Université de Nancy 1

U.E.R. Sciences Mathématiques

LE PERT-COUT
METHODE ET PROGRAMME

THESE

pour l'obtention du

Doctorat de spécialité d'informatique

soutenue le 12 décembre 1975

par

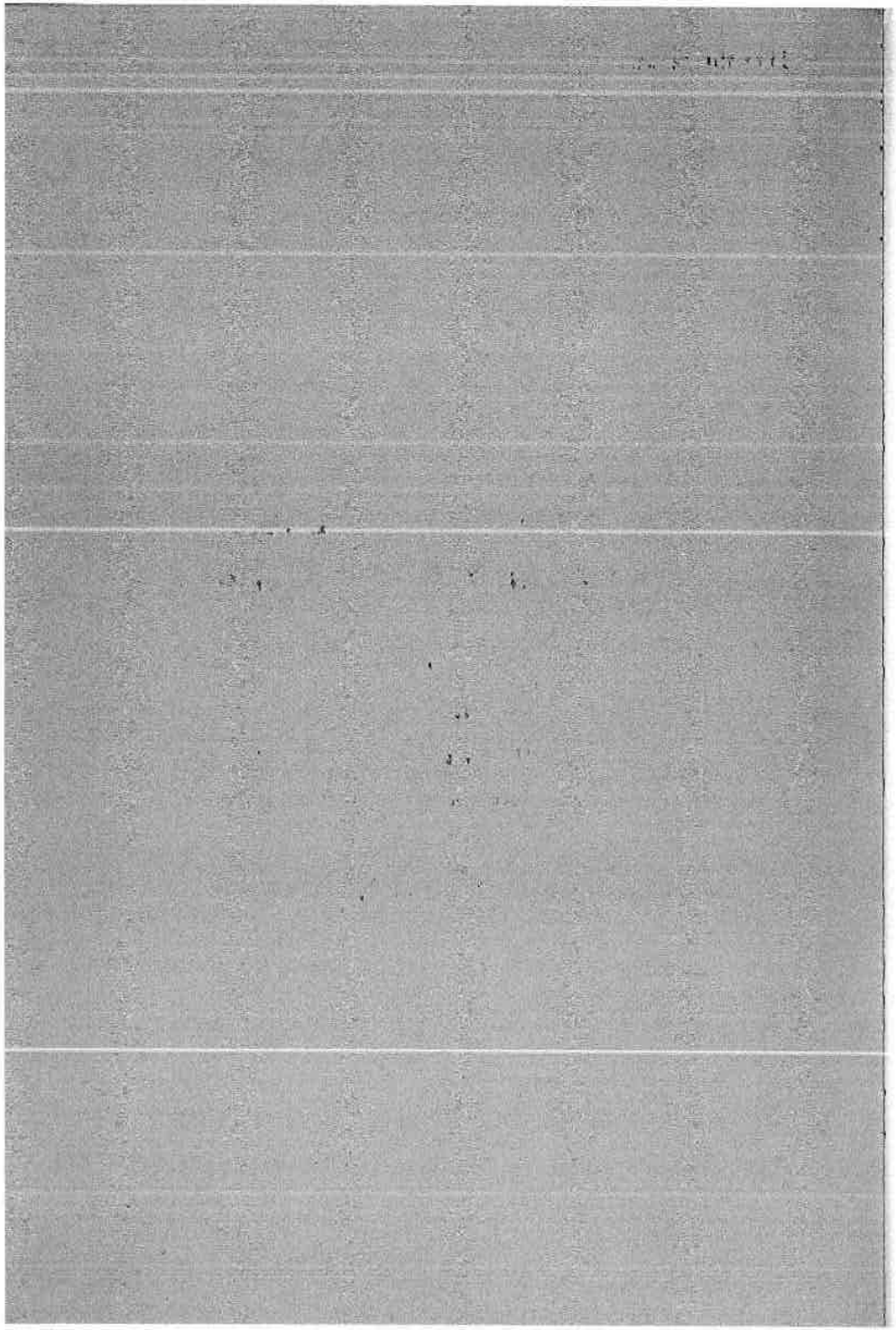
CHARLES JENNER

Membres du jury:

