N° d'ordre 85

THÈSE

présentée à la

FACULTÉ DES SCIENCES DE L'UNIVERSITÉ DE LILLE

pour obtenir le

Titre de Docteur de Spécialité

(Mathématiques Appliquées)

par

JEAN-PIERRE STEEN

Algorithme de Recherche d'un Isomorphisme entre deux Graphes

Thèse soutenue le 26 Février 1968, devant la Commission d'Examen

Monsieur P. BACCHUS	**Président**
Monsieur P. POUZET	**Examinateur**
Monsieur C. BERGE	**Invité**
Monsieur J.-C. HERZ	**Rapporteur**

FACULTE DES SCIENCES — UNIVERSITE DE LILLE

LISTE DES PROFESSEURS

-oOo-

DOYENS HONORAIRES

MM. H. LEFEBVRE, M. PARREAU

PROFESSEURS HONORAIRES

MM. ARNOULT, BEGHIN, BROCHARD, CAU, CHAPPELON, CHAUDRON, CORDONNIER, DEHEUVELS, DEHORNE, DOLLE, FLEURY, P. GERMAIN, KAMPE DE FERIET, KOURGANOFF, LAMOTTE, LELONG, Mme LELONG, MM. MAZET, MICHEL, NORMANT, PARISELLE, PASCAL, PAUTHENIER, ROIG, ROSEAU, ROUBINE, ROUELLE, WIEMAN, ZAMANSKY

PROFESSEURS

BACCHUS P.	Mathématiques appliquées
BEAUFILS J.P.	Chimie
BONNEMAN P.	Chimie
BECART M.	Physique
BLOCH V.	Biologie et Physiologie Animales
BONTE A.	Sciences de la Terre
BOUGHON P.	Mathématiques pures
BOUISSET S.	Biologie et Physiologie Animales
BOURIQUET R.	Biologie Végétale
CELET P.	Sciences de la Terre
CONSTANT E.	Electronique, Electrotechnique et Automatique
CORSIN P.	Sciences de la Terre
DECUYPER M.	Mathématiques pures
DEDECKER P.	Mathématiques Pures
DEFRETIN R.	Biologie et Physiologie Animales
DEHORS R.	Electronique, Electrotechnique et Automatique
DELATTRE C.	Sciences de la Terre
DELEAU P.	Sciences de la Terre

DELHAYE M.	Chimie
DESCOMBES R.	Mathématiques pures
DURCHON M.	Biologie et Physiologie Animales
FOURET R.	Physique
GABILLARD R.	Electronique, Electrotechnique et Automatique
GLACET C.	Chimie
GONTIER G.	Mathématiques appliquées
HEIM DE BALSAC H.	Biologie et Physiologie Animales
HEUBEL J.	Chimie
HOCQUETTE M.	Biologie végétale
LEBEGUE A.	Botanique
Mme LEBEGUE G.	Physique
LEBRUN A.	Electronique, Electrotechnique et Automatique
Mlle LENOBLE J.	Physique
LIEBAERT R.	Electronique, Electrotechnique et Automatique
LINDER R.	Biologie végétale
LUCQUIN M.	Chimie
MARION E.	Chimie
MARTINOT-LAGARDE A.	Mathématiques appliquées
Mlle MARQUET S.	Mathématiques pures
MENNESSIER G.	Géologie
MONTARIOL F.	Chimie
MONTREUIL J.	Chimie
MORIAMEZ M.	Physique
MOUVIER G.	Chimie
PARREAU M.	Mathématiques pures
PEREZ J.P.	Physique
PHAM MAU QUAN	Mathématiques pures
POUZET P.	Mathématiques appliquées
PROUVOST J.	Sciences de la Terre
SAVARD J.	Chimie
SCHILTZ R.	Physique
SCHALLER F.	Biologie et Physiologie Animales
Mme SCHWARTZ M.H.	Mathématiques pures
TILLIEU J.	Physique
TRIDOT G.	Chimie
VAZART B.	Botanique

VIVIER E.	Biologie et Physiologie Animales
WATERLOT G.	Sciences de la Terre
WERTHEIMER R.	Physique

MAITRES DE CONFERENCES

BELLET J.	Physique
BENABOU J.	Mathématiques pures
BILLARD J.	Physique
BOILLET P.	Physique
BUI TRONG LIEU	Mathématiques pures
CHERRUAULT Y.	Mathématiques pures
CHEVALIER A.	Mathématiques
DERCOURT J.M.	Sciences de la Terre
DEVRAINNE P.	Chimie
Mme DIXMIER S.	Mathématiques
Mme DRAN R.	Chimie
DUQUESNOY A.	Chimie
GOUDMAND P.	Chimie
GUILBAULT P.	Biologie et Physiologie Animales
GUILLAUME J.	Biologie végétale
HENRY L.	Physique
HERZ J.C.	Mathématiques appliquées
HEYMAN M.	Physique
HUARD DE LA MARRE P.	Mathématiques appliquées
JOLY R.	Biologie et Physiologie Animales
LABLACHE-COMBIER A.	Chimie
LACOSTE L.	Biologie végétale
LAMBERT G.	Physique
LANDAIS J.	Chimie
LEHMANN D.	Mathématiques pures
Mme LEHMANN J.	Mathématiques pures
LOUCHEUX C.	Chimie
MAES S.	Physique
METTETAL C.	Zoologie
MONTEL M.	Physique

NGUYEN PHONG CHAU	Mathématiques
PANET M.	Electronique, Electrotechnique et Automatique
PARSY F.	Mathématiques pures
RACZY L.	Physique
SAADA G.	Physique
SEGARD E.	Chimie
TUDO J.	Chimie minérale appliquée
VAILLANT J.	Mathématiques pures
VIDAL P.	Electronique, Electrotechnique et Automatique
Mme ZINN-JUSTIN N.	Mathématiques pures

Je remercie vivement Monsieur le Professeur BACCHUS de l'honneur qu'il me fait de présider le Jury de cette thèse.

J'exprime toute ma gratitude à Monsieur le Professeur HERZ qui a bien voulu diriger mes travaux de recherches, m'a encouragé et aidé de ses précieux conseils.

Je suis tout particulièrement reconnaissant à Monsieur le Professeur BERGE - dont les ouvrages font autorité en Théorie des Graphes - et à Monsieur le Professeur POUZET qui ont bien voulu juger ce travail.

Mes remerciements vont aussi à Madame DUSART ainsi qu'à toute l'équipe technique du Laboratoire de Calcul de la Faculté des Sciences de LILLE pour le soin qu'ils ont apporté à la présentation de cette thèse.

Mes remerciements vont également à ma femme pour ses encouragements et sa participation efficace à la réalisation du programme et du manuscrit.

I. INTRODUCTION

1 - Introduction

Ce chapitre, qui a pour but de faire découvrir le sujet traité, a été volontairement rédigé en termes intuitifs de façon à permettre aux lecteurs non-mathématiciens, en particulier les spécialistes des sciences humaines, chez qui la théorie des graphes trouve une audience particulière, de se familiariser avec le problème et d'en avoir une vue d'ensemble. Le mathématicien trouvera dès le chapitre 2 les termes auxquels il est habitué.

Tous les graphes considérés sont des graphes finis.

1.1. Le problème

1.1.1 - Définition intuitive d'un graphe

Définition 1 Un graphe peut-être représenté par une figure géométrique formée de points reliés par des lignes orientées.
Les points représentent les sommets.
Les lignes orientées représentent les arcs
On appellera n le nombre de sommets et m le nombre d'arcs

1.1.2 - Enoncé du problème

La figure géométrique représentative d'un graphe n'est pas unique. Il y en a une infinité qui se déduisent de l'une d'elles par déformation continue.

Problème La question qui se pose naturellement est de savoir reconnaître si deux figures représentent le même graphe, c'est-à-dire si elles sont superposables par déformation continue. (Exemple : Fig. 1)
Il est évident qu'elles doivent avoir même nombre d'arcs et de sommets.

1.1.3 - Difficulté et expression du problème

Pour des graphes de quelques sommets et quelques arcs, l'oeil permet de donner une solution, mais dès que les nombres d'arcs et de sommets augmentent il devient nécessaire de repérer les sommets (et peut-être même, les arcs) en leur affectant un identificateur, qui en général sera un nombre ou une lettre.
On parlera d'indexation des sommets.

Expression Le problème devient alors : trouver une correspondance biunivoque entre l'ensemble des sommets de l'une des figures et l'ensemble des sommets de l'autre figure qui respecte les liaisons par les arcs qui existent entre eux (Exemple : Fig. 2)
On résout en fait un problème plus compliqué, puisqu'on cherche une transformation de l'une des figures en l'autre pour établir si elles sont ou non superposables.
Notons, en remarque, que cette application biunivoque n'est pas nécessairement unique

1.1.4 - Difficulté supplémentaire : La représentation d'un graphe

Plus un graphe a de sommets et d'arcs, plus il est difficile de l'étudier à cause de la taille des problèmes de mathématiques combinatoires qu'il faut résoudre. On a alors recours à des procédés mécaniques comme ceux qu'utilisent les calculateurs électroniques. L'inconvénient est que ceux-ci utilisent des graphes qui ne sont pas des figures géométriques et qui impliquent une indexation des sommets.

Définition Un graphe comportera alors un ensemble de sommets indicés et une fonction exprimant les liaisons existant entre ces sommets, et la représentation géométrique aura complètement disparue. Mais il faut se rendre compte que les points d'une même figure peuvent être indexés de multiples façons par un même ensemble d'indices.

Nouvelle expression Involontairement on prendra pour différents deux graphes obtenus à partir d'une même figure, mais indexée de deux façons différentes. Le problème sera encore de reconnaître s'il s'agit bien du même graphe (Exemple : Fig. 3).
Il est le même que dans le cas précédent.

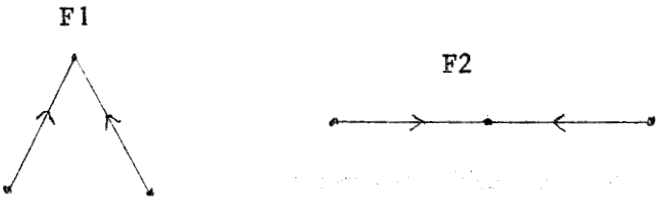
1.1.5. Graphes isomorphes. Définition intuitive

Partant de la notion de graphe donnée par la Définition 2(61.4) on dira que

Définition 3 deux graphes sont isomorphes s'il existe une correspondance biunivoque de l'ensemble des indices des sommets de l'un, sur l'ensemble des indices des sommets de l'autre qui respecte les liaisons.

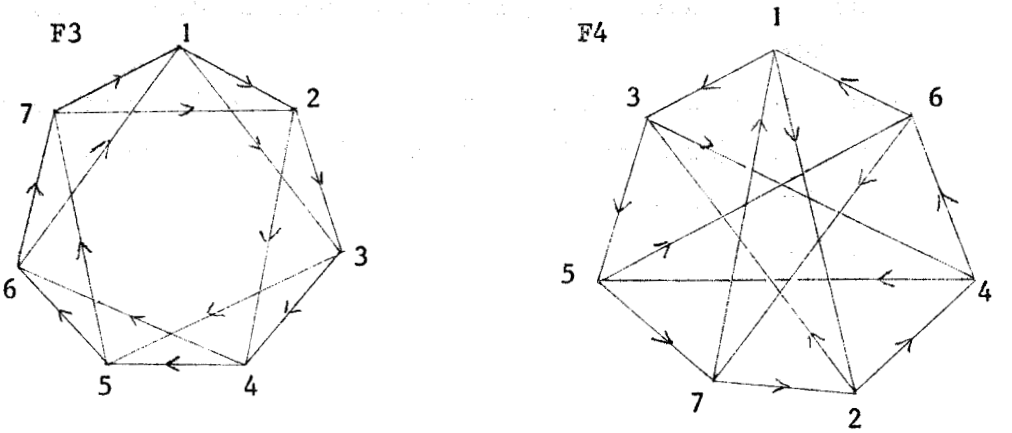
Si les indices sont les mêmes, cette correspondance est une permutation. Les graphes isomorphes permettent de résoudre le problème posé

Fig. 1



F1 et F2 sont deux figures qui représentent un même graphe

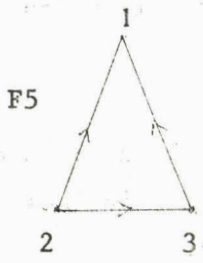
Fig. 2

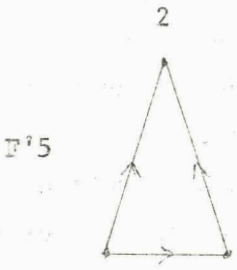


F3 et F4 sont 2 représentations du même graphe

Il faut faire correspondre les sommets de même indice.

Fig. 3





Représentation

Indice	sommets successeurs
1	
2	1,3
3	1

Indices	Sommets successeurs
1	2,3
2	
3	2

F 5 et F'5 sont le même graphe sous des indexations différentes

1.1.6 - Cas concrets

Rencontre-t-on effectivement ce problème?

Oui, chaque fois que l'on engendre des graphes par un procédé quelconque.

Exemple : construction par calculateur de graphes ayant une propriété, qui peut-être

- avoir un nombre fixe de sommets, d'arcs.
- être hypohanultonien (Problème de banquet du Club des Irrascibles)
- être susceptible de supporter des flots qui seront aussi des tensions (Problème de la décomposition de rectangles en carrés de côtés différents)

etc...

1.2. Principe de résolution

1.2.1 - Des tests

On peut évidemment inclure des tests appropriés à la méthode employée au cours de la construction des graphes, mais le plus souvent c'est devant les graphes obtenus que se pose le problème de reconnaître ceux qui sont isomorphes.

1.2.2 - Et des Permutations

Si on prend la précaution d'indicer les sommets avec les mêmes identificateurs, c'est alors un problème de permutation, et la méthode la plus naturelle, celle qui vient à l'esprit est de permuter de toutes les façons possibles les sommets de l'un des graphes et de le comparer à l'autre. Malheureusement cette méthode est très longue.

Pour des graphes de n = 12 sommets, et un calculateur qui engendre et essaye * une permutation de 0,1 seconde, il faudra dans les plus mauvais cas, calculer n! = 12! = 479 001 600 permutations pour déterminer si deux graphes sont isomorphes. Si les graphes ne le sont pas, on ne le saura qu'après avoir essayé ces n! permutations. Pour un tel calculateur cela représente un fonctionnement ininterrompu pendant 1 an et demi ou 2 ans.

1.2.3 - Respectant les liaisons

Aussi, ne va-t-on pas essayer toutes les permutations, et par différents procédés essayer d'en éliminer le plus possible.

---

* Un tel calculateur pourrait travailler comme suit :
- Représenter les graphes par leurs matrices caractéristiques : A1 pour le 1er, A2, pour le second.
- Pour chaque permutation construire la matrice permutation associée P
- Vérifier si A1 = P^{t}.A2.P-

Ces opérations nécessitent pour chaque permutation $2n^3$ additions + $2n^3$ multiplications et $2n^2$ comparaisons sans compter les opérations sur les indices

Pour cela on utilisera :

la propriété fondamentale

> Si deux sommets sont reliés par un arc dans l'un des graphes leurs correspondants le sont aussi dans l'autre avec respect de l'orientation de l'arc
> qui intuitivement signifie que la correspondance respecte la "structure du graphe", constituée par les liaisons.

1.3. La méthode

1.3.1. - Les difficultés

On se pose alors trois questions
- comment s'exprime la "structure du graphe"
- comment apparaît l'isomorphisme
- comment le construire en utilisant cette "structure"

1.3.2 - Les réponses

On verra au chapitre 4
- que la "structure" se traduit en fonction de structure dont la plus naturelle est la classique fonction Γ qui a chaque sommet associe ses successeurs
- que l'isomorphisme se construit à l'aide du calcul des fonctions intrinsèques qui déterminent sur les graphes des partitions de l'ensemble des sommets, partitions de plus en plus fin dont les classes dans les deux graphes sont mises en correspondance
- qu'une méthode dite "méthode des deux fonctions" permet d'affiner à l'aide des fonctions de structure les partitions associées aux fonctions intrinsèques.

1.4. - Les Résultats et la méthode des hypothèses

1.4.1 - Aboutissement

On aboutira à plusieurs possibilités :

- on trouve un isomorphisme. Le problème est alors résolu.
- on montre qu'il n'y a pas d'isomorphisme. Le problème a encore une réponse
- on ne parvient pas à répondre parce qu'on ne sait pas, parmi les permutations qui restent en choisir ou en éliminer.

1.4.2 - Méthode des hypothèses

La méthode des hypothèses , qui n'est autre qu'une méthode de "Backtracking", suppose une certaine correspondance entre les sommets, et vérifie par la méthode des deux fonctions si cette supposition est valable, et en fait une autre si nécessaire.

Elle fait implicitement les essais qui restaient à faire sur les permutations qui n'avaient pas été éliminées. Elle pourrait à elle seule résoudre le problème.

1.5 - Présentation de l'ouvrage

1.5.1 - Théorie (chapitre 2)

C'est l'étude analytique des isomorphes des graphes. En particulier on y trouvera la définition du groupe d'automorphismes d'un graphe.

1.5.2 - Critères d'existence d'isomorphisme entre deux graphes (Chapitre 3)

C'est un survol des essais de déterminations de ces critères.

1.5.3 - Méthode générale de recherche d'un isomorphisme entre 2 graphes (Chapitre 4)

On y trouvera la description des outils utilisés (Fonctions intrinsèques, Fonctions de structure) et leurs emplois.
Un paragraphe spécial est consacré à la description de la méthode des hypothèses et à sa nécessité qui ne semble pas avoir été clairement établie jusqu'ici.

1.5.4 - Comparaison avec les autres méthodes (Chapitre 5)

Ce sont les méthodes qui ont déjà été publiées. Certaines s'en approchent, d'autres, pas spécialement étudiées en vue de la recherche de l'isomorphisme de deux graphes, peuvent quand même résoudre partiellement le problème qui nous intéresse.

1.5.5 - Réalisation pratique d'un algorithme (Chapitre 6)

En fonction de la représentation matricielle des graphes, construction et utilisation concrète des outils utilisés par la méthode.
On trouvera des procédures ALGOL pour la plupart des opérations de l'Algorithme.

En particulier on verra la réalisation de la méthode des deux fonctions à l'aide de la technique du produit scalaire inachevé

1.5.6 - Le programme (Chapitre 7)

Il realise l'algorithme à quelques restrictions près.

1.5.7 - Conclusion (Chapitre 8)

Dans quel sens peut-on améliorer l'algorithme.

II - THEORIE

2 - Théorie

2.1. Définition mathématique d'un graphe

2.1.1.1 - Définition

Un graphe G est un triplet (X, U, e) où
X est l'ensemble des sommets
U est l'ensemble des arcs
e est la fonction extrémité

On appellera $n = |X| = \text{card}(X)$ le nombre des sommets
$m = |U| = \text{card}(U)$ le nombre des arcs

2.1.1.2 - Fonction extrémité

C'est une application $e : U \to X \times X$ qui à l'arc n associe le couple de sommets (x,y) de X x X, tel que x soit l'origine de l'arc u et y son extrémité. Soit : eu = (x,y).

Par projection sur les ensembles du produit X x X, on construit les application
$e^+ : U \to X$ telle que e^+u = x = origine de l'arc u
$e^- : U \to$ telle que e^-u = y = extrémité de l'arc u

2.1.1.3 - Graphe plein, graphe sans arcs

Un graphe plein est un graphe tel que entre deux sommets il existe un arc les reliant dans chaque sens.
Pour un tel graphe la fonction e est surjective
Dans un graphe sans arcs chaque sommet est isolé : $U = \emptyset$

2.1.1.4 - Multigraphe

C'est un graphe tel qu'il peut exister plusieurs arcs de même sens entre deux sommets. S'il y en a au plus p, on parle de

p -graphe. Un graphe ordinaire serait un 1-graphe

Pour les multigraphes ($\rho > 1$), la fonction e n'est pas injective.
Pour un graphe ordinaire la fonction e est injective.

2.1.2 - Numérotation des sommets et des arcs.

2.1.2.1 - Numérotation des sommets.

Pour représenter un graphe il faut avant tout pouvoir distinguer les sommets et les arcs.
Pour réaliser cette condition on numérote d'abord les sommets de 1 à n. On dira que les sommets ont été indicés.

2.1.2.2 - Déduction de la numérotation des arcs dans un graphe ordinaire

Pour un tel graphe il est possible de distinguer les arcs par leurs extrémités.
En effet la fonction $e : U \rightarrow e(U) \subset X \times X$ est bijective.
Les arcs seront alors numérotés comme leur image par l'application e ; celles-ci sont des couples distincts d'entiers.
On peut les classer et les numéroter de 1 à m.

2.1.2.3 - Déduction de la numérotation des arcs dans un multigraphe

Dans un p-graphe ($\rho \geq 2$) on peut distinguer deux arcs qui ont des extrémités distinctes, mais, à priori, il faudra pouvoir distinguer deux arcs ayant mêmes extrémités et même sens.
On construira l'application :

$$e : U \longrightarrow X \times X \times \{1,2,\ldots,\rho\}$$

telle que $eu = (x,y,i)$ où

- x est l'origine de l'arc u
- y son extrémité
- i un nombre qui permet de distinguer u des autres arcs ayant mêmes extrémités.

Une telle fonction est injective. En appliquant le processus ci-dessus on pourra déduire la numérotation des arcs de la numérotation des triplets, images par la fonction e.

Notons que dans les multigraphes la numérotation des arcs d'après les sommets nécessite une numérotation partielle, à priori, des arcs ayant mêmes extrémités et même sens. C'est le rôle du troisième élément du triplet.

Nous ne porterons que peu d'intérêt à ce dernier problème. En effet, ce qui importe dans cette étude, c'est qu'entre les sommets correspondants de deux graphes susceptibles d'être isomorphes, il y ait le même nombre d'arcs dans chaque sens. Nous verrons souvent que le nombre d'élements ayant une propriété donnée est plus utile que les éléments eux-mêmes.

2.1.3 - Représentation d'un graphe

Il existe de nombreuses représentations des graphes En voici trois parmi les plus courantes.

2.1.3.1 - Représentation par la fonction e : tableau E

En tenant compte des modifications précédentes, la fonction e sera déterminée dès que les couples (ou triplets) de son image seront numérotés.
Une première représentation d'un graphe consistera alors à se donner la liste des éléments de X, celle de U et à décrire la fonction e (Fig. 4)

En observant le résultat, on se rend compte qu'il y a redondance d'information et qu'en fait seulement la fonction e présente un intérêt. On peut remplacer X et U par leurs cardinaux n et m, et ramener la fonction e à un tableau E à m lignes et deux colonnes. Les lignes seront indicées par les arcs et les colonnes seront les images des fonctions e^+ et e^-. On peut avoir plusieurs lignes identiques dans le cas des multigraphes.

2.1.3.2 - Représentation par la matrice caractéristique A

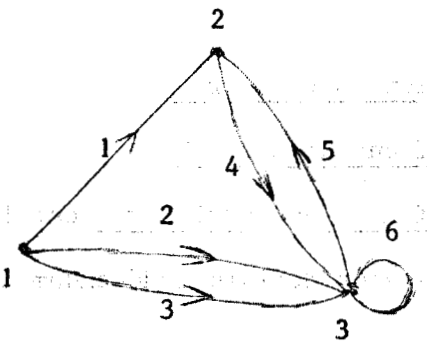
2.1.3.2.1 - Cas des graphes ordinaires

C'est la matrice A de la fonction caractéristique de e(U) dans X x X. Elle est indicée en ligne et en colonne par l'ensemble des sommets et

$$A_{x\,y} = \begin{cases} 1 & \text{s'il existe un arc de x à y} \\ 0 & \text{sinon} \end{cases}$$

Elle caractérise le graphe. n est l'ordre de la matrice, m est le nombre de ses éléments non nuls. Elle détermine chaque arc par ses extrémités et ne nécessite même pas leur numérotation.
Elle permet de dessiner le graphe.

Fig. 4 Représentation d'un graphe



X = { 1,2,3 }

U = { 1,2,3,4,5,6 }

e1 = (1,2,1) (1,2)

e2 = (1,3,1) (1,3)$_1$

e3 = (1,3,2) ou (1,3)$_2$

e4 = (2,3,1) (2,3)

e5 = (3,2,1) (3,2)

e6 = (3,3,1) (3,3)

tableau E

E	e^+	e^-
1	1	2
2	1	3
3	1	3
4	2	3
5	3	2
6	3	3

matrice A

A	1	2	3
1	0	1	2
2	0	0	1
3	0	1	1

liste de successeurs Γ

tableau Γ

x	Γ
1	2,3,3,
2	3,
3	2,3,

liste Γ

1(2,3,3) 2(3) 3(2,3)

2.1.3.2.2 - Cas des multigraphes

On généralise la matrice A. On posera
A_{xy} = le nombre d'arcs de x à y.
Elle caractérise toujours le graphe, en ce sens, qu'elle permet de le dessiner. m devient alors la somme des éléments de la matrice A.

2.1.3.3 - Représentation par la fonction successeur Γ.

2.1.3.3.0-Monoïde commutatif engendré par X

Déf. 2.1.3.3.0 - Le monoïde commutatif $X^{\circledast}$ engendré par X est l'ensemble des mots $\{\Pi\, x_i^{\alpha_i} \mid x_i \in X\ \alpha_i \in \mathbb{N}\}$ sont définis indépendamment de l'ordre de leurs lettres.

A chaque mot de ce monoïde correspond une application α appartenant à $\mathbb{N}^{\alpha}$ de $X \longrightarrow \mathbb{N}$ telle que $\alpha(x_i) = \alpha_i$

On appellera $X^{\circledast}_{p}$ le sous-monoïde de $X^{\circledast}$ formé des mots pour lesquels $\forall i\ 0 \leq \alpha_i \leq p$. Aux mots de $X^{\circledast}_{p}$ correspondent des applications de $\{0,\ldots,p\}^{X}$

2.1.3.3.1-Définition

Déf. 2.1.3.3.1 - Un sommet y est successeur d'un sommet de x s'il existe un arc de x à y. x est alors un prédécesseur de y.

2.1.3.3.2 - Définition

Déf. 2.1.3.3.2 - La fonction successeur $\Gamma : X \longrightarrow X_1^{\circledast}$ associe à chaque sommet x de X le mot Γx est formé par les successeurs.

Remarquons que les mots $X_1^{\circledast}$ sont en correspondance avec les applications de $\{0,1\}^{X}$ qui est l'ensemble des fonctions caractéristiques des sous-ensembles de X. La fonction successeur peut se ramener à une application : $X \longrightarrow P(X)$

2.1.3.3.3 - Représentation

Elle se fait par la liste des successeurs qui est soit
- au tableau Γ à simple entrée, indicé en lignes par les sommets et tel que chaque ligne contient le mot associé.
- une liste Γ obtenue en juxtaposant les lignes du tableau Γ, précédées de leur indice.

Pour un multigraphe on répètera un sommet dans les successeurs d'un autre autant de fois qu'il y a d'arcs le reliant dans le bon sens. Pour un p-graphe, la fonction successeur est alors une fonction : $X \longrightarrow X^{\circledast}_{p}$

Le tableau Γ et la matrice A se déduisent l'un de l'autre sans calcul, puisqu'il s'agit de la représentation d'un même graphe La ligne x de la matrice A est l'image de la fonction associée au mot x.

2.1.3.4 - Les problèmes de la représentation

2.1.3.4.1 - La forme de la représentation

C'est un problème de stockage d'information, celle que contient le graphe. Il faut la mettre dans un volume restreint et en retrouver rapidement la partie utile au problème qui nous intéresse.
Plus l'information est concentrée, plus il est long de la reconstituer sous une forme pratique pour l'utilisation, mais il ne faut toutefois pas tomber dans l'excès inverse où par une trop grande dispersion, on prend du temps à la retrouver.
La liste Γ, puis les tableaux E, Γ et A forment une progression en ce qui concerne la place utilisée.
Pour notre part, bien qu'elle soit la plus encombrante nous avons choisi la représentation matricielle A, car elle est celle qui se prête le plus facilement à l'explication des méthodes que nous allons décrire.

2.1.3.4.2 - La numérotation et le problème de l'isomorphie.

Nous avons vu en 1.1.4 que la numérotation ou l'indexation des sommets était la difficulté principale de la représentation des graphes. Le problème de l'isomorphie en résulte.
Il se pose en deux étapes. D'abord établir l'existence d'un isomorphisme, ensuite en construire un. En résolvant la deuxième étape, on résout naturellement le problème au niveau de la première.

2.2. - Quelques éléments de théorie des graphes (lexique)

On se reportera avec intérêt à un cours de théorie des graphes. Voici un rappel de définitions.

graphes — graphe $G = (X, U, e)$ avec $e : U \longrightarrow X^2$; $e = (e^+, e^-)$

sous-graphe de $G = (X'U', e)$ avec $X' \subset X$ et $U' = \{u \mid e'u \in X' \times X'\}$

arcs — arête deux arcs de sens contraires entre deux sommets. On les représente par un arc non orienté.

graphe symétrique : tout arc appartient à une arête. C'est un graphe non orienté.

arcs adjacents : ils ont une extrémité en commun.

arc adjacent à un sommet : ce sommet est l'une de ses extrémités.

boucle : $e^+u = e^-u$; arc dont les extrémités sont confondues.

sommets — demi degré extérieur (intérieur) : nombre d'arcs partant (resp. tortant) d'un sommet.

degré : somme des deux demi-degrés

successeur d'un sommet : sommet extrémité d'un arc issu de ce sommet.

prédécesseur d'un sommet : sommet origine d'un arc aboutissant à ce sommet

voisin : prédécesseur ou successeur

suites d'arcs : chaîne : suite d'arcs tels que chacun d'eux est adjacent au précédent. Sa longueur est le nombre d'arcs.

cycle : chaîne fermée.

chemin : chaîne d'arcs orientés dans le même sens.

circuit : chemin fermé

cycle élémentaire : cycle ne passant pas deux fois par le même sommet;

base de cycles : ensemble minimal de cycles élémentaires permettant de construire tous les autres.

chemin ou circuit élémentaire : ne passe pas deux fois par le même sommet.

graphes spéciaux

graphe connexe : entre deux sommets quelconques il existe une chaîne.

composante connexe : sous-graphe connexe, non relié au reste du graphe.

composante fortement connexe : entre deux sommets quelconques il existe un chemin.

arbre : graphe connexe, sans cycles.

arborescence : arbre tel qu'en chaque sommet n'aboutit qu'un arc au plus.

racine : sommet unique auquel n'aboutit aucun arc dans une arborescence.

Propriété de la matrice caractéristique A.

graphe symétrique = matrice A symétrique

graphe plein = Vi, Vj, $A_{ij} = 1$.

graphe sans arcs = Vi, Vj, $A_{ij} = 0$.

boucle sur le sommet i = $A_{ii} = 1$

2.3 - Définition de l'isomorphie de graphes

2.3.1 - Définition

Def. 2.3.1 | Deux graphes $G_1 = (X_1, U_1, e_1)$ et $G_2 = (X_2, U_2, e_2)$ sont <u>isomorphes</u> s'il existe une correspondance biunivoque entre les sommets des ensembles X_1 et X_2 telle que si deux sommets sont reliés par des arcs dans l'un des graphes, leurs correspondants le sont aussi dans l'autre graphe, avec respect du nombre et du sens des arcs.
Un <u>isomorphisme</u> est une application qui permet de transformer G_1 en G_2

Il en résulte immédiatement que si deux graphes sont isomorphes ils ont même nombre de sommets n. D'autre part cette correspondance biunivoque sous entend en pratique une indexation des sommets dans chacun des graphes.
Si cette indexation est faite avec le même ensemble de n indices pour les deux graphes, à la correspondance biunivoque entre les sommets est associée une permutation ν des indices des sommets.
Voir figure 5. la schématisation de ce raisonnement.
La propriété d'isomorphisme entre deux graphes est une relation d'équivalence dans tout ensemble fini de graphes.

2.3.2 - 1ère conséquence : Correspondance entre les arcs.

Nous avons vu que les arcs sont identifiés par leurs extrémités. Il en résulte de la correspondance biunivoque entre les sommets, une correspondance biunivoque entre les couples de sommets d'où une correspondance entre les groupes d'arcs ayant mêmes extrémités, correspondance qui est biunivoque entre les arcs dans le cas des graphes ordinaires, et qu'on peut rendre biunivoque dans le cas des multigraphes.

Fig. 5 : Isomorphisme et permutation

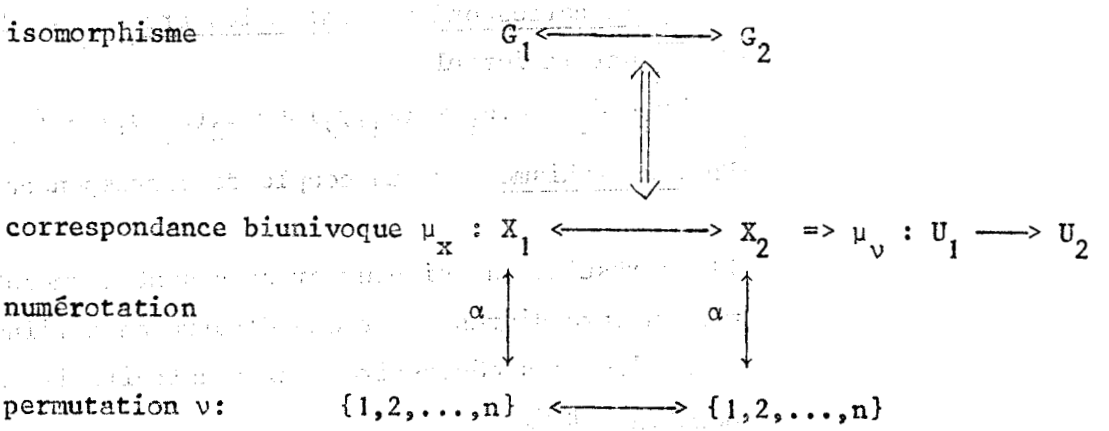


Fig.6 P-équivalence de matrices

- notation : A est une matrice d'ordre 4 dont les couples de lignes et colonnes de même indice sont numérotés. $A_{1,2,3,4}$ est la matrice dont les rangées sont dans l'ordre 1234
- soit la permutation ν : $\{1,2,3,4\} \longrightarrow \{2,3,1,4\}$
- permutation des rangées par ν^{-1} | correspondance entre les indices

$A'_{1,2,3,4}$ $\xrightarrow{P_\nu^t . P_\nu}$ $P_\nu^t A' P_\nu = A'_{3,1,2,4}$ = A" a,b,c,d

dont les rangées sont dans l'ordre $\{1,2,3,4\} \xrightarrow{\nu^{-1}}$

dont les rangées sont dans l'ordre $\{3,1,2,4\} \xrightarrow{\mu}$

dont les rangées sont dans l'ordre {b,c,a,d}

- Rappel $P_\nu^t = P_\nu^{-1} = P_{\nu-1}$

d'où la correspondance $\mu = \{1,2,3,4\} \longrightarrow \{b,c,a,d\}$

Def. 2.3.2 Appelons μ_x la correspondance biunivoque entre les sommets et μ_u la correspondance entre les arcs. μ_u se définit à partir de μ_x par la formule

$$\forall u_1 e U_1 : e_1 u_1 = (x_1, y_2) \Leftrightarrow e_2(\mu_u\, u_1) = (\mu_x\, x_1,\ \mu_x y_1).$$

Un isomorphisme sera un couple de correspondance (μ_x, μ_u)

Il en résulte que si deux graphes sont isomorphes ils ont même nombre d'arcs m, et que d'autre part l'indexation des arcs n'est pas nécessaire pour construire la correspondance entre U_1 et U_2.

L'ambiguité qui existe pour les arcs de mêmes extrémités et de même sens dans les multigraphes n'est pas une difficulté car il n'est pas nécessaire que la correspondance soit biunivoque pour ces arcs de mêmes extrémités. Il suffit qu'il y en ait le même nombre entre les sommets correspondants.

2.3.3 - 2ème conséquence - P-équivalence de matrices caractéristiques

2.3.3.1 - P - équivalence de matrices

Def. 2.3.3. | Deux matrices A' et A" carrées de même ordre sont P-équivalentes, s'il existe une matrice P de permutation telle que $A'' = P^t A' P$

La matrice d'une permutation s'obtient en permutant par ν les lignes d'une matrice unité.
L'opération $P^t A' P$ revient à permuter simultanément les rangées (lignes et colonnes) de A', par la permutation inverse ν^{-1} de celle qui a donné P. En établissant l'égalité entre A" et $P^t A' P$ on obtient une application μ qui fait passer des rangées de A' aux rangées de A". (voir Fig.6)

2.3.3.2 - Application aux matrices caractéristiques

La correspondance biunivoque entre les sommets de deux graphes isomorphes, existe entre les rangées des matrices caractéristiques des deux graphes, et la condition sur les arcs reste vérifiée. Elle s'exprime par : "le nombre qui se trouve à l'intersection de deux rangées (une ligne et une colonne) de l'une des matrices est égal à celui qui se trouve à l'intersection des rangées correspondantes de l'autre matrice".
Nous avons là l'égalité des deux matrices à condition de permuter les rangées de l'une d'elle, d'où

prop. 2.3.3. | Les matrices caractéristiques de deux graphes isomorphes sont P-équivalentes.

2.4 - Définition des automorphismes de graphes

Il peut exister plusieurs correspondances biunivoques entre les sommets de deux graphes. Leur étude se fait en utilisant les automorphismes de chacun des graphes.

2.4.1. - Définition

Déf. 2.4.1. Un automorphisme d'un graphe est un isomorphisme de ce graphe sur lui-même.

Si on veut transporter ici la définition des isomorphismes, on constate que l'application biunivoque est une permutation des sommets et non pas de leurs indices, et on peut alors directement exprimer la définition à l'aide des permutations ν_x et ν_u construites comme μ_x et μ_u

Prop. 2.4.1 Soit G =(X,U,e). Un automorphisme σ de G est défini par une permutation ν_x^σ des sommets de X telle que si u ∈ U est un arc orienté de x vers y alors il existe un arc u' ∈ U entre $\nu_x^\sigma x$ et $\nu_x^\sigma y$, orienté de $\nu_x^\sigma x$ vers $\nu_x^\sigma y$.

$$\forall u \in U : e(u)=(x,y) \ \exists u' \in U \ \ e(u')=(\nu_x^\sigma x, \nu_x^\sigma y) \ \ (u' = \nu_u^\sigma u).$$

On dira d'une permutation ν_x qui vérifie la condition ci-dessus, qu'elle respecte la structure du graphe

2.4.2. - Groupe d'automorphismes

Il peut exister plusieurs telles permutations différentes. Elles forment un groupe H qui est un sous-groupe du groupe Sn des permutations de n =|x| objets. La vérification en est aisée. Comme à chacune d'elle est associée un automorphisme,

prop. 2.4.2 L'ensemble des automorphismes d'un graphe G forme un groupe H(G) qui est isomorphe au sous-groupe H de Sn des permutations des sommets qui respectent la structure du graphe.

