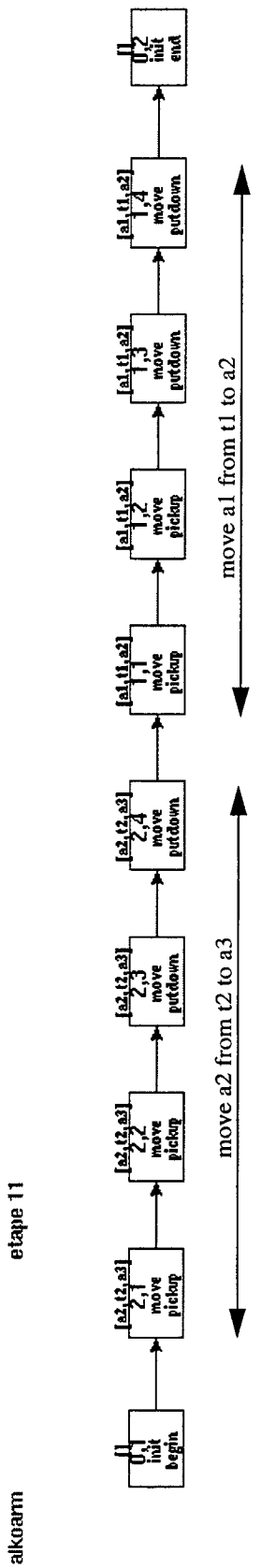
```
incompatibilite(holding(X),on(Y,U)) :: egal(X,U).
incompatibilite(holding(X),on(Y,U)) :: egal(X,Y).
incompatibilite(holding(X),clear(U)) :: egal(X,U).
incompatibilite(clear(X),on(Y,U)) :: egal(X,U).
incompatibilite(on(T,U1),on(Y,U2)) :: (egal(T,Y), diff(U1,U2)).
incompatibilite(on(Y1,T),on(Y2,U)) :: (egal(Y1,Y2),diff(T,U)).
incompatibilite(arm_is_free,holding(X)).
```

- **Plan résultant**

Plan initial:



Plan final:


alkoarm
etape 11
[] 0,1 init begin
[a2,t2,a3] 2,1 move pickup
[a2,t2,a3] 2,2 move pickup
[a2,t2,a3] 2,3 move putdown
[a2,t2,a3] 2,4 move putdown
[a1,t1,a2] 1,1 move pickup
[a1,t1,a2] 1,2 move pickup
[a1,t1,a2] 1,3 move putdown
[a1,t1,a2] 1,4 move putdown
[] 0,2 init end
move a1 from t1 to a2
move a2 from t2 to a3

Version 2: 2 bras de robot

- **Description des actions**

```
move ::liste_variables [X,F,T]
        composantes
                (A,pickup,liste_variables[X,F]) &
                (B,putdown, liste_variables[X,T])
        index  index(B) et index(A)
        contraintes_variables
                        when((nonvar(F),nonvar(T)), \+(T=F))
        contraintes_temporelles A av B
        debut  debut(A)
        fin    fin(B)
        maintien (holding(X),fin(A),debut(B)).

pickup ::liste_variables [X,Y]
        composantes
                (A, instant,precond clear(X) et on(X,Y)
                                effet  add    []
                                       del    []
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        ) &
                (B, instant,precond []
                                effet  add    -clear(X)
                                              et (clear(Y),s)
                                              et holding(X)
                                              et -on(X,Y)
                                       del    clear(X) et on(X,Y)
                                              et -holding(X)
                                              et -clear(Y)
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        )
        index  index(B)
        contraintes_variables true
        contraintes_temporelles A av B
        debut  A
        fin    B
        maintien (on(X,Y),A,B).

putdown ::liste_variables [X,Y]
        composantes
                (A, instant,precond holding(X) et clear(Y)
                                effet  add    on(X,Y) et -clear(Y)
                                       del    -on(X,Y) et clear(Y)
                                contraintes
                        when((nonvar(X),nonvar(Y)), \+(X=Y))
                        ) &
                (B, instant,precond []
                                effet  add    (clear(X),s)
                                              et (-holding(X),s)
                                       del    -clear(X)
```

```
                                        et holding(X)
                                contraintes true
                        )
        index  index(A) et index(B)
        contraintes_variables true
        contraintes_temporelles A av B
        debut  A
        fin    B
        maintien (on(X,Y),A,B).
```

- **Description des mondes**

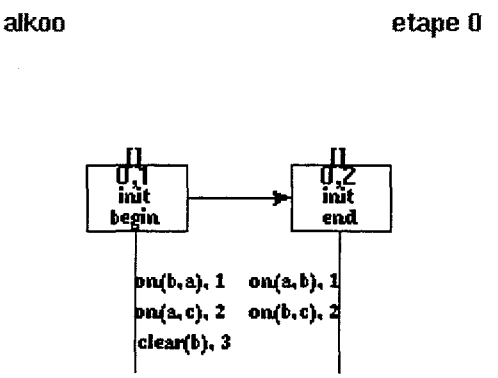
```
init ::liste_variables []
                composantes
                        (BEGIN, begin, liste_variables []) &
                        (END, end, liste_variables [])
                index  []
                contraintes_variables true
                contraintes_temporelles (BEGIN av END)
                debut  BEGIN
                fin    END
                maintien [].

begin ::liste_variables []
                composantes
                        (A1, instant,precond []
                                        effet  add    on(b,a) et
                                                      on(a,c) et
                                                      clear(b)
                                               del    []
                                        contraintes true)
                index  []
                contraintes_variables true
                contraintes_temporelles []
                debut  A1
                fin    A1
                maintien [].

end ::liste_variables []
                composantes
                        (A1, instant,precond on(a,b) et on(b,c)
                                        effet  add    []
                                               del    []
                                        contraintes true)
                index  []
                contraintes_variables true
                contraintes_temporelles []
                debut  A1
                fin    A1
                maintien [].
```

- **Incompatibilités**

```
incompatibilite(holding(X),on(Y,U)) :: egal(X,U).
incompatibilite(holding(X),clear(U)) :: egal(X,U).
incompatibilite(clear(X),on(Y,U)) :: egal(X,U).
incompatibilite(on(T,U1),on(Y,U2)) :: (egal(T,Y), diff(U1,U2)).
incompatibilite(on(Y1,T),on(Y2,U)) :: (egal(Y1,Y2),diff(T,U)).
```

- **Plan résultant**

Plan initial



Plan final