Président: M.J. Legras

Examinateurs: Mme C. Rolland

M.P. Boyer



Université de Nancy 1

U.E.R. Sciences Mathématiques

LE PERT-COUT
METHODE ET PROGRAMME

THESE

pour l'obtention du

Doctorat de spécialité d'informatique

soutenue le 12 décembre 1975

par

CHARLES JENNER

Membres du jury:

Président: M. J. Legras

Examinateurs: Mme C. Rolland

M. P. Boyer

NOM DE L'ETUDIANT : JENNER Charles

NATURE DE LA THESE : Doctorat de SPECIALITE en INFORMATIQUE

VU, APPROUVE

& PERMIS D'IMPRIMER

NANCY, le 3/12/1975

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

A Salomé,

Je remercie Monsieur LEGRAS d'avoir bien voulu me prendre sous sa direction ; ses critiques et ses conseils m'ont été d'une précieuse nécessité tout au long de ce travail.

Je remercie Madame ROLLAND et Monsieur BOYER de m'avoir fait l'honneur de faire partie du jury.

Que Roland STROSSER trouve ici l'expression de ma reconnaissance pour la sympathie qu'il n'a cessé de me témoigner.

Ma gratitude va également à Marie-France pour le soin qu'elle a apporté à la réalisation pratique de cet ouvrage.

P L A N

I. INTRODUCTION

Les problèmes d'ordonnancement ont fait l'objet de recherches nombreuses et diverses. Le présent exposé ne prétend pas innover dans ce domaine. Il se contente de donner une méthode de résolution du PERT-COUT, basée sur la théorie du PRIMAL-DUAL.

Dans une première partie, nous présenterons plusieurs possibilités de résolution du problème d'ordonnancement, et nous mettrons en valeur les causes qui nous ont amenés au choix de la représentation des tâches par les sommets du graphe.

Dans la deuxième partie, nous étendrons l'étude au PERT-COUT ; nous y exposerons d'abord la résolution mathématique par la méthode du PRIMAL-DUAL, pour présenter ensuite les problèmes posés par la programmation de l'algorithme proposé.

Le PERT classique est surtout un problème en soi. Mais il est aussi à la base du PERT-COUT. C'est pourquoi il serait inconcevable de vouloir traiter le PERT-COUT, sans être passé auparavant par une étude détaillée du PERT classique.

II. LE PERT CLASSIQUE

A. GENERALITES.

1. Historique : La méthode PERT est une méthode d'ordonnancement née aux Etats-Unis, pour pallier aux retards apportés aux réalisations industrielles. En hommes pratiques, les Américains inventèrent cette méthode discutable en théorie, au moins partiellement, mais grâce à laquelle ils conservèrent dans certains domaines, le premier rôle sur la scène internationale.
   Bientôt les praticiens se métamorphosèrent en spécialistes. Informatique et mathématiques devinrent de rigueur. S'il existe des organismes assez fortunés pour utiliser le PERT et ses variantes les plus perfectionnées, pour beaucoup d'autres l'emploi de méthodes permettant de réaliser dans les délais les plus brefs un ensemble complexe d'opérations de production, de recherche, administrative, financière ou commerciale, n'est pas sans importance.

2. Exemple succinct : Supposons que nous désirions remettre en état une conduite de vapeur. On peut penser qu'il s'agit d'une opération sans grande complication. Pourtant dès que l'on détaille les différentes tâches à accomplir, on constate que leur nombre est déjà grand. On devra ainsi :
   - mettre la conduite hors service,
   - placer un échafaudage,
   - enlever le revêtement,
   - examiner la conduite pour localiser la fuite,
   - commander les matériaux nécessaires à la réparation,
   - se procurer l'équipement pour le découpage et la soudure,
   - se procurer l'équipement pour la dépose du tuyau,
   - enlever le tuyau,
   - enlever les soupapes,
   - monter le nouveau tuyau,
   - ajuster les soupapes,
   - vérifier la pression du tuyau,
   - remettre le revêtement,
   - enlever l'équipement utilisé pour la dépose du tuyau,
   - laisser sècher le revêtement,
   - passer la peinture,
   - nettoyer ce qui a été sali,
   - remettre en marche l'installation,

- enlever l'échafaudage,

soit au total 19 tâches différentes.