Pour un graphe plein (entre deux sommets quelconques il existe un arc dans chaque sens) et pour un graphe sans arcs (chaque sommet est isolé) les groupes d'automorphismes H(G) sont isomorphes à S_n car toutes les permutations des sommets respectent les arcs.
Un graphe G, son complémentaire (ils ont mêmes sommets et leurs ensembles d'arcs sont complémentaires dans XxX) et son inverse (obtenu en inversant chaque arc) ont même groupe d'automorphismes. Une permutation qui respecte les arcs dans un de ces trois graphes, les respecte dans les autres.

2.5. Isomorphismes et automorphismes

On établit trois propriétés

2.5.1. - Proposition 1

Si deux graphes sont isomorphes, leurs groupes d'automorphismes sont isomorphes

Démonstration

Soient $G_1 = (X_1, U_1, e_1)$ et $G_2 = (X_2, U_2 e_2)$ deux graphes isomorphes et soit μ_x la correspondance biunivoque qui applique X_1 sur X_2. Soient $H(G_1)$ et $H(G_2)$ les groupes d'automorphismes des graphes Soient H_1 et H_2 les groupes de permutations qui leur sont isomorphes. Ce sont des permutations qui respectent les arcs. L'application $\phi : H_1 \rightarrow H_2$ qui a ν_1 de H_1 associe ν_2 de H_2.

définie par $\nu_2 = \mu_x . \nu_1 . \mu_x^{-1}$ est un isomorphisme de H_1 sur H_2.

2.5.2 - Proposition 2

Tout isomorphisme entre deux graphes, se construit en faisant le produit d'un isomorphisme entre les deux graphes par un automorphisme quelconque de l'un des deux graphes, le produit se faisant à droite ou à gauche suivant que l'automorphisme est défini sur le graphe de départ ou sur le graphe d'arrivée de l'isomorphisme.

Démonstration :

A chaque automorphisme est associé au moins une correspondance biunivoque μ_x entre les sommets des deux graphes : $\mu_x : X_1 \longrightarrow X_2$. Soit ν_2 une permutation des sommets de X_2 du groupe H_2 isomorphe au groupe d'automorphismes $H(G_2)$ du graphe G_2. $\mu'_x = \nu_2 \mu_x$ est encore une correspondance biunivoque entre les sommets des deux graphes. Comme μ_x et ν_2 respectent les arcs, il en est de même de μ'_x. D'où à μ'_x est associé un isomorphisme qui s'obtient en faisant le produit à gauche de l'isomorphisme associé à μ_x par l'automorphisme associé à ν_2. On peut faire le même raisonnement evec un automorphisme ν_1 de H_1 et le produit à droite.

D'autre part pour tout couple d'isomorphismes de G_1 dans G_2 il existe un automorphisme de G_1 tel que l'un d'eux s'obtient à l'aide du produit à droite de l'autre par l'automorphisme de G_1 (même raisonnement avec un automorphisme de G_2 et le produit à gauche). En appelant μ_x et μ'_x les correspondances biunivoques qui associent X_1 à X_2, associées aux automorphismes de G_1 dans G_2 alors $\mu'_x \cdot \mu_x^{-1}$ est une application de X_1 dans X_2, biunivoque, qui respecte les arcs. Il lui est donc associé un automorphisme de G_1.
On peut alors construire à partir d'un isomorphisme et du groupe d'automorphismes de l'un des graphes tous les isomorphismes qui existent entre les deux graphes.

2.5.3. - Proposition 3

Il existe une correspondance biunivoque entre les graphes isomorphes à un graphe donné G et les classes dans Sn du sous-groupe H isomorphe au groupe des automorphismes H(G) du graphe donné à condition que tous ces graphes soient indicés par le même ensemble.

Démonstration

Indiçons par le même ensemble les sommets G et ceux de ses graphes isomorphes. Si μ_x est la correspondance biunivoque entre les sommets de G et ceux d'un graphe G' qui lui est isomorphe, associée à un isomorphisme, les correspondances associées à tous les isomorphismes entre G et G' constituent le sous-ensemble $\mu_x H$ de Sn, qui est une classe dans Sn du sous-groupe H (μ_x est une permutation). Inversement, une permutation quelconque des indices des sommets d'un graphe transforme ce graphe en un graphe isomorphe. La correspondance biunivoque μ_x est justement la permutation, et tous les autres isomorphismes se construiront à partir de la classe de la permutation.

Conséquence

Le nombre de graphes isomorphes à un graphe donné G est l'indice

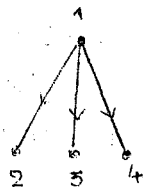
du groupe H isomorphe au groupe H(G) des automorphismes de G, dans le groupe des permutations Sn

2.5.4 - Exemple

Soit le graphe G = (X,U,e) de 4 sommets représenté par le vecteur Γ: 1(2,3,4). On a n = 4, m = 3, d'où

X = {1,2,3,4} , U = {1,2,3} et

e(U) = {1,2,), (1,3), (1,4)}



On constate qu'on peut échanger les indices des sommets 2,3,4 sans modifier le graphe. Le groupe H(G) est isomorphe au sous-groupe H du groupe S_4 des permutations de 4 objets qui laissent l'objet 1 invariant. H est isomorphe au groupe S_3 des permutations de 3 objets.

Les graphes isomorphes à ce graphe, se déduisent de ce dernier par des permutations qui ne laissent pas le sommet 1 invariant. Comme on ne peut appliquer que les sommets 2,3,4 sur le sommet 1, à toutes les permutations qui appliquent un de ces sommets sur le sommet 1 correspond un graphe isomorphe, dont le groupe d'automorphismes est isomorphe au sous-groupe des permutations de 4 objets qui laissent le sommet appliqué sur le sommet 1 invariant.

Il existe trois graphes isomorphes à G, et 6 isomorphismes pour chacun d'eux. Si on représente le graphe par 1(2,3,4) on aura

a) ses automorphismes qui appliquent le graphe sur

H(G) = {1(2,3,4), 1(3,4,2),1(4,2,3),1(3,2,4),1(2,4,3),1(4,3,2)}

b) ses isomorphismes qui appliquent le graphe sur

$H(G_1)$ = {2(3,4,1),2(4,1,3),2(1,3,4),2(4,3,1),2(3,1,4),2(1,4,3)}

$H(G_2)$ = {3(4,1,2),3(1,2,4),3(2,4,1),3(1,4,2), 3(4,2,1), 3(2,1,4)}

$H(G_3)$ = {4(1,2,3),4(2,3,1),4(3,1,2),4(2,1,3),4(1,3,2),4(3,2,1)}

2.6. - Sommets équivalents

A l'aide du groupe d'automorphismes d'un graphe G on peut définir sur l'ensemble des sommets des relations d'équivalence

2.6.1. - H-équivalence

Déf. 2.6.1 Soit H le sous-groupe du groupe des permutations Sn isomorphe au groupe H(G) des automorphismes d'un graphe G = (X,U,e) de n sommets.
2 sommets x et y de X sont H-équivalents s il existe une permutation ν de H telle que $y = \nu x$

$$x \underset{H}{\sim} y \iff \exists \nu \in H \quad y = \nu x$$

Alors la H-classe H_x d'un sommet x pour cette relation d'équivalence sera définie par $H_x = \{y \mid y \in X, \exists \nu \in H : y = \nu x\} = H.x$.
Il n'est pas immédiat de calculer la H-partition de X associée à cette équivalence sans connaître le groupe H.

2.6.2 - S-équivalence

Déf. 2.6.2 2 sommets x et y de X sont S-équivalents si la transposition τ_{xy} appartient à H

$$x \underset{S}{\sim} y \iff \tau_{xy} \in H$$

En considérant que l'identité est une transpposition particulière ($\tau_{xx} = I$) on a une relation d'équivalence
La S-classe S_x d'un sommet x sera définie par

$$S_x = \{y \mid y \in X , \ y = \tau_{xy}, \ \tau_{xy} \in H\}$$

De la définition des automorphismes on déduit

prop. 2.6.2 τ_{xy} appartient à H signfie que x et y ont
- mêmes successeurs,
- mêmes prédécesseurs,
- et sont ou ne sont pas tous les deux support d'une boucle.

il y en a autant dans chaque sens.

Ce résultat a une grande importance pratique car il sera aisé de construire la S-partition de X, ayant découvert les transpositions de H par les sommets sur lesquels ils agissent.

2.6.3. - Comparaison des équivalences

prop.
2.6.3.1 |La S-équivalence implique la H-équivalence mais non l'inverse

d'où la S-partition est plus fine que la H-partition de X. En effet si x est S-équivalent à y, il existe $\nu = \tau_{xy}$ e H telle que $y = \nu x$, d'où x est H-équivalent à y. L'inverse n'est pas vrai. Même si les permutations de H peuvent e décomposer en produits de transpositions il n'en est pas moins vrai que ces transpositions individuellement ne sont pas nécessairement des éléments.

prop.
2.6.3.2 |Les S-classes contenues dans une H-classe ont même cardinal

Démonstration - Si un sommet est contenu dans une H-classe, sa S-classe y est entièrement contenue d'après la proposition 2.6.3. Soient alors deux sommets x et y qui ne sont pas équivalents pour S mais qui le sont pour H. Ils sont dans une même H-classe et il existe une permutation ν de H telle que $\nu x = y$. Si ν est une application biunivoque entre S_x et S_y, elles auront même cardinal.
Soient z de S_x différents de x. Il existe une transposition τ_{xz} de H telle que $z = \tau_{xz} x$. Posons $t = \nu z$. t est S-équivalent à

y car $\tau_{yt} = \nu \tau_{xz} \nu^{-1}$. A deux sommets z_1 et z_2 de S_x différents, seront associés deux sommets t_1 et t_2 différents de S_y, car ν est biunivoque sur X.
L'importance de cette propriété apparaîtra dans l'étude des fonctions intrinsèques.

2.6.4 - Sommets invariants par H

Déf. 2.6.4. | I_4 est l'ensemble des sommets invariants par H

| $I_4 = \{x | x \in X, \forall \nu \in H \quad \nu x = x\}$

Prop. 2.6.4 | Chaque sommet de I_4 est une H-classe et une S-classe

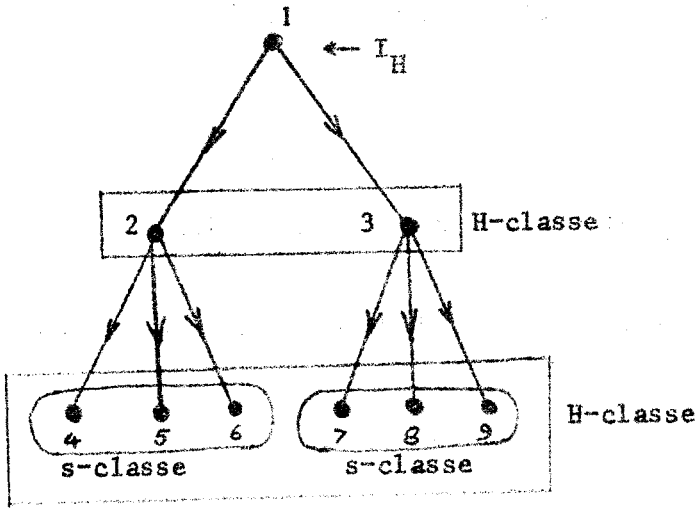
En effet H.x = x pour tout x appartenant à I_H, d'autre part $\tau_{xx} = I$ pour chacun d'eux.

2.6.5 - Exemples

Voir Figures 6,7 et 8.

S-équivalence, H-équivalence, I_H;

Fig.6



deux S-classes non réduites à 1 sommet

deux H-classes non réduites à 1 sommet $I_H = \{1\}$

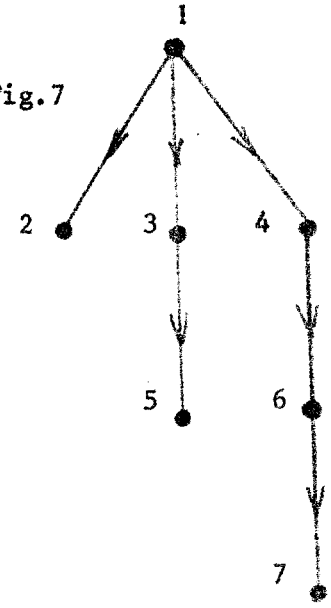
Le groupe H se schématise comme suit :

$H = S_3(\text{sur } 4,5,6) \times S_3(\text{sur } 7,8,9) \times r_1$

où $r_1 = [2,3][4,7][5,8][6,9]$.

note : [a,b] est un cycle de permutation

Fig.7

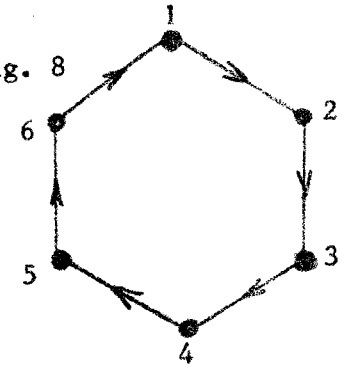


$I_H = X$

Toutes les S et H-classes se réduisent à un sommet

H = I{(identité)}

Fig. 8



pas de s-classes non réduites à 1 sommet

1 seule H-classe : X

$I_H = \emptyset$

H est par la permutation cyclique des 6 sommets

$H = \{I, [1,2,3,4,5,6]^p,\ p \leq 6\}$

2.6.6 - Comparaison de partitions.

2.6.6.1 - Précision de vocabulaire

On dira qu'une partition p est plus fine qu'une partition q si toute classe de p est contenue dans une classe de p.
On dira qu'une partition p est strictement plus fine qu'une partition q si p est plus fine que la partition q et si elle contient plus de classes que la partition q. Il en résulte alors que au moins une classe de q contient au moins deux classes de p. La proposition 2.6.3.1 signifie que la S - partition est strictement plus fine que la H - partition.

2.6.6.2 - Théorème

Soit un ensemble de n objets : X.
Soit une partition p de ces objets et la p-équivalence correspondante. f est la fonction associée à la partition p :
$x \underset{p}{\sim} y \rightarrow f(x) = f(y)$
Soient les sous-groupes k de S_n dont les k-partitions associées sont plus fines que la partition p. Soit h un des sous-groupes k tel qu'il n'existe pas de k-partition strictement moins fine.

Soit $I : X \rightarrow P(X)$ une fonction vérifiant $\forall x \in X, \forall \nu \in k$
$I(\nu x) = \nu (Ix)$ une fonction vérifiant cette propriété sur un groupe contenant h, la vérifie sur h.

f est une fonction vérifiant $\forall x \in X, \forall \nu \in h \quad f(\nu x) = f(x)$.

Elle vérifie cette propriété pour tout sous-groupe de h et même pour tout sous-groupe k.

Soit l'équivalence q :
$x \underset{q}{\sim} y \rightarrow x \underset{p}{\sim} y$ et $\forall A$ une classe de p $| bx \cap A| = |by \cap A|$

Théorème

La q-partition est moins fine que la h-partition.
On trouvera fig. 18 (chapitre 4) une démonstration de ce théorème. Il explique les insuffisances de la méthode des deux fonctions et justifie la méthode des hypothèses.

2.7 - Equivalence et Isomorphie

2.7.1 - Correspondance des classes

prop 2.7.1 | L'isomorphisme respecte les classes.

C'est-à-dire que les isomorphismes qui existent entre deux graphes établissent une correspondance biunivoque entre les ensembles de classes de même cardinal.

Cette propriété résulte de ce que les groupes d'automorphismes de deux graphes isomorphes sont isomorphes.

Soient deux graphes $G_1 = (X_1, U_1, e_1)$ et $G_2 = (X_2, U_2, e_2)$ isomorphes.

Soient x et y deux sommets d'une même classe de G_1, et soient z et t leurs correspondants respectifs dans G_2. Soit ν_1 la permutation de H qui fait passer de x à y dans G_1, alors $\nu_2 = \mu\, \nu_1 \mu^{-1}$ est une permutation de H qui fait passer de z à t. Ce raisonnement est aussi valable si ν_1 est une transposition, auquel cas ν_2 est aussi une transposition. D'où il ressort que z et t sont dans une même classe dans G_2. Du fait que μ est une correspondance biunivoque la classe de x dans G_1 et celle de son correspondant y dans G_2 sont alors de même cardinal. On constate donc qu'une classe de G_1 ne peut correspondre qu'à une classe de G_2 de même cardinal.

Cette propriété sera utilisée pour simplifier la recherche de la correspondance biunivoque associée à un isomorphisme.

2.7.2 - Correspondance entre les sommets de S-classes correspondantes

prop 2.7.2 | Les sommets de deux S-classes correspondantes peuvent être mis en correspondance d'une façon arbitraire.

Ceci résulte de ce que les transpositions associées à des S-classes engendrent un sous-groupe de H isomorphe au groupe S_p des permutations de p objets si les S-classes ont p pour cardinal, et de la propriété 2 de 2.5. On passera d'un isomorphisme à un autre par produit de ce premier par une transposition, ce qui revient à échanger deux sommets d'une S-classe de l'un ou l'autre graphe dans la correspondance.

2.8. - Restrictions du problème

2.8.1 - Simplifications

On a intérêt, comme dans tous les problèmes de combinatoire à diminuer au maximum le nombre de choix à faire dans le cheminement vers la solution, ou à se ramener par décomposition du problème à des cas de taille plus réduite. En général le problème n'en reste pas moins encombrant et nécessite de puissants moyens pour être résolu.

Dans le cas qui nous intéresse, il est possible éventuellement de remplacer les graphes par d'autres graphes dont on sait qu'ils ont même groupe d'automorphismes, en utilisant en particulier les indications données au paragraphe 2.4.2.

Le graphe complémentaire peut avoir moins d'arcs, par exemple, ce qui diminue le temps de calcul des chemins, toutefois il risque de ne plus être connexe.

2.8.2. - Graphes non connexes

Pour de tels graphes on conçoit très bien que l'isomorphisme entre les graphes, implique une correspondance biunivoque entre les composantes connexes. Deux composantes correspondantes sont des sous-graphes isomorphes. On sera donc amené à chercher des isomorphies entre les composantes connexes. D'où il résulte que le problème de l'isomorphisme de graphe doit se résoudre dans les graphes connexes.

Pour un graphe non-connexe, le problème garde tout son encombrement si le nombre de composantes est élevé, car il y a beaucoup d'isomorphies ; éventuels à tester.

2.9. - Représentation d'un isomorphisme

Il s'agit de représenter une correspondance biunivoque entre les éléments de deux ensembles. Il existe de nombreuses méthodes.
Par exemple une matrice indicée en ligne par les éléments de l'un des ensembles, en colonne par les éléments de l'autre ensemble, et n'ayant qu'un élément non nul par ligne et par colonne : une matrice unité permutée.
Cependant voici deux méthodes qui permettent aussi de représenter une correspondance entre deux ensembles X_1 et X_2, qu'on sait biunivoque, mais dont on ne sait seulement faire correspondre biunivoquement que des sous-ensembles (de même cardinal).

2.9.1. - Liste d'associations

C'est une liste des couples de sous-ensembles correspondants. On peut écrire les sommets du premier sous-ensemble de X_1 suivis entre parenthèses des sommets du sous-ensemble correspondant de X_2, puis placer le sous-ensemble suivant de X_1 suivi de son correspondant dans X_2.

Exemple : Pour les graphes suivants étudiés en 2.5.3 :

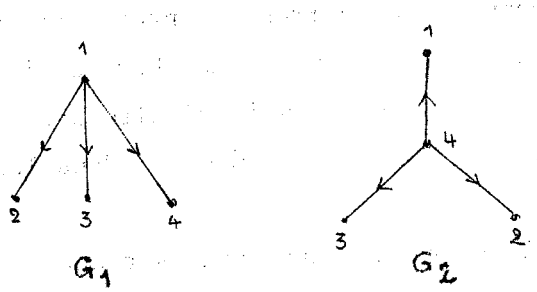


Fig. 11

On peut établir la correspondance
1(4) 2,3,4(1,2,3),
mais on ne peut pas dire si le sommet 2 de G_1 correspond nécessairement au sommet 1 de G_2. Cela résulterait d'un choix à faire du fait que les sommets 2,3,4 de G_1 forment une S-classe, ainsi que 1,2,3 de G_2.
Un isomorphisme serait par exemple
1(4)2(1)3(3)4(2)

2.9.2.- tableaux d'associations

Affectons des valeurs différentes aux sous-ensembles mis en correspondance, et choisissons arbitrairement une correspondance biunivoque entre les sommets de ces sous-ensembles.

Le tableau d'associations aura alors trois rangées dont les deux premières contiendront les couples de sommets correspondants et la troisième la valeur affectée à leurs sous-ensembles.

Dans un tel tableau, si les valeurs dans la troisième rangée sont différentes, la correspondance entre les sommets sera biunivoque, sinon alors on pourra arbitrairement échanger les sommets de l'une des deux premières rangées si les valeurs en troisième rangée sont égales.

Exemple : la liste d'associations précédente devient :

G_1 : 1 2 3 4

G_2 : 4 1 2 3

Val.: 1 2 2 2

tandis que l'isomorphe serait

G_1 : 1 2 3 4

G_2 : 4 1 2 3

Val.: 1 2 3 4

III. EXISTENCE D'UN ISOMORPHISME

3 - Existence d'un isomorphisme

C'est la première étape du problème. On ne cherche pas encore à construire un isomorphisme mais seulement son existence. Pour cela nous allons essayer d'établir des conditions d'isomorphie entre deux graphes.

3.1- Conditions d'isomorphie

Une condition suffisante vérifiée permettra de dire que deux graphes sont isomorphes. Si une condition nécessaire n'est pas vérifiée, alors les deux graphes ne seront pas isomorphes.
En fait, il n'a pas encore été trouvées de conditions suffisantes effectivement faciles à vérifier. Elles nécessitent souvent de longs calculs.
Par contre on connait de nombreuses conditions nécessaires, faciles à établir et à vérifier. Il est alors possible de résoudre le problème inverse : Déterminer si deux graphes ne sont pas isomorphes.
Ce problème a son importance quand il s'agit de reconnaître les graphes isomorphes parmi un ensemble de graphes engendrés automatiquement. En utilisant les conditions nécessaires on pourra sur les graphes en question définir une relation d'équivalence qui dira que deux graphes sont équivalents s'ils vérifient les conditions nécessaires qu'on a considérées. Deux graphes non-équivalents ne seront alors pas isomorphes. Il ne restera à comparer à l'aide des conditions suffisantes que les graphes d'une même classe, ce qui limite d'autant le nombre des comparaisons.
L'idéal serait d'avoir une condition nécessaire et suffisante d'isomorphisme

3.2 - Caractéristiques (DUIJVESTIJN)

3.2.1 - Définition

Déf. 3.2.1 Une caractéristique est un être mathématique, comparable à un autre de même nature, associé à un graphe et déterminé d'une façon unique à partir de sa représentation.
On se servira d'elles pour établir des conditions d'isomorphisme à l'aide de leur comparaison. Pour cela on les classe en deux types.

3.2.2 - Type des Caractéristiques

Déf.3.2.2 Une caractéristique est de type 1 si des graphes ayant des caractéristiques différentes ne sont pas isomorphes.
Une caractéristique est de type 2 si des graphes ayant des caractéristiques égales sont isomorphes.

Avec les caractéristiques de type 1 on pourra construire des conditions nécessaires d'isomorphie. avec les caractéristiques de type 2 on aura des conditions suffisantes.
Réciproquement en exprimant des conditions d'isomorphisme sous des formes particulières, on obtiendra des caractéristiques.

3.3 - Conditions nécessaires

Elles se déduisant de la définition de l'isomorphie de deux graphes.

3.3.1 - Degré des sommets correspondants

3.3.1.1 - Proposition

prop.3.3.1.1 Les sommets correspondants de deux graphes isomorphes ont mêmes demi-degrès intérieur et extérieur, et sont support du même nombre de boucles.

Démonst. Pour deux graphes isomorphes G_1 et G_2 considérons deux sommets correspondants x_1 et x_2. Tout successeur (prédécesseur) de x_1 correspond à un successeur (prédécesseur) de x_2, et vice-versa, à cause de la définition de l'isomorphie (2.3.1), puisqu'entre un sommet et chacun de ces successeurs (prédécesseurs) il existe un arc, et cet arc est respecté par tout l'isomorphisme. Il en résulte que deux sommets correspondants ont même nombre de successeurs (prédécesseurs) d'où même demi-degré extérieur (intérieur).

Si le sommet x_1 est support d'une boucle il est successeur et prédécesseur de lui-même. Son correspondant sera aussi support d'une boucle car cet arc est aussi respecté par tout l'isomorphisme.

Pour les multigraphes si entre deux sommets il existe plusieurs arcs de même sens, ils sont autant de fois prédécesseurs et successeurs l'un de l'autre suivant le sens des arcs.

3.3.1.2 - Conséquences

On en déduit tout de suite

Prop 3.3.1.2

Si deux graphes sont isomorphes ils ont même nombre de sommets de mêmes demi-degrés, et même nombre de sommets supports du même nombre de boucles.

Une autre conséquence sera utilisée dans la construction d'un isomorphisme.... Dans la recherche de la correspondance entre sommet il sera inutile d'essayer l'association de deux sommets de demi-degrés différents. Par contre on associera les successeurs (prédécesseurs) d'un sommet de l'un des graphes avec les successeurs (prédécesseurs) de son correspondant dans l'autre graphe.

3.3.1.3 - Caractéristique associée

Dans la proposition 3.3.1.2 les éléments à comparer sont les nombres de sommets de même demi-degrés, ou les nombres de sommets support du même nombre de boucles. Ce sont ces nombres qu'il faut rassembler dans la caractéristique associée.
Ainsi on pourra par exemple pour les demi-degrés extérieurs construire pour chaque graphe un vecteur indicé par ces demi-degrés tel que la coordonnée d'indice i soit le nombre de sommets du graphe de demi-degré extérieur i. Ce vecteur aura au plus m composantes, si le graphe a m arcs.
On pourra pareillement en construire pour les demi-degrés extérieurs et pour le nombre de boucles par sommets.
Ce sont des caractéristiques de type 1 d'après leur construction. Il est aisé de construire des graphes non isomorphes qui auront mêmes caractéristiques de ce type. Ils prouveront que ces caractéristiques ne sont pas de type 2.
Pour les degrés et demi-degrés on peut utiliser les deux graphes suivants :

G_1 G_2

fig. 12

3.3.2 - Longueur des circuits

Nous avons vu qu'un isomorphisme entrainait une correspondance entre les arcs. De cette correspondance on peut tirer des résultats semblables à ceux obtenus pour les sommets.

3.3.2.1 proposition

prop 3.3.2.1

> Pour deux graphes isomorphes, si deux arcs sont adjacents dans l'un des graphes leurs correspondants le sont aussi dans l'autre graphe, de la même façon. (arrivant, partant ou se suivant en un sommet)

Ceci résulte de la définition de la correspondance entre les arcs, qui est construite à partir de la correspondance de leurs extrémités.

Il en résultera alors que

prop 3.3.2.2

> L'isomorphie entre deux graphes induit une correspondance biunivoque entre les cycles et entre les circuits.

Pour s'en rendre compte il suffit d'appliquer la proposition précédente à chacun des sommets d'un circuit ou d'un cycle d'un graphe. On construira dans l'autre graphe un circuit ou un cycle correspondant, de même longueur. Comme deux circuits ou deux cycles distincts diffèrent au moins par un arc, les circuits ou cycles correspondants diffèreront aussi par les arcs correspondant à ceux utilisés pour la différenciation dans l'autre graphe. Il en résulte que la correspondance est biunivoque.

3.3.2.2 - Conséquences

Comme pour les sommets on peut dire que

prop 3.3.2.2.1

> Si deux graphes sont isomorphes ils ont même nombre de cycles et de circuits de même longueur.

Une autre conséquence à utiliser dans la recherche de la correspondance entre les sommets est que

prop 3.3.2.2.2

> Pour deux graphes isomorphes, par des sommets correspondants il passe le même nombre de circuits et cycles de même longueur.

3.3.2.3 - Caractéristique associée

Elle se construira pour chaque graphe de la même façon qu'au paragraphe 3.3.1.3. Pour les circuits, par exemple, ce sera un vecteur indicé par les longueurs de circuits tel que la composante d'indice i soit le nombre de circuits de longueur i dans le graphe. La dimension maximale de ce vecteur est encore m, le nombre d'arcs.

Remarquons que pour chaque sommet on pourrait construire des vecteurs du même type indiquant le nombre de circuits (cycles) de chaque longueur passant par ce sommet, et au cours de la construction d'un isomorphisme, essayer d'associer des sommets de mêmes vecteurs.

Ces caractéristiques de type 1 sont malheureusement longues à calculer. Les circuits, par exemple, peuvent se calculer à l'aide des éléments diagonaux des puissances de la matrice caractéristique du graphe.

On vérifiera encore sur des exemples qu'elles ne sont pas de type 2. (voir fig. 13)

3.3.3 - Une caractéristique de type 1 : la matrice D

3.3.3.1 - Arcs et degrés

Les caractéristiques que nous avons vues consistent à indiquer le nombre de sommets d'un graphe qui vérifient une propriété quantitative. En voici une construite sur les arcs qui vérifient une propriété quantitative.

Si la propriété quantitative ne dépend que des arcs, on construira un vecteur, mais si elle dépend des sommets, comme les arcs ont deux extrémités il faudra utiliser une matrice.

Supposons que la propriété en question soit qu'"un arc relie un sommet de degré i à un sommet de degré j." On construira alors la matrice D, indicée en lignes et en colonnes par les degrés des sommets et telle que D_{ij} sera le nombre d'arcs reliant un sommet de degré i à un sommet de degré j.

Il est aisé de démontrer à partir de la proposition 3.3.1.1, de la définition 2.3.1 de l'isomorphie et de la définition de la correspondance entre les arcs, que la matrice D est une caractéristique de type 1.

Les graphes de la fig. 12 qui ont même matrice D et qui ne sont pas isomorphes montrent qu'elle n'est pas de type 2.

On trouvera fig. 14 un autre contre-exemple.

3.3.3.2 - Généralisation

Le processus de construction de ces caractéristiques est maintenant suffisamment démonté, pour qu'il ne soit plus nécessaire d'en donner des exemples. Indiquons simplement qu'on peut en inventer un grand nombre, et qu'il n'est pas nécessaire d'indicer par le même genre de quantité les lignes et colonnes de la matrice D.

Nous verrons plus loin (méthode des deux fonctions. 4.4.2.2.) une autre généralisation, et une méthode simple de calcul. Nous y verrons également l'incidence de cette caractéristique sur la recherche d'un isomorphisme.

3.3.4 - Les relations d'équivalence

3.3.4.1 - Un type de relations d'équivalence

Si deux graphes sont isomorphes, les relations d'équivalences entre les sommets doivent être respectées, à condition que celles-ci soient définies directement ou indirectement à l'aide des arcs. C'est le cas des H-et S-équivalences qui dépendent des arcs par le groupe d'automorphismes. Il est isomorphe au groupe des permutations des sommets qui respectent les arcs. Ce serait aussi le cas d'une relation qui indiquerait comme équivalents deux sommets qui ont même degré. Mais si on dit que deux sommets sont équivalents si leurs indices sont pairs, on obtiendra une équivalence qui ne rentre pas dans le genre envisagé. Ce genre de définition d'équivalences est nécessité par le fait qu'un isomorphisme est une correspondance entre les sommets associée à une correspondance entre les arcs.

prop 3.3.4.1

> Un isomorphisme respecte les équivalences entre les sommets si elles sont construites à l'aide des arcs.

3.3.4.2 - Conséquences

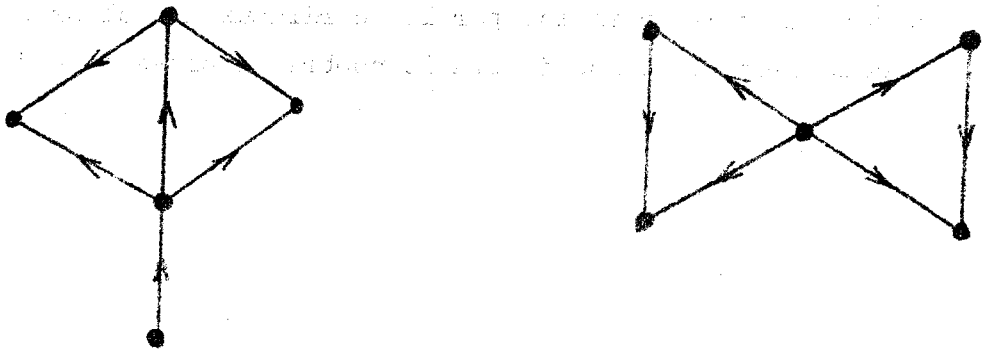
L'isomorphisme induira alors une correspondance biunivoque entre les classes d'équivalence des deux graphes, parce que, comme pour la H et S-équivalence, deux sommets équivalents dans l'un des graphes ne pourront correspondre qu'à deux sommets équivalents dans l'autre graphe. On peut établir

prop 3.3.4.2

> Si deux graphes sont isomorphes, ils ont pour toute relation d'équivalence du type 3.3.4.1 même nombre de classes de même cardinal.

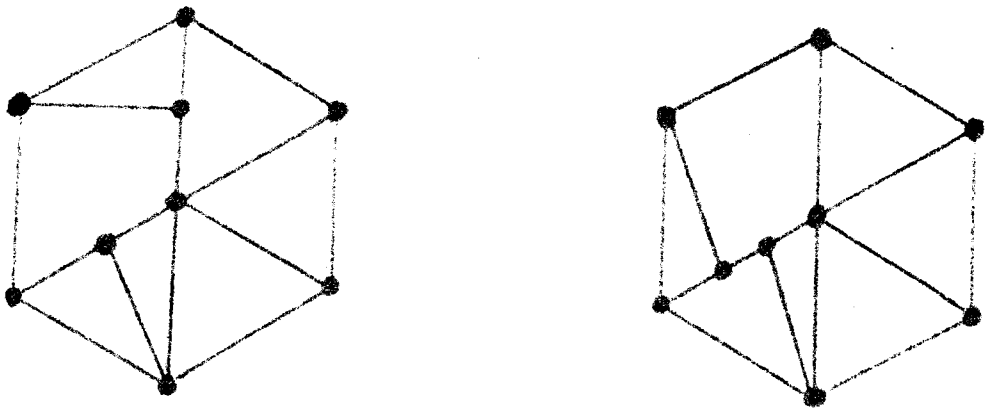
En suivant le même raisonnement que pour les degrés des sommets on tirera d'autres conséquences utiles à la recherche d'un'isomorphisme, et on pourra construire une caractéristique.

fig. 13 2 graphes n'ayant que 2 cycles élémentaires de longueur 3 non-isomorphes.



G_1 G_2

fig. 14 2 graphes symétriques (on confond en un arc non orienté deux arcs de sens contraires)



G_1 G_2

Pour ces graphes les degrés, les circuits et la matrice D ne sont pas suffisants pour indiquer qu'ils ne sont pas isomorphes.

3.3.4.3 - Caractéristique

Ce sera un vecteur indicé par les cardinaux des classes et dont la composante d'indice j sera le nombre de classes de j sommets.

3.3.5 - Les propriétés des graphes

Pour la détermination de non-isomorphie de deux graphes il est naturel d'utiliser toutes les informations que l'on a, ou que l'on peut avoir sur les graphes.
Aussi de la définition on déduira que

prop 3.3.5

> Si deux graphes sont isomorphes ils sont ensemble symétriques ou non-symétriques

et donc qu'un graphe symétrique et un graphe non-symétrique ne sont pas isomorphes.
De même de l'étude précédente on saura qu'un arbre (graphe connexe sans cycle) et qu'un graphe hamiltonien (il existe un circuit passant par tous les sommets) ne peuvent être isomorphes. L'inconvénient est que de telles propriétés ne se constatent pas aisément.
Même si on pouvait trouver un grand nombre de telles propriétés, elles n'auront pas un intérêt pratique car le plus souvent le problème de l'isomorphie se pose pour des graphes construits intentionnellement pour vérifier une propriété, qui en général en exclue beaucoup d'autres.
Remarquons toutefois ceci : bien que nous ne l'ayons pas exprimé, l'isomorphie entre deux graphes induit une correspondance entre les chaînes et chemins des graphes. Si on sait que les cycles et circuits sont des chaînes et chemins fermés, on n'aura pas de difficultés à le montrer. Or pour rechercher un isomorphisme entre deux graphes on utilise directement ou indirectement les chemins et les chaînes dans les deux graphes, qu'il faut alors calculer explicitement ou implicitement. Moyennant quelques calculs supplémentaires on peut faire apparaître des propriétés qui pourront être utilisées pour tester la non-isomorphie .

On peut ainsi faire apparaître la dimension et le nombre des composantes fortement connexes. Il existe même une relation d'équivalence entre les sommets apparentée aux composantes fortement connexes qui se construit à partir de l'égalité de couples de lignes et colonnes de la matrice de la fermeture transitive du graphe. Cette dernière est la somme des puissances de la matrice caractéristique du graphe, et il existe des algorithmes très rapides pour la calculer. Deux graphes isomorphes ont mêmes nombres de classes de même cardinal pour la relation d'équivalence ci-dessus.

3.4 - Conditions suffisantes

Nous revenons au problème de l'existence d'un isomorphisme. La première condition suffisante est la définition même de l'isomorphie de deux graphes. Les autres ne seront que des expressions de cette condition, comme par exemple la P-équivalence des matrices caractéristiques des deux graphes.
Pour être effectivement utile une condition suffisante doit permettre d'éviter d'envisager les permutations éventuelles de l'ensemble des sommets. Pratiquement il n'en existe pas, du moins, à notre connaissance.
Les caractéristiques de type 2 qui sont associées aux conditions suffisantes sont aussi assujetties aux mêmes contraintes, et celles qui sont connues ne sont pas non plus d'un emploi aisé. En voici, toutefois, une intéressante pour son principe de construction.

3.4.1 - Expression par un nombre de la matrice caractéristique

Juxtaposons les lignes de la matrice caractéristique d'un graphe. On obtient un vecteur dont les composantes sont les chiffres d'un nombre écrit en binaire s'il s'agit d'un graphe ordinaire, en base P+1 s'il s'agit d'un p-graphe. L'expression de ce nombre en base 10 représente d'une façon unique la matrice, et est pour les graphes une caractéristique de type 2.
Pour un graphe simple, son expression est

$$K = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} \cdot 2^{(n-i)\,n\,+n-j}$$

On peut même, pour gagner du temps en calcul, quand le graphe est symétrique, sans boucle, appliquer ce processus à la partie triangulaire supérieure de la matrice caractéristique.

L'utilisation pratique de ce nombre en calculateur n'est faisable que si tout le nombre peut être représenté, et comme c'est un grand nombre (de l'ordre de 2^{n^2}) il faut que mémoire n'ait pas une structure de mot, ce qui actuellement est rare. L'encombrement de ce nombre est évident si on se souvient qu'il représente une matrice.

Si deux graphes ont même nombre K ils seront isomorphes, car cela signifie que les matrices caractéristiques des deux graphes sont égales. Il ne faut pas être grand clerc pour se rendre compte que la comparaison directe des matrices caractéristiques sera plus rapide que la comparaison de leur nombre K qui pour être calculé, nécessite d'isoler successivement les éléments de la matrice, comme dans une comparaison.

3.5 - Conditions nécessaires et suffisantes

Pas plus que pour les conditions suffisantes, il n'existe de conditions nécessaires et suffisantes d'existence d'isomorphie qui soit valable.

Ici encore il faut éviter d'utiliser toutes permutations des sommets. On cherche alors à numéroter les sommets des graphes d'une façon uniforme dans tous les graphes. On pourrait essayer par exemple de faire en sorte que la matrice caractéristique ait le plus d'éléments non nuls en haut et à gauche. Il n'existe pas de méthode valable.

La magnitude d'identification qui est le maximum du nombre K pour les différentes permutations de la matrice caractéristique est un essai dans ce sens, mais pour le calculer il faut envisager les permutations. Il n'en est pas moins vrai, qu'elle sera indépendante de l'ordre des sommets pour un graphe donné et de ce fait sera une caractéristique de type 1 et 2.

3.6 - Conclusion

Il résulte en fin de compte qu'il est relativement facile de conclure à l'inexistence d'un isomorphisme entre graphes, mais que la meilleure façon de prouver son existence est d'en exhiber un. Pour cela il faut le construire, et c'est l'objet du chapitre suivant.

IV - METHODE DE RECHERCHE D'UN ISOMORPHISME

4 - Méthode de recherche d'un isomorphisme.

4. I. Principe

Ce chapître contient d'abord la description du matériel mathématique et ensuite son emploi.

On cherche une correspondance biunivoque qui respecte les arcs, entre les sommets de deux graphes susceptibles d'être isomorphes. Pour cela on établit dans chacun des graphes une relation d'équivalence entre les sommets et on construit entre les classes des partitions associées une correspondance biunivoque compatible avec l'isomorphisme éventuel des deux graphes, c'est-à-dire que les sommets d'une classe dans l'un des graphes ne pourront correspondre qu'aux sommets de la classe correspondante dans l'autre graphe.

On affine ensuite les partitions obtenues à l'aide d'une autre relation d'équivalence ou d'un procédé qui fait intervenir les classes des voisins (successeurs et prédécesseurs) de chacun des sommets jusqu'à obtenir la partition associée à la S - équivalence. Si on a pris la précaution de conserver constamment la correspondance entre les classes des deux graphes, on obtient à la fin une correspondance entre les S-classes. En se rappelant que les sommets de deux S-classes correspondantes peuvent être associés arbitrairement, on obtiendra l'isomorphisme cherché (prop. 2.6.5.2).

Il se peut qu' à une étape donnée on ne puisse plus affiner la partition. On utilisera alors un procédé qui consiste à associer arbitrairement des sommets. Il faudra dans la suite de la recherche vérifier que ces associations sont acceptables, et éventuellement en choisir d'autres.

L'isomorphisme peut être obtenu également dans le cas où la partition a été affinée jusqu'à ne contenir qu'un sommet par classe. La correspondance b-iunivoque entre les sommets est la même que celle qui existe entre les classes.

4. 2. Fonctions intrinsèques.

Ce sont elles qui permettent d'obtenir des relations d'équivalence entre les sommets des graphes.

4. 2. I. Définition F.I.

Déf.4.2.I. Soient un graphe $G = (X,U,e)$ et un ensemble F quelconque.
Une fonction $f : X \longrightarrow F$ est <u>intrinsèque</u> si l'image d'un sommet par cette fonction est invariante par les automorphismes du graphe.
C'est-à-dire
$\forall \gamma \in H, \forall x \in X \qquad f(\gamma x) = f(x)$.

L'ensemble F qui en général est l'ensemble N des entiers ne sert qu'à distinguer des sommets qui ont des images distinctes par la fonction f. Il ne présente en fait que peu d'intérêt et on peut le choisir de façon à ce que son emploi soit le plus agréable possible.

4. 2. 2. Exemples de fonctions intrinsèques.

On les construit à l'aide de conditions nécessaires d'isomorphismes, plus précisément de leurs conséquences sur les sommets.

En prenant pour ensemble F l'ensemble des demi-degrés intérieurs possibles sur les graphes (l'ensemble $\mathbb{N}^+$) à la fonction $f : X \longrightarrow F$ qui à chaque sommet associe son demi-degré intérieur est une fonction intrinsèque. En effet un automorphisme est un isomorphisme de ce graphe dans lui-même et d'après la prop. 3.3.I. les sommets de mêmes 1/2 degrés intérieurs ne peuvent se permuter qu'entre eux.

Par ce même procédé et à l'aide des conditions nécessaires définies en 3.3. , on pourra construire des fonctions intrinsèques. Si on exprime par la flèche $x \longmapsto$ la phrase "qui à x associe", on pourra établir la liste suivante

$f : x \longmapsto$ n où n = nombre de sommets du graphe

$f : x \longmapsto$ degré du sommet x

$f : x \longmapsto$ demi-degré intérieur du sommet x

$f' : x \longmapsto$ demi-degré extérieur du sommet x

$f : x \longmapsto$ le nombre de boucles sur le sommet x

$f_1 : x \longmapsto$ I si x appartient à un circuit de longueur 1, 0 sinon

$f : x \longmapsto$ le nombre de circuits passant par x

etc....

On verra au paragraphe 4. 3. d'autres fonctions intrinsèques.

4. 2. 3. F.I. - équivalence.

A une fonction est associée la relation d'équivalence suivante "deux éléments sont équivalents si et seulement si ils ont même image par la fonction". Pour une fonction intrinsèque cette équivalence définie sur X sera appelée la F.I - équivalence. Les F.I. - classes seront les images réciproques de chaque sommet par la fonction intrinsèque.

A chaque fonction intrinsèque sera associée une F.I. - partition .

A partir de deux d'entre-elles on peut en créer une troisième formée par les intersections des classes des deux partitions. La relation d'équivalence associée est "deux sommets d'un graphe sont équivalents si et seulement s'ils ont même image par chacune des deux fonctions". En généralisant à toutes les fonctions intrinsèques on obtiendra une partition de l'ensemble X des sommets d'un graphe qui sera plus fine que toutes les autres.

4. 2. 4. FI-équivalence et H -équivalence.

On peut connaitre une partition plus fine que toutes les F.I.-partitions.

Prop. 4.2.4. La H - équivalence implique la FI-équivalence.

Ceci résulte de ce que d'après la définition 4.2.I une fonction intrinsèque a même valeur pour tous les sommets des S - classes.

On en déduira que la H - partition est plus fine que toutes les F.I.-partitions.

4. 2. 5. Fonctior intrinsèque associée à une partition.

De la proposition 4.2.4. il résulte que si une partition est moins fine que la H - partition, on pourra construire une nouvelle fonction intrinsèque qui est la fonction associée à la partition.

Pour cela on numérote les classes de la partition, et on affecte à chaque sommet le numéro de sa classe.

Pratiquement les seuls cas où on est assuré que la partition est moins fine que la H - partition est celui où on construit cette partition à partir de deux F.I.-partitions. Au lieu de numéroter les classes de la partition obtenue on peut numéroter les couples des valeurs des deux fonctions intrinsèques pour chaque sommet. On obtiendra directement une nouvelle fonction qui est encore une fonction intrinsèque. (comparer avec 4.2.3)

Appelons ce processus "la juxtaposition de deux fonctions". En itérant cette juxtaposition de fonctions dans un ensemble de fonctions intrinsèques on obtient une fonction intrinsèque qui a un éventail de valeurs beaucoup plus large que toutes les autres.

4. 2. 6. F.I.-équivalence et isomorphie

4. 2. 6. I. Correspondance biunivoque entre les F.I.-classes de deux graphes.

Une fonction intrinsèque associe à chaque sommet d'un graphe une valeur qui exprime la mesure d'une propriété quantitative de ce sommet. Mesurons de la même façon ces propriétés dans les deux

graphes susceptibles d'être isomorphes. On fera correspondre les classes de chacun des graphes qui ont même valeur par la fonction intrinsèque.

Cette correspondance est compatible avec la correspondance biunivoque que l'on cherche à obtenir entre les sommets, c'est à dire qu'un sommet d'une F.I. - classe de l'un des graphes ne peut être associé qu'avec un sommet de la F.I.-classe correspondante dans l'autre graphe car on a vu en 2.6.5 qu'un isomorphisme respecte les H - classes.

Quand on utilise la "juxtaposition de deux fonctions" il faut prendre la précaution de numéroter simultanément les couples des valeurs des deux fonctions dans les deux graphes.

4.2.6.2. Condition nécessaire d'isomorphie.

Les F.I.-équivalences dépendent des arcs par le groupe d'automorphismes comme les H et S - équivalences. On peut leur établir les résultats obtenus en 3.3.4. et en particulier

Prop. 4.2.6.2. I.

> Si deux graphes sont isomorphes, ils ont même nombre de F.I.-classes de même cardinal.

En utilisant les remarques du paragraphe précédent on a un résultat plus fin.

Prop. 4.2.6.2. 2.

> Si deux graphes sont isomorphes les F.I.-classes correspondantes ont même cardinal.

4. 2. 6. 3. Fonctions intrinsèques et matrice D.

En se rappelant la définition "3.3.3." de la matrice D et sa généralisation on constate qu'on peut indicer les lignes par les valeurs d'une fonction intrinsèque, les colonnes par les valeurs de la même ou d'une autre fonction intrinsèque et définir D_{ij} par le nombre d'arcs reliant un sommet ayant la valeur i pour la fonction intrinsèque à un sommet ayant la valeur j pour la seconde fonction intrinsèque.

4. 3. Fonction de structure.

Ce sont les fonctions qui permettent d'affiner les partitions obtenues par les fonctions intrinsèques. Elles doivent répondre àdeux contraintes : conserver la correspondance biunivoque entre les classes, exprimer les liaisons entre les sommets dans chacun des graphes. C'est à cette seconde contrainte qu'elles doivent leur nom.

4. 3. I. Définition (FS).

Def. 4. 3. I. Pour un graphe $G = (X,U,e)$ une fonction de structure est une fonction $\gamma : X \dashrightarrow X^{\Rightarrow}$ (où $X^{\Rightarrow}$ est le monoïde commutatif engendré par X) qui à chaque sommet x de X associe un mot γx tel que pour toute permutation ν de H on ait $\gamma(\nu x) = \nu(\gamma x)$.

Dans cette définition $\nu(\gamma x)$ est le mot formé des transformés par la permutation ν des sommets de γx.

4. 3. 2. Exemple.

La fonction de structure la plus naturelle est la fonction successeur (2.I.3.3). A chaque sommet x on associe le mot formé par les sommets y qui sont exrémités des arcs issus de x. La définition des automorphismes permet d'affirmer qu'il s'agit bien d'une fonction de structure.

En voici d'autres :

Par les puissances de la fonction Γon envisagera les sommets y qui peuvent être atteints à partir de x par un chemin de longueur 1, s'il s'agit de Γ^1 ; et les sommets y', à partir desquels on peut atteindre x par un chemin de longueur 1, s'il s'agit de Γ^{-1} .

La notion de chemin peut être remplacée par celle de chaîne. Pour les construire on superpose au graphe étudié son inverse. Les chaînes sont les chemins dans les graphes obtenus.

Un raffinement de ce qui précède consiste à prendre les sommets qui peuvent être atteints par un chemin de longueur l, mais pas par un chemin de longueur plus petite. Ces sommets forment ce qu'on appelle l'écaille de taille l associée au sommet x qui a servi à la construire. Ces écailles ont un lien avec les chemins élémentaires les plus courts et les circuits élémentaires.

On peut également utiliser les cycles et circuits: associer à x les sommets qui appartiennent à un circuit passant par x. Ces sommets appartiennent à l'intersection des fermetures transitives

$$\Gamma^*(x) \text{ et } (\Gamma^{-1})^*(x)$$

On peut encore envisager parmi les successeurs d'ordre 1 d'un sommet, ceux qui sont d'un degré donné, ou ceux qu'on peut atteindre par plusieurs chemins de longueur 1, etc...

Toutes ces fonctions de structure se construisent à partir de la fonction Γ. Elles sont liées aux arcs par le groupe d'automorphismes, et la fonction Γ qui est une représentation du graphe contient toute l'information sur les liaisons des sommets par les arcs. Il n'est pas toujours possible de trouver une formule simple pour exprimer les fonctions de structure à partir des puissances de la fonction Γ.

4. 3. 3. Graphe associé à une fonction de structure.

4.3.3.1 Représentation des fonctions de structure.

Si on compare la définition des fonctions de structure avec celle de la fonction successeur Γ (2.1.3.3) on constate que dans les deux cas il s'agit d'associer un mot de X à un sommet x. On peut alors conclure que leurs représentations seront les mêmes. Et comme il s'agit de représentations de graphes, les autres méthodes de représentation des graphes seront aussi représentatives des fonctions de structure.

4. 3. 3. 2. Graphe associé.

A partir de la représentation de la fonction de structure on peut donc construire un graphe qui sera le graphe associé à la fonction de structure.

Déf.4.3.3.2 | Le graphe G_γ associé à une fonction de structure est le graphe qui admet pour fonction successeur la fonction de structure.
γ est en particulier une fonction de structure pour G_γ.

4. 3. 3. 3. Comparaison des groupes d'automorphismes.

4.3.3.3.1 Proposition.

Ce graphe G_γ a même nombre de sommets que G. On peut donc comparer son groupe d'automorphismes avec celui du graphe G. On a

prop. 4.3.3.3.1

Le groupe d'automorphismes H (G) d'un graphe G est isomorphe a un sous-groupe du groupe d'automorphismes H (G_γ) d'un graphe G_γ associé au graphe G et à une fonction de structure γ.

Demonst. :

On sait que le groupe d'automorphismes H (G) d'un graphe G est isomorphe à un sous-groupe H du groupe S_n des permutations de n sommets du graphe. Soit H_γ le sous-groupe de S_n isomorphe à H (G_γ). Si H est un sous-groupe de H_γ le résultat sera prouvé. Pour cela exprimons le groupe H à l'aide de la fonction Γ. La proposition 2.4.1 qui donnait la définition des permutations ν de H permet d'écrire :

$$\nu \in H \Longleftrightarrow \{\forall x,y : y \in \Gamma x \longrightarrow \nu y \in \Gamma \nu x\}$$

où la formule $y \in \Gamma x$ signifie qu'il existe un arc α de x à y.

De même :

$$\nu_\gamma \in H_\gamma \Longleftrightarrow \{\forall x,y : y \in \gamma x \longrightarrow \nu_\gamma\ y \in \nu_\gamma x\}.$$

Vérifions enfin que toute permutation ν de H est une permutation de H_γ. Soient x et y deux sommets tels que $y \in \gamma x$. Comme ν est une permutation, ν y appartient à $\nu(\gamma x)$; et par la définition de γ on a ν y appartient à $\gamma(\nu x)$. ν se comporte donc comme une permutation ν_γ.

4. 3. 3. 3. 2. Conséquences.

a) Une conséquence de cette proposition est que

Prop. 4.3.3.3.2

La H-partition est plus fine que la H_γ - partition.

D'où une fonction intrinsèque pour G ne sera pas nécessairement une fonction intrinsèque pour G_γ alors que l'inverse est vrai.

b) Il y a une nuance à apporter à la première proposition.

On peut admettre que le sous-groupe soit le groupe tout entier. Cette éventualité n'a jamais été exclue jusqu'ici, même quand il

s'agit de H, sous-groupe de Sn.

Parmi les graphes G_γ on distingue ceux pour lesquels H est un sous-groupe propre (différent) de H_γ et ceux pour lesquels H se confond avec H_γ. On verra plus tard (4.4.2.) que l'éfficacité d'une fonction γ dépend de l'inclusion $H \subset H_\gamma$. On aurait intérêt à savoir d'avance si cette inclusion est stricte ou large. Mais H_γ dépend de G et de γ ; et on ne peut pas tester facilement cette inclusion sans connaitre les groupes H et H_γ.

Il peut même se produire qu'une fonction de structure donne avec un graphe une inclusion stricte, et avec un autre graphe une inclusion large. Voir Fig. 15.

4. 3. 3. 4. Graphe associé et isomorphe.

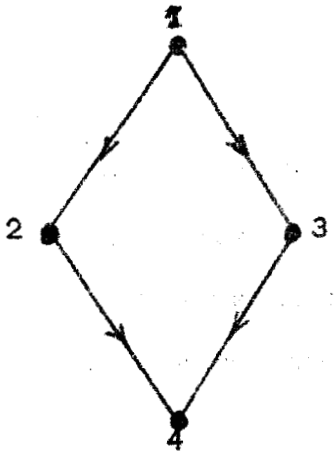
L'emploi des fonctions de structure dans la recherche des isomorphismes entre graphes est conditionné par la proposition suivante :

Prop. 4.3.3.4

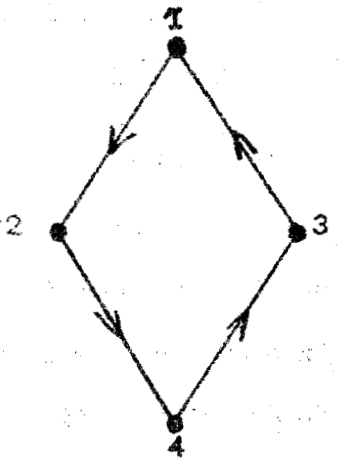
> Les graphes associés par la même fonction de structure, à deux graphes isomorphes sont isomorphes.

Cette proposition est une conséquence de l'existence d'une correspondance biunivoque entre les sommets de deux graphes isomorphes.

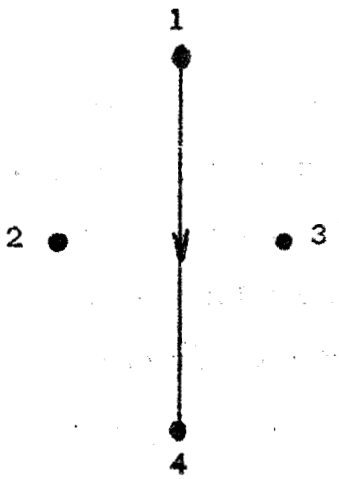
Fig. 15 Comparaison des groupes H et H_γ pour deux graphes et la fonction de structure $\gamma = \Gamma^2$



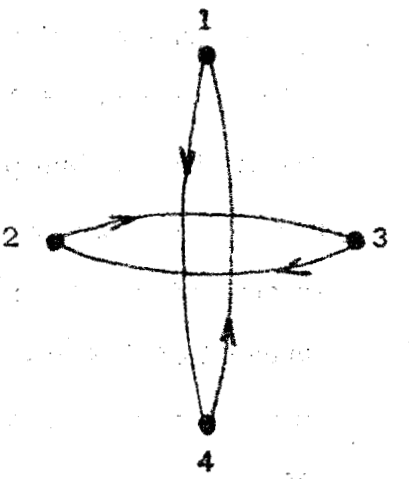
Graphe G_1



Graphe G_2



Graphe $G_{1_{\Gamma^2}}$



Graphe $G_{2_{\Gamma^2}}$ (2 composantes connexes isomorphes)

Groupe H_1 : (I, r_1)
avec $r_1 = [2,3]$
et $r_1^2 = I$

Groupe $H_{1_{\Gamma^2}}$: $(I, r_1) = H_1$

$\boxed{H_1 = H_{1_{\Gamma^2}}}$

Groupe H_2 : (I, r'_1)
avec $r'_1 = [1,2,4,3]$
et $r'^4_1 = I$

Groupe $H_{2_{\Gamma^2}}$: (I, r'_1, r'_2)
avec $r'_1 = 1,2,4,3 \quad r'_2 = [1,4]$
et $r'^2_2 = I$, $(r'_1 r'_2 r'_1)^2 = I$

$\boxed{H_2 \subset H_{2_{\Gamma^2}}}$

4. 4. Fonctions de structure et fonctions intrinsèques.

4. 4. 1. Construction de fonctions de structure.

4. 4. 1. 1. Construction

A l'aide d'une fonction de structure γ et d'une fonction intrinsèque f on peut construire une nouvelle fonction de structure γ' par la méthode suivante : γ'_a associe à un sommet x les sommets y de γx pour lesquels f (y) a une valeur donnée a. Ce sont les sommets de γx qui appartiennent à une FI_f- classe donnée.

4. 4. 1. 2. Propriété.

Prop. 4.4.1.2

γ'_a est une fonction de structure pour G.

Soit z appartenant à $\gamma' x$. Alors $z \in \gamma x$ et $f(z) = a$ où a est la valeur donnée. Une permutation ν de H transforme x en νx et z en νz. z et νz ont même valeur par la fonction intrinsèque. D'autre part νz appartient à $\nu(\gamma x)$, c'est à dire à $\gamma \nu x$. Il en résulte que νz appartient à $\gamma' \nu x$. Comme ceci est vrai pour tout z de $\gamma' x$ et pour toute permutation ν de H, on a le résultat attendu, et $H\gamma'_a$ contient H.

4. 4. 1. 3. Comparaison de H_γ et $H_{\gamma'a}$.

Il n'est pas possible de donner un résultat à priori sur la comparaison des groupes H_γ et $H_{\gamma'}$, ni sur l'éventualité de γ' fonction de structure pour G_γ. On peut construire des fonctions γ' où $H_{\gamma'}$ n'a aucune relation d'inclusion avec H_γ. Voir fig. I6.

La situation réciproque de $H_{\gamma'}$ et H_γ dépend de la finesse de la F.I - partition associée à la fonction intrinsèque par rapport à

la H_γ - partition. Il se produira que si la F.I - partition est plus fine que la H_γ - partition alors la $H_{\gamma'}$ - partition sera plus fine que la F.I - Partition.

Les H_γ - partitions plus fines que les F.I - partitions ont un rôle important dans l'amélioration des F.I - partitions, comme on le verra dans la méthode des deux fonctions.

On verra un exemple de construction de fonction de structure γ' fig. 16. On remarquera, dans le deuxième cas, que γ' n'est pas une fonction de structure pour G_γ. Ceci résulte de ce qu'une fonction intrinsèque pour G ne l'est pas en général pour G_γ. (Justification en 4.5.3)

4. 4. 2. Construction des fonctions intrinsèques.

4. 4. 2. 1. Fonction intrinsèque sur G_γ.

Soit une fonction de structure γ. La fonction qui à x associe le cardinal de γx est une fonction intrinsèque. C'est la fonction demi-degré extérieur sur le graphe G_γ. Elle est une fonction intrinsèque pour ce graphe. Comme $H_\gamma \supset H$, la H - partition est plus fine que la H_γ - partition qui est plus fine que toute F.I - partition sur G_γ.

On obtient donc que toute fonction intrinsèque sur G_γ est une fonction intrinsèque sur G.

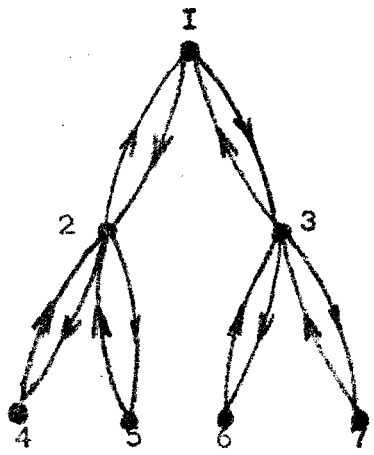
Il en sera de même pour les graphes $G_{\gamma'}$, où en particulier la fonction demi-degré extérieur est le cardinal de l'ensemble des sommets y de γx qui vérifient $f(y) = a$, où a est une valeur donnée.

Fig. I6 Exemple de construction de γ' et comparaison des H_γ et $H_{\gamma'}$

---

a) $G = (X,U,e)$ est défini par la fonction Γ suivante

Γ = I(2,3) 2(I,4,5) 3(I,6,7) 4(2) 5(2) 6(2) 7(2)



b) Groupe H : générateurs $r_1 = [4,5]$

$r_2 = [2,3]\ [4,6]\ [5,7]$

H = $I, r_1, r_2, r_1r_2, r_2r_1, r_1r_2r_1, r_2r_1r_2, r_1r_2r_1r_2$.

H - partition = (I) (2,3) (4,5,6,7)

(en faisant la réunion des cycles des permutations qui ont une intersection non vide).

c) FI - partition = (I,2,3) (4,5,6,7) (f(I)= 1, f(4) = 2)

Premier cas.

$\gamma = \Gamma^2$ = 1(1,4,5,6,7) 2(2,3) 3(2,3) 4(1,4,5) 5(1,4,5) 6(1,6,7) 7(1,6,7)

Générateurs : $r_1 = [4,5]$ H_γ - partition = (1) (2,3) (4,5,6,7)

$r_2 = [4,6]\ [5,7]$

$r_3 = [2,3]$

$H_\gamma = I, r_1, r_2, r_1r_2, r_2r_1, r_3, r_3r_1, r_3r_2, r_3r_2r_1, r_1r_2r_3$.

γ'_1 = 1(1) 2(2,3) 3(2,3) 4(1) 5(1) 6(1) 7(1).

$H_{\gamma'_1} = S_4$ (sur 4,5,6,7) x S_2 (sur 2,3); $H_{\gamma'_1}$ - partition = (1) (2,3) (4,5,6,7).

γ'_2 = 1(4,5,6,7) 2( ) 3( ) 4(4,5) 5(4,5) 6(6,7) 7(6,7).

$\boxed{H_{\gamma'_2} = H_\gamma}$ $\boxed{H_{\gamma'_1} \supset H_\gamma}$

Deuxième cas.

$\gamma = \Gamma^3$ = 1(2,3) 2(1,4,5,6,7) 3(1,4,5,6,7) 4(2,3) 5(2,3) 6(2,3) 7(2,3).

$H_\gamma = S_5$ (sur 1,4,5,6,7) x S_2 (sur 2,3); H_γ - partition = (1,4,5,6,7) (2,3)

γ'_1 = 1(2,3) 2(1) 3(1) 4(2,3) 5(2,3) 6(2,3) 7(2,3).

$H_{\gamma'_1} = S_4$ (sur 4,5,6,7) x S_2 (sur 2,3); $H_{\gamma'_1}$ - partition = (1) (2,3)(4,5,6,7)

γ'_2 = 1( ) 2(4,5,6,7) 3(4,5,6,7) 4( ) 5( ) 6( ) 7( )

$\boxed{H_{\gamma'_2} = H_{\gamma'_1} \subset H_\gamma}$

4. 4. 2. 2. Méthode des deux fonctions.

Cette méthode avec celle des hypothèses forme la partie essentielle de l'algorithme qu'on déduira de cette étude.

Elle dérive de la méthode de construction de fonctions intrinsèques ci-dessus indiquée.

4. 4. 2. 2. 1 Description.

Soit une fonction de structure γ et une fonction intrinsèque f qui prend ses valeurs dans un ensemble fini F. Soit l'ensemble des fonctions de structure γ'_a où a décrit F. Appelons f_a la fonction intrinsèque de G donnée par le demi-degré extérieur dans $G_{\gamma'_a}$.

La méthode des deux fonctions γ et f consiste à construire par le processus de juxtaposition (vu en 4.2.5) de f et des f_a une nouvelle fonction intrinsèque f'.

La F.I - partition associée à f' est plus fine que celle associée à f. f' est une fonction intrinsèque par ce que le processus de juxtaposition construit des fonctions intrinsèques à partir de fonctions intrinsèques. On trouvera une autre démonstration de cette affirmation fig. 17 et 18.

Fig. 17 La méthode des deux fonctions construi des fonctions intrinsèques

Démonstration.

f est une fonction intrinsèque : $\forall \nu \in H,\ \forall x \in X \quad f(\nu x) = f(x)$

est une fonction de structure : $\forall x \in X,\ \forall \nu \in H \quad \gamma(\nu x) = \nu(\gamma x)$

H - équivalence $\quad x \underset{H}{\sim} y \iff \exists\, \nu \in H \quad y = \nu x$

FI- équivalence $\quad x \underset{FI}{\sim} y \iff f(x) = f(y)$

Propriété $\quad x \underset{H}{\sim} y \implies x \underset{FI}{\sim} y$

La méthode des deux fonctions construit une fonction qui affecte la même valeur à deux sommets s'ils sont FI - équivalent s et si ils ont autant de sommets associés par γ dans chaque FI - classe.

L'équivalence associée est la FI' - équivalence.

$x \underset{FI'}{\sim} y \iff x \underset{FI}{\sim} y$ et $\forall A$ une FI - classe $|\gamma_x \cap A| = |\gamma_y \cap A|$

Montrons que H - équivalence $\implies$ FI' - équivalence.

Démonst.

$x \underset{H}{\sim} y \implies \exists\, \nu \in H \mid y = \nu x$ et $\gamma y = \nu(\gamma x)$.

soit $t \in \gamma x \cap A$ alors

$t \in \gamma x \implies z = \nu t \in \nu \gamma x = \gamma y$

$t \in A \implies z = \nu t \implies z \in A$ $\Big\}$ $\implies z \in \gamma y \cap A$

soit $t' \in \gamma x$ $t' \neq t$ alors

$z' = \nu t'$ est différent de z

d'où $\forall\, t \in \gamma x \cap A,\ \exists!\ z_t \in \gamma y \cap A$

ce qui établit une correspondance b iunivoque entre x A et y A

d'où $|\gamma x \cap A| = |\gamma y \cap A|$, et ceci $\forall A$.

Comme d'autre part H - équivalence $\implies$ FI - équivalence on a bien le résultat attendu.

Fig. 18 Forme algébrique de la méthode des deux fonctions.

- Soit un ensemble de n objets : X.
- Soit une partition p de ces objets et la p - équivalence correspondante. f est la fonction associée à la partition p :

  $x \underset{p}{\sim} y \Longrightarrow f(x) = f(y)$
- Soient les sous-groupes k de Sn dont les k - partitions associées sont plus fines que la partition p. Soit h un des sous-groupes k tel qu'il n'existe pas de k - partition strictement moins fine.
- Soit $1 : X \longrightarrow X^*$ une fonction vérifiant $\forall x \in X, \forall \nu \in h$

  $1(\nu x) = \nu(1x)$

  C'est une fonction du type "de structure". Remarquons qu'une fonction vérifiant cette propriété sur un groupe contenant h, la vérifie sur h.
- f est une fonction vérifiant $\forall x \in X, \forall \nu \in h \quad f(\nu x) = f(x)$.

  C'est une fonction du type "intrinsèque". Elle vérifie cette propriété pour tout sous-groupe de h, et même pour tout sous-groupe k.
- La méthode des deux fonctions consiste à construire la partition q telle que

  $x \underset{q}{\sim} y \Longleftrightarrow x \underset{p}{\sim} y$ et $\forall A$ une classe de p $|1x \cap A| = |1y \cap A|$.

Théorème La q - partition est moins fine que la h - partition.

Démonst.

$x \underset{h}{\sim} y \Longrightarrow \exists \nu \in h$ tel que $y = \nu x$ et $1y = \nu(1x)$

$\forall A$, soit $t \in 1x \cap A$, alors

$$\left.\begin{array}{l} t \in 1x \Longrightarrow z = \nu t \in \nu 1x = 1y \\ t \in A \Longrightarrow z = \nu t \Longrightarrow z \in A \end{array}\right| \Longrightarrow z \in 1y \cap A$$

soit $t' \in 1x \cap A$, $t' \neq t$, alors $z' = \nu t'$ est différent de z

Fig. 18 (suite)

ce qui établit une correspondance biunivoque entre les éléments de $\mathrm{l}x \cap A$ et $\mathrm{l}y \cap A$ d'où $|\mathrm{l}x \cap A| = |\mathrm{l}y \cap A|$.

Comme $x \underset{h}{\sim} y \Longrightarrow x \underset{p}{\sim} y$ on a bien

$x \underset{h}{\sim} y \Longleftrightarrow x \underset{p}{\sim} y$ et $\forall A\ p, |\mathrm{l}x \cap A| = |\mathrm{l}y \cap A|$.

4. 4. 2. 2. 2. Emploi.

a) Itération

La méthode des deux fonctions peut être itérée avec la même fonction de structure γ et en utilisant la nouvelle fonction intrinsèque. Implicitement ceci revient à utiliser les puissances de la fonction γ qui se calculent comme les puissances de la fonction successeur Γ.

Si Γ est la fonction de structure, comme on sait que Γ^1 indique les sommets qui peuvent être atteints à partir de chaque sommet par des chemins de longueur 1, on constate qu'en itérant la méthode des deux fonctions, on construi implicitement tous les chemins issus de tous les sommets du graphe. L'ensemble des chemins issus d'un sommet peut être schématisé, avec quelques précautions, par une arborescence. La méthode des deux fonctions itérée revient alors à donner par les nouvelles fonctions intrinsèques la même valeur aux sommets racines d'arborescences isomorphes.

Comme pour le processus de juxtaposition, il faut prendre la précaution d'affecter ces valeurs silmutanément dans les deux graphes dont on cherche l'isomorphisme de façon à respecter la correspondance biunivoque entre les FI - classes des deux graphes.

b) Stabilisation

Les fonctions intrinsèques obtenues sont associées à des FI - partitions de plus en plus fines, mais on a vu en 4.2.4 que les FI - partitions étaient toutes moins fines que la H - partition. Comme l'ensemble X des sommets est fini on aboutit au bout d'un certain nombre d'itérations à l'impossibilité d'affiner la FI - partition. On dira qu'elle est stabilisée. Ceci ne sous-entend pas que la FI - partition soit devenue égale à la H - partition. Il existe des cas où l'égalité se produit.

En tenant compte des remarques précédentes, on comprendra que le nombre d'itérations nécessaire pour arriver à la stabilité d'une FI - partition est au plus égal à la longueur du plus long chemin élémentaire existant dans le graphe, et donc, est inférieur où égal à n. C'est aussi le nombre de partitions distinctes de plus en plus fines qu'on peut construire.

c) Amorce des itérations.

La première itération se fait avec une fonction intrinsèque quelconque. Remarquons qu'on peut partir d'une fonction intrinsèque associant à chaque sommet la même valeur. Si la fonction de structure utilisée est la fonction successeur Γ on obtient à la première itération la fonction demi-degré extérieur du graphe. Si on prend la puissance Γ^{-1} on obtient la fonction demi-degré extérieur. On n'obtiendrait pas la fonction intrinsèque du nombre de boucles par sommet à l'aide de ce procédé.

d) Changement de γ.

Il est intéressant d'utiliser quand la stabilisation des fonctions intrinsèques est obtenue, une autre fonction de structure et de rechercher avec elle une nouvelle stabilisation sur une F.I.partition plus fine que la précédente.

On peut même itérer directement la méthode des deux fonctions en changeant la fonction γ à chaque itération. Ce procédé ne présente pas grand intérêt, car il nécessite quand la stabilisation est obtenue de vérifier qu'elle est pour toutes les fonct ions de structure utilisées. Cependant l'utilisation simultanée des fonctions de structure Γ et Γ^{-1} permet d'accélérer la convergence des FI - partitions vers la stabilisation, en utilisant implicitement non seulement les chemins, mais aussi les chaînes, issues de chaque sommet.

4. 4. 2. 3. Généralisation de la matrice D.

4. 4. 2. 3. 1. Tableau T.

On construit le tableau T par juxtaposition des fonctions intrinsèques f_a dans la méthode des deux fonctions γ et f.

Il est indicé en ligne par les sommets et en colonne par les valeurs de la fonction intrinsèque f, et est tel que l'élément Txa est le nombre de sommets y de γx qui vérifient $f(y) = a$.

La construction de ce tableau est simple si on a une représentation matricielle de la fonction de structure γ. On l'obtient en sommant les colonnes de cette matrice dont les indices sont les sommets auxquels la fonction intrinsèque donne la valeur a.

Fig . I9 Exemple de graphes ayant des tableaux T différents et des matrices D égales.

Graphes

Ces deux graphes ne sont pas isomorphes puisque il existe dans G_2 une boucle (circuit de longueur 1) qui n'existe pas dans G_1.

Matrice caractéristique Fonction $\gamma = \Gamma$.

A	1	2	3	4	DDE
1		I	I		2
2	I			I	2
3					0
4					0
DDI	1	1	1	1	

A	1	2	3	4	DDE
1	I	I			2
2			I	I	2
3					0
4					0
DDI	1	1	1	1	

Fonction intrinsèque f = juxtaposition de DDE et DDI

sommets		1	2	3	4
pour G_1	f =	1	1	2	2
pour G_2	f =	1	1	2	2

tableaux T

G_1	1	2
1	1	1
2	1	1
3	0	0
4	0	0

G_2	1	2
1	2	0
2	0	2
3	0	0
4	0	0

matrices D

G_1	1	2
1	2	2
2	0	0

=

G_2	1	2
1	2	2
2	0	0

4. 4. 2. 3. 2. Matrice D généralisée.

La matrice D va s'obtenir à partir du tableau T en sommant les lignes dont les indices sont des sommets qui ont même valeur par la fonction intrinsèque.

La matrice obtenue est indicée en ligne et en colonne par les valeurs de la fonction caractéristique et on a D_{ij} égal au nombre de sommets, de valeur j par la fonction intrinsèque, associés par la fonction de structure aux sommets de valeur i.

C'est la matrice D pour la fonction intrinsèque f, sur le graphe G_γ.

Remarquons qu'il peut se produire des cas où les tableaux T de deux graphes sont différents, quelque soit la permutation de lignes qu'on applique à l'un deux , alors que les matrices D sont égales.
Or si les tableaux T de deux graphes sont différents, les fonctions intrinsèques f' qu'on construit à partir de ces tableaux par la méthode des deux fonctions seront différentes aussi et les caractéristiques qu'on leur associera indiqueront la non-isomorphie des graphes.