Ces tâches ne se succèdent pas toutes.

Certaines d'entre elles, par exemple "enlever l'équipement pour la dépose du tuyau" et "laisser sècher le revêtement" peuvent être accomplies parallèlement.

Par ailleurs chaque tâche nécessite des moyens pour pouvoir être accomplie et demande un certain délai.

La remise en état de notre conduite, demandera au moins trois jours si nous n'avons pas oublié de passer à temps les commandes de matériel. Elle coûtera plusieurs dizaines de francs.

On conçoit donc que pour un projet de grande envergure - construire une usine par exemple - le nombre de tâches soit considérable, et que la supervision du projet pose des problèmes ardus.

3. Définitions : Un projet est un ensemble d'opérations qui doivent permettre d'atteindre un objectif clairement exprimable et présentant un certain caratère d'unicité. Ainsi construire une usine sidérurgique, installer un service informatique dans une entreprise, lancer une nouvelle gamme de produits de beauté, envoyer un homme sur la lune, constituent autant de projets.

Dans le cas d'un projet, un certain nombre d'opérations doivent être effectuées pour que l'objectif fixé au départ soit atteint. Nous avons ainsi dénombré 19 opé-

rations pour remettre en état une conduite de vapeur.

Nous appellerons tâches ou encore activités, ces opérations. Il ne suffit pas, bien entendu, de connaître les différentes tâches à accomplir pour réaliser un projet. Il faut encore avoir une claire connaissance de l'articulation de ces tâches entre elles, puis établir un plan complet d'action permettant de réaliser le projet dans les conditions de coût et de délais qui sont imposées.

L'ensemble des décisions que supposent ces différentes actions, constitue l'ordonnancement du projet. Nous appellerons technique d'ordonnancement, une méthode ou un ensemble de méthodes qui permettent au responsable du projet de prendre les décisions nécessaires dans les meilleures conditions possibles.

Une technique d'ordonnancement doit donc aider à :

- analyser le projet, c'est-à-dire le décomposer en tâches,
- élaborer un plan d'action permettant de réaliser le projet en respectant les diverses contraintes qui sont imposées,
- contrôler le déroulement du projet.

4. Critères d'efficacité d'une technique d'ordonnancement :

En premier lieu, un projet doit être analysé. Cette analyse recense souvent un grand nombre de tâches et de liaisons entre celles-ci. Une telle masse d'informa-

tions, pour être maniée facilement, doit être ordonnée et synthétisée. La première qualité d'une technique d'ordonnancement sera d'offrir un moyen simple de résumer complètement et clairement l'analyse du projet. L'analyse étant achevée, il faut bâtir un plan d'action et mettre en place une organisation qui permette d'atteindre l'objectif fixé dans les délais demandés et avec les moyens disponibles.

Une telle méthode sera particulièrement utile, si elle permet de détecter l'ensemble des tâches-clés qui conditionnent la réalisation du projet. Toutes les tâches ne jouent pas le même rôle, en effet, dans la réalisation du projet. Certaines peuvent être accomplies à des époques différentes, ou demander des délais supérieurs aux prévisions sans que le résultat final soit compromis. D'autres, par contre, doivent être entreprises et achevées aux dates prévues et tout retard pris dans la réalisation de ces activités se répercute sur la date finale. Il est donc intéressant que ce dernier type de tâches - nous les qualifierons de critiques - soit clairement mis en évidence.

Remarquons que le bon sens exige enfin que le coût de la mise en oeuvre d'une technique d'ordonnancement soit raisonnable, c'est-à-dire ne représente qu'une faible fraction du coût du projet lui-même.

B. MÉTHODES D'ORDONNANCEMENT.

Les nombreuses méthodes d'ordonnancement existantes peuvent se regrouper en deux grandes familles selon le principe de base qu'elles utilisent. On distingue d'une part les méthodes du type diagramme à barres ou diagramme de Gantt, et d'autre part les méthodes à chemin critique, telle la méthode PERT.

1. Le diagramme de Gantt : Cette méthode s'attache avant tout à mettre en évidence les durées. On dresse un tableau quadrillé dans lequel chaque colonne correspond à une unité de temps et chaque ligne à une tâche. Chaque tâche est représentée par une barre horizontale dont la longueur correspond à la durée de la tâche. Cette barre occupe une place correspondant à la période durant laquelle la tâche doit se dérouler.

   Considérons ainsi un projet comportant 5 tâches :

   A : 3 semaines ; B : 6 semaines ; C : 4 semaines ; D : 7 semaines ; E : 5 semaines ; telles que B et D succèdent à A, C succède à B et E à D.

   Le diagramme de Gantt d'un tel projet est représenté par la figure suivante :

TEMPS / TACHES	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A																
B																
C																
D																
E																

Comme on le voit, le principe de ce système est extrêmement simple, et la représentation en diagramme particulièrement claire. Les qualités de clarté et de simplicité de cette méthode sont telles que son utilisation est extrêmement répandue. Par contre rien ne nous permet de savoir si une erreur de forme ou de fond a été commise au niveau de l'analyse. De même les tâches critiques ne sont pas mises en évidence, et l'élaboration d'un plan d'action se révèle malaisée dans la mesure où l'on ne voit pas simplement quelles sont les conséquences des modifications apportées aux tâches non critiques.

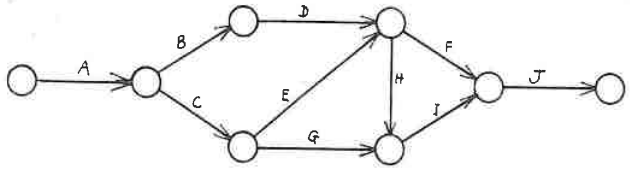
La méthode du diagramme de Gantt apparaît donc, malgré des avantages importants, comme insuffisante. On lui reproche essentiellement de ne pas faire apparaître les liaisons qui existent entre les tâches.

2. <u>La méthode PERT - Représentation des tâches par arcs</u> :

La méthode PERT s'attache essentiellement à mettre en évidence les liaisons qui existent entre les tâches. Ces dernières sont représentées par des flèches dont la longueur n'a pas de signification et n'est donc pas proportionnelle au temps.

Deux tâches qui se succèdent immédiatement sont représentées par deux flèches placées l'une à la suite de l'autre.

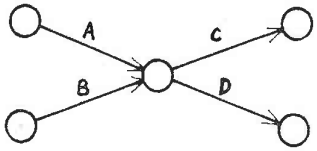
Dire que B succède immédiatement à A, signifie que la tâche B peut être entreprise sans autre condition que l'achèvement de A. Afin de bien distinguer la fin de la tâche A du début de la tâche B, un cercle est dessiné entre les deux flèches. Ce cercle concrétise une discontinuité dans le déroulement du travail, une étape. Ces étapes sont mises en évidence pour toutes les tâches : chaque tâche aura une étape "début" et une étape "fin", l'étape "fin" d'une tâche étant l'étape "début" de la tâche suivante. Les flèches seront assemblées de manière à figurer la succession des tâches qu'elles représentent. La figure suivante représente une succession de dix tâches.