En conclusion, la méthode des deux fonctions permet, tout en le recherchant, de vérifier à chaque étape l'inexistence éventuelle d'isomorphisme entre deux graphes G, celui-ci résultant de l'isomorphisme entre les graphes G_γ associés pour toute fonction de structure γ.

4. 5. Méthode des hypothèses.

C'est une méthode progressant suivant une structure d'arborescence et permettant de faire éclater la partition sur laquelle se sont stabilisées les partitions associées aux fonctions intrinsèques quand ces partitions ne permettent pas d'établir l'isomorphisme (un sommet par classe). La correspondance biunivoque entre les sommets est alors celle qui existe entre les classes.

4. 5. 1. Description.

Pour deux graphes entre lesquels on recherche un isomorphisme ont été trouvées et stabilisées des FI - partitions telles qu'on connait une correspondance biunivoque entre les classes des deux graphes. On a vu que chaque sommet de l'une des classes dans l'un des graphes ne peut être appliqué que sur un sommet de la classe correspondante dans l'autre graphe, et peut-être, même pas sur tous.

Comme on ne peut plus affiner les partitions obtenues par les méthodes employées jusqu'ici, on va faire "éclater" arbitrairement les classes de ces partitions. Pour cela on va admettre l'hypothèse qu'un sommet donné d'une classe donnée du premier graphe doit s'appliquer sur un sommet donné de la classe correspond ante dans l'autre graphe. On fait là une hypothèse de niveau 1.

On crée ainsi une partition plus fine que la précédente dans chacun des graphes. On l'appellera la FH - partition. On va affiner cette FH - partition à l'aide de la méthode des deux fonct ions jusqu'à obtenir l'un des trois cas suivants :

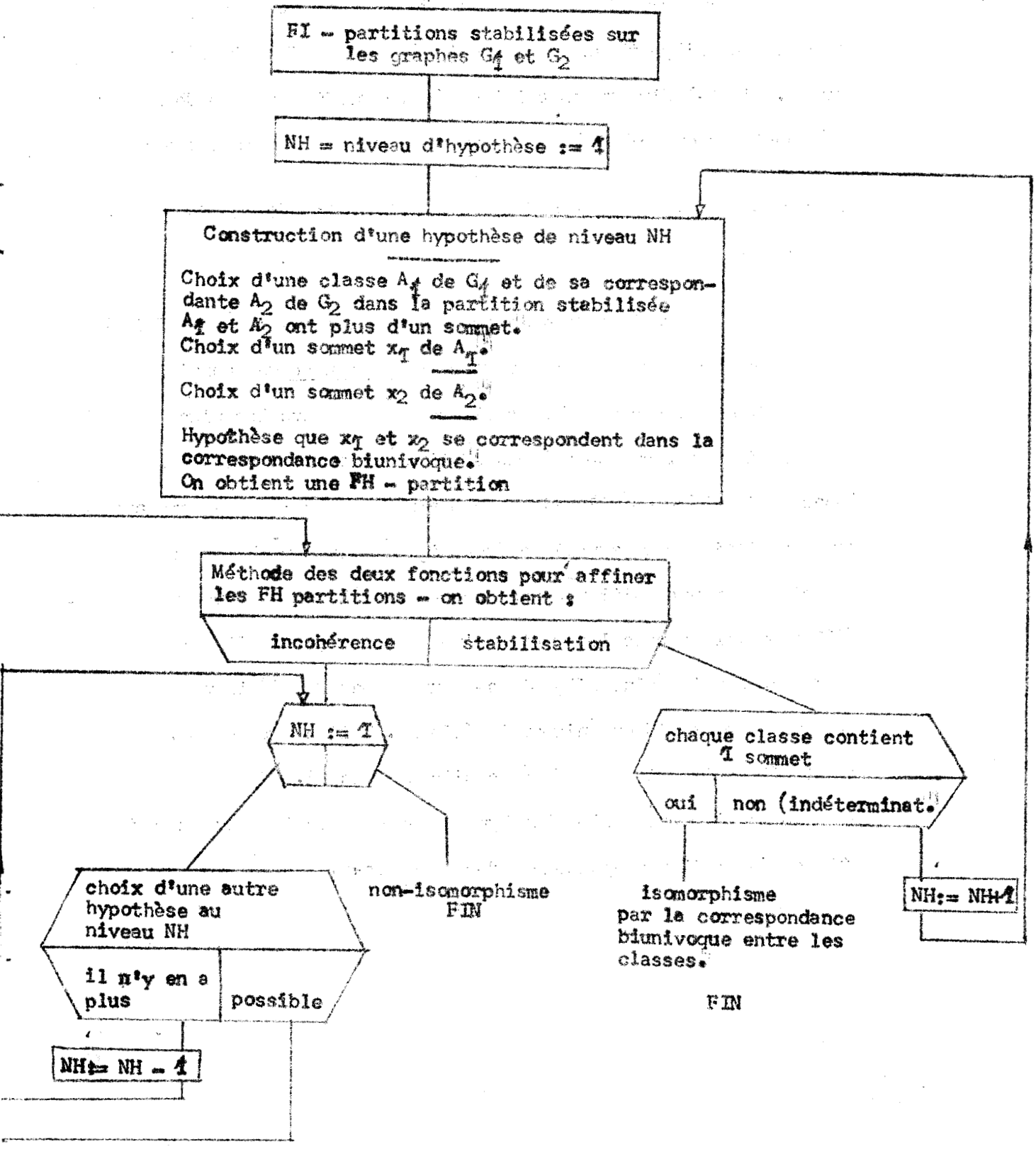
- L'isomorphie . Chaque classe ne contient qu'un sommet. L'hypothèse était juste.
- Une incohérence : des FH classes correspondantes ont des cardinaux différents dans les deux graphes. L'hypothèse était fausse. Il faut alors reprendre les partitions sur lesquelles l'hypothèse a été faite, et en faire une autre : associer le même sommet de la classe du premier graphe sur un autre sommet de la classe correspondante.
- Une indétermination : la FH - partition a été affinée et stabilisée sur une partition dont une classe contient au moins deux sommets. On se retrouve dans la même situation. On fera encore une hypothèse, d'un niveau plus élevé.

4. 5. 2. Incohérence et niveau d'hypothèse.

Remarquons la nécessité de pouvoir retrouver pour les deux graphes les partitions sur lesquelles la dernière hypothèse a été faite et la liste des associations qui ont été essayées afin de permettre une reprise dans le cas de l'obtention d'une incohérence. Il faut conserver les hypothèses.

Il se peut que,dans le cas de l'incohérence il ne soit plus possible de trouver un autre sommet de la classe du second graphe : ils ont tous été essayés et chacun a conduit à une incohérence.
S'il s'agit du premier niveau d'hypothèses, celles faites sur les FI - partitions stabilisées, on pourra conclure que les deux graphes ne sont pas isomorphes, puisqu'un sommet du premier graphe ne peut être associé à aucun des sommets de la FI-classe correspondante.

Fig. 20 Emploi de la méthode des hypothèses



Par contre s'il s'agit d'hypothèses d'un niveau plus élevé, on devra conclure que l'hypothèse de niveau immédiatement inférieur était fausse. C'est-à-dire qu'on a l'incohérence sur l'hypothèse de niveau précédent, et que c'est à ce niveau qu'il faut en choisir une autre.

On trouvera fig. 20 un organigramme décrivant la méthode des hypothèses.

La méthode des hypothèses progresse parmi les partitions stabilisées de la même façon qu'un voyageur dans un labyrinthe qui aurait la forme d'une arborescence dont seulement les sommets les plus éloignés de la racine correspondent à des sorties. Qunad il aboutit à une impasse (incohérence) il fait marche arrière jusqu'au dernier carrefour (partition sur laquelle a été faite la dernière hypothèse) et choisit une route qu'il n'a pas encore explorée (une autre hypothèse). S'il a exploré toutes les routes d'un carrefour, et qu'il n'a pas trouvé la sortie (plus d'hypothèse à un niveau donné) il retourne au carrefour précédent (niveau inférieur). S'il ne trouve pas de sortie il revient à l'entrée (pas d'isomorphisme).
Dans ce dernier cas toutes les hypothèses de tous les niveaux ont été testées : il y a épuisement des hypothèses.

4. 5. 3. Etude de l'indétermination.

4. 5. 3. 1. FI et FH - partitions.

Quand une hypothèse est faite, la FH - partition obtenue n'est pas en général une FI - partition, car elle peut être plus fine que la H - partition.

L'exemple de fig. 18 le montre.

Ce résultat se justifie par la remarque suivante. Quand on fait l'hypothèse qu'un sommet x du premier graphe doit être associé à un sommet du second graphe, on élimine a priori les autres sommets des deux classes correspondantes sur lesquelles on a f ait l'hypothèse. Ceci revient à supposer que ce sommet x ne peut être permuté avec les autres, et donc qu'il est invariant. On ne considère donc plus le groupe H, mais un de ces sous-groupes H'_x qui laisse x invariant, c'est-à-dire tel que $I_{H'_x} \supset I_H \cup \{x\}$.

La fonction associée à la FH - partition (elle donne des valeurs différentes aux classes et affecte à chaque sommet les valeurs de sa classe) est une fonction intrinsèque vis à vis de la H'_x - partition, qui est plus fine que la H - partition. Ce n'est donc pas une fonction intrinsèque pour le graphe en général.

4. 5. 3. 2. Stabilisation des FH - partitions.

Les éléments du paragraphe précédent permettent de connaître une partition plus fine que toutes les FH - partitions obtenues après une hypothèse par la méthode des deux fonctions. C'est une H'_x - partition, et on pourra conclure que les FH - partitions se stabiliseront sur une FH - partition moins fine qu'une H'_x - partition.

La fig. 21 montre un cas où l'on obtient exactement la H'_x - partition.

La démonstration de ce résultat serait la même que celle proposée fig. 17 pour montrer que la méthode des deux fonctions construit une fonction intrinsèque, les FH - partition prenant la place des FI - partitions et H'_x la place de H.

Le résultat obtenu dans cette démonstration est vrai en général pour les cas exposés fig . 18. On peut déduire des conditions du théorème que H'_x est tel qu'il n'existe pas de sous-groupes de H qui laissent $I_H \cup \{x\}$ invariant et qui contiennent strictement le sous-groupe H'_x. On peut aussi le déduire de la façon dont on construit H'_x.

4. 5. 4. Nombre d'hypothèses.

4. 5. 4. 1. Nombre minimum d'hypothèses.

C'est un nombre intéressant car il correspond au cas où l'on obtient le résultat le plus rapidement possible.

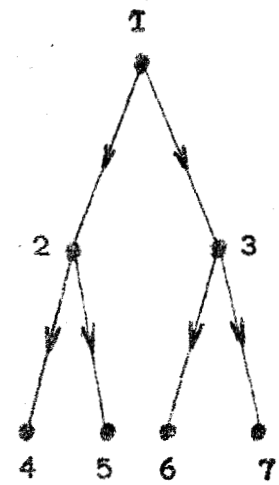
Si les deux graphes sont isomorphes on obtiendra le résultat plus rapidement si à chaque niveau on fait une hypothèse juste, entrainant la stabilisation rapide des FH - partitions, et si on minimise le niveau de la dernière hypothèse. On ne peut pas connaître a priori le "pouvoir séparateur" d'une hypothèse.

Si les deux graphes ne sont pas isomorphes, avant d'obtenir le résultat il faut essayer les p hypothèses qu'il est possible de faire sur des FI - classes correspondantes de p sommets. On choisira alors à chaque niveau les FI - classes (ou FH - classes) qui ont le moins

Fig. 2I Application de la méthode des deux fonctions et de la méthode des hypothèses pour obtenir des partitions sur un seul graphe.

$G = (X,U,e)$

$= 1(2,3)\ 2(4,5)\ 3(6,7)\ 4(\ )\ 5(\ )\ 6(\ )\ 7(\ ).$



Le groupe H est le même que celui du graphe Fig. I6.

H - partition = (I) (2,3) (4,5,6,7)

I° Création de FI - partition par la construction de fonctions intrinsèques

Sommets	1	2	3	4	5	6	7	
fonct. demi-degr. ext.	2	2	2	0	0	0	0	2 fonctions intrinsèques
fonct. demi-degr. int.	0	1	1	1	1	1	1	
juxtaposition	1	2	2	3	3	3	3	numérotation des couples de valeurs des deux lignes précédentes

FI - partition (1) (2,3) (4,5,6,7) c'est la H - partition

La méthode des deux fonctions ne pourra pas donner d'amélioration

Essais avec $\gamma = \Gamma$								
	0	0	0	0	0	0	0	$\gamma_2'x$ = nb. de sommets
tableau T	2	0	0	0	0	0	0	de valeur a pour la dernière
	0	2	2	0	0	0	0	fonction intrinsèque dans Γx
Juxtaposition	1	2	2	3	3	3	3	C'est la même fonction intrinsèque.

Ce travail ayant été fait simultanément sur un autre graphe, et ayant obtenu des partitions correspondantes, on doit faire une hypot-hèse.

Hypothèse : association de 2 de ce graphe avec un sommet de la classe correspondante de l'autre graphe.

FH - partition : (1) (2) (3) (4,5,6,7)

fonction associée f_a	1	2 4	3	3	3	3	ce n'est pas une fonction intrinsèque.

Fig. 2I (suite)

Sommets	1	2	3	4	5	6	7	
Méthode des deux fonctions avec $\gamma = \Gamma$ et la fonction associée								
par γ'_1	0	0	0	0	0	0	0	tableau T
par γ'_2	1	0	0	0	0	0	0	
par γ'_3	0	2	2	0	0	0	0	
par γ'_4	1	0	0	0	0	0	0	
Juxtaposition f_1	1	2	2	3	3	3	3	la FH-partition est moins fine que la précédente
Juxtaposition f_1 et f_a : f_2	1	2	3	4	4	4	4	on a une nouvelle numérotation de f_a.

Essais avec $\gamma = \Gamma^{-1}$

$\gamma = 1(\ )\ 2(1)\ 3(1)\ 4(2)\ 5(2)\ 6(3)\ 7(3)$

par γ'_1	0	1	1	0	0	0	0	tableau T
par γ'_2	0	0	0	1	1	0	0	
par γ'_3	0	0	0	0	0	1	1	
par γ'_4	0	0	0	0	0	0	0	
Juxtaposition f_3	1	2	2	3	3	4	4	Cette FH-partition n'est pas comparable à la précédente.
Juxtaposition f_2, f_3 : f_4	1	2	3	4	4	5	5	Cette FH-partition est plus fine que celle associée à la fonction f_2

D'autres essais montreront qu'on a une stabilisation sur la FH-partition

(1) (2) (3) (4,5) (6,7)

Elle correspond à la H' - partition d'un sous-groupe H' du groupe H.

Pour la faire éclater il faudra encore faire une hypothèse qui sera de niveau 2.

La solution apparaitra après une hypothèse de niveau 3, car alors on aura un sommet par classe.

Remarquons que $I_{H'}$ contient deux éléments (2 et 3) en plus de I_H.

de sommets, et parmi celles là, celles qui entraineront moins de niveaux supérieurs d'hypothèses; ou bien simplement les classes qui entraîneront un nombre total minimum d'hypothèses. On ne peut pas encore les connaître a priori.

Devant tant d'impondérable , la pratique et le bon sens semblent indiquer qu'il vaut mieux choisir à chaque niveau d'hypothèse des FH - classes (non réduites à un sommet) de cardinal minimum.
Il existe des graphes où cette pratique ne minimise pas le niveau maximum d'hypothèses. Voir fig. 22.

4. 5. 4. 2. Niveau maximum d'hypothèses.

Son intérêt est de donner le nombre maximum de partitions stabilisées et d'hypothèses faites dessus, qu'il faut conserver en vue d'une incohérence éventuelle à un niveau quelconque qui entrainerait successivement l'incohérence à tous les niveaux inférieurs.

Ce nombre dépend du plus grand nombre de sous-groupes propres emboités qu'on peut trouver dans H. On étudiera cela dans le paragraphe suivant.

On peut trouver une borne à ce nombre indépendamment de H. Chaque hypothèse permet d'obtenir une partition plus fine que la précédente. Le niveau maximum d'hypothèses sera inférieur au nombre de partitions de plus en plus fines qu'on peut trouver dans un ensemble de n objets, ce nombre étant diminué d'une unité. (Quand il ne reste qu'une classe de deux objets, une hypothèse suffit).

Les graphes pleins et sans arcs qui vérifient $H = S_n$ nécessitent effectivement n-1 niveaux d'hypothèses. En effet quand on associe deux sommets pris dans deux graphes sans arcs, on ne peut rien conclure pour les autres sommets, et le problème reste le même pour eux.
Le graphe plein est le complémentaire du graphe sans arcs. En fait de tels graphes ont des matrices caractéristiques particulières qu'on sait reconnaître immédiatement. On verra plus loin un moyen d'éviter les derniers niveaux d'hypothèses, et dans le cas de ces deux graphes, de n'en faire aucune.

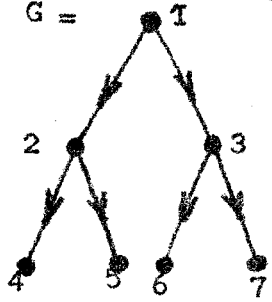
4. 5. 4. 3. Niveau minimum d'hypothèses.

Bien que ce nombre n'ait pas une utilité pratique pour nous, si ce n'est que pour diminuer les recherches de l'isomorphisme dans le cas où on pourrait mesurer le "pouvoir séparateur" d'une hypothèse, nous donnerons des indications le concernant. On obtiendra en même temps des indications sur le niveau maximum d'hypothèses.

Sous-groupe spécial H'.

Soit un groupe H de permutations de S_n et soit I_H l'ensemble des objets invariants associés.
Soit x un objet n'appartenant pas à I_H.
H' est le sous-groupe propre de H tel que $I_{H'} \supset I_H \cup \{x\}$ et qu'il n'en existe d'autre vérifiant cette propriété et le contenant strictement.
On dira qu'un sous-groupe vérifiant $I_{H'} \supset I_H \cup \{x\}$ est un sous-groupe spécial et que H' est un sous-groupe spécial maximal.

Fig. 22 Sous-groupes spéciaux.

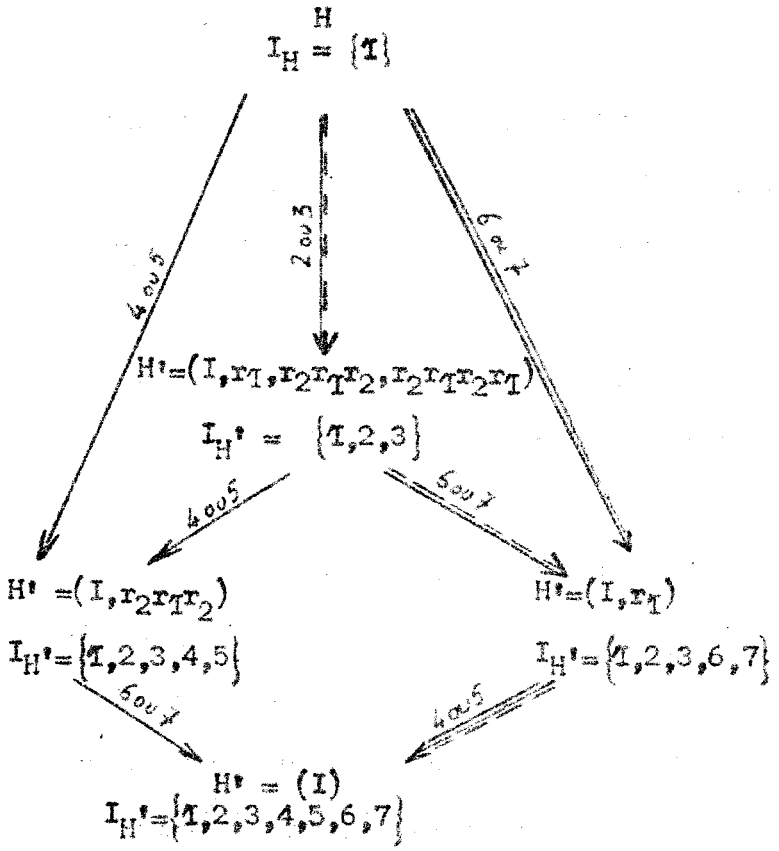


Groupe H →	I	r_1	r_2	r_1r_2	r_2r_1	$r_1r_2r_1$	$r_2r_1r_2$	$r_2r_1r_2r_1 = r_1r_2r_1r_2$
1	1	1	1	1	1	1	1	1
2	2	2	3	3	3	3	2	2
4	4	5	6	6	7	7	4	5
5	5	4	7	7	6	6	5	4
3	3	3	2	2	2	3	3	3
6	6	6	4	5	4	5	7	7
7	7	7	5	4	5	4	6	6

H-partition = (1)(2,3)(4,5,6,7).

2 g-générateurs : r_1, r_2.

3 relations : $r_1^2 = I$, $r_2^2 = I$, $(r_1r_2r_1)^2 = I$.



Les nombres inscrits sur les arcs sont les sommets qu'il faut ajouter au $I_{H'}$ à l'origine de l'arc pour obtenir le sous-groupe H' spécial maximal de l'extrémité.

Tous les sommets non invariants apparaissent sur les arcs sortant de chaque sous-groupe H'.

Pour le graphe G il faut au moins r = 2 niveaux d'hypothèses = 2 générateurs et au plus s = 3 niveaux d'hypothèses = 3 relations.

Le chemin le plus court nécessite une hypothèse sur la H-classe ayant le plus de sommets.

Suite de sous-groupes-spéciaux emboités.

C'est une suite de sous-groupes spéciaux inclus strictement les uns dans les autres:

$$H \not\supset H'_1 \not\supset H'_2 \not\supset \text{-----} \not\supset H'_q = I$$

I est le sous-groupe spécial plus petit que tous les autres.

q est la longueur de la suite. H'_i est maximal pour H'_{i+1} ;

Sur l'ensemble de ces suites on peut définir r = min(q) ; s = max(q).

Hypothèses et sous-groupes spéciaux.

On a vu qu'une hypothèse consiste à ajouter un élément à I_H (soit x cet élément) et de passer alors de H à un sous-groupe spécial maximal H'_x.

Les hypothèses de niveaux successifs nous permettent de construire une suite de sous-groupes spéciaux emboités. Il résultera que pour un graphe donné, le niveau minimum d'hypothèses est r, le niveau maximum est s.

On peut connaître r et s à partir de l'étude des chemins dans un graphe représentant la clôture de la relation d'inclusion dans le treillis des sous-groupes spéciaux de H.

Voir fig. 22 la schématisation pour un cas concret.

Je pense qu'il y a une relation entre r et le nombre minimal de générateurs d'un groupe, et entre s et le nombre minimal de relations qui définissent un groupe. L'étude des générateurs d'un groupe, et en particulier de l'ensemble des relations le définissant est un problème délicat sur lequel on n'a que peu de résultats actuellement.

4. 5. 6. Remarques sur la méthode des hypothèses.

4. 5. 6. 1. La nécessité de cette méthode.

Il est nécessaire en général de faire appel à la méthode des hypothèses pour rechercher un isomorphisme éventuel entre deux graphes. En effet les indications données en 4.3.2.2.1, et en particulier le théorème démontré dans la fig. I8, montrent que la méthode des deux fonctions stabilise les partitions sur une H' - partition moins fine parmi celles qui sont strictement plus fines.

La méthode des hypothèses permet de rendre plus fine la partition obtenue en la faisant "éclater".

Si on étudie les conditions du théorème de la fig. 18 on constate qu'il serait peut-être possible d'atteindre une H' - partition plus fine que la H - partition en utilisant des fonctions du type "de structure" relatives à un sous-groupe H' de H, et sans faire d'hypothèse. Mais qu'elle serait la nature d'une telle fonction vis à vis du graphe? D'autre part, on n'est pas certain d'atteindre la H - partition (ou la H' - partition), et il faudrait trouver une fonction du type "de structure" et une fonction intrinsèque idéale, qui donneraient systématiquement la H' - partition par la méthode des deux fonctions. En attendant on ne peut qu'augmenter l'arsenal de ces fonctions.

4. 5. 6. 2. La suffisance de cette méthode.

La méthode des hypothèses utilisée avec ou sans la méthode des deux fonctions est suffisante pour résoudre le problème dès qu'on possède un test pour vérifier que la correspondance biunivoque obtenue entre les sommets de deux graphes susceptibles d'être isomorphes, est celle d'un isomorphisme. On peut se passer complètement des fonctions intrinsèques.

On considère que les sommets sont tous dans la même classe. Une hypothèse de niveau NH consiste alors à partir d'une partition formée par NH classes, toutes de un sommet sauf une, à passer à la partit ion strictement plus fine de NH + 1 classes, toutes de un sommet sauf une, obtenue en isolant dans une classe un sommet de la classe en contenant plusieurs. Cette opération étant faite simultanément sur les deux graphes étudiés à la fin on obtient une correspondance biunivoque quelconque entre les sommets des deux graphes. C'est alors qu'il faut tester si elle correspond à un isomorphisme.

Par ce moyen on doit faire systématiquement n-1 niveaux d'hypothèses et à chaque niveau on doit choisir parmi les (n-NH+1) possibilités qu'offrent les classes non - réduites à un sommet.
On fera au plus nx(n-1)x ---- x(n-NH+1)x ---- x2 = n ! hypothèses, ce qui correspond à calculer et tester la correspondance biunivoque entre les sommets de l'un des graphes et une des n! permutations des sommets de l'autre graphe.

La méthode des deux fonctions permet de diminuer considérablement le nombre des hypothèses, ce qui revient à éliminer d'avance un grand nombre de permutations pour les tests.

4. 5. 6. 3. Recherche des automorphismes.

Les automorphismes d'un graphe sont des isomorphismes de ce graphe dans lui même. Il est possible de les calculer en utilisant les méthodes décrites ici, en les appliquant à ce graphe et à sa copie.
Le processus est le suivant. A chaque fois qu'on obtient un isomorphisme, c'est à dire ici une permutation du groupe H, on réagit comme s'il s'agissait d'une incohérence, et on modifie la dernière hypothèse faite. Quand obtiendra le non isomorphisme par épuisement des hypothèses alors toutes les permutations trouvées constitueront le groupe H.

Si on se rappelle la comparaison avec le labyrinthe (4.5.2) on peut voir que ce serait le cheminement d'un voyageur qui veut connaître tout le labyrinthe, et qui, quand il trouve une sortie considère qu'il s'agit d'une impasse et fait demi-tour. Vu la structure d'arbre du labyrinthe il se retrouvera à la fin à l'entrée (non - isomorphisme).

4. 5. 6. 4. Conservation des partitions et des hypothèses.

Celle ci peut se faire en ut ilisant une structure de pile (notion de programmation). C'est en effet le système le plus naturel pour explorer un arbre.

4. 6. S - équivalence et hypothèses.

4. 6. 1. "Court-circuiter" des hypothèses.

Nous avons déjà vu que le sous-groupe des permutations de H qui laissent tous les sommets invariants sauf ceux d'une S - classe est isomorphe au groupe symétrique S_p si la S - classe est de cardinal p. Pour des S - classes correspondantes, prises dans deux graphes susceptibles d'être isomorphes, on peut choisir une correspondance biunivoque quelconque entre les sommets. On peut en utilisant les S - classes "court-circuiter" un certain nombre d'hypothèses ce qui diminue le niveau maximum, mais il faut prendre la précaution de stabiliser les FH - partitions sur la S - partition.

4. 6 . 2. FI - partition associée à la S - partition.

On réalise cela en associant une fonction intrinsèque, et donc une FI - partition à la S - partition. Il est facile de calculer la S - partition et on sait que les S - classes continues dans chaque H - classe sont de même cardinal. A la S - partition on associera la FI - partition correspondant à l'équivalence suivante

$x \underset{FI}{\sim} y \iff$ x et y appartiennent à des S - classes de même cardinal. C'est la FI - partition qu'on obtient en regroupant les S - classes de même cardinal. C'est une partition moins fine que la H - partition.

4. 6. 3. Hypothèses sur les S - classes.

Pour orienter la progression de la méthode des hypothèses vers la

S - partition on respectera la condition suivante: à chaque hypothèse choisir une FH - partition moins fine que la S - partition. Cela se réalise facilement en ajoutant à I_H non pas un sommet isolé, mais la S - classe de ce sommet qu'on choisira parmi celles d'une classe de la partition stabilisée sur laquelle on fait l'hypothèse.

Tout ceci revient à travailler sur les S - classes du graphe qui ont été virtuellement remplacées par un sommet, au lieu d'utiliser les sommets directement.

L'intérêt de cette modification apparait dans la recherche d'un isomorphisme entre deux graphes pleins, ou deux graphes sans arcs. Pour ces graphes le groupe d'automorphismes est S_n , et en appliquant la définition de la S - équivalence à tout couple de sommets on constate qu'il n'y a qu'une seule S - classe englobant tout X. Au lieu de faire successivement une hypothèse (juste d'ailleurs) à chacun des n-1 niveaux différents, on choisira une correspondance arbitraire entre les sommets des deux graphes.

Cette modification ne gêne pas la recherche des automorphismes, puisqu'on connait le sous-groupe de H qui ne permute que les sommets de chaque S - classe.

V - COMPARAISON AVEC LES AUTRES METHODES

5. Comparaison avec les autres méthodes.

Nous n'allons pas comparer la méthode proposée avec toutes celles qui existent, mais seulement avec cellesqui sont du même genre.

5. 1. Méthodes semblables.

5. 1. 1. Algorithme de SH. UNGER. (1964)

On trouvera sa référence en bibliographie.

Il utilise la représentation des graphes par la liste Γ et la méthode qu'il emploie pour représenter l'isomorphisme ou la correspondance entre les partitions est la représentation naturellement associée à la liste Γ (parce que du même type) : la liste d'associations qu'il appelle "possible node pairing list" .

Les fonctions intrinsèques s'appellent "nodal function" (fonction de noeud). Elles correspondent aux fonctions données en exemple en 4.2.2. et celles construites en 4.4.2.1.

Les fonctions de structure qui se représentent par des listes γ semblables à la liste Γ s'appellent "correspondence list" ou liste de correspondants. Elles sont du même type que celles en 4.3.2.

La méthode des deux fonctions : "extending" (extension) permet de calculer de nouvelles "nodal functions" qui ne sont d'ailleurs pas en général des fonctions intrinsèques. Elles consistent à affecter

à chaque sommet la somme des valeurs d'une "nodal function" sur les sommets qui lui sont associés par la "correspondence list". Nous verrons en 5.1.3 la nuance qui existe avec la méthode des deux fonctions.

La méthode des hypothèses est signalée pour faire "éclater" les partitions qui se stabilisent, sans toutefois en donner la justification.

5. 1. 2. Algorithme de A.J.W. DUIJVESTIJN (I962).

La référence est donnée en bibliographie.

Le problème de l'isomorphisme s'est posé pour ce chercheur à l'occasion du calcul de graphes dans lesquels on pouvait faire passer un flot qui soit une tension. Il pouvait ainsi résoudre le problème du découpage des rectangles en carrés de côtés différents.

Le problème se pose dans les graphes planaires symétriques sans boucles. La représentation qu'il emploie est particulière. Elle est basée sur un codage des faces du graphe, c'est à dire de l'une des bases de cycles. A partir de ce codage il reconstruit une représentation en tableau E. La représentation de l'isomorphisme est proche du tableau d'associations.

Les fonctions intrinsèques sont des "weights" (poids) qu'il donne aux sommets. Il les choisit parmi les puissances de 2 (pour construire plus facilement les hypothèses). Il part du poids constant 2 sur tous les sommets.

Il utilise comme fonction de structure la fonction Γ du graphe et celle du graphe complémentaire (un sommet par face - un arc entre deux sommets si les faces sont adjacentes).

La méthode des deux fonctions qu'il appelle "weights and score" (poids et sommation) consiste à affecter à chaque sommet le poids obtenu en faisant la somme des poids de ses voisins. C'est en tenant compte du fait que le graphe est symétrique la même méthode que celle de S.H UNGER.

Quand il se produit une stabilisation des "poids" la méthode des hypothèses consiste à augmenter d'une unité les poids de deux sommets pris dans deux classes correspondantes. Aucune justification n'est donnée.

5. 2. Méthode du produit scalaire inachevé.

Cette méthode permet de retrouver la méthode des deux fonctions à partir des méthodes des deux auteurs ci-dessus. Elle est employée par BOHM et SANTOLINI qui ne la citent pas explicitement.

5. 2. 1. Forme matricielle de méthodes de 5.1 et 5.2.

La méthode des deux fonctions pour les deux auteurs précédents admet une représentation unique sous forme mat ricielle.

La fonction de structure se représente par une matrice du type matrice caractéristique d'un graphe. La fonction intrinsèque par un vecteur indicé par les sommets du graphe et tel que la coordonnée de sommet x est la valeur de la fonction intrinsèque pour ce sommet.

La matrice de la fonction de structure est, pour les graphes ordinaires, une matrice de 0 et de 1. Faire la somme des valeurs de la fonction intrinsèque pour les sommets associés par la fonction de structure à un sommet donné, revient à faire le produit scalaire du vecteur fonction intrinsèque par le vecteur ligne de la matrice fonction de structure associé au sommet donné.

La méthode des deux fonctions est alors le produit de la matrice fonction de structure par le vecteur fonction intrinsèque. Il reste à faire la juxtaposition des deux vecteurs fonctions intrinsèques pour obtenir une fonction intrinsèque associée à une partition plus fine.

Cette méthode est moins puissante que la méthode des deux fonctions que nous avons étudiée en 4.3.2.2.

5. 2. 2. Le produit scalaire inachevé.

5. 2. 2. 1. Description.

Quand on fait le produit scalaire de deux vecteurs on perd l'information sur la répartition réciproque des coordonnées, c'est à dire, sur la valeur et même la forme de chacun des produits partiels.

Le produit scalaire inachevé consiste à associer les coordonnées des deux vecteurs sans faire la somme finale. Théoriquement on construit une matrice C indicée en ligne par les valeurs des coordonnées d'un vecteur, en colonne par les valeurs des coordonnées de l'autre vecteur, et on posera Cij = le nombre de fois où le produit ixj apparaît dans le produit scalaire des deux vecteurs.

Exemple :

$$(1,1,2,2) \times \begin{pmatrix} 3 \\ 4 \\ 2 \\ 2 \end{pmatrix} =$$

C	2	3	4
1	0	1	1
2	2	0	0

La valeur habituelle du produit scalaire est $\sum_{i,j} Cij \times i \times j$.

5. 2. 2. 2. Variante.

Dans le cas où le premier vecteur a ses composantes nulles ou égales à 1, on ne conservera de la matrice C que la ligne correspondant à la valeur 1. En effet quand une colonne est non nulle dans une ligne, elle est nulle dans l'autre. On obtient un vecteur C dont la coordonnée Ci = le nombre de fois où une composante de valeur i a même indice que (ou correspond à) une composante non nulle dans le premier.

On peut alors envisager le produit d'une matrice de O et de l'par un vecteur quelconque, puisque à chaque ligne de la matrice on associera un vecteur C. On verra en 5.5 comment faire un produit de matrices à l'aide de ce produit scalaire inachevé.

5. 2. 2. 3. Application à la méthode des deux fonctions.

L'ensemble des vecteurs obtenus en faisant le produit d'une matrice par un vecteur à l'aide du produit scalaire inachevé forme une matrice T, indicé en ligne comme la matrice et en colonne par les valeurs de la fonction intrinsèque, telle que Tij = le nombre de fois où une composante de valeur j du vecteur correspond à un élément non nul dans la ligne i. Si on veut que le produit d'une matrice par un vecteur soit encore un vecteur on affectera aux lignes différentes de la matrice T des valeurs différentes, et on conservera ces valeurs dans un vecteur indicé de la même façon que les lignes de la matrice T.

Si la matrice est celle d'une fonction de structure, si le vecteur est celui d'une fonction intrinsèque, la matrice T est le tableau T de 4.4.2.3.

On a retrouvé la méthode des deux fonctions à laquelle nous étions habitués.

5. 2. 2. 4. Avantage du produit scalaire inachevé.

L'avantage du produit scalaire inachevé est qu'il permet de distinguer non seulement les nombres, mais aussi leurs partitions*.
Par exemple il considèrera comme distinctes

$$4 = 3 + 1 , \qquad 4 = 2 + 2 , \qquad 4 = 2 + 1 + 1.$$

C'est ce qui lui confère une supériorité sur le produit scalaire ordinaire quand il s'agit d'étudier la répartition des valeurs des coordonnées de vecteurs. Voir fig. 23.

S.H UNGER s'était rendu compte de l'insuffisance du produit scalaire ordinaire. Il sentait la nécessite d'avoir des nombres dont la décomposition soit unique. Il avait envisagé une arithmétique basée sur les nombres premiers, qui vérifient justement cette propriété.

Le produit $Y = A,X$ où A est une matrice de 0 et de 1 et X un vecteur dont les composantes sont des nombres premiers, ne serait plus

---

* partition d'un nombre c.

C'est une suite d'entiers a_i vérifiant les relations.

$$1 \leqslant a_1 \leqslant a_2 \leqslant \cdots \leqslant a_r \leqslant c$$

telle que $\qquad a_1 + a_2 + \cdots + a_r = c$

$$Y_i = \sum_k A_{ik} \cdot X_k = \sum_{A_{ik} \neq o} X_k$$

mais

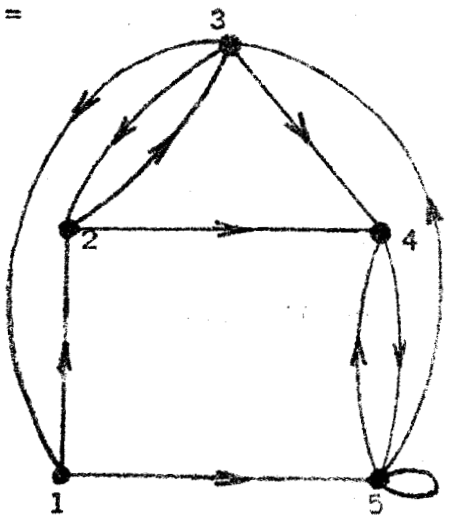
$$Y_i = \prod_k A_{ik} \cdot X_k = \prod_{A_{ik} \neq o} X_k$$

L'inconvénient est que ces nombres deviennent très vite trop grands pour être utilisés facilement surtout en calculateur électronique où les grands nombres sont souvent en "arithmétique dite flottante" (notion de programmation). En effet pour un graphe de n sommets dans les plus mauvais cas on serait amené à calculer le produit des n premiers nombres premiers. Pour n = 11 ce produit est égal à 6. 469. 693. 410 qui est un entier de 10 chiffres. Les représentations "flottante" des nombres n'ont le plus souvent que 10 chiffres exacts en mantisse, ce qui nous limiterait aux graphes de 11 sommets au plus.

Notons enfin que la forme matricielle de la méthode des deux fonctions montre comment au cours des itérations, interviennent les puissances de la fonction γ. En effet : si matriciellement $f' = \gamma . f$, alors $\gamma . f' = \gamma . ^2 f$.

Fig. 23 Comparaison des produits scalaires ordinaire et inachevé.

G =



Matrice caractéristique

A	1	2	3	4	5	d°	f
1		1			1	4	1
2			1	1		4	1
3	1	1		1		5	2
4	1				1	5	2
5			1	1	1	6	3

Fonction de structure = représenté par la matrice A.
Fonction intrinsèque f = degré du sommet. Il y a 3 FI - classes numérotées de 1 à 3.

Méthode des deux fonctions par

Produit scalaire ordinaire.

	1	2	3	4	5	Sommets
	1	1	2	2	3	f
	4	4	4	4	7	A . f
pas d'amélioration c'est la	1	1	2	2	3	f_1

tableau T

Produit scalaire inachevé

1	2	3	4	5	
1	1	2	2	3	
1		2	1		
	2	1		2	
1			1	1	
1	2	3	4	5	chaque sommet est isolé dans sa classe.

même partition. On ne pourra pas affiner cette partition si on ne change pas les valeurs de la fonction intrinsèque. Prenons les

degrés:	4	4	5	5	6	D
	10	10	13	10	16	A.D
	1	1	2	3	4	f'_1

Fig. 23 (Suite)

cas du produit scalaire ordinaire :

Sommets	1	2	3	4	5	
f'_1	1	1	2	3	4	
$A.f'_1$	5	5	5	5	9	
f'_2	1	1	2	3	4	pas d'amélioration

prenons des valeurs quelconques

f'_3	1	1	3	6	IO	
$A.f'_3$	11	9	8	11	19	
f'_4	1	2	3	4	5	chaque sommet est isolé dans sa classe.

Remarque : La juxtaposition de fonctions intrinsèques élémentaires permettait d'atteindre immédiatement la fonction intrinsèque f'_1

Sommets	1	2	3	4	5
demi degré extérieur	2	2	3	2	3
demi degré intérieur	2	2	2	3	2
Boucles	0	0	0	0	1
f'_1	1	1	2	3	4

Echec du produit scalaire ordinaire :

Quand on calcule A.f. on a : ligne 1 et 4 : $4 = 1+3$.
ligne 2 : $4 = 2+2$.
ligne 3 : $4 = 2+1+1$.

Si on augmente les valeurs de la fonction intrinsèque d'une même quantité p on a : ligne 1 et 4 $(1+p)+(3+p)=(1+3)+2p=4+2p$
ligne 2 $(2+p)+(2+p)=(2+2)+2p=4+2p$
ligne 3 $(2+p)+(1+p)+(1+p)=(2+1+1)+3p=4+3p$.

Pour passer de f aux degrés on a pris $p=3$.

C'est en prenant des quantités p différentes pour des valeurs différentes de la fonction intrinsèque qu'on a obtenu le résultat.

5. 3. Méthode approchée.

C'est la méthode de D. R. DEUEL et A. GILL. Elle s'applique aux graphes ordinaires pondérés, c'est à dire, pour lesquels les arcs ont une valeur. Les automates en particulier se représentent par de tels graphes. Voir en bibliographie.

La méthode consiste à associer a chaque sommet un graphe représentant les chemins issus de ce sommet et tenant compte des valeurs des arcs. En travaillant en même temps sur les deux graphes dont on recherche l'isomorphisme on peut comparer les graphes associés aux sommets.

L'intérêt de cette méthode est de construire explicitement les chemins et les circuits des graphes susceptibles d'être isomorphes.

Indiquons que la méthode que nous avons étudiée peut s'appliquer aux graphes pondérés, moyennant un certain nombre de précaution. En particulier remplacer les demi-degrés (intérieur ou extérieur) par l'ensemble des valeurs affectées aux arcs adjacents (intérieurement ou extérieurement).

5. 4. Méthode propre à la représentation matricielle.

On déborde ici apparamment le cadre du problème de l'isomorphisme de graphes pour aborder celui de la P - équivalence de matrices (2.3.3). Un algorithme a été étudié par C. BÖHM et A. SANTOLINI (I964). Il est basé sur le produit scalaire inachevé.

5. 4. 1. Produit matriciel par produit scalaire inachevé.

Le produit matriciel calcule à l'aide du produit scalaire chaque coefficient de la matrice produit. Si on emploie la méthode du produit scalaire inachevé on obtient une matrice au lieu d'un scalaire. La matrice produit alors peut être de taille considérable ($n^2 \times n^2$ si la matrice de départ était n x n). Pour se ramener à une matrice n x n on affecte des valeurs différentes aux matrices différentes obtenues. Ce qui importe c'est de pouvoir distinguer les produits qui ne sont pas égaux. Ces valeurs suffisent et constituent la matrice produit.

5. 4. 2. Algorithme de la P - équivalence.

Pour trouver une correspondance entre les indices des rangées de deux matrices on calcule leurs puissances successives à l'aide du produit scalaire inachevé. A chaque étape la numérotation des matrices différentes obtenues se fait simultanément sur les deux matrices étudiées, de façon à avoir une correspondance entre des rangées se ressemblant dans les deux matrices. On utilise le même principe de correspondance entre les classes

de partitions des indices des deux graphes, ces partitions étant données par une équivalence obtenue en comparant les lignes et colonnes d'un même indice aux rangées correspondantes des autres indices.

On établit sur les indices une équivalence du même type que la S - équivalence pour les sommets. Les auteurs proposent de remplacer par un seul couple de rangées, les couples de ligne et colonne de chaque S - classe, quelque soit la puissance de la matrice où ces S - classes apparaissent. Cette méthode reviendrait à remplacer par un seul d'entre eux des sous-graphes isomorphes. Elle est en fait l'expression de la méthode des hypothèses dans leur théorie. Elle diffère de celle que nous avons étudiée par le fait que l'association est faite au moment de la construction de la correspondance, et que d'autre part on fait toujours une hypothèse vraie.

Notons que la méthode des deux fonctions correspond au calcul des puissances de la matrice, et pour terminer ce paragraphe remarquons que à une matrice à valeur entière positive on peut associer un multigraphe, et donc, théoriquement, le problème de l'isomorphisme entre ces multigraphes est celui de la P - équivalence des matrices.

VI. L'ALGORITHME

6. L'Algorithme

6.1 Description générale. Automatisation

Il contient trois parties principales

- création d'une FI partition de départ.
- méthode des deux fonctions.
- méthode des hypothèses.

Nous allons étudier l'automatisation de chacune de ces parties. Voir fig. 24 l'organigramme général. Un programme en language ALGOL a été réalisé.

6.1.1 Représentations utilisées.

La représentation choisie pour les graphes est la représentation matricielle (2.1.3.2).

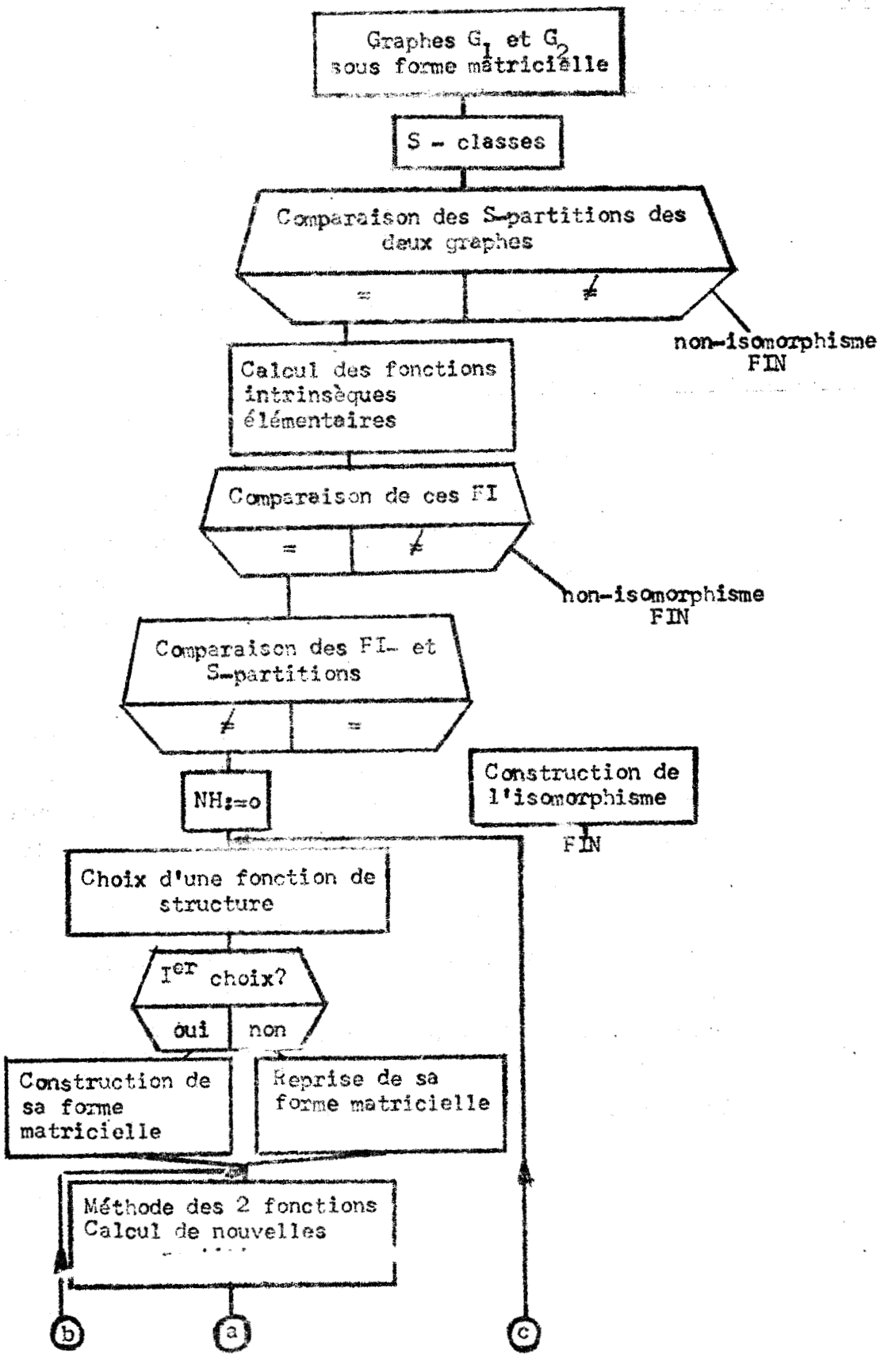
Les représentations des fonctions utilisées en découlent. Les fonctions intrinsèques et les fonctions associées aux FH partitions sont mises sous forme vectorielle (voir 5.2.1). Les fonctions de structure sont représentées par des matrices comme les graphes (voir 4.3.3.1).

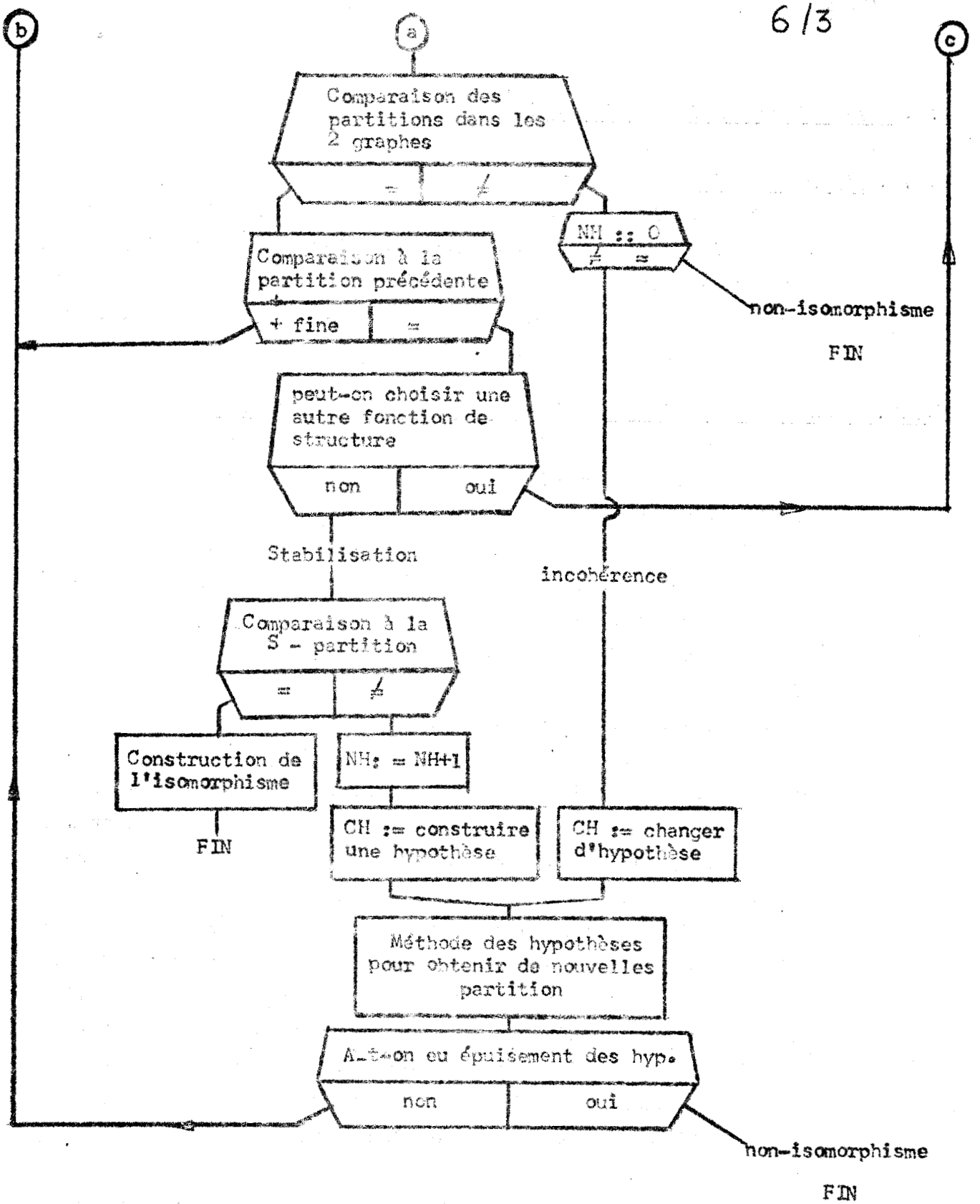
A tout instant les classes correspondantes des partitions des deux graphes ont même valeur par la fonction associée (4.2.5). Seules les S-classes échappent à cette règle car il n'est pas possible de connaître à priori une correspondance entre ces classes. Elles sont nécessaires pour accélérer l'algorithme (voir 4.6), on les calculera au début.

La correspondance biunivoque entre les sommets sera représentée par le tableau d'associations (2.9.2). Les valeurs intervenant en dernière rangée sont celles de la dernière fonction calculée, associée à une partition.

L'intérêt de ces représentations est d'automatiser facilement la méthode des deux fonctions et la méthode des hypothèses.

Fig. 24 Algorithme.




b
a
c
Comparaison des partitions dans les 2 graphes
=
≠
NH :: 0
≠
=
non-isomorphisme
FIN
Comparaison à la partition précédente
+ fine
=
peut-on choisir une autre fonction de structure
non
oui
Stabilisation
incohérence
Comparaison à la S - partition
=
≠
Construction de l'isomorphisme
FIN
NH := NH+1
CH := construire une hypothèse
CH := changer d'hypothèse
Méthode des hypothèses pour obtenir de nouvelles partition
A-t-on eu épuisement des hyp.
non
oui
non-isomorphisme
FIN

6.1.2 Opérations sur les partitions

6.1.2.1 Représentation

La fonction associée à une partition est suffisante pour la définir. Nous avons vu (6.1.1) que c'est au vecteur indicé par les sommets. La composante correspondant à un sommet x est le numéro de la classe de x dans la partition. On retrouve la partition en groupant les sommets qui ont des composantes égales.

6.1.2.2 Quantités associées à une partition. Caractéristiques

Deux types de quantités associées à une partition sont intéressants : le cardinal de chaque classe, le nombre de classes de même cardinal.

Ces nombres sont faciles à calculer à partir du vecteur, de la fonction associée. Ils permettent de construire des caractéristiques (voir3.2). On calcule d'abord un vecteur (1) indicé par les numéros des classes et dont les composantes sont les cardinaux des classes correspondantes. On refait sur ce vecteur le même calcul que celui qui a permis de le construire à partir du vecteur de la fonction associée, et on obtient un vecteur (2) indicé par différents cardinaux des classes et dont les composantes donnent le nombre de classes de cardinal correspondant.

La proposition 3.3.4.2 entraîne que les derniers vecteurs calculés sont des caractéristiques de type 1.

Pour les FI et FH partitions il n'est pas nécessaire de calculer le second vecteur. En effet pour ces partitions les classes de même numéro se correspondent, et comme l'isomorphie implique une correspondance biunivoque entre les sommets des FI et FH-classes, il est nécessaire qu'elles aient même cardinal. Le premier vecteur est pour les FI et FH partitions une caractéristique de type 1. C'est seulement pour les S-partitions qu'on calculera le second vecteur.

Si on utilise ces caractéristiques, il n'est pas nécessaire de

connaître le nombre de classes des partitions des deux graphes. Ces nombres, qui sont des caractéristiques de type 1 sont, soit la dimension des premiers vecteurs, soit la somme des composantes des seconds. Si on prend la précaution de numéroter les classes de FI et FH-partitions à partir de 1, et sans laisser passer de nombres dans au moins un des deux graphes, le nombre maximum utilisé sera le nombre de classes. Cela simplifie la recherche de nombre de classes quand on ne connaît que le vecteur de la fonction associée à la partition.

On trouvera dans le programme ALGOL page P3, ligne 109 une procédure qui automatise le calcul du premier vecteur.

```
Procédure CALCUL(FIM)POUR:(FI)DE TAILLE:(N)NOMBRE DE CLASSES:(CFI)
                                RESULTAT(ER);
   valeur N;
   entier N,CFI;
   entier tableau FIM,FI;
   boolean ER;
```

FI est le vecteur associé à une partition. Il est de dimension N, le nombre de sommets du graphe. CFI est le plus grand nombre utilisé pour numéroter les classes. Si aucun nombre n'a été oublié, c'est le nombre de classes. Pour les nombres oubliés la procédure indique que les classes correspondantes sont de cardinal 0. ER est un booleen qui prend la valeur vrai si CFI est strictement plus grand que N. Cela se produit dans le graphe où on a accepté de passer des nombres et signifie que les caractéristiques ne sont pas égales. Les partitions des deux graphes ne sont pas égales parce que dans le premier graphe une des classes a un cardinal plus grand que celui de sa correspondante dans l'autre graphe. Les classes ne peuvent évidemment pas se correspondre. Le nombre maximal de classes dans une partition est N=n, le nombre de sommets, c'est pour cela que CFI>N ne peut pas se produire dans le premier graphe (celui où on ne laisse pas passer de nombres).

D'autre part il est préférable que FI1 ne soit pas un tableau de dimension supérieure à N pour ne pas perdre inutilement de la place.

6.1.2.3 Comparaison des partitions

On compare les partitions à l'aide des vecteurs FI.

Les comparaisons sur le nombre de classes ne sont pas suffisantes. Il peut arriver que les classes se correspondent avec des cardinaux différents.

La procédure ALGOL se trouve page P3-ligne 124 du programme.

```
procédure COMPFI (N) GRAPHE1:(CFI1,FM1)
                     GRAPHE2:(CFI2,FI2)
                     RESULTAT:(EG);
   valeur N,CFI1,CFI2;
   entier N,CFI1,CFI2;
   entier tableau FI1,FI2;
   booleen EG;
```

Le booleen EG prend la valeur vrai si les deux vecteurs FI1 et FI2 sont égaux. Les partitions des deux graphes sont alors égales.

6.1.2.4 Partition plus fine que deux partitions

La partition obtenue en juxtaposant les fonctions associées à deux partitions est, parmi les plus fines, une des moins fines. Aux fonctions correspondent des vecteurs FI et la fonction de la nouvelle partition s'obtient en numérotant les couples différents de valeurs de même indice des FI (voir 4.2.5).

La procédure ALGOL est page P4 ligne 93.

```
procédure FID2FI (N)GRAPHE 1:(FI11,FI21,FI31)
                       GRAPHE 2:(FI12,FI22,FI32);
   valeur N;
   entier N;
   entier tableau FI11,FI21,FI31,FI12,FI22,FI32;
```

Cette procédure calcule simultanément sur les deux graphes une nouvelle partition dont le vecteur associé est FI31 dans le

premier graphe et FI32 dans le second. Les partitions précédentes avaient pour vecteurs associés FI11 et FI21 dans le premier graphe, FI12 et FI22 dans le second.

La plus grande finesse de la nouvelle partition par rapport aux précédentes, se traduit par un CFI31 plus grand qu'un CFI32. En effet la procédure numérote, sans passer de nombres, les classes de la nouvelle partition du premier graphe. C'est donc dans ce graphe que le numéro le plus élevé est le nombre de classes.

Cette procédure donne le principe d'automatisation du processus de juxtaposition dans un cas simple. Il apparaîtra dans une forme plus générale dans d'autres procédures (61.42 - 61.6).

6.1.3 Calcul des S-classes

Ce calcul se fait indépendemment dans les deux graphes dont on cherche un isomorphisme, puisqu'il n'y a pas de correspondance entre les S-classes.

D'après leur définition et la proposition 2.6.2

prop. 6.1.3

> x est S-équivalent à y si et seulement si leurs lignes et leurs colonnes dans la matrice caractéristique A sont partout égales sauf pour les éléments à leurs intersections qui vérifient.
>
> $A_{xy} = A_{yx}$ et $A_{xx} = A_{yy}$

Il s'agit d'un transcodage en notation matricielle de la proposition 2.6.2.

Leur calcul devient alors simple car il est facile de vérifier les conditions de la proposition ci-dessus. C'est la procédure page P4 ligne 188 qui le réalise.

```
procédure SCLASSES (N)SUR LE GRAPHE:(G)PARTITION:(S)NOMBRE DE
          CLASSES:(CS)CARDINAL PAR SOMMETS:(SD)NOMBRE DE
          CLASSES DE MEME CARDINAL:(SM);
   valeur N;
   entier N,CS;
     entier tableau G,S,SD,SM;
```

La matrice caractéristique du graphe est le tableau G. S est le vecteur de la fonction associé à la partition. Les valeurs de cette fonction sont les numéros des classes ordonnées suivant le plus petit indice des sommets qu'elles contiennent. C'est un ordre qui pratiquemment est arbitraire. SD est une fonction qui à chaque sommet associe le cardinal de sa classe. La partition associée à la fonction SD est moins fine que la H-partition (voir en 4.6.2). SD est une fonction intrinsèque. SM indique le nombre de classes qui ont même cardinal. C'est la caractéristique de type 1 correspondant au deuxième vecteur de 6.1.2.2. Il sert à comparer les S-partitions de deux graphes à l'aide de la procédure COMPFI (6.1.2.3).

6.1.4 FI-partitions

6.1.4.1 Fonctions intrinsèques élémentaires

Pour amorcer les itérations de la méthode des deux fonctions il faut connaître une FI-partition. Nous en possédons déjà une : la SD-partition (6.1.3). Le paragraphe 4.2.2 en donne une série. Dans l'algorithme nous les appellerons fonctions intrinsèques élémentaires car elles sont naturelles et faciles à calculer, en opposition aux autres.

Le programme les calcule page P2 ligne 65.

```
procédure FIELEM(N,G,DDI,DDE,BOUCLE);
   valeur N;
   entier N;
     entier tableau G,DDI,DDE,BOUCLE;
```

G est la matrice caractéristique du graphe, DDI et DDE sont les fonctions demi-degrés intérieur et extérieur. On les obtient en faisant la somme des éléments des colonnes et des lignes de la matrice. BOUCLE comme son nom l'indique est le nombre de boucles par sommets.

Le calcul de ces fonctions peut se faire indépendemment sur les deux graphes.

6.1.4.2 Juxtaposition de ces fonctions

Une procédure se charge de juxtaposer les quatre fonctions intrinsèques que nous connaissons maintenant. C'est sur la partition obtenue que la méthode des deux fonctions sera appliquée. Voir page P2 ligne 76.

```
procédure FIDFIELEM (N)GRAPHE1:(FI1,DDE1,DDI1,BOUCLE1,SD1)
                          GRAPHE2:(FI2,DDE2,DDI2,BOUCLE2,SD2);
   valeur N; entier N;
   entier tableau FI1,DDE1,DDI1,BOUCLE1,SD1,FI2,DDE2,DDI2,
                  BOUCLE2,SD2;
```

FI1 du graphe 1 et FI2 du graphe 2 sont des fonctions intrinsèques dont les FI-partitions sont plus fines que les quatre autres. La juxtaposition est réalisée en numérotant différemment les différents 4-uples de composantes de même indice dans les quatre

vecteurs des deux graphes. C'est la première généralisation de la juxtaposition de deux fonctions à un nombre quelconque de fonctions.

6.1.4.3 Comparaison des FI-partitions

Elle se fait comme pour toutes les partitions avec les procédures décrites en 6.1.2.2 et 6.1.2.3.

6.1.4.4 Test de l'isomorphisme

Il y a isomorphisme quand les partitions se sont stabilisées sur la S-partition.

Le fait qu'elles soient stabilisées est important. Il existe des graphes pour lesquels on obtient une correspondance biunivoque entre les S-classes, sans que cette correspondance soit celle impliquée par un isomorphisme. Voir fig. 34. Cela peut se produire quand les plus longs chemins n'ont pas été explorés. En itérant la méthode des deux fonctions si la S-partition est stabilisée on ne peut plus avoir de plus fine partition. S'il n'y a pas d'isomorphisme on ne peut pas avoir la correspondance habituelle entre les S-classes et l'itération fait numéroter d'une façon différente (en passant des nombres) les S-classes du second graphe.

La stabilisation est décelée à l'aide de tests dans l'algorithme. Ces tests ont été placés dans le programme principal. L'égalité de la partition à la S-partition se fait par la procédure page P3 ligne 135.

```
procédure TESTISO (N)GRAPHE1(FI1,FIM1,SD1)
                   GRAPHE2(FI2, FIM2,SD2)
                   RESULTAT (ISO);
```

valeur N; entier N ;
entier tableau FI1, FIM1, SD1, FI2, FIM2, SD2 ;
booleen ISO ;

ISO est un booleen qui est vrai si pour chaque sommet la FI - classe est de même cardinal que sa S - classe.

Il n'est pas nécessaire que la comparaison se fasse en même temps sur les deux graphes. Etant donné le caractère particulier de la S - partition, il n'est pas suffisant de comparer les nombres de classes des partitions. Le plus souvent les S - classes se réduisent à un sommet. De ce fait la procédure ne fait pas, en moyenne, trop de calculs inutiles. Pour des classes de plusieurs sommets, elle fait plusieurs fois la comparaison de leurs cardinaux (à cause de la représentation choisie).

6.1.5 - Fonctions de structure

Le calcul d'une fonction de structure dépend de la formule la représentant. Il existe de nombreux algorithmes pour rechercher des chemins et des circuits dans les graphes. On en trouvera dans la littérature sur la théorie des graphes.

En principe il est inutile de calculer les fonctions de structure du type γ'_a de 4.4.1.1. La méthode des deux fonctions les évite dans la construction du tableau T (4.4.2.3.)

Le programme que nous proposons n'utilise que la fonction successeur Γ et la fonction prédécesseur Γ^{-1}. La matrice associée à cette seconde est la transposée de la matrice caractéristique du graphe. La procédure TRANSP page P4 ligne 181 se charge de la transposition. Ces deux fonctions sont théoriquement suffisantes et les applications du programme le prouvent, puisqu'elles contiennent toute l'information sur le graphe. Les autres fonctions de structure ne pourraient qu'accélérer l'algorithme.

6.1.6 Methode des deux fonctions

L'étude faite en 5.2.2 sur le produit scalaire inachevé indique la reprédentation matricielle de la méthode des deux fonctions.

Elle se ramène à calculer le tabeeau T et à le juxtaposer à la fonction utilisée pour obtenir la fonction associée à la nouvelle partition, plus fine que la précédente. C'est la généralisation du processus de juxtaposition au cas d'un vecteur et d'un tableau. La composante du vecteur et la ligne de même indice du tableau forment le "couple de valeurs" à numéroter.

L'automatisation de la fonction de structure est réalisée dans le procédure page P6 ligne 24?

```
procédure BIMET2F (N)GRAPHE1:(G1,FI11,FI21,TA1,CFI11)
                            GRAPHE2:(G2,FI12,FI22,TA2);
   valeur N,CFI11;
   entier N,CFI11;
   entier tableau G1,FI11,FI21,G2,FI12,FI22,TA1,TA2;
```

TA1 et TA2 sont les tableaux T des graphes. Ils sont calculés en sommant les lignes dont les indices sont des sommets de même valeur par les fonctions FI11 et FI12 dans chaque graphe. CFI11 est le nombre de colonnes de ces tableaux. FI21 et FI22 sont les nouvelles fonctions obtenues après juxtaposition.

Remarquons que la façon de calculer les tableaux T permet de traiter des matrices dont les éléments ne sont pas seulement des 0 et des 1, et donc de traiter les multigraphes. Ceci sous-entend une modification du produit scalaire inachevé qui ne perd pas pour autant son efficacité.

Nous avons vu en 4.4.2.3 qu'il n'est pas nécessaire de calculer les matrices D.

6.1.7 - Méthode des hypothèses

L'automatisation pose quelques difficultés qui se situent dans deux domaines : d'abord le choix des sommets à associer, ensuite : la conservation des hypothèses.

6.1.7.1 - Choix des sommets

L'étude faite en 4.5.4.1 propose de faire à chaque niveau l'hypothèse sur des sommets appartenant à une classe ayant le plus petit cardinal. L'algorithme choisit parmi les classes ayant le plus petit cardinal, celle qui contient le sommet qui a le plus petit indice dans l'ensemble des sommets qu'elles contiennent. C'est avec ce sommet qu'elle fera la première hypothèse.

Pour changer l'hypothèse, dans le cas où elle donne une incohérence, il faut choisir dans la classe du second graphe un sommet qui n'ait pas encore servi à faire une hypothèse. L'algorithme les prend dans l'ordre croissant de leurs indices.

6.1.7.2 - Conservation des choix

La conservation de ces choix est faite dans un tableau H à trois colonnes et n-1 lignes (on ne voit au plus que n-1 niveaux d'hypothèses d'après 4.5.4.2). La première colonne contient dans la ligne NH le sommet du premier graphe choisi pour faire l'hypothèse de niveau NH, la seconde contient le sommet du second graphe, la troisième contient leur valeur par la fonction associée à la partition avant l'hypothèse. La valeur après l'hypothèse est le nombre de classes CFI augmenté d'une unité.

Si on change d'hypothèse à ce niveau on choisira un sommet du second graphe qui a un indice strictement supérieur au nombre en deuxième colonne et qui appartient à la classe dont le numéro est en troisième colonne. Le nombre en deuxième colonne sera changé, si on trouve une nouvelle hypothèse.

6.1.7.3 Conservation des partitions.

Elle permet de construire une nouvelle hypothèse à chaque niveau. On conserve les partitions sous la forme des fonctions associées dans un tableau indicé en ligne par les sommets et en colonne par les n-1 niveaux possibles d'hypothèses. Une ligne supplémentaire est ajoutée pour indiquer le nombre de classes. Elle permet de gagner du temps dans les calculs. Il existe un tableau pour chaque graphe.

Ces tableaux et le tableau précédent de conservation du choix suffisent pour conserver les hypothèses. Dans le programme on a ajouté des tableaux identiques à ces deux-ci pour conserver les partitions après hypothèses. L'intérêt est de gagner du temps dans le cas où il faut changer d'hypothèses, en rectifiant la modification faite à la dernière hypothèse de ce niveau à l'aide de la troisième colonne du tableau de conservation des choix. Notons que les quatre tableaux de conservation des partitions suffisent pour conserver les hypothèses (à condition d'utiliser le principe de choix de 6.1.7.1) mais sont d'un emploi beaucoup plus lourd.

6.1.7.4 Automatisation

Elle est réalisée dans le programme par deux procédures.

procédure CONSTHYPOT (N)GRAPHE1:(FI11,CFI11,FIM11,FI21,CFI21,S1,SD1)
GRAPHE2:(FI12,FIM12,FI22,S2,SD2)
HYPOTHESE (H)DENIVEAU:(NH)DETYPE:(CH)
RESULTAT:(EG);

valeur N,CFI11;
entier N,CFI11,CFI21,NH;
entier tableau FI11,FIM11,FI21,S1,SD1,FI12,FIM12,FI22,S2,H;
booleen CH,EG;

Cette procédure réalise le choix et le conserve. (page P6 ligne 276)

FI11,CFI11,FIM11 sont les informations relatives à la partition du premier graphe avant hypothèse ; FI12,FIM12 sont celles relatives au second graphe (CFI11 = CFI21). S1,SD1,S2,SD2, sont les

informations relatives aux S-partitions des graphes. FI21 et FI22 sont les nouvelles fonctions obtenues. H est le tableau de conservation des choix, NH le niveau de la dernière hypothèse faite. CH est un booleen égal à vrai s'il s'agit de construire une première hypothèse à un niveau donné. Dans ce cas NH augmente d'une unité. CH est faux dans le cas d'un changement d'hypothèse.
En sortant de la procédure CH est faux dans le cas où il n'a pas été possible de trouver une hypothèse au niveau proposé. EG est un booleen de sécurité qui prend la valeur vrai si la partition proposée est la S-partition.

Au cours du paragraphe 4.6 nous avons remarqué qu'il était plus rapide de travailler sur les S-classes plutôt que sur les sommets. Pour appliquer ce résultat le programme ne choisit pas un sommet dans chaque graphe, mais une S-classe. Il la prend dans une des FH-classes qui en contient le moins. La conservation peut encore se faire dans le tableau de conservation des choix en ne gardant qu'un sommet des S-classes choisies. Les fonctions associées aux FH-partitions sont modifiées pour tous les sommets des S-classes choisies (Organigramme fig28)

La deuxième procédure conserve les partitions. Elle englobe la première au moment de l'utilisation. page P7 ligne 305.

```
procédure HYPOT (N)GRAPHE1:(FI11,FI21,FIM11,CFI11,SD1,S1)
                       GRAPHE2:(FI12,FI22,FIM12,S2)
                       HYPOTHESE:(NH,CH)
                       CONSERVATION:(H,FIAVH1,FIAPH1,FIAVH2,FIAPH2);
   valeur N;
   entier N,NH,CFI11;
```

entier tableau FI11, FI21, FI12, FI22, H, FIAVH1, FIAPH1,
FIAVH2, FIAPH2, FIM11, SD1, S1, S2, FIM12 ;

booleen CH ;

Les symboles sont les mêmes que dans la procédure HYPOT. Les tableaux FIAVH et FIAPH sont ceux qui servent à la conservation des partitions. C'est le type de tableau à déclarer rémanent dans un programme ALGOL. Le compilateur employé ne le permettrait pas. (Organigramme fig. 27).

6.1.8 - Construction de l'isomorphisme

La correspondance entre les sommets qui donne l'isomorphisme est calculée dans un vecteur SAS. Il est indicé par les sommets du premier graphe et sa composante d'indice x est l'indice du sommet correspondant dans le second graphe. C'est une représentation du type tableau d'association. Ce vecteur de sommets associés correspond aux deux premières lignes du tableau en 2.9.2.

L'automatisation est réalisée par la procédure ISOMORPHISME page P4 ligne 145.

6.2 - Le programme

Il enchaîne les procédures, que nous avons décrites en 6.1, suivant l'organigramme de l'algorithme (fig. 24). Les différences entre le programme et l'algorithme sont dûes aux restrictions apportées à celui-ci. Il contient d'autre part un certain nombre de procédures de service.

6.2.1 - Les procédures de services

6.2.1.1 - Fonction intrinsèque préalable

Une séquence de programme permet d'introduire des partitions préalablement calculées sur les deux graphes. Ces partitions sont testées et comparées à la S - partition comme si elles avaient été calculées par le programme. Elles sont ensuite juxtaposées aux fonctions intrinsèques élémentaires.

Ceci permet de rectifier les insuffisances dues aux restrictions de l'organigramme et de faire intervenir des propriétés qui ne sont jamais (ou seulement après de nombreux calculs) testées, et qui sont connues de l'utilisateur du programme.
Ce procédé permet aussi d'orienter la solution en proposant une hypothèse qui ne pourra pas être changée. (voir page P9, ligne 413 et page P10, ligne 485).

6.2.1.2 - Procédures d'introduction

Ces procédures ont pour but d'introduire dans la mémoire du calculateur les matrices caractéristiques des graphes à traiter. Elles sont trois et introduisent les matrices à partir de représentations différentes à l'extérieur du calculateur. Un code associé au graphe qui donne son numéro, le nombre de ses sommets et de ses arcs, indique également la procédure à utiliser.

Le corps de ces procédures peut être modifié pour permettre d'utiliser une représentation d'un type différent à l'extérieur.

Les procédures sont page P1 lignes 2,9 et 17. La forme extérieure des matrices caractéristiques correspondante est donnée en 7,1,2.

6.2.1.3 Les séquences d'impression

6.2.1.3.1 Impression des données

Leur intérêt est d'indiquer sur quelles matrices travaille le programme.

Ce sont des procédures d'impression d'une matrice. Elles sont sorties ligne par ligne à raison de 16 éléments par ligne d'impression. La procédure SMATRICE (page P1 ligne 29) imprime une matrice* carrée d'ordre quelconque (voir pages p 17 et P 27 la présentation de ces matrices).

Le corps de la procédure SDONNEES a été sacrifié dans le programme. Le gain de place obtenu (300 mots de mémoire) a permis au programme compilé de tenir dans la mémoire du calculateur.

6.2.1.3.2 Impression des Resultats

Dans le cas d'isomorphisme c'est une séquence de programme qui réalise l'impression des résultats. Elle commence page P13 ligne 610. Elle se charge d'imprimer la liste des sommets, puis le contenu du tableau SAS (6.1.7) qui donne le correspondant dans le second graphe de chaque sommet de premier graphe, enfin en troisième ligne la liste des S-classes des graphes par le vecteurS.
On obtient un tableau d'associations du type en 2.9.2 (voir pages P25 et P29).