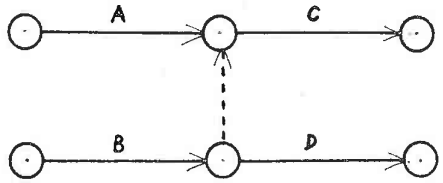
Nous désirons toutefois qu'à chaque tâche corresponde une flèche et une seule. Cette condition simple oblige à compliquer légèrement le dessin des réseaux dans certains cas.

Ainsi supposons que nous désirions représenter l'enchaînement suivant :

- D succède à B
- C succède à A et B



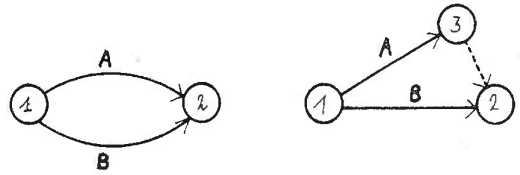
La figure ci-dessus n'est pas satisfaisante. Elle exprime bien les contraintes données, mais elle indique également que D succède à A, ce qui n'est nullement imposé. En réalité, on montre qu'il n'existe aucune façon de représenter un tel ensemble de contraintes à l'aide de flèches symbolisant des tâches réelles. Aussi utilise-t-on des flèches tracées en pointillés qui ont pour seul but de traduire une liaison logique entre deux tâches :



Sur la figure ci-dessus, la flèche en pointillés indique seulement qu'il est nécessaire pour que C soit entrepris, que B soit achevé. On peut interpréter ces flèches comme correspondant à des tâches "fictives" de durée nulle et ne mettant en jeu aucun moyen matériel et financier.

Pour lancer le projet, une décision doit être prise, et la ou les premières tâches ne peuvent commencer qu'une fois cette décision arrêtée. Le réseau de flèches part donc d'une étape et d'une seule correspondant à cette décision. De la même façon, il doit aboutir à une étape et une seule, correspondant à la constatation que le projet est achevé. Afin de faciliter la consultation du graphe, on convient généralement d'orienter toutes les flèches de la gauche vers la droite ; on peut ainsi considérer que le temps s'écoule globalement dans ce sens. Pour les mêmes raisons, on numérotera les étapes, et l'on codera les tâches bien que le numérotage des étapes suffise à lui seul pour repérer les tâches. Celles-ci sont en effet définies lorsqu'on connaît le numéro de leur étape début et de leur étape fin. Il n'y a ambiguïté que lorsque deux ou plusieurs tâches peuvent être menées exactement en parallèle. On intro-

duit alors une étape et une tâche fictive supplémentaire.



Pour que la représentation de projets par des graphes soit utilisable, il faut encore que les temps de réalisation puissent être pris en compte. Là réside en effet un des principaux intérêts de ce type de représentation qui permet un calcul simple des délais de réalisation du projet, met en évidence les tâches critiques, autorise de nombreuses modifications ultérieures et dégage les conséquences de celles-ci.

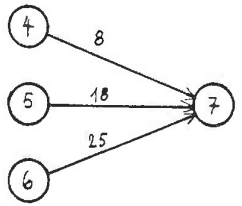
Les avantages d'une telle méthode d'ordonnancement apparaîtront alors bien supérieurs à ceux du diagramme de Gantt, malgré une certaine perte de simplicité. Mais dès à présent nous pouvons constater que le mode de représentation des projets qu'utilise la méthode PERT, met en évidence les liaisons logiques qui existent entre les tâches. Cette caractéristique constitue à elle seule un avantage considérable sur les méthodes traditionnelles.