Dans le cas du non-isomorphisme une procédure TYPORESULT (page P4 ligne 156) présente un libellé qui indique par quel moyen a été découvert ce non-isomorphisme. Ces libellés sont imprimés par le programme.

* carrée d'ordre inférieur ou égal à 16. La procédure SDONNEES (page P2-ligne 60) imprime une matrice.

6.2.1.3.3 Resultats intermédiaires

Ils sont imprimés à la demande de l'utilisateur en donnant au booleen CRESULT la valeur vrai par introduction du nombre 1 avant les données relatives aux graphes à traiter. On introduit 0 quand on ne les désire pas.

Leur intérêt est d'avoir facilité la mise au point du programme et maintenant de montrer comment procéde l'algorithme. Ils ont l'inconvénient d'augmenter le temps d'exploitation (environ 10 fois).

Deux procédures se chargent de ce travail. La procédure SFONCTION (page P5 ligne 206) qui imprime les fonctions associées aux partitions et la procédure STABLEAU (page P5 ligne 223) qui imprime les tableaux T calculés dans la méthode des deux fonctions. Dans le cas où une hypothèse est faite les sommets choisis dans chacun des graphes sont imprimés par une séquence de calcul (page P 12 ligne 575).

Pour des raisons de présentation les résultats intermédiaires ne sont imprimés que si N≤16. (voir la présentation à partir de la page 19)

6.2.1.4 Procédure de tests

Les tests sur les fonctions de partitions ont été rassemblés dans une procédure, pour éviter la répétition de la séquence de programme les contenant. Voir la procédure TESTFI (page P4 ligne 163). Ce systeme n'empêche pas l'emploi individuel des procédures de tests. Organigramme : fig 25.

6.2.2 L'enchaînement des procédures

C'est ce que réalise le programme suivant l'organigramme fig 26 Cet organigramme ressemble à l'organigramme de l'algorithme. Le programme commence page P 19 ligne 349.

Fig. 25 Procédure TESTFI

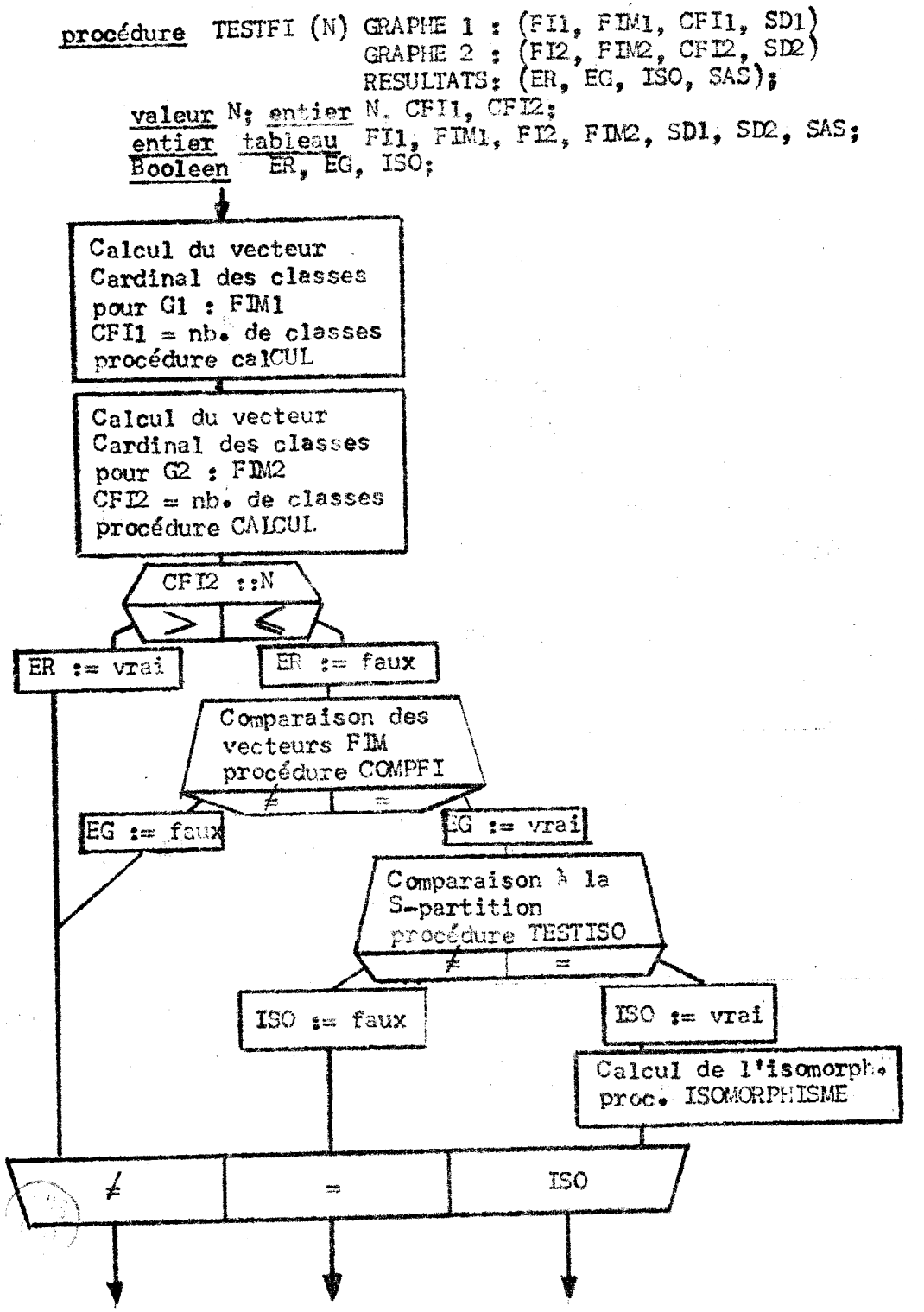


Fig 26 LE PROGRAMME -

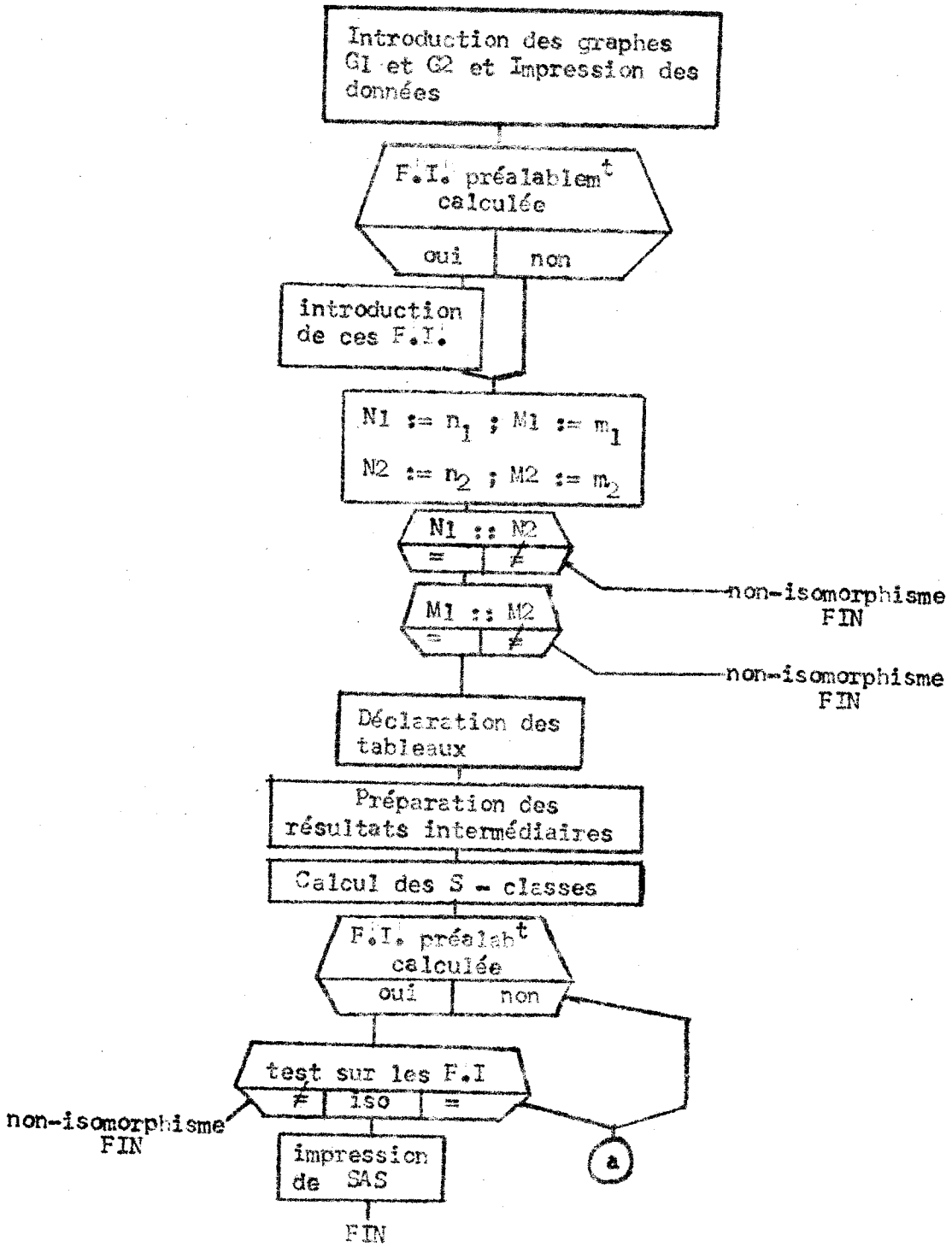
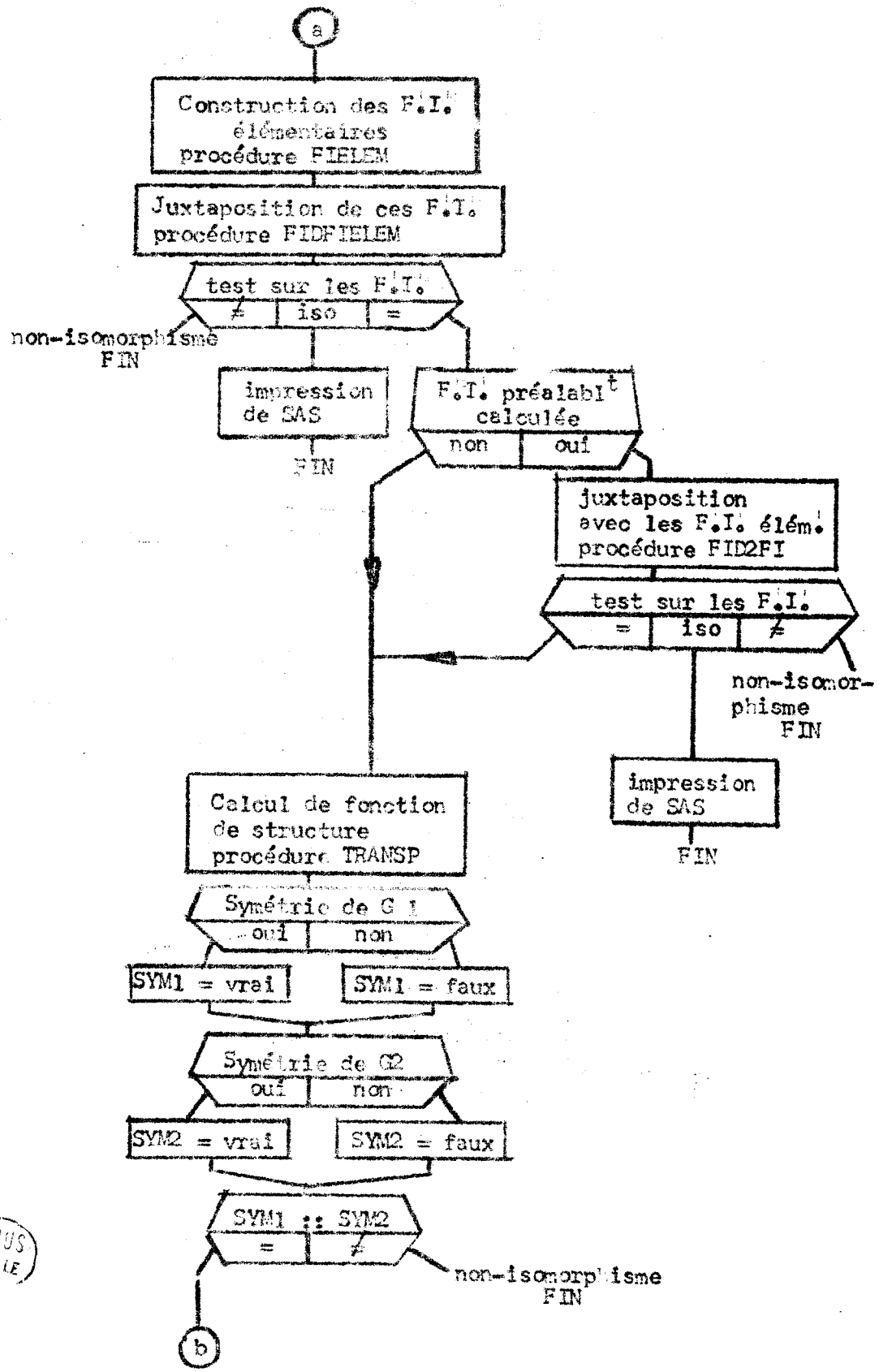


Fig. 26 (Suite) LE PROGRAMME (Suite)



.Fig. 26 (Fin) LE PROGRAMME (Fin)

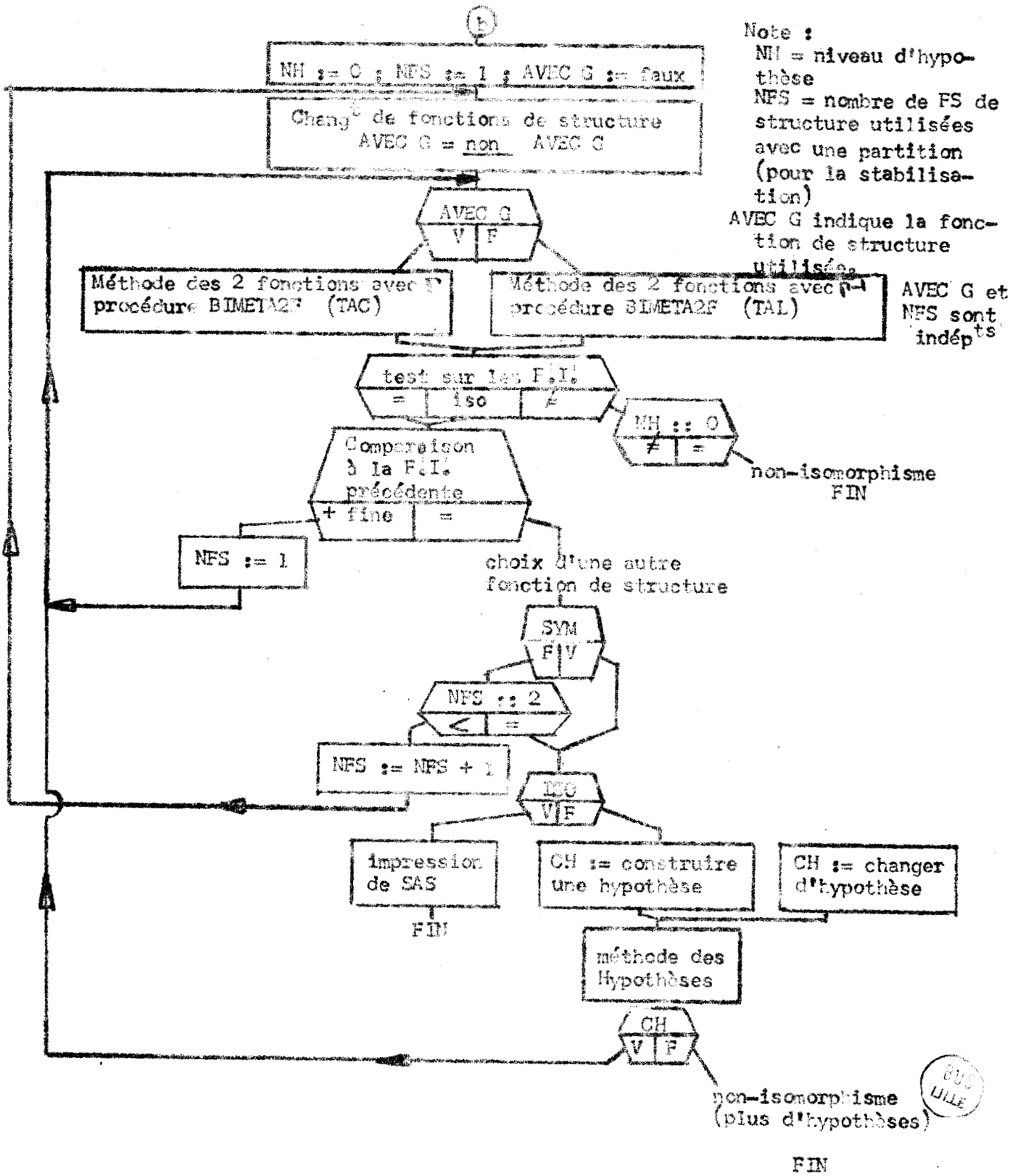


Fig. 27 **procédure** HYPOT

FI1 est la partition avant hypothèse

FI2 est la partition après hypothèse

EG = vrai si la FI1-partition était la S-partition.

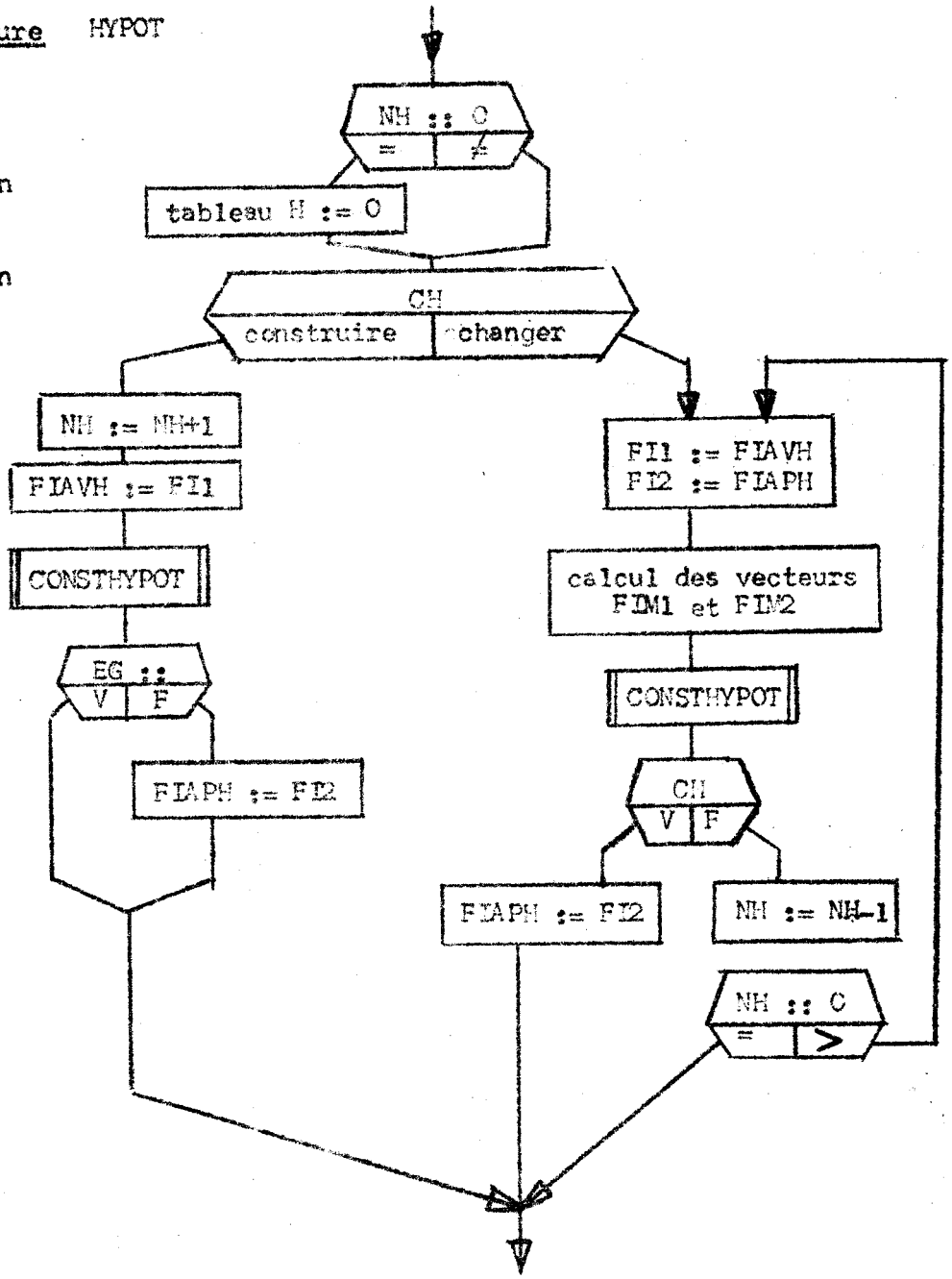
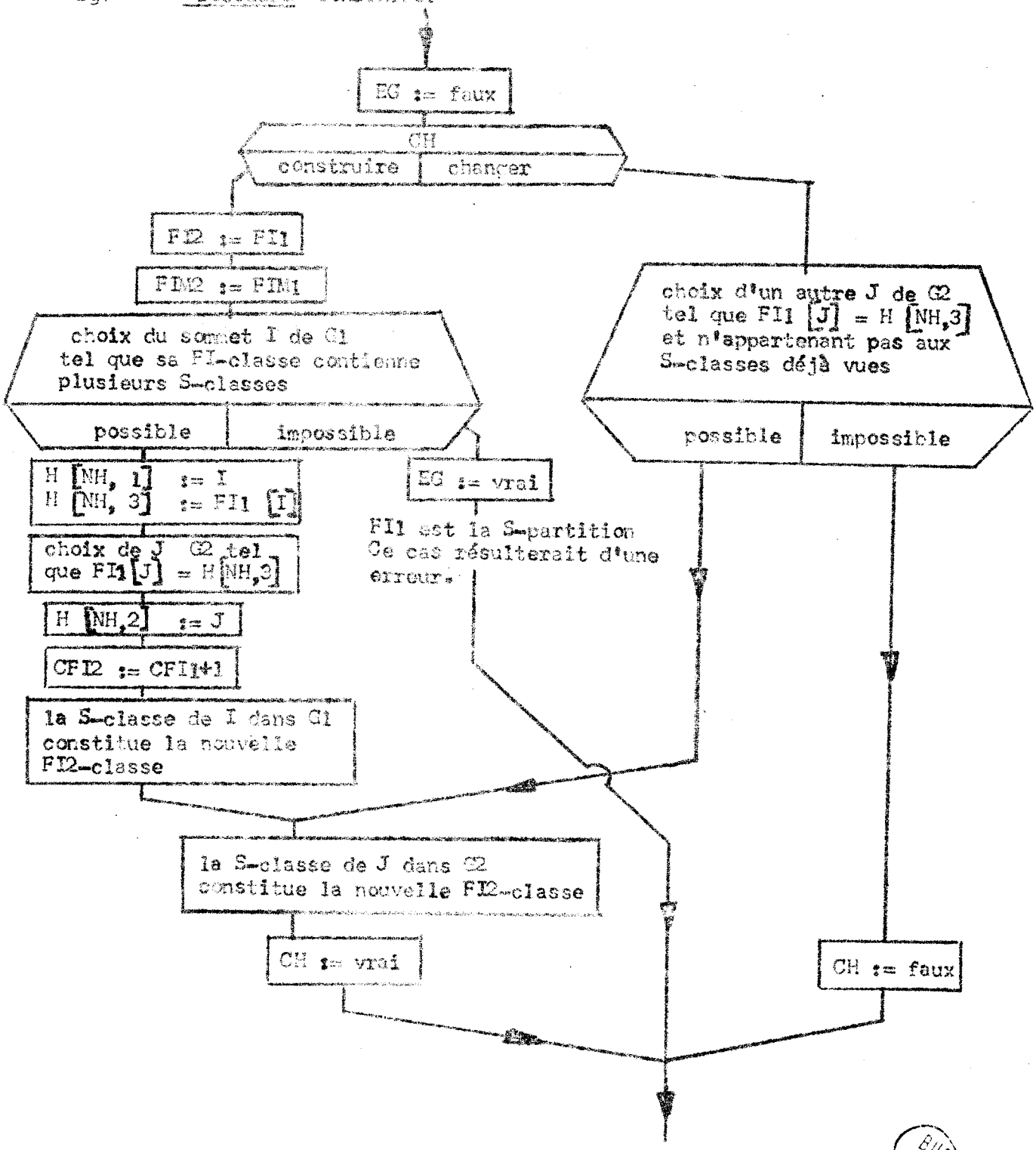


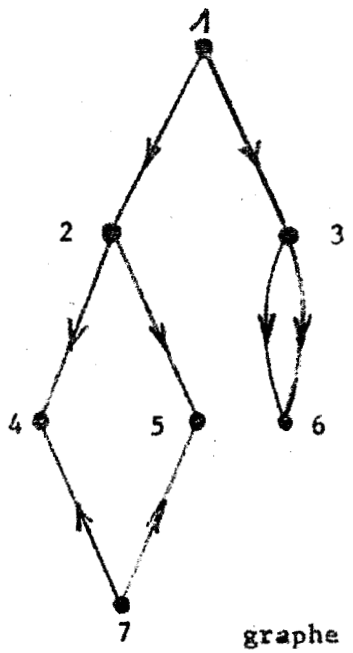
Fig.° 28 Procédure CONSTRUCT



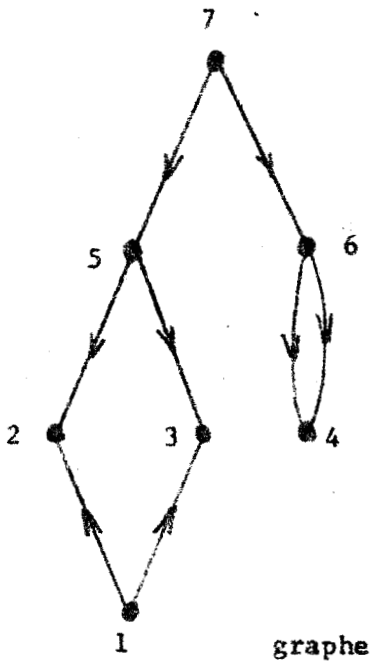
6.3. Exemples d'utilisation du programme

On trouvera dans les figures qui suivent des graphes qui ont été traités par le programme, et entre lesquels on a trouvé ou on n'a pas trouvé d'isomorphisme. Pour chacun des essais est indiquée la ligne générale du chemin suivi par le programme pour arriver au résultat qu'il a donné.

fig. 29



graphe 1



graphe 2

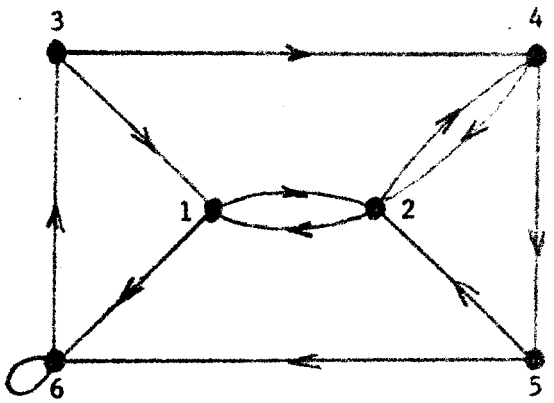
Traitement des graphes 1 et 2

Isomorphisme (3 itérations de la méthode des deux fonctions)

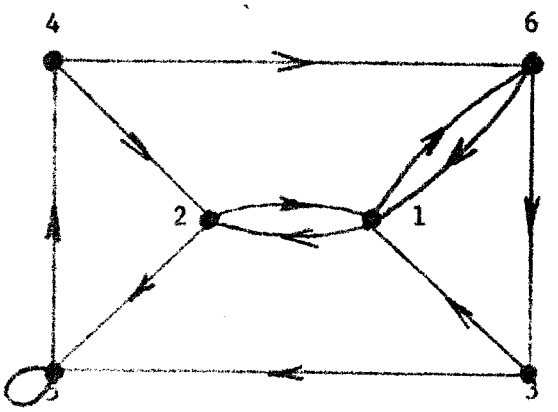
G1	1	2	3	4	5	6	7
G2	7	6	5	2	3	4	1
S-classes	1	2	3	4	4	5	6

On peut échanger les sommets 2 et 3 du graphe 2.

fig. 30



graphe 30



graphe 31

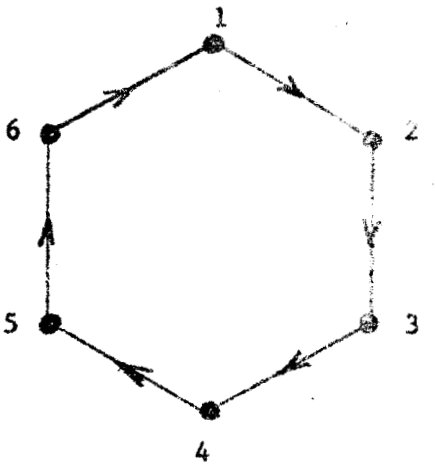
Traitement des graphes 30 et 31

Isomorphisme

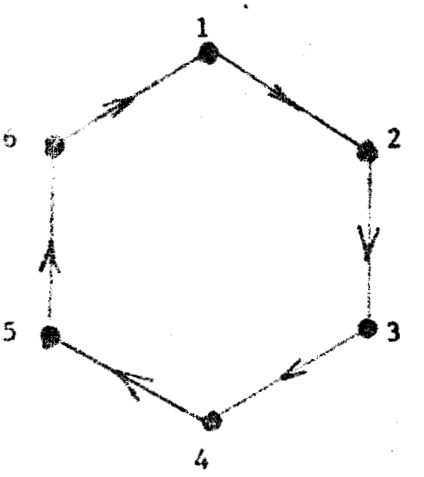
G30	1	2	3	4	5	6
G31	2	1	4	6	3	5
S-classes	1	2	3	4	5	6

Le programme a fait 3 itérations de la méthode des deux fonctions.

fig. 31



graphe 10



graphe 11

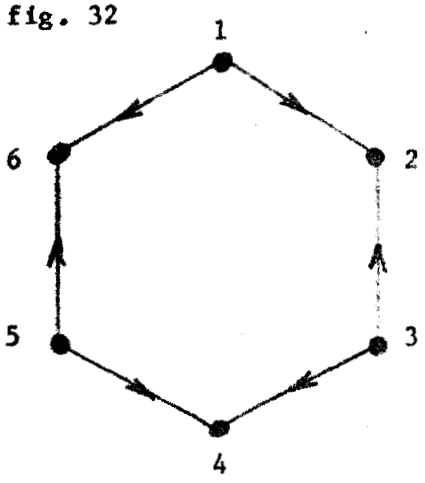
Ces graphes sont des circuits d'ordre n. Le groupe H est cyclique d'ordre n. Il y a n hypothèses, toutes justes, au niveau 1 qui est le seul niveau. Il faut ensuite n-1 itérations de la méthode des deux fonctions.

Traitement des graphes 10 et 11

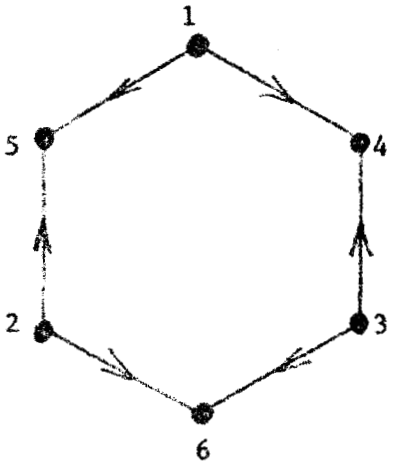
Isomorphisme

G10	1	2	3	4	5	6
G11	1	2	3	4	5	6
S-classes	1	2	3	4	5	6

fig. 32



graphe 5



graphe 6

Traitement des graphes 5 et 6 (voir page P 17)

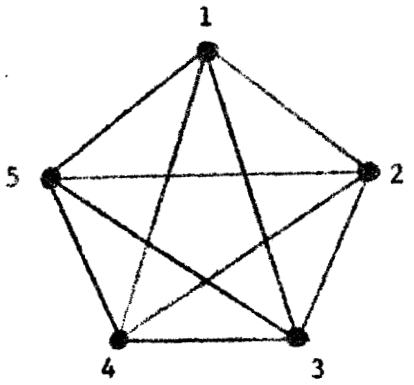
Isomorphisme

G5	1	2	3	4	5	6
G6	1	4	3	6	2	5
S-classes	1	2	3	4	5	6

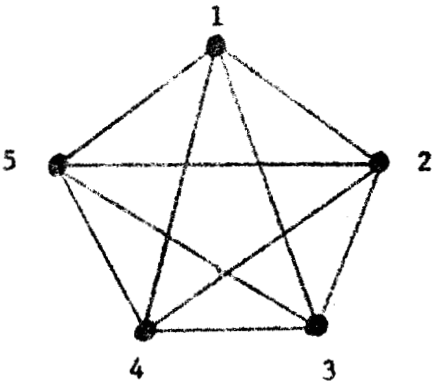
Le programme rencontre deux niveaux d'hypothèses.

Au premier niveau il y a 3 hypothèses possibles toutes justes.
Au second niveau il y a 2 hypothèses pour chaque hypothèse de niveau 1, toutes justes. Il faut 3 itérations (avec les contrôles de stabilité) de la méthode des deux fonctions entre les 2 niveaux et aussi après la dernière hypothèse.

fig.33



graphe 12



graphe 13

graphes symétriques, pleins, sans boucles.

Pour ces graphes il n'y a qu'une S-classe, les groupes H sont les groupes symétriques S_5.

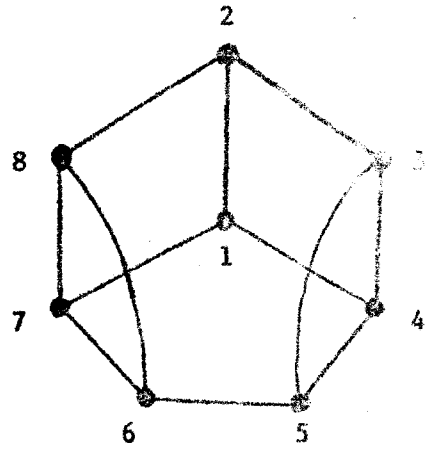
Traitement des graphes 12 et 13

Isomorphisme

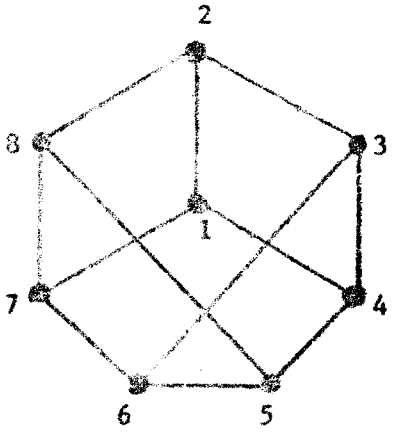
G1	1	2	3	4	5
G2	1	2	3	4	5
S-classes	1	1	1	1	1

Le programme s'arrête dès le calcul des S-classes.
La correspondance est arbitraire.

fig.34



graphe 20



graphe 21

graphes symétriques

2 circuits de longueur 3 — pas de circuits de longueur 3

Le graphe 21 s'obtient à partir du graphe 20 en échangeant les extrêmités 5 et 6 des arêtes issues de 3 et 8.

Traitement des graphes 20 et 21

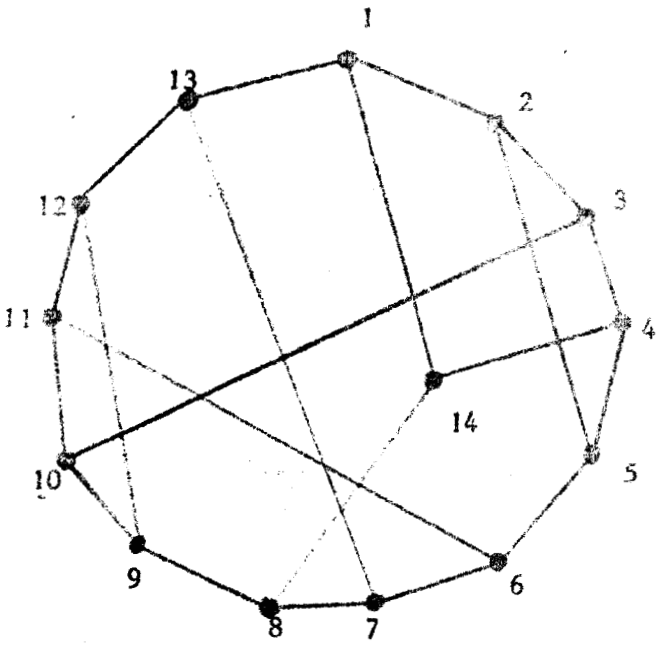
Non-isomorphisme

Par épuisement des hypothèses.

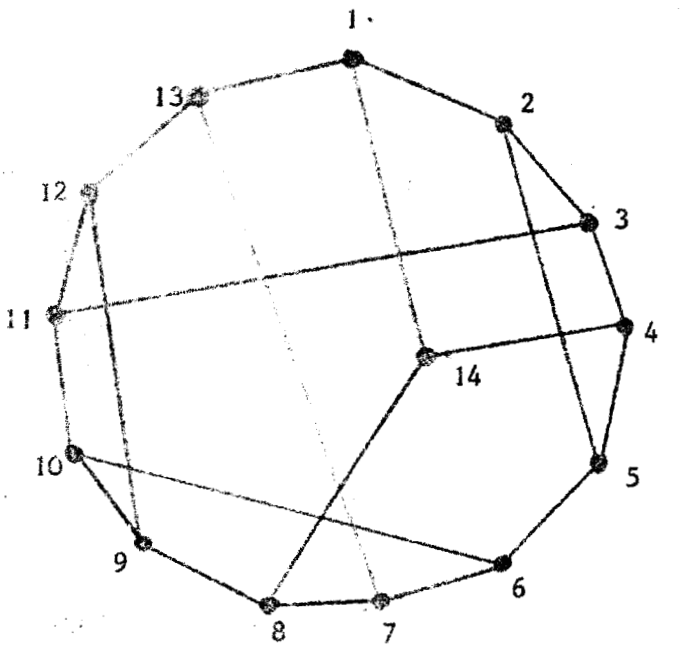
La plus fine FI-partition ne contient qu'une classe (graphes symétriques de degré 3). Il y a deux niveaux d'hypothèses. Au premier niveau il y a 8 hypothèses possibles. Au second niveau, il y en a 2 pour chaque hypothèse de niveau 1. Elles ont toutes été testées, et 90 partitions ont été calculées.