C. LA METHODE PERT - REPRESENTATION DES TACHES PAR SOMMETS OU METHODE DES POTENTIELS TACHES.

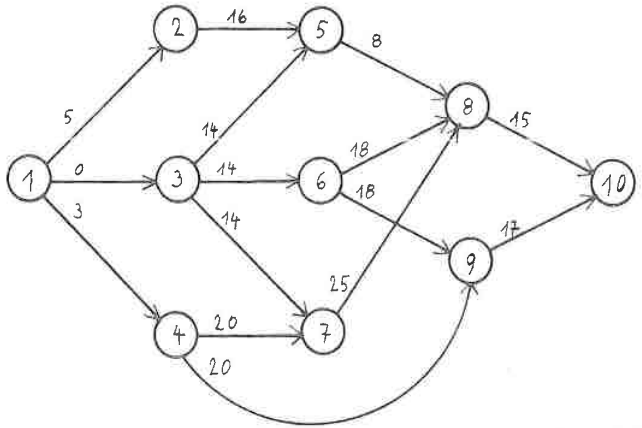
L'introduction d'étapes et de tâches fictives dans la méthode précédente nous conduit à une représentation graphique lourde. Nous développerons dans ce paragraphe une méthode qui pallie à ces difficultés.

La méthode est connue sous le nom de méthode des potentiels tâches, et les tâches y sont représentées par des sommets. Les arcs joignant les sommets matérialisent les conditions de succession et portent les durées.

Exemple : 7 (durée 15) succède à 4 (d=8), à 5 (d=18), et à 6 (d=25)



1. Représentation graphique : Considérons l'exemple représenté par le graphe suivant :



2. Contraintes : Dans le cas général, il existe des relations de succession qui se distinguent en :

- Contraintes de localisation :

$t_i \leqslant e'_i$ et surtout $t_i \geqslant e''_i$

t_i date de début de la tâche i

e'_i et e''_i dates données.

- Contraintes de succession :

$t_j \geqslant t_i + d_i$

t_i début de la tâche i

d_i durée d'exécution de la tâche i

t_j date de début de la tâche j

Sur l'exemple, ces conditions se traduisent par :

$$
\left.
\begin{array}{l}
t_2 \geqslant 5 \\
t_3 \geqslant 0 \\
t_4 \geqslant 3 \\
t_5 \geqslant t_2 + 16 \ ; \ t_5 \geqslant t_3 + 14 \\
t_6 \geqslant t_3 + 14 \ ; \\
t_7 \geqslant t_3 + 14 \ ; \ t_7 \geqslant t_4 + 20 \\
t_8 \geqslant t_5 + 8 \ ; \ t_8 \geqslant t_6 + 18 \ ; \ t_8 \geqslant t_7 + 25 \\
t_9 \geqslant t_4 + 20 \ ; \ t_9 \geqslant t_6 + 18 \\
t_{10} \geqslant t_8 + 15 \ ; \ t_{10} \geqslant t_9 + 17
\end{array}
\right\} \quad (A)
$$

3. But du PERT : Etant donné ces systèmes d'inégalités linéaires, il s'agit de définir les t_i, dates de départ de chaque tâche, de façon que la durée totale des travaux soit minimum.

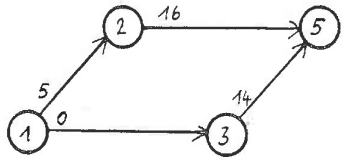
Cette durée totale s'interprète comme date de départ de la tâche "fin". C'est t_{10} dans notre exemple.
Le problème consiste donc à chercher

$$\min t_{10} \text{ sous les contraintes (A)}$$

C'est un problème de programmation linéaire, qu'il serait maladroit de traiter par le Simplex : nous allons exposer une technique qui est plus rapide et donne beaucoup plus de renseignements.

4. Méthode PERT : Exposé sur la représentation graphique :

a) Considérons le début des travaux schématisé ci-dessous :



et cherchons les conditions imposées au début de la tâche (5) :

- elle succède à (2) qui elle-même ne commence que 5 jours après l'origine ($t_2 \geqslant 5$) et qui dure 16 jours,

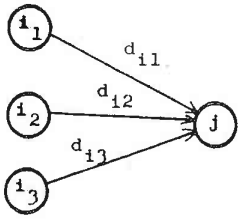
$$\Rightarrow \quad t_5 \geqslant 5 + 16 = 21$$

- elle succède à (3) commencée à l'origine et qui dure 14 jours,

$$\Rightarrow \quad t_5 \geqslant 14$$

t_5 peut donc prendre toute valeur $\geqslant$ max (21,14)