Dans le traitement de tels graphes on rencontre des correspondances biunivoques entre les sommets (ou les S-classes : chacune n'a qu'un sommet) non-stabilisées, qui donc ne correspondent pas à un isomorphisme (voir 6.1.4.4.)

Figure 35



graphe 32



graphe 33

Ces graphes sont symétriques.

Le graphe 33 s'obtient en échangeant les extrémités 10 et 11 des arêtes issues de 3 et 6 dans le graphe 32.

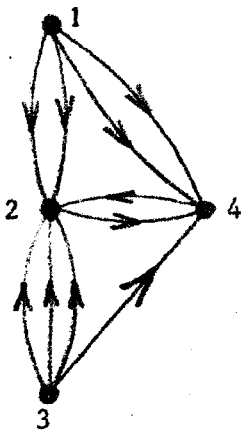
Traitement des graphes 32 et 33 (voir P27)

Isomorphisme.

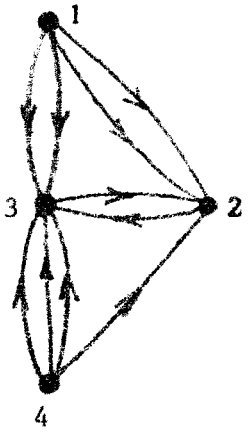
G 32	1	2	3	4	5	6	7	8	9	10	11	12	13	14
G 33	14	4	3	2	5	6	7	13	12	11	10	9	8	1
S-classes	1	2	3	4	5	6	7	8	9	10	11	12	13	14

La plus fine F-I-partition ne contient qu'une classe. Etant donné l'ordre des sommets le programme a envisagé toutes les hypothèses de niveau 1 possible. Après la dernière, qui était la seule juste, il a fallu 6 itérations de la méthode des deux fonctions. Au total 55 partitions ont été calculées.

Fig. 36

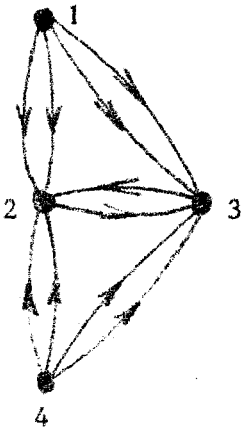


graphe 40



graphe 41

(multigraphes)



graphe 42

Traitement des graphes 40 et 41.

Isomorphisme (après 3 itérations de la méthode des deux fonctions).

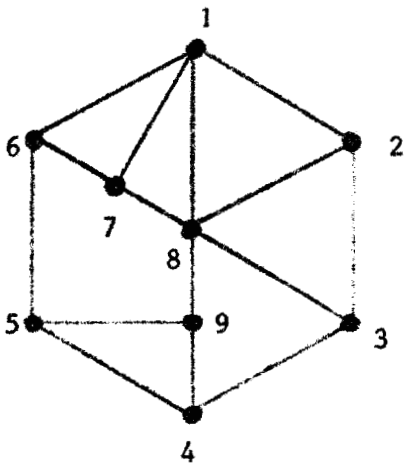
G 40	1	2	3	4
G 41	1	3	2	4
S-classes	1	2	3	4.

Traitement des graphes 40 et 42

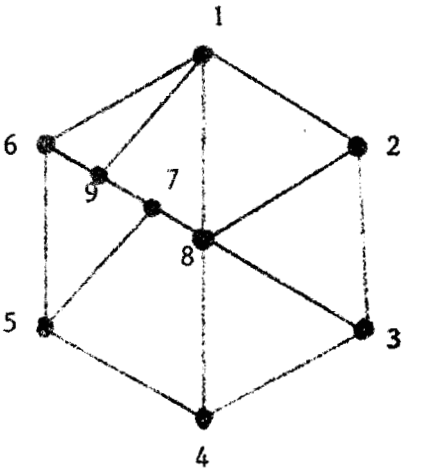
Non-isomorphisme

Les S-classes sont différentes.

Fig. 37



graphe 50



graphe 51

(graphes symétriques)

Traitement des graphes 50 et 51.

Non-isomorphisme

F-I-partitions différentes à la 2ème itération de la méthode des deux fonctions.

(Voir figure 14)

VII - L E P R O G R A M M E

E T S O N E X P L O I T A T I O N

7 - Le programme et son exploitation

7.1. Eléments du dossier du programme

Ce paragraphe contient les indications nécessaires à l'utilisateur du programme.

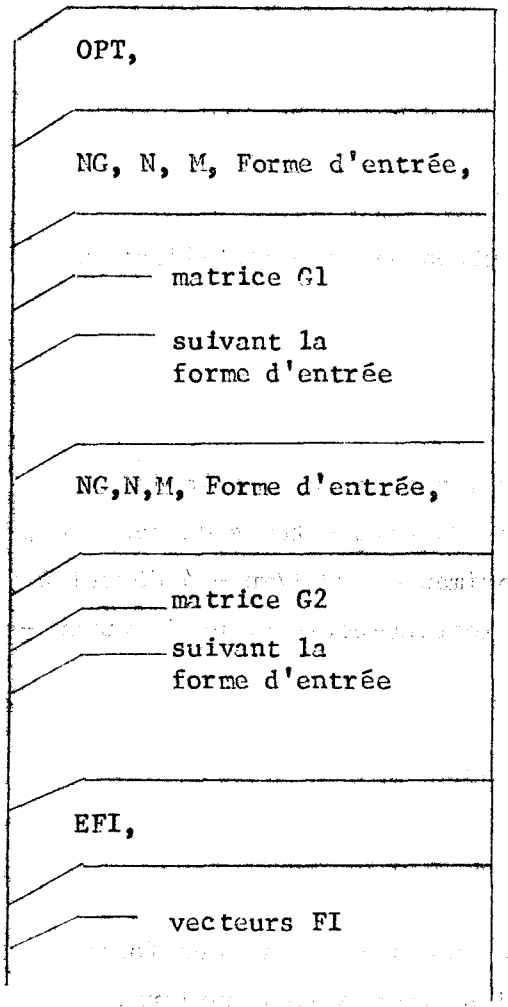
7.1.1. - Langage et machine

Le programme est écrit en ALGOL 60, version adaptée au calculateur BULL G.E. M40 (mémoire de 16 000 mots de 24 bits - temps de base 24 µs - un lecteur de cartes 600 c/mn - une imprimante 600 l/mn - 4 dérouleurs de bande - un tambour de 32 000 mots - lecteur-perforateur de ruban - 4 machines à écrire).

Les entrées-sorties sont du type GAMMA 60

7.1.2. - Données - Résultats

Les données sont entrées par cartes, les nombres étant sous forme ALGOL, séparés par des virgules. Tous les nombres sont entiers, positifs.



si OPT ≠ 0 alors sortie des résultats intermédiaires

Code du graphe 1

NG = numéro du graphe

N = nombre de sommets

M = nombre d'arcs

Forme d'entrée = code choisissant une des 3 procédures d'entrée

Code du graphe 2

si EFI = 0 alors il n'y a pas de fonction intrinsèque préalablement calculée

un vecteur associé à une FI partition pour chaque graphe si EFI ≠ 0.

Forme d'entrée

si Forme d'entrée = 1 alors

les cartes contiennent les éléments de la matrice caractéristique du graphe, ordonnés ligne par ligne

si Forme d'entrée = 2 alors

Les cartes contiennent les deux indices de chaque élément non nuls (leur valeur est 1)

si Forme d'entrée = 3 alors

Les cartes contiennent les deux indices et la valeur de chaque élément non nul

si Forme d'entrée a une autre valeur alors

Le programme n'introduit pas les deux graphes proposés.
Les résultats sont expliqués à leur sortie sur imprimante.

7.1.3. - Encombrement

Le programme compilé occupe pratiquement toute la place qui lui est réservé en mémoire. On peut évidemment l'optimiser encore, ou gagner de la place en supprimant des séquences d'intérêt secondaire comme la sortie des résultats intermédiaires.

Le nombre d'entier (sens ALGOL) réservé pour les tableaux est donné par la formule suivante, où n est le nombre de sommets de chacun des graphes traités :

$$nb = 10\ n^2 + 27\ n - 7$$

Le calculateur proposant 4 000 entier pour les tableaux, la valeur maximale pour n est alors n = 18 (nb = 3719).

7.1.4. - Le temps de calcul

Le listage, la compilation et l'assemblage avec les procédures de la bibliothèque nécessitent environ 5/100^{e} d'heure.

Le traitement de deux graphes nécessite, avec la sortie des résultats, au plus 2/100^{e} d'heure, sans résultats intermédiaires moins de 5".

7.2. Texte du programme - Résultats

Le document qui suit est celui qui a été obtenu au cours d'une exploitation. Aucune retouche n'a été faite sauf celle nécessitée par le cadrage.

Significations des identificateurs du programme

Les nombres renvoient à la ligne du programme où l'identificateur est déclaré.

B = booléen, E = entier, Eti = étiquette, P = procédure, T = entier tableau.

Le dernier chiffre d'un identificateur est le numéro du graphe.

L'avant dernier chiffre est le numéro de la fonction.

A	AUTFS	Eti	523	Changement de fonction de structure
	AVECG	B	1	Indique la fonction de structure utilisée
B	B	T	384	Nombre de boucles
	BIMETA2F	P	248	Méthode des 2 fonctions appliquée sur les 2 graphes
C	CALCUL	P	109	Calcul du cardinal des classes
	CFI	E	0	Nombre de classes de la FI correspondante
	CH	B	1	Construction d'hypothèse - Hypothèse construite
	COMPFI	P	124	Comparaison des partitions des deux graphes
	CONSTHYPOT	P	276	Construction ou changement d'une hypothèse
	CRESULT	B	1	Commande les résultats intermédiaires
	CS	E	0	Nombre de S-classes
D	DDE	T	384	Demi-degré extérieur
	DDI	T	384	Demi-degré intérieur
	DINSUF	Eti	654	Donnée insuffisante (Erreur sur FE)
E	EG	B	1	Partitions égales dans les deux graphes (ou dans CONSTHYPOT, partition égale à la S-partition)
	EINDELEM	P	17	Procédure d'entrée (3) des indices et des éléments
	EINDMAT	P	9	Procédure d'entrée (2) des indices
	EMATCAR	P	2	Procédure d'entrée (1) des matrices caractéristiques
	ER	B	1	Partition numérotée au-delà de N
F	FE	E	0	Forme d'entrée
	FI	T	384	Fonction associée à une partition - Fonction intrinsèque

	FIAPH	T	384	Partition après une hypothèse
	FIAVH	T	384	Partition avant une hypothèse
	FID2FI	P	93	Juxtaposition de deux partitions
	FIDFIELEM	P	76	Juxtaposition des FI-élémentaires
	FIELEM	P	65	Fonctions intrinsèques élémentaires
G	G	T	557-566	Matrice caractéristique
	GT	T	384	Fonction de structure, Matrice caractéristique transposée
H	H	T	384	Conservation des choix des hypothèses
	HYP	Eti	567	Méthode des hypothèses dans le programme
	HYPOT	P	305	Méthode des hypothèses
I	I	E	0	Indice
	ISO	B	1	Indique l'égalité d'une partition avec la S-partition
	ISOMORPHISME	P	145	Calcul de l'isomorphisme (SAS)
J	J	E	0	Indice
K	K	E	0	Indice
L	LAFIN	Eti	656	La fin du programme
M	M	E	0	Nombre d'arcs
N	N	E	0	Nombre de sommets
	NFS	E	0	Nombre de FS essayées à la stabilisation d'une partition
	NG	E	0	Numéro du graphe G
	NH	E	0	Niveau d'hypothèses
	NUFI	E	0	Nombre de FI calculées
O				
P				
Q				
R	REBIMETA2F	Eti	524	Reprise de la méthode des deux fonctions
S	S	T	384	S-partition
	SAS	T	384	Sommets associés dans l'isomorphisme
	SCLASSE	P	188	Calcul des S-classes
	SD	T	384	Cardinal de la classe de chaque sommet
	SDONNEES	P	60	Sortie de matrice caractéristique (N>16)
	SFONCT	P	206	Impression d'une fonction associée à une partition
	SM	T	384	Nombre de S-classes de même cardinal

	SMATRICE	P	29	Sortie de matrice caractéristique (N≤16)
	SORTREG	Eti	610	Sortie régulière : impression de l'isomorphisme
	STABLEAU	P	223	Impression d'un tableau T
	SYM	B	1	Indique la symétrie du graphe G (et le nombre de FS)
T	TA	T	384	Tableau T dans la méthode des deux fonctions
	TAC	libellé	Résultats	Tableau T construit sur G (colonne)
	TAL	libellé	intermédiaires	Tableau T construit sur G transposée (ligne)
	TESTFI	P	163	Tests sur les partitions
	TESTISO	P	135	Comparaison d'une partition à la S-partition
	TRANSP	P	181	Transposition de matrice
	TYPORESULT	P	156	Typographie préparatoire aux résultats (non-iso)

U

V

W

X

Y

Z

```
$JOB,STEEN-ISOMORPHISME-1012,00002,099,9999;

$ALGO,LIST;

MPILATION ALGOL M 40    DATE 060268&
 0000     'BEGIN''COMMENT'RECHERCHED'UNISOMORPHISMEENTREUNGRAPHEG1ETUNGRAPHEG
 0000     2;
 0000     'COMMENT'DECLARATIONS;
 0000     'INTEGER'N1,M1,NG1,FE1,N2,M2,NG2,FE2,CF111,CF112,CF121,CF122,CF131,C
 0000     F132,CS1,CS2,I,J,K,NUF1,NFS,NH,EF1;
 0001     'BOOLEAN'ISO,ER,EG,SYM1,SYM2,AVECG,CRESULT,CH;
 0002     'COMMENT'INTRO1;
 0002     'PROCEDURE'EMATCAR(G,N);
 0003     'VALUE'N;
 0004     'INTEGER'N;
 0005     'INTEGER''ARRAY'G;
 0006     'BEGIN''INTEGER'I,J;
 0007     'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'N'DO'G[I,J]:=DATA
 0007     ;
 0008     'END'DEEMATCAR;
 0009     'COMMENT'INTRO2;
 0009     'PROCEDURE'EINDMAT(G,N,M);
 0010     'VALUE'N,M;
 0011     'INTEGER'N,M;
 0012     'INTEGER''ARRAY'G;
 0013     'BEGIN''INTEGER'I,J;
 0014     'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'N'DO'G[I,J]:=0;
 0015     'FOR'J:=1'STEP'1'UNTIL'M'DO'G[DATA,DATA]:=1;
 0016     'END'DEEINDMAT;
 0017     'COMMENT'INTRO3;
 0017     'PROCEDURE'EINDELEM(G,N,M);
 0018     'VALUE'N,M;
 0019     'INTEGER'N,M;
 0020     'INTEGER''ARRAY'G;
 0021     'BEGIN''INTEGER'I,J,K,L;
 0022     'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'N'DO'G[I,J]:=0;
 0023     'FOR'I:=1'STEP'1'UNTIL'M'DO''BEGIN'K:=DATA;
 0024     L:=DATA;
 0025     G[K,L]:=DATA;
 0026     'IF'G[K,L]#1'THEN'I:=I+G[K,L]-1;
 0027     'END';
 0028     'END'DEEINDELEM;
 0029     'COMMENT'IMPRESSIONDESDONNEES(N<=16);
 0029     'PROCEDURE'SMATRICE(NUMEROG,G,N,M);
 0030     'VALUE'N,M,NUMEROG;
 0031     'INTEGER'N,M,NUMEROG;
 0032     'INTEGER''ARRAY'G;
 0033     'BEGIN''INTEGER'I,J;
 0034     SPACE(3);
 0035     TEXT("GRAPHE?\);
 0036     EDIT("F6.0\,NUMEROG);
 0037     TEXT("?DE?\);
 0038     EDIT("F6.0\,N);
 0039     TEXT("?SOMMETS?ET\);
 0040     EDIT("F6.0\,M);
 0041     TEXT("?ARCS\);
 0042     PRINT(4);
 0043     SPACE(3);
```

```
0044    TEXT("MATRICE-CARACTERISTIQUE\);
0045    PRINT(2);
0046    SPACE(9);
0047    TEXT("SOMMETS\);
0048    SPACE(5);
0049    'FOR'I:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,I);
0050    PRINT(2);
0051    PRINT(1);
0052    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'SPACE(9);
0053    EDIT("F6.0\,I);
0054    SPACE(6);
0055    'FOR'J:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,G[I,J]);
0056    PRINT(2);
0057    'END';
0058    PRINT(1);
0059    'END'DESMATRICE;
0060    'COMMENT'IMPRESSIONDESDONNEES;
0060    'PROCEDURE'SDONNEES(NUMEROG,G,N,M);
0061    'VALUE'N,M,NUMEROG;
0062    'INTEGER'N,M,NUMEROG;
0063    'INTEGER''ARRAY'G;
0064    ;
0065    'COMMENT'FONCTIONSINTRINSEQUESELEMENTAIRESPOURIGRAPHE;
0065    'PROCEDURE'FIELEM(N,G,FIDDI,FIDDE,BOUCLE);
0066    'VALUE'N;
0067    'INTEGER''ARRAY'G,FIDDI,FIDDE,BOUCLE;
0068    'INTEGER'N;
0069    'BEGIN''INTEGER'I,J;
0070    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIDDI[I]:=FIDDE[I]:=0;
0071    'FOR'J:=1'STEP'1'UNTIL'N'DO''BEGIN'FIDDI[I]:=FIDDI[I]+G[J,I];
0072    FIDDE[I]:=FIDDE[I]+G[I,J];
0073    'END';
0074    BOUCLE[I]:=G[I,I];
0075    'END''END'DEFIELEM;
0076    'COMMENT'CONSTRUCTIONDELAFIASSOCIEEAUXFIELEM;
0076    'PROCEDURE'FIDFIELEM(N,FI1,FIDDE1,FIDDI1,BOUCL1,SD1,FI2,FIDDE2,FIDD
0076    2,BOUCL2,SD2);
0077    'VALUE'N;
0078    'INTEGER''ARRAY'FI1,FIDDE1,FIDDI1,BOUCL1,SD1,FI2,FIDDE2,FIDDI2,BOUC
0078    2,SD2;
0079    'INTEGER'N;
0080    'BEGIN''INTEGER'I,J,CFI;
0081    'FOR'I:=1'STEP'1'UNTIL'N'DO'FI1[I]:=FI2[I]:=0;
0082    CFI:=1;
0083    'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FI1[I]=0'THEN''BEGIN'FI1[I]:=CFI;
0084    'FOR'J:=I+1'STEP'1'UNTIL'N'DO''IF'FI1[J]=0'AND'FIDDE1[J]=FIDDE1[I]'
0084    ND'FIDDI1[J]=FIDDI1[I]'AND'BOUCL1[J]=BOUCL1[I]'AND'SD1[J]=SD1[I]'TH
0084    N'FI1[J]:=CFI;
0085    'FOR'J:=1'STEP'1'UNTIL'N'DO''IF'FI2[J]=0'AND'FIDDE2[J]=FIDDE1[I]'AN
0085    'FIDDI2[J]=FIDDI1[I]'AND'BOUCL2[J]=BOUCL1[I]'AND'SD2[J]=SD1[I]'THEN
0085    FI2[J]:=CFI;
0086    CFI:=CFI+1;
0087    'END';
0088    'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FI2[I]=0'THEN''BEGIN'FI2[I]:=CFI;
0089    'FOR'J:=I+1'STEP'1'UNTIL'N'DO''IF'FI2[J]=0'AND'FIDDE2[J]=FIDDE2[I]'
0089    ND'FIDDI2[J]=FIDDI2[I]'AND'BOUCL2[J]=BOUCL2[I]'AND'SD2[J]=SD2[I]'TH
0089    N'FI2[J]:=CFI;
0090    CFI:=CFI+1;
```

```
0091      'END';
0092      'END'DEFIDFIELEM;
0093      'COMMENT'REUNIONDE2F.I.;
0093      'PROCEDURE'FID2FI(NI,FI1I,FI2I,FI3I,FII2,FI22,FI32);
0094      'VALUE'NI;
0095      'INTEGER'NI;
0096      'INTEGER''ARRAY'FI1I,FI2I,FI3I,FII2,FI22,FI32;
0097      'BEGIN''INTEGER'I,J,CFI;
0098      'FOR'I:=I'STEP'I'UNTIL'NI'DO'FI3I[I]:=FI32[I]:=0;
0099      CFI:=I;
0100      'FOR'I:=I'STEP'I'UNTIL'NI'DO''IF'FI3I[I]=0'THEN''BEGIN'FI3I[I]:=CFI;
0101      'FOR'J:=I+I'STEP'I'UNTIL'NI'DO''IF'FI3I[J]=0'AND'FI1I[J]=FI1I[I]'AND
0101      'FI2I[J]=FI2I[I]'THEN'FI3I[J]:=CFI;
0102      'FOR'J:=I'STEP'I'UNTIL'NI'DO''IF'FI32[J]=0'AND'FII2[J]=FI1I[I]'AND'F
0102      I22[J]=FI2I[I]'THEN'FI32[J]:=CFI;
0103      CFI:=CFI+I;
0104      'END';
0105      'FOR'I:=I'STEP'I'UNTIL'NI'DO''IF'FI32[I]=0'THEN''BEGIN''FOR'J:=I+I'S
0105      TEP'I'UNTIL'NI'DO''IF'FI32[J]=0'AND'FII2[J]=FII2[I]'AND'FI22[J]=FI22
0105      [I]'THEN'FI32[J]:=CFI;
0106      CFI:=CFI+I;
0107      'END';
0108      'END'DEFID2FI;
0109      'COMMENT'CALCULDESFIM;
0109      'PROCEDURE'CALCUL(FIM,FI,N,CFI,EG);
0110      'VALUE'N;
0111      'INTEGER'N,CFI;
0112      'INTEGER''ARRAY'FIM,FI;
0113      'BOOLEAN'ER;
0114      'BEGIN''INTEGER'I;
0115      CFI:=FI[I];
0116      ER:='FALSE';
0117      'FOR'I:=2'STEP'I'UNTIL'N'DO''IF'FI[I]>CFI'THEN'CFI:=FI[I];
0118      'IF'CFI>N'THEN''BEGIN'ER:='TRUE';
0119      'GOTO'SORTCALCUL;
0120      'END';
0121      'FOR'I:=I'STEP'I'UNTIL'N'DO'FIM[I]:=0;
0122      'FOR'I:=I'STEP'I'UNTIL'N'DO'FIM[FI[I]]:=FIM[FI[I]]+I;
0123      SORTCALCUL:'END';
0124      'COMMENT'COMPARAISONDESFONCTIONSINTRINSEQUESDE2GRAPHES;
0124      'PROCEDURE'COMPFI(N,CFI1,FIM1,CFI2,FIM2,EG);
0125      'VALUE'N,CFI1,CFI2;
0126      'INTEGER'N,CFI1,CFI2;
0127      'INTEGER''ARRAY'FIM1,FIM2;
0128      'BOOLEAN'EG;
0129      'BEGIN''INTEGER'I;
0130      EG:='FALSE';
0131      'IF'CFI1#CFI2'THEN''GOTO'SORTCOMPFI;
0132      'FOR'I:=I'STEP'I'UNTIL'CFI1'DO''IF'FIM1[I]#FIM2[I]'THEN''GOTO'SORTCO
0132      MPFI;
0133      EG:='TRUE';
0134      SORTCOMPFI:'END'DECOMPFI;
0135      'COMMENT'RECHERCHEDELISOMORPHISMEAVECLESFONCTIONSINTRINSEQUES;
0135      'PROCEDURE'TESTISO(N,FI1,FIM1,SD1,FI2,FIM2,SD2,ISO);
0136      'VALUE'N;
0137      'INTEGER'N;
0138      'INTEGER''ARRAY'FI1,FIM1,SD1,FI2,FIM2,SD2;
0139      'BOOLEAN'ISO;
```

```
0140    'BEGIN''INTEGER'I;
0141    ISO:='FALSE';
0142    'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FIMI[FI[[I]]#SDI[[I]'OR'FIM2[FI2[[I]]#
0142    SD2[[I]'THEN''GOTO'SORTESTISO;
0143    ISO:='TRUE';
0144    SORTESTISO:'END'DETESTISO;
0145    'COMMENT'CONSTRUCTIONISOMORPHISME;
0145    'PROCEDURE'ISOMORPHISME(N,FI[,FI2,SAS);
0146    'VALUE'N;
0147    'INTEGER'N;
0148    'INTEGER''ARRAY'FI[,FI2,SAS;
0149    'BEGIN''INTEGER'I,J;
0150    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN''FOR'J:=1'STEP'1'UNTIL'N'DO''IF'F
0150    I2[J]=FI[[I]'THEN''BEGIN'SAS[I]:=J;
0151    FI2[J]:=0;
0152    'GOTO'SORTJ;
0153    'END';
0154    SORTJ:'END';
0155    'END'DEISOMORPHISME;
0156    'COMMENT'TYPOGRAPHIE;
0156    'PROCEDURE'TYPORESULT;
0157    'BEGIN'JUMP;
0158    TEXT("RESULTATS\");
0159    PRINT(6);
0160    TEXT("???NON-ISOMORPHISME\");
0161    PRINT(2);
0162    'END'DETYPORESULT;
0163    'COMMENT'TESTSSURLESF.I.;
0163    'PROCEDURE'TESTFI(N,FI[,FIMI,CFI[,SDI,FI2,FIM2,CFI2,SD2,ER,EG,ISO,SA
0163    S);
0164    'VALUE'N;
0165    'INTEGER'N,CFI[,CFI2;
0166    'INTEGER''ARRAY'FI[,FIMI,FI2,FIM2,SDI,SD2,SAS;
0167    'BOOLEAN'ER,EG,ISO;
0168    'BEGIN'CALCUL(FIMI,FI[,N,CFI[,ER);
0169    'IF'ER'THEN''BEGIN'TYPORESULT;
0170    TEXT("???F.I.?NON-ACCEPTABLE?G[\);
0171    PRINT(2);
0172    'GOTO'STESTFI;
0173    'END';
0174    CALCUL(FIM2,FI2,N,CFI2,ER);
0175    'IF'ER'THEN''GOTO'STESTFI;
0176    COMPFI(N,CFI[,FIMI,CFI2,FIM2,EG);
0177    'IF''NOT'EG'THEN''GOTO'STESTFI;
0178    TESTISO(N,FI[,FIMI,SDI,FI2,FIM2,SD2,ISO);
0179    'IF'ISO'THEN'ISOMORPHISME(N,FI[,FI2,SAS);
0180    STESTFI:'END'DETESTFI;
0181    'COMMENT'CALCULDELATRANSPOSEEDELAMAT.CARACT;
0181    'PROCEDURE'TRANSP(GI,GTI,N);
0182    'VALUE'N;
0183    'INTEGER'N;
0184    'INTEGER''ARRAY'GI,GTI;
0185    'BEGIN''INTEGER'I,J;
0186    'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'N'DO'GTI[I,J]:=GI
0186    [J,I];
0187    'END'DETRANSP;
0188    'COMMENT'RECHERCHEDESCLASSES;
0188    'PROCEDURE'SCLASSES(N,G,S,CS,SD,SM);
```

```
0189      'VALUE'N;
0190      'INTEGER'N,CS;
0191      'INTEGER''ARRAY'G,S,SD,SM;
0192      'BEGIN''INTEGER'I,J,K,T;
0193      'FOR'I:=1'STEP'1'UNTIL'N'DO'S[I]:=SD[I]:=SM[I]:=0;
0194      CS:=1;
0195      'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'S[I]=0'THEN''BEGIN'S[I]:=CS;
0196      T:=1;
0197      'FOR'J:=I+1'STEP'1'UNTIL'N'DO''IF'S[J]=0'AND'G[I,I]=G[J,J]'AND'G[I,J
0197      ]=G[J,I]'THEN''BEGIN''FOR'K:=1'STEP'1'UNTIL'I-1,I+1'STEP'1'UNTIL'J-1
0197      ,J+1'STEP'1'UNTIL'N'DO''IF'G[J,K]#G[I,K]'OR'G[K,J]#G[K,I]'THEN''GOTO
0197      'SORTS;
0198      S[J]:=CS;
0199      T:=T+1;
0200      SORTS:'END';
0201      'FOR'K:=1'STEP'1'UNTIL'N'DO''IF'S[K]=CS'THEN'SD[K]:=T;
0202      CS:=CS+1;
0203      SM[T]:=SM[T]+1;
0204      'END';
0205      'END'DESCLASSES;
0206      'COMMENT'IMPRESSIOND'UNEFONCTION(N<=16);
0206      'PROCEDURE'SFONCT(F1,F2,N);
0207      'VALUE'N;
0208      'INTEGER'N;
0209      'INTEGER''ARRAY'F1,F2;
0210      'BEGIN''INTEGER'I;
0211      SPACE(7);
0212      TEXT("G1\");
0213      SPACE(12);
0214      'FOR'I:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,F1[I]);
0215      PRINT(2);
0216      SPACE(7);
0217      TEXT("G2\");
0218      SPACE(12);
0219      'FOR'I:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,F2[I]);
0220      PRINT(2);
0221      PRINT(1);
0222      'END'DESFONCT;
0223      'COMMENT'IMPRESSIONTABLEAUTA(N<=16);
0223      'PROCEDURE'STABLEAU(TA1,TA2,N,CF1);
0224      'VALUE'N,CF1;
0225      'INTEGER'N,CF1;
0226      'INTEGER''ARRAY'TA1,TA2;
0227      'BEGIN''INTEGER'I,J;
0228      SPACE(7);
0229      TEXT("G1\");
0230      PRINT(2);
0231      'FOR'I:=1'STEP'1'UNTIL'CF1'DO''BEGIN'SPACE(9);
0232      EDIT("F6.0\,I);
0233      SPACE(6);
0234      'FOR'J:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,TA1[J,I]);
0235      PRINT(2);
0236      'END';
0237      SPACE(7);
0238      TEXT("G2\");
0239      PRINT(2);
0240      'FOR'I:=1'STEP'1'UNTIL'CF1'DO''BEGIN'SPACE(9);
0241      EDIT("F6.0\,I);
```

```
0242      SPACE(6);
0243      'FOR'J:=1'STEP'1'UNTIL'N'DO'EDIT("F6.0\,TA2[J,I]);
0244      PRINT(2);
0245      'END';
0246      PRINT(1);
0247      'END'DESTABLEAU;
0248      'COMMENT'METHODEDES2FONCTIONS;
0248      'PROCEDURE'BIMETA2F(N,G1,FI1I,FI2I,TA1,CFIII,G2,FII2,FI22,TA2);
0249      'VALUE'N,CFIII;
0250      'INTEGER'N,CFIII;
0251      'INTEGER''ARRAY'G1,FIII,FI2I,G2,FII2,FI22,TA1,TA2;
0252      'BEGIN''INTEGER'I,J,K,CFI;
0253      'INTEGER'T1,T2,T3,T4;
0254      'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'CFIII'DO'TA1[I,J
0254      :=TA2[I,J]:=0;
0255      'FOR'I:=1'STEP'1'UNTIL'N'DO'FI2I[I]:=FI22[I]:=0;
0256      'FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1'STEP'1'UNTIL'N'DO''BEGIN'TA1[
0256      ,FIII[J]]:=TA1[I,FIII[J]]+G1[I,J];
0257      TA2[I,FII2[J]]:=TA2[I,FII2[J]]+G2[I,J];
0258      'END';
0259      CFI:=1;
0260      'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FI2I[I]=0'THEN''BEGIN'FI2I[I]:=CFI;
0261      'FOR'J:=I+1'STEP'1'UNTIL'N'DO''IF'FI2I[J]=0'AND'FIII[J]=FIII[I]'THE
0261      ''BEGIN''FOR'K:=1'STEP'1'UNTIL'CFIII'DO''IF'TA1[J,K]#TA1[I,K]'THEN'
0261      GOTO'SBII;
0262      FI2I[J]:=CFI;
0263      SBII:'END';
0264      'FOR'J:=1'STEP'1'UNTIL'N'DO''IF'FI22[J]=0'AND'FII2[J]=FIII[I]'THEN'
0264      BEGIN''FOR'K:=1'STEP'1'UNTIL'CFIII'DO''IF'TA2[J,K]#TA1[I,K]'THEN''G
0264      TO'SBI2;
0265      FI22[J]:=CFI;
0266      SBI2:'END';
0267      CFI:=CFI+1;
0268      'END';
0269      'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FI22[I]=0'THEN''BEGIN'FI22[I]:=CFI;
0270      'FOR'J:=I+1'STEP'1'UNTIL'N'DO''IF'FI22[J]=0'AND'FII2[J]=FII2[I]'THE
0270      ''BEGIN''FOR'K:=1'STEP'1'UNTIL'CFIII'DO''IF'TA2[J,K]#TA2[I,K]'THEN'
0270      GOTO'SBI3;
0271      FI22[J]:=CFI;
0272      SBI3:'END';
0273      CFI:=CFI+1;
0274      'END';
0275      'END'DEBIMETA2F;
0276      'COMMENT'CONSTRUCTIONDHYPOTHESE;
0276      'PROCEDURE'CONSTRYPOT(N,FIII,CFIII,FIMII,FI2I,CFI2I,SI,SDI,FII2,FIM
0276      2,FI22,S2,H,NH,CH,EG);
0277      'VALUE'N,CFIII;
0279      'INTEGER'N,CFIII,CFI2I,NH;
0279      'INTEGER''ARRAY'FIII,FIMII,FI2I,SI,SDI,FII2,FIMI2,FI22,S2,H;
0280      'BOOLEAN'CH,EG;
0281      'BEGIN''INTEGER'I,J,K;
0282      EG:='FALSE';
0283      'IF'CH'THEN''BEGIN''FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FI2I[I]:=FIII
0283      I];
0284      FI22[I]:=FII2[I];
0285      'END';
0286      'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FIMII[FIII[I]]>SDI[I]'THEN''BEGIN'H
0286      NH,I]:=1;
```

```
0287    H[NH,3]:=F1[[I];
0288    'GOTO'SORTIP1'END';
0289    EG:='TRUE';
0290    'GOTO'SORTHYP;
0291    SORTIP1:'FOR'I:=1'STEP'1'UNTIL'N'DO''IF'FI12[I]=H[NH,3]'THEN''BEGIN'
0291    H[NH,2]:=I;
0292    'GOTO'SORTOP2;
0293    'END';
0294    SORTOP2:CFI2I:=CFIII+1;
0295    'FOR'I:=H[NH,1]'STEP'1'UNTIL'N'DO''IF'SI[I]=SI[H[NH,1]]'THEN'FI2[[I]
0295    :=CFI2I;
0296    'END''ELSE''BEGIN''FOR'J:=H[NH,2]+1'STEP'1'UNTIL'N'DO''IF'FI12[J]=H[
0296    NH,3]'AND'S2[J]#S2[H[NH,2]]'THEN''BEGIN'K:=J;
0297    CH:='TRUE';
0298    'FOR'I:=H[NH,2]'STEP'1'UNTIL'N'DO''IF'S2[I]=S2[H[NH,2]]'THEN'FI22[I]
0298    :=H[NH,3];
0299    H[NH,2]:=K;
0300    'GOTO'SORTAP3'END';
0301    'GOTO'SORTHYP;
0302    'END';
0303    SORTAP3:'FOR'I:=H[NH,2]'STEP'1'UNTIL'N'DO''IF'S2[I]=S2[H[NH,2]]'THEN
0303    'FI22[I]:=CFI2I;
0304    SORTHYP:'END'DECONSTHYPOT;
0305    'COMMENT'METHODEDESHYPOTHESES;
0305    'PROCEDURE'HYPOT(N,FIII,FI2I,FIMII,CFIII,SDI,SI,FII2,FI22,FIMI2,S2,N
0305    H,CH,H,FIAVHI,FIAPHI,FIAVH2,FIAPH2);
0306    'VALUE'N;
0307    'INTEGER'N,NH,CFIII;
0308    'INTEGER''ARRAY'FIII,FI2I,FII2,FI22,H,FIAVHI,FIAPHI,FIAVH2,FIAPH2,FI
0308    MII,SDI,SI,S2,FIMI2;
0309    'BOOLEAN'CH;
0310    'BEGIN''INTEGER'I,J,CFI2I;
0311    'BOOLEAN'EG;
0312    'IF'NH=0'THEN''FOR'I:=1'STEP'1'UNTIL'N'DO''FOR'J:=1,2,3'DO'H[I,J]:=0
0312    ;
0313    'IF'CH'THEN''BEGIN'NH:=NH+1;
0314    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIAVHI[NH,I]:=FIII[I];
0315    FIAVH2[NH,I]:=FII2[I]'END';
0316    FIAVHI[NH,N+1]:=CFIII;
0317    FIAVH2[NH,N+1]:=CFIII;
0318    CONSTHYPOT(N,FIII,CFIII,FIMII,FI2I,CFI2I,SI,SDI,FII2,FIMI2,FI22,S2,H
0318    ,NH,CH,EG);
0319    'IF'EG'THEN''GOTO'FINHYPOT;
0320    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIAPHI[NH,I]:=FI2[[I];
0321    FIAPH2[NH,I]:=FI22[I];
0322    'END';
0323    FIAPHI[NH,N+1]:=CFI2I;
0324    FIAPH2[NH,N+1]:=CFI2I;
0325    'GOTO'FINHYPOT;
0326    'END';
0327    REPHYPOT:'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIII[I]:=FIAVHI[NH,I];
0328    FII2[I]:=FIAVH2[NH,I];
0329    FI2I[I]:=FIAPHI[NH,I];
0330    FI22[I]:=FIAPH2[NH,I];
0331    'END';
0332    CFIII:=FIAVHI[NH,N+1];
0333    CFI2I:=FIAPHI[NH,N+1];
0334    'FOR'I:=1'STEP'1'UNTIL'N'DO'FIMII[I]:=FIMI2[I]:=0;
```

```
0335    'FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIM1[FI[I]]:=FIM1[FI[I]]+
0336    FIM2[FI2[I]]:=FIM2[FI2[I]]+1;
0337    'END';
0338    CONSTHYPOT(N,FI1,CFI1,FIM1,FI2,CFI2,S1,SD1,FI2,FIM2,FI22,S2
0338    ,NH,CH,EG);
0339    'IF'CH'THEN''BEGIN''FOR'I:=1'STEP'1'UNTIL'N'DO''BEGIN'FIAPH1[NH,I]
0339    FI2[I];
0340    FIAPH2[NH,I]:=FI22[I];
0341    'END';
0342    'GOTO'FINHYPOT;
0343    'END';
0344    NH:=NH-1;
0345    'IF'NH>=1'THEN''GOTO'REPHYPOT;
0346    FINHYPOT:'END'DEHYPOT;
0347    'COMMENT'TYPOGRAPHIE;
0347    HEADING("ISOMORPHISME???DE???GRAPHES\);
0348    CRESULT:=DATA#0;
0349    'COMMENT'INTRODUCTIONDESGRAPHES;
0349    JUMP;
0350    TEXT("DONNEES\);
0351    PRINT(6);
0352    TEXT("G1\);
0353    PRINT(6);
0354    NG1:=DATA;
0355    N1:=DATA;
0356    M1:=DATA;
0357    'BEGIN''INTEGER''ARRAY'G1[1:N1,1:N1],FI3[1:N1];
0358    FE1:=DATA;
0359    'IF'FE1=1'THEN'EMATCAR(G1,N1)'ELSE''IF'FE1=2'THEN'EINDMAT(G1,N1,M1
0359    ELSE''IF'FE1=3'THEN'EINDELEM(G1,N1,M1)'ELSE''GOTO'DINSUF;
0360    'IF'N1<=16'THEN'SMATRICE(NG1,G1,N1,M1)'ELSE'SDONNEES(NG1,G1,N1,M1)
0361    TEXT("G2\);
0362    PRINT(6);
0363    NG2:=DATA;
0364    N2:=DATA;
0365    M2:=DATA;
0366    'BEGIN''INTEGER''ARRAY'G2[1:N2,1:N2],FI32[1:N2];
0367    FE2:=DATA;
0368    'IF'FE2=1'THEN'EMATCAR(G2,N2)'ELSE''IF'FE2=2'THEN'EINDMAT(G2,N2,M2
0368    ELSE''IF'FE2=3'THEN'EINDELEM(G2,N2,M2)'ELSE''GOTO'DINSUF;
0369    'IF'N2<=16'THEN'SMATRICE(NG2,G2,N2,M2)'ELSE'SDONNEES(NG2,G2,N2,M2)
0370    EFI:=DATA;
0371    'IF'EFI#0'THEN''BEGIN''FOR'I:=1'STEP'1'UNTIL'N1'DO'FI3[I]:=DATA;
0372    'FOR'I:=1'STEP'1'UNTIL'N2'DO'FI32[I]:=DATA;
0373    'END';
0374    'COMMENT'PREMIERSTESTS;
0374    'IF'N1#N2'THEN''BEGIN'TYPORESULT;
0375    TEXT("???NOMBRES?DE?SOMMETS?DIFFERENTS\);
0376    PRINT(6);
0377    'GOTO'LAFIN;
0378    'END';
0379    'IF'M1#M2'THEN''BEGIN'TYPORESULT;
0380    TEXT("???NOMBRES?D?ARCS?DIFFERENTS\);
0381    PRINT(6);
0382    'GOTO'LAFIN;
0383    'END';
0384    'COMMENT'DECLARATIONSDETABLEAUX;
0384    'BEGIN''INTEGER''ARRAY'GT1,GT2,TA1,TA2[1:N1,1:N1],S1,S2,SD1,SD2,FI
```

```
0384    I,FI12,FI22,DDE1,DDE2,DD11,DD12,B1,B2,SM1,SM2,FIM11,FIM12,FIM21,FIM2
0384    2,FIM31,FIM32,FI21,SAS[1:N1],H[1:3,1:N1-1],FIAVH1,FIAVH2,FIAPH1,FIAP
0384    H2[1:N1-1,1:N1+1];
0385    'COMMENT'RESULTATSINTERMEDIAIRES;
0385    'IF'CRESULT'THEN''BEGIN'JUMP;
0386    TEXT("RESULTATS?INTERMEDIAIRES\);
0387    PRINT(6);
0388    'IF'N1>16'THEN''BEGIN'CRESULT:='FALSE';
0389    TEXT("???N>16?PAS?DE?RESUL?INTERM\);
0390    'END''ELSE''BEGIN''COMMENT'PREPARATIONDESRESULTATSINTERMEDIAIRES(N<=
0390    16);
0390    TEXT("???NUMEROS?DES?SOMMETS?\);
0391    PRINT(6);
0392    SPACE(21);
0393    'FOR'I:=1'STEP'1'UNTIL'N1'DO'EDIT("F6.0\,I);
0394    PRINT(2);
0395    PRINT(1);
0396    'END';
0397    'END';
0398    'COMMENT'CALCULDESS-CLASSES;
0398    SCLASSES(N1,G1,S1,CS1,SD1,SM1);
0399    SCLASSES(N1,G2,S2,CS2,SD2,SM2);
0400    'IF'CRESULT'THEN''BEGIN'TEXT("???S-CLASSES\);
0401    PRINT(2);
0402    SFONCT(S1,S2,N1);
0403    TEXT("???TAILLES?DES?S-CLASSES\);
0404    PRINT(2);
0405    SFONCT(SD1,SD2,N1);
0406    'END';
0407    COMPFI(N1,CS1,SM1,CS2,SM2,EG);
0408    'IF''NOT'EG'THEN''BEGIN'TYPORESULT;
0409    TEXT("???S-CLASSES?DIFFERENTES\);
0410    PRINT(2);
0411    'GOTO'LAFIN;
0412    'END';
0413    'COMMENT'FI-PREEXISTENTES;
0413    NUFI:=0;
0414    'IF'EFI=0'THEN''GOTO'FISUITE;
0415    'IF'CRESULT'THEN''BEGIN'NUFI:=NUFI+1;
0416    TEXT("???F.I.??\);
0417    EDIT("F6.0\,NUFI);
0418    PRINT(2);
0419    SFONCT(FI31,FI32,N1);
0420    'END';
0421    'COMMENT'COMPARAISONDECESFI;
0421    CALCUL(FIM31,FI31,N1,CFI31,ER);
0422    'IF'ER'THEN''BEGIN'TYPORESULT;
0423    TEXT("???F.I.?INTRODUITE?NON?ACCEPTABLE(G1)\);
0424    PRINT(2);
0425    'GOTO'LAFIN;
0426    'END';
0427    CALCUL(FIM32,FI32,N1,CFI32,ER);
0428    'IF'ER'THEN''BEGIN'TYPORESULT;
0429    TEXT("???F.I.?INTRODUITE?NON?ACCEPTABLE(G2)\);
0430    PRINT(2);
0431    'GOTO'LAFIN;
0432    'END';
0433    COMPFI(N1,CFI31,FIM31,CFI32,FIM32,EG);
```

```
0434    'IF''NOT'EG'THEN''BEGIN'TYPORESULT;
0435    TEXT("???F.I.?DIFFERENTES\);
0436    PRINT(2);
0437    'GOTO'LAFIN;
0438    'END';
0439    TESTISO(NI,FI3I,FIM3I,SDI,FI32,FIM32,SD2,ISO);
0440    'IF'ISO'THEN''BEGIN'ISOMORPHISME(NI,FI3I,FI32,SAS);
0441    'GOTO'SORTREG;
0442    'END';
0443    FISUITE:;
0444    'COMMENT'CONSTRUCTIONDESFIELEM;
0444    FIELEM(NI,GI,DDII,DDEI,BI);
0445    FIELEM(NI,G2,DDI2,DDE2,B2);
0446    'IF'CRESULT'THEN''BEGIN'TEXT("???DEMI-DEGRES?INTERIEURS\);
0447    PRINT(2);
0448    SFONCT(DDII,DDI2,NI);
0449    TEXT("???DEMI-DEGRES?EXTERIEURS\);
0450    PRINT(2);
0451    SFONCT(DDEI,DDE2,NI);
0452    TEXT("???BOUCLES\);
0453    PRINT(2);
0454    SFONCT(BI,B2,NI);
0455    'END';
0456    FIDFIELEM(NI,FIII,DDII,DDEI,BI,SDI,FII2,DDI2,DDE2,B2,SD2);
0457    'IF'CRESULT'THEN''BEGIN'TEXT("???F.I.?\);
0458    NUFI:=NUFI+1;
0459    EDIT("F6.0\,NUFI);
0460    PRINT(2);
0461    SFONCT(FIII,FII2,NI);
0462    'END';
0463    CALCUL(FIMII,FIII,NI,CFIII,ER);
0464    'IF'ER'THEN''BEGIN'TYPORESULT;
0465    TEXT("???VERIFIER?F.I.ELEM?GI\);
0466    PRINT(2);
0467    'GOTO'LAFIN;
0468    'END';
0469    CALCUL(FIMI2,FIII,NI,CFII2,ER);
0470    'IF'ER'THEN''BEGIN'TYPORESULT;
0471    TEXT("???VERIFIER?F.I.ELEM?G2\);
0472    PRINT(2);
0473    'GOTO'LAFIN;
0474    'END';
0475    COMPFI(NI,CFIII,FIMII,CFII2,FIMI2,EG);
0476    'IF''NOT'EG'THEN''BEGIN'TYPORESULT;
0477    TEXT("???F.I.ELEM.?DIFFERENTES\);
0478    PRINT(2);
0479    'GOTO'LAFIN;
0480    'END';
0481    TESTISO(NI,FIII,FIMII,SDI,FII2,FIMI2,SD2,ISO);
0482    'IF'ISO'THEN''BEGIN'ISOMORPHISME(NI,FIII,FII2,SAS);
0483    'GOTO'SORTREG;
0484    'END';
0485    'COMMENT'REUNIONDESFI;
0485    'IF'EFI#0'THEN''BEGIN'FID2FI(NI,FI3I,FIII,FI2I,FI32,FII2,FI22);
0486    'IF'CRESULT'THEN''BEGIN'TEXT("???F.I.??\);
0487    NUFI:=NUFI+1;
0488    EDIT("F6.0\,NUFI);
0489    PRINT(2);
```

```
0490    SFONCT(FI2I,FI22,NI);
0491    'END';
0492    TESTFI(NI,FI2I,FIM2I,CFI2I,SDI,FI22,FIM22,CFI22,SD2,ER,EG,ISO,SAS);
0493    'IF'ER'THEN''BEGIN'TYPORESULT;
0494    TEXT("???F.I.?NON-ACCEPTABLE?G2\);
0495    PRINT(2);
0496    'GOTO'LAFIN;
0497    'END';
0498    'IF''NOT'EG'THEN''BEGIN'TYPORESULT;
0499    TEXT("???F.I.?DIFFERENTES\);
0500    PRINT(2);
0501    'GOTO'LAFIN;
0502    'END';
0503    'IF'ISO'THEN''GOTO'SORTREG;
0504    'FOR'I:=I'STEP'I'UNTIL'NI'DO''BEGIN'FII[I]:=FI2I[I];
0505    FII2[I]:=FI22[I];
0506    'END';
0507    CFII:=CFI2I;
0508    'END';
0509    'COMMENT'LAFIDEDEPARTESTTOUJOURSFII;
0509    'COMMENT'MATRICESTRANSPOSEES;
0509    TRANSP(GI,GTI,NI);
0510    TRANSP(G2,GT2,NI);
0511    SYMI:=SYM2:='TRUE';
0512    'FOR'I:=I'STEP'I'UNTIL'NI'DO''FOR'J:=I'STEP'I'UNTIL'NI'DO''BEGIN''IF
0512    'GI[I,J]#GTI[I,J]'THEN'SYMI:='FALSE';
0513    'IF'G2[I,J]#GT2[I,J]'THEN'SYM2:='FALSE';
0514    'END';
0515    'IF''NOT'(SYMI'EQUI'SYM2)'THEN''BEGIN'TYPORESULT;
0516    'IF'SYMI'THEN'TEXT("GISYMETRIQUE?ET?G2?NON-SYMETRIQUE\)'ELSE'TEXT("G
0516    I?NON-SYMETRIQUE?ET?G2?SYMETRIQUE\);
0517    PRINT(6);
0518    'GOTO'LAFIN;
0519    'END';
0520    'COMMENT'CALCUL;
0520    NH:=0;
0521    NFS:=I;
0522    AVECG:='FALSE';
0523    AUTFS:AVECG:='NOT'AVECG;
0524    REBIMETA2F:'IF'AVECG'THEN'BIMETA2F(NI,GI,FIII,FI2I,TAI,CFIII,G2,FII2
0524    ,FI22,TA2)'ELSE'BIMETA2F(NI,GTI,FIII,FI2I,TAI,CFIII,GT2,FII2,FI22,TA
0524    2);
0525    'IF'CRESULT'THEN''BEGIN''IF'AVECG'THEN'TEXT("???TAC\)'ELSE'TEXT("???
0525    TAL\);
0526    PRINT(2);
0527    STABLEAU(TAI,TA2,NI,CFIII);
0528    NUFI:=NUFI+I;
0529    TEXT("???F.I.??\);
0530    EDIT("F6.0\,NUFI);
0531    PRINT(2);
0532    SFONCT(FI2I,FI22,NI);
0533    'END';
0534    'FOR'I:=I'STEP'I'UNTIL'NI'DO'FI32[I]:=FI22[I];
0535    TESTFI(NI,FI2I,FIM2I,CFI2I,SDI,FI32,FIM22,CFI22,SD2,ER,EG,ISO,SAS);
0536    'IF'ER'THEN''BEGIN''IF'NH=0'THEN''BEGIN'TYPORESULT;
0537    TEXT("???F.I.?NON-ACCEPTABLE?G2\);
0538    PRINT(2);
0539    'GOTO'LAFIN;
```

```
0540      'END''ELSE'EG:='FALSE';
0541      'END';
0542      'IF''NOT'EG'THEN''BEGIN''IF'CRESULT'THEN''BEGIN'TEXT("???F.I.?DIFFE
0542      ENTES\);
0543      PRINT(2);
0544      PRINT(1);
0545      'END';
0546      'IF'NH=0'THEN''BEGIN'TYPORESULT;
0547      TEXT("???F.I.?DIFFERENTES\);
0548      PRINT(2);
0549      'GOTO'LAFIN;
0550      'END';
0551      CH:='FALSE';
0552      'GOTO'HYP;
0553      'END';
0554      'IF'CFII[I]<CFI2[I]'THEN''BEGIN''FOR'I:=1'STEP'1'UNTIL'NI'DO''BEGIN'FI
0554      I[I]:=FI2I[I];
0555      FII2[I]:=FI22[I];
0556      'END';
0557      CFII[I]:=CFI2[I];
0558      NFS:=1;
0559      'GOTO'REBIMETA2F;
0560      'END';
0561      'IF''NOT'SYMI'THEN''BEGIN''IF'NFS<2'THEN''BEGIN'NFS:=NFS+1;
0562      'GOTO'AUTFS;
0563      'END';
0564      'END';
0565      'IF'ISO'THEN''GOTO'SORTREG;
0566      CH:='TRUE';
0567      HYP:HYPOT(NI,FI2I,FIII,FIM2I,CFI2I,SDI,SI,FI22,FII2,FIM22,S2,NH,CH,
0567      ,FIAVHI,FIAVH2,FIAPHI,FIAPH2);
0568      'IF''NOT'CH'THEN''BEGIN''IF'CRESULT'THEN'TEXT("???PLUS?D?HYPOTHESE\
0568      ;
0569      PRINT(2);
0570      TYPORESULT;
0571      TEXT("???EPUISEMENT?DES?HYPOTHESES\);
0572      PRINT(2);
0573      'GOTO'LAFIN;
0574      'END';
0575      'IF'CRESULT'THEN''BEGIN'TEXT("???HYPOTHESE???\);
0576      EDIT("F6.0\,NH);
0577      TEXT("??:??ASSOCIATION?DE?\);
0578      EDIT("F6.0\,H[NH,1]);
0579      TEXT("???DE?G1???AVEC?\);
0580      EDIT("F6.0\,H[NH,2]);
0581      TEXT("???DE?G2???(VALEUR?FIAVH:??\);
0582      EDIT("F6.0\,H[NH,3]);
0583      TEXT("?)\);
0584      PRINT(2);
0585      PRINT(1);
0586      TEXT("???F.I.??\);
0587      NUFI:=NUFI+1;
0588      EDIT("F6.0\,NUFI);
0589      PRINT(2);
0590      SFONCT(FIII,FII2,NI);
0591      'END';
0592      'FOR'I:=1'STEP'1'UNTIL'NI'DO'FI32[I]:=FII2[I];
0593      TESTFI(NI,FIII,FIMII,CFIII,SDI,FI32,FIMI2,CFII2,SD2,ER,EG,ISO,SAS);
```

```
0594    'IF'ER'THEN''BEGIN'TYPORESULT;
0595    TEXT("???F.I.?NON-ACCEPTABLE?G2\);
0596    PRINT(2);
0597    'GOTO'LAFIN;
0598    'END';
0599    'IF''NOT'EG'THEN''BEGIN''IF'GPESULT'THEN''BEGIN'TEXT("???F.I.DIFFERE
0599    NTES\);
0600    PRINT(2);
0601    PRINT(1);
0602    'END';
0603    TYPORESULT;
0604    TEXT("???F.I.DIFFERENTES\);
0605    PRINT(2);
0606    'GOTO'LAFIN;
0607    'END';
0608    NFS:=1;
0609    'GOTO'REBIMETA2F;
0610    'COMMENT'RESULTATSISO;
0610    SORTREG:JUMP;
0611    TEXT("RESULTATS\);
0612    PRINT(6);
0613    TEXT("???ISOMORPHISME\);
0614    PRINT(2);
0615    TEXT("???CORRESPONDANCE?ENTRE?LES?SOMMETS?\);
0616    PRINT(2);
0617    SPACE(3);
0618    EDIT("F6.0\,NG1);
0619    TEXT("???G1\);
0620    SPACE(6);
0621    'FOR'I:=1'STEP'1'UNTIL'('IF'N1<=16'THEN'N1'ELSE'16)'DO'EDIT("F6.0\,I
0621    );
0622    PRINT(2);
0623    SPACE(3);
0624    EDIT("F6.0\,NG2);
0625    TEXT("???G2\);
0626    SPACE(6);
0627    'FOR'I:=1'STEP'1'UNTIL'('IF'N1<=16'THEN'N1'ELSE'16)'DO'EDIT("F6.0\,S
0627    AS[I]);
0628    PRINT(2);
0629    SPACE(6);
0630    TEXT("S.CLASSES\);
0631    SPACE(5);
0632    'FOR'I:=1'STEP'1'UNTIL'('IF'N1<=16'THEN'N1'ELSE'16)'DO'EDIT("F6.0\,S
0632    I[I]);
0633    PRINT(2);
0634    'IF'N1>16'THEN''FOR'J:=1'STEP'1'UNTIL'N1%16'DO''BEGIN'SPACE(7);
0635    TEXT("SUITE?G1\);
0636    SPACE(6);
0637    'FOR'K:=1'STEP'1'UNTIL'('IF'J<=N1%16-1'THEN'16'ELSE'N1-(N1%16)*16)'D
0637    O'EDIT("F6.0\,J*16+K);
0638    PRINT(3);
0639    SPACE(7);
0640    TEXT("SUITE?G2\);
0641    SPACE(6);
0642    'FOR'K:=1'STEP'1'UNTIL'('IF'J<=N1%16-1'THEN'16'ELSE'N1-(N1%16)*16)'D
0642    O'EDIT("F6.0\,SAS[J*16+K]);
0643    PRINT(2);
0644    SPACE(5);
```

```
0645      TEXT("SUITE?S=CL\);
0646      SPACE(6);
0647      'FOR'K:=1'STEP'1'UNTIL'('IF'J<=NI%16-1'THEN'16'ELSE'NI-(NI%16)*16)'D
0647      O'EDIT("F6.0\,S1[J*16+K]);
0648      PRINT(2);
0649      'END';
0650      'GOTO'LAFIN;
0651      'END';
0652      'END';
0653      'END';
0654      DINSUF:TEXT("PRECISER?FORME?DONNEE\);
0655      PRINT(2);
0656      LAFIN:;
0657      PRINT(3);
0658      'END'

FIN DE COMPILATION
```

```
    $CUTS:

IMPLANTATION
    'LGOL    : AD.P=10004   AD.D=10006
    EDIT     : AD.P=24313   AD.D=25057
    DATA     : AD.P=25103   AD.D=25276
    TEXT     : AD.P=25406   AD.D=25453
    PRINT    : AD.P=25456   AD.D=25543
    SPACE    : AD.P=25551   AD.D=25571
    JUMP     : AD.P=25572   AD.D=25627
    HEADIN   : AD.P=25627   AD.D=25663
    INITIA   : AD.P=25664   AD.D=25713
    ALGOL2   : AD.P=25725   AD.D=25727
    ALGOL4   : AD.P=26027   AD.D=26031
    RESTAB   : AD.P=26072   AD.D=26606
    INTA     : AD.P=26625   AD.D=26633
    ENTIER   : AD.P=26635   AD.D=26644
    SYSUL    : AD.P=26646   AD.D=27041
    EXTFLO   : AD.P=27041   AD.D=27241
    INTFLO   : AD.P=27270   AD.D=27511
    INT      : AD.P=27573   AD.D=27620
    VIDE     : AD.P=27626   AD.D=27631
 AD.F=27630   AD.L=10330
 ADFC=37774   AD.C=00000

   FIN DE CHARGEMENT
```