Ce raisonnement est général : soient i_1, i_2, i_3 les prédecesseurs de j, on doit avoir :



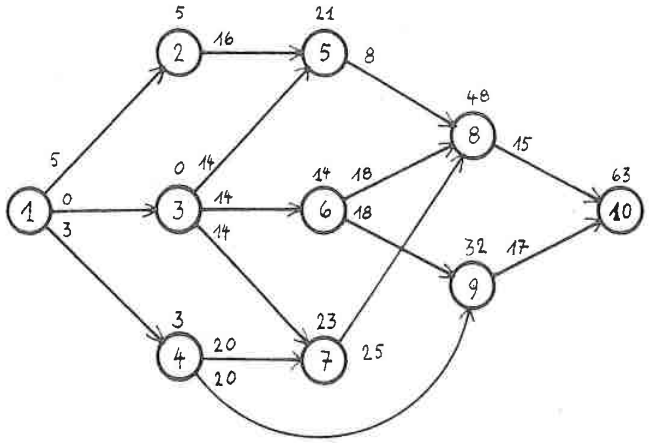
$$t_j \geqslant \begin{cases} t_{i1} + d_{i1} \\ t_{i2} + d_{i2} \\ t_{i3} + d_{i3} \end{cases}$$

donc $\boxed{t_j \geqslant \max (t_{i1} + d_{i1},\ t_{i2} + d_{i2},\ t_{i3} + d_{i3})}$

Nous voyons ainsi apparaître en chaque sommet une "date au plus tôt", date avant laquelle la tâche ne peut pas être commencée. Cette date au plus tôt pourra se calculer si nous connaissons les dates de début d'exécution des tâches prédécesseurs de la tâche étudiée.

b. Mise en oeuvre sur l'exemple - Recherche des "dates au plus tôt".

Nous reprenons le graphe et au-dessus de chaque sommet figurera la date au plus tôt calculée selon la méthode précédemment exposée.



Nous voyons que $t_{10} \geqslant 63$.

c. Prenons à présent le problème inverse. Sachant que $t_{10} = 63$, déterminé dans la première phase, peut-on trouver t_2, t_3, ..., t_9 dans ces conditions ?

Reprenons l'interprétation graphique.

Pour que $t_{10} = 63$, il faut et il suffit que

$$\begin{cases} t_8 \leqslant 63 - 15 = 48 \\ t_9 \leqslant 63 - 17 = 46 \end{cases}$$

Comme d'après le calcul précédent

$32 \leqslant t_9$ nous voyons que

$32 \leqslant t_9 \leqslant 46$

La borne supérieure des t_i (ici 46) sera appelée

"date au plus tard".

Donc $\begin{cases} 32 = \text{date au plus tôt} \\ 46 = \text{date au plus tard.} \end{cases}$

Comme $48 \leqslant t_8$ (calcul précédent) et que

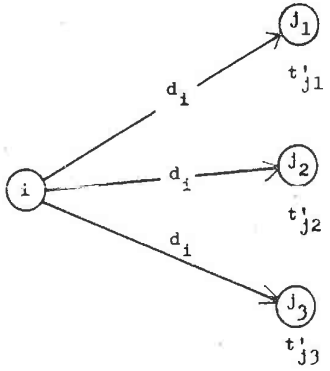
$t_8 \leqslant 48$ (dernier calcul)

$t_8 = 48$

Pour le sommet (8) la date au plus tard est égale à la date au plus tôt. Pour cette raison (8) est dit sommet ou tâche critique. Tout retard sur la date de début ou tout allongement de la durée de la tâche augmente la durée totale des travaux.

L'ensemble des tâches critiques formera le chemin critique.

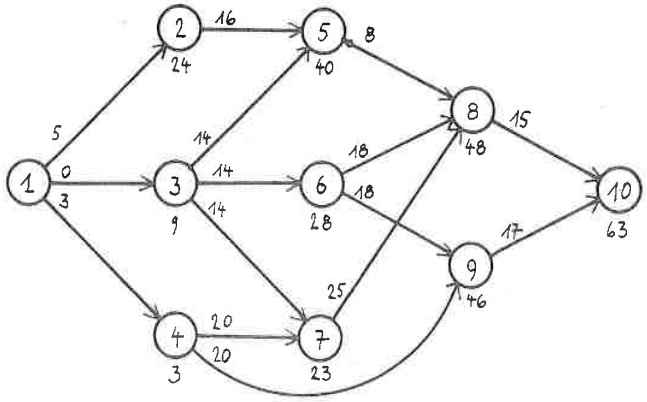
Pour calculer la date au plus tard du sommet (i), nous supposons déterminées les dates au plus tard de tous les successeurs de (i).



Nous voyons que $t'_i + d_i \leqslant \begin{cases} t'_{j1} \\ t'_{j2} \\ t'_{j3} \end{cases}$

$$t'_i = \min (t'_{j1} - d_i,\ t'_{j2} - d_i,\ t'_{j3} - d_i)$$

d. Mise en oeuvre sur l'exemple.

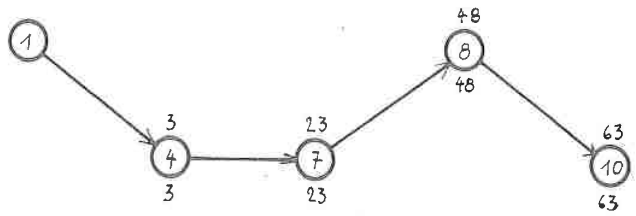


Les dates au plus tard sont calculées d'après la formule précédente et sont notées sous les sommets correspondants.

e. En regroupant les résultats des deux graphes, nous résumons dans le tableau suivant :

n° de tâche	date au plus tôt	date au plus tard	chemin critique	inégalités
2	5	24		$5 \leqslant t_2 \leqslant 24$
3	0	9		$0 \leqslant t_3 \leqslant 9$
4	3	3	oui	$t_4 = 3$
5	21	40		$21 \leqslant t_5 \leqslant 40$
6	14	28		$14 \leqslant t_6 \leqslant 28$
7	23	23	oui	$t_7 = 23$
8	48	48	oui	$t_8 = 48$
9	32	46		$32 \leqslant t_9 \leqslant 46$
10	63	63	oui	$t_{10} = 63$
				min $t_{10} = 63$

Le chemin critique, ou sous-graphe critique, est forme des arcs du graphe précédent et joigant l'origine, les sommets critiques, le sommet fin.
Ici le sous-graphe critique se réduit au chemin



f. Aspects mathématiques.

Le PERT est un problème de programmation linéaire, mais il y a une grande indétermination de la solution ; la valeur de l'optimum est évidemment unique ; certaines valeurs t_i (sommets critiques) sont uniques, mais pour tous les sommets non critiques, nous avons $\alpha j \leqslant t_j \leqslant \beta j$. Ces informations ne pourraient pas être obtenues par le Simplex, du moins sous sa forme classique.

L'algorithme PERT que nous venons de décrire, permet donc de trouver la solution d'un système de la forme :

$$\begin{cases} t_i \geqslant e_i \\ t_j \geqslant t_k + d_k \\ \min t \text{ final} \end{cases}$$

g. Aspects informatiques.

Le calcul d'une date au plus tôt demande de mettre en oeuvre l'algorithme

$$t_j \geqslant \begin{cases} t_{i1} + d_{i1} \\ \\ t_{i2} + d_{i2} \end{cases} \qquad t_j = \max\,(t_{i1}+d_{i1},\ t_{i2}+d_{i2})$$

Pour cela il faut obtenir tous les prédécesseurs i_1, i_2 .... de chaque tâche et ne calculer la date

au plus tôt d'un sommet qu'après avoir calculé la date au plus tôt de tous les prédécesseurs de ce sommet