BLANC

ISOMORPHISME DE GRAPHES

)ONNEES

:1

GRAPHE 5 DE 6 SOMMETS ET 6 ARCS

MATRICE CARACTERISTIQUE

SOMMETS	1	2	3	4	5	6
1	0	1	0	0	0	1
2	0	0	0	0	0	0
3	0	1	0	1	0	0
4	0	0	0	0	0	0
5	0	0	0	1	0	1
6	0	0	0	0	0	0

:2

GRAPHE 6 DE 6 SOMMETS ET 6 ARCS

MATRICE CARACTERISTIQUE

SOMMETS	1	2	3	4	5	6
1	0	0	0	1	1	0
2	0	0	0	0	1	1
3	0	0	0	1	0	1

4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

RESULTATS INTERMEDIAIRES

NUMEROS DES SOMMETS	1	2	3	4	5	6
S-CLASSES						
G1	1	2	3	4	5	6
G2	1	2	3	4	5	6
TAILLES DES S-CLASSES						
G1	1	1	1	1	1	1
G2	1	1	1	1	1	1
DEMI-DEGRES INTERIEURS						
G1	0	2	0	2	0	2
G2	0	0	0	2	2	2
DEMI-DEGRES EXTERIEURS						
G1	2	0	2	0	2	0
G2	2	2	2	0	0	0
BOUCLES						
G1	0	0	0	0	0	0
G2	0	0	0	0	0	0
F.I. 1						
G1	1	2	1	2	1	2
G2	1	1	1	2	2	2

TAC

G1

1	0	0	0	0	0	0
2	2	0	2	0	2	0

G2

1	0	0	0	0	0	0
2	2	2	2	0	0	0

F.I. 2

G1	1	2	1	2	1	2
G2	1	1	1	2	2	2

TAL

G1

1	0	2	0	2	0	2
2	0	0	0	0	0	0

G2

1	0	0	0	2	2	2
2	0	0	0	0	0	0

F.I. 3

G1	1	2	1	2	1	2
G2	1	1	1	2	2	2

HYPOTHESE 1 : ASSOCIATION DE 1 DE G1 AVEC 1 DE G

F.I. 4

G1	3	2	1	2	1	2
G2	3	1	1	2	2	2

TAL

G1

	1	0	1	0	2	0	1
	2	0	0	0	0	0	0
	3	0	1	0	0	0	1
G2							
	1	0	0	0	1	1	2
	2	0	0	0	0	0	0
	3	0	0	0	1	1	0
F.I.	5						
G1		1	2	3	4	3	2
G2		1	3	3	2	2	4

TAL

G1

1	0	1	0	0	0	1
2	0	0	0	0	0	0
3	0	1	0	2	0	1
4	0	0	0	0	0	0

G2

1	0	0	0	1	1	0
2	0	0	0	0	0	0
3	0	0	0	1	1	2
4	0	0	0	0	0	0

F.I. 6

G1	1	2	3	4	3	2
G2	1	3	3	2	2	4

TAC

G1

1	0	0	0	0	0	0
2	2	0	1	0	1	0

	3	0	0	0	0	0	0
	4	0	0	1	0	1	0
G2							
	1	0	0	0	0	0	0
	2	2	1	1	0	0	0
	3	0	0	0	0	0	0
	4	0	1	1	0	0	0
F.I.	7						
G1		1	2	3	4	3	2
G2		1	3	3	2	2	4

HYPOTHESE 2 : ASSOCIATION DE 2 DE G1 AVEC 4 DE

F.I.	8						
G1		1	5	3	4	3	2
G2		1	3	3	5	2	4
TAC							
G1							
	1	0	0	0	0	0	0
	2	1	0	0	0	1	0
	3	0	0	0	0	0	0
	4	0	0	1	0	1	0
	5	1	0	1	0	0	0
G2							
	1	0	0	0	0	0	0
	2	1	1	0	0	0	0
	3	0	0	0	0	0	0
	4	0	1	1	0	0	0
	5	1	0	1	0	0	0

F.I. 9

G1	1	2	3	4	5	6
G2	1	5	3	2	6	4

TAC

G1

1	0	0	0	0	0	0
2	1	0	1	0	0	0
3	0	0	0	0	0	0
4	0	0	1	0	1	0
5	0	0	0	0	0	0
6	1	0	0	0	1	0

G2

1	0	0	0	0	0	0
2	1	0	1	0	0	0
3	0	0	0	0	0	0
4	0	1	1	0	0	0
5	0	0	0	0	0	0
6	1	1	0	0	0	0

F.I. 10

G1	1	2	3	4	5	6
G2	1	5	3	2	6	4

TAL

G1

1	0	1	0	0	0	1
2	0	0	0	0	0	0
3	0	1	0	1	0	0
4	0	0	0	0	0	0
5	0	0	0	1	0	1

	6	0	0	0	0	0	0
G2							
	1	0	0	0	1	1	0
	2	0	0	0	0	0	0
	3	0	0	0	1	0	1
	4	0	0	0	0	0	0
	5	0	0	0	0	1	1
	6	0	0	0	0	0	0

F.I. 11

G1	1	2	3	4	5	6
G2	1	5	3	2	6	4

ISOMORPHISME DE GRAPHES

RESULTATS

ISOMORPHISME

CORRESPONDANCE ENTRE LES SOMMETS

5	G1	1	2	3	4	5	6
6	G2	1	4	3	6	2	5
S.CLASSES		1	2	3	4	5	6

FIN D EXECUTION ALGO

$LANC;

GRAPHE 32 Dr 14 SOMMETS ET 42 ARCS

MATRICE CARACTERISTIQUE

SOMMETS	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	1	0	0	0	0	0	0	0	0	0	0	1	0
2	1	0	1	0	1	0	0	0	0	0	0	0	0	0
3	0	1	0	1	0	0	0	0	0	1	0	0	0	0
4	0	0	1	0	1	0	0	0	0	0	0	0	0	1
5	0	1	0	1	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	1	0	0	0	1	0	0	0
7	0	0	0	0	0	1	0	1	0	0	0	0	1	0
8	0	0	0	0	0	0	1	0	1	0	0	0	0	1
9	0	0	0	0	0	0	0	1	0	1	0	1	0	0
10	0	0	1	0	0	0	0	0	1	0	1	0	0	0
11	0	0	0	0	0	1	0	0	0	1	0	1	0	0
12	0	0	0	0	0	0	0	0	1	0	1	0	1	0
13	1	0	0	0	0	0	1	0	0	0	0	1	0	0
14	1	0	0	1	0	0	0	1	0	0	0	0	0	0

G2

G2

GRAPHE 33 DE 16 SOMMETS ET 42 ARCS

MATRICE CARACTERISTIQUE

SOMMETS	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	1	0	0	0	0	0	0	0	0	0	0	1	1
2	1	0	1	0	1	0	0	0	0	0	0	0	0	0
3	0	1	0	1	0	0	0	0	0	0	1	0	0	0
4	0	0	1	0	1	0	0	0	0	0	0	0	0	1
5	0	1	0	1	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	1	0	1	0	0	1	0	0	0	0
7	0	0	0	0	0	1	0	1	0	0	0	0	1	0
8	0	0	0	0	0	0	1	0	1	0	0	0	0	1
9	0	0	0	0	0	0	0	1	0	1	0	1	0	0
10	0	0	0	0	0	1	0	0	1	0	1	0	0	0
11	0	0	1	0	0	0	0	0	0	1	0	1	0	0
12	0	0	0	0	0	0	0	0	1	0	1	0	1	0
13	1	0	0	0	0	0	1	0	0	0	0	1	0	0
14	1	0	0	1	0	0	0	1	0	0	0	0	0	0

ISOMORPHISME DE GRAPHES

ESULTATS

ISOMORPHISME

CORRESPONDANCE ENTRE LES SOMMETS

32	G1	1	2	3	4	5	6	7	8	9	10	11	12	13
33	G2	14	4	3	2	5	6	7	13	12	11	10	9	8
S.CLASSES		1	2	3	4	5	6	7	8	9	10	11	12	13

FIN D EXECUTION ALGO

TREE JOB = 7 NB. LIGNES = 1073 NB. CARTES = 0

$JOB**FIN DE MONITEUR**;

VIII - C O N C L U S I O N

- 8 Conclusions

Cet algorithme qui résoud le problème de l'isomorphisme entre deux graphes laisse apparaître un certain nombre d'insuffisances

8.1 Methode des deux fonctions

Tout d'abord, en ce qui concerne les fonctions de structure, et la méthode des deux fonctions, nous avons vu qu'on ne pouvait pas être certain d'affiner une partition jusqu'à une des H ou H'-partitions les moins fines qu'elle contient (4.4.2.2.2.b).

Il serait intéressant de posséder une fonction de structure idéale qui donnerait systématiquement cette H ou H'-partition en itérant la méthode des deux fonctions, ou à défaut d'une seule fonction de structure idéale, un échantillon réduit de fonctions de structure à utiliser.

Une autre question qu'on peut se poser au sujet de la méthode des deux fonctions est de savoir dans quel ordre utiliser les fonctions de structures qu'on possède pour arriver le plus rapidement possible à un résultat.

Il semble actuellement qu'on ne puisse pas donner de réponses globales à ces objections, et que cela dépend des graphes traités. Mais alors, pour chaque graphe à étudier n'existe-t-il pas un type de graphes correspondants pour lesquels le groupe d'isomorphismes soit le même, et pour lesquels on puisse résoudre le problème de l'isomorphisme en employant toujours la ou les mêmes fonctions de structure ?

8.2 Methode des hypothèses

Les études faites au paragraphe 4.5.4 sur le nombre d'hypothèses, donnent des idées sur la façon dont il faut orienter les recherches pour accélérer l'algorithme.

Le problème est de mesurer le "pouvoir séparateur d'une hypothèse " c'est à dire de déterminer si cette hypothèse entraînera moins de

niveaux supérieurs d'hypothèses qu'une autre hypothèse et où la stabilisation s'obtiendra à chaque fois en le moins possible d'itérations de la méthode des deux fonctions.

Si on pousse les investigations encore plus loin on se demandera s'il n'est pas possible d'éviter la méthode des hypothèses.

Avec la méthode des deux fonctions nous avons vu qu'elle était nécessaire puisque cette méthode ne donne pas de partition plus fine qu'une H'-partition. Avec un autre algorithme on retrouvera une technique correspondant à la méthode des hypothèses qui fera intervenir un choix arbitraire, car si, dans chacun des deux graphes susceptibles d'être isomorphes, il existe des sous-graphes isomorphes entre eux, qui se correspondront dans l'isomorphisme, chacun des sous-graphes du premier graphe peut prétendre correspondre à n'importe lequel des sous-graphes du second graphe. Le choix arbitraire indique à chacun un correspondant. Il y a une relation entre ces sous-graphes et les sous-groupes du groupe d'automorphismes. Ce sont eux qui transporteront le problème dans un autre algorithme.

8.3 Existence d'isomorphisme.

Nous avons montré au cours du chapitre 3 l'inexistence actuelle de conditions nécessaires et suffisantes simples d'isomorphisme. Des recherches peuvent être faites dans ce sens.

8.4 Conclusion

L'existence de cet algorithme ne peut pas satisfaire le chercheur. La méthode théorique (par test de toutes les permutations des sommets d'un graphe) ne le satisfaisait pas à cause du temps nécessaire à son emploi même sur des graphes de peu de sommets. Cet algorithme, plus rapide, repousse à un nombre de sommets plus élevé la limitation imposée par le temps. Les machines qui ont des temps de calcul de plus en plus courts, repoussent aussi cette limite. Mais on peut construire des graphes pour lesquels malgré la taille et la vitesse des calculateurs cet algorithme et ses semblables ne suffisent pas. Il faut donc en chercher un nouveau plus performant.

IX

B I B L I O G R A P H I E

T A B L E D E S M A T I E R E S

S Y M B O L E S E M P L O Y E S

I N D E X

BIBLIOGRAPHIE

P.S. ALEXANDROFF

Introduction à la théorie des groupes
DUNOD Paris 1965.

C. BERGE

Théorie des graphes et ses applications
DUNOD Paris 1963.

C. BOHM, A. SANTOLINI

A quasi decision algorithm for the P - equivalence of two matrices.
Icc Bull., N°1, 1964, p 57 à 59.

I. BOLLIET, N. GASTINEL, P.J. LAURENT

Un nouveau langage scientifique : ALGOL, manuel pratique
HERMANN Paris 1964.

D.R. DEVEL, A. GILL

Some decision problems associated with weighted, directed graphs.
j.of SIAM (App. Math.), Vol. 14, N°5, sep. 1966, p 970 à 979.

A.J.W DUIJVESTIJN

Electronic Computation of squared rectangles (thèse)
PHILIPS Computing Centre. Eindhoven, The Netherlands 1962.

R.W FLOYD

Non-deterministic Algorithms.
j. of ACM, Vol. 14, N°4, oct. 1967, p 636 à 644.

I. GROSSMANN, W. MAGNUS

Groups and their graphs
NEW MATHEMATICAL LIBRARY, 1964.

W. LEDERMANN

Introduction to the theory of finite groups
OLIVER and BOYD London, 1957.

O. ORE

Theory of graphs
American Mathematical Society
Colloquium Publications Vol. XXXVIII, 1962.

H.J. RYSER

Combinatorial mathematics
JOHN WILEY and SONS, 1963.

S.H. UNGER

GIT. A heuristic program for testing pairs of directed line graphs for isomorphism.
Com. of ACM, Vol. 7, N°1, jan. 1964, p 26 à 34.

Table des matières

Symboles employés

Les renvois indiquent les paragraphes où les symboles sont définis ou employés pour la première fois.

A.	A	2.1.3.2.	Matrice caractéristique.
B.			
C.	C	5.2.1.	Produit scalaire inachevé d'un vecteur par une matrice de 0 et de 1.
D.	D	3.3.3.1.	Matrice. Caractéristique de type 1.
E.	E	2.1.3.1.	Tableau de la fonction extrémité.
	e	2.1.1.1.	Fonction extrémité.
	e^+, e^-	2.1.1.1.	Composant de la fonction extrémité.
F.	F.I.	4.2.3.	Abréviation de Fonction Intrinsèque.
	F.H.	4.5.1.	Défini une partition obtenue après une hypothèse.
	f	4.2.1.	Fonction intrinsèque.
	f'	4.4.2.2.1	Fonction intrinsèque obtenue par la méthode des deux fonctions.
	fa	4.4.2.2.1	Fonction demi-degré extérieur sur le graphe $G\gamma$.
G.	G	2.1.1.1	Graphe.
	G	3.3.2.	Graphe associé à une fonction intrinsèque γ.
	$\Gamma.^{\gamma}$	2.1.3.3.	Fonction successeur.
	Γ^1, Γ^{-1}	4.3.2.	Puissance de la fonction successeur.
	Γ^{-1}		Fonction prédécesseur.
	γ	4.3.1	Fonction de structure.
	γ_a'	4.4.1.1.	Fonction de structure obtenue à partir de G_γ.
H.	H	2.4.2.	Groupe des permutations associées aux automorphismes du graphe G.
	H(G)	2.4.2.	Groupe d'automorphismes de G.
	H_γ	4.3.3.3.1	Groupe des permutations associées aux automorphismes du graphe G_γ.

	H_x	2.6.1.	Classe de x pour la H - équivalence.
	H'	4.5.4.3.	Sous-groupe spécial de H.
	H'_x	4.5.4.3.	Sous-groupe spécial de H laissant x invariant.
I.	I		Permutation identité.
	I_H	2.6.4.	Sommets invariant par H.
J.	K	3.4.1.	Nombre associé à une matrice.
L.			
M.	m	2.1.1.1	Cardinal de U = nombre d'arcs.
	μ_u	2.3.2.	Correspondance biunivoque entre les arcs (pour l'isomorphisme)
	μ_x	2.3.2.	Correspondance biunivoque entre les sommets (isomorphisme)
N	NH	4.5.2.	Niveau d'hypothèse.
	n	2.1.1.1.	Cardinal de X = nombre de sommets.
	ν	2.3.1.	Permutation du groupe H.
	ν_u	2.4.1.	Permutation des arcs associée à un automorphisme.
	ν_x	2.4.1.	Permutation des sommets associée à un automorphisme.
O.			
P.	P	2.3.3.	P - équivalence de matrice. Matrice de permutation.
	ϕ	2.5.1.	Isomorphisme entre groupe H.
Q.			
R.	r	4.5.4.3.	Nombre minimum de niveaux d'hypothèses.
	r_i	4.5.4.3.	Générateur d'un groupe.
S.	S_n	2.4.2.	Groupe Symétrique des permutations de n objets.
	S_x	2.6.2.	Classe de x dans la S- équivalence.
	s	4.5.4.3.	Nombre maximum de niveaux d'hypothèses.
	σ	2.4.1.	Isomorphisme entre deux graphes.
T.	T	4.4.2.3.	Tableau dans la méthode des deux fonctions. Matrice résultat. dans le produit scalaire inachevé.
	τ_{xy}	2.6.2.	Transpositions de x et y.
U.	U	2.1.1.1.	Ensemble des arcs.
	u		Arc.

V.

W.

X. X 2.1.1.1. Ensemble des sommets.

x, y, z, t Sommets.

$X^{\otimes}$ 2.1.3.3. Monoïde commutatif engendré par X

Y.

Z.

Index

Les renvois indiquent le paragraphe où le concept est défini ou utilisé pour la première fois. Voir aussi le lexique en 2.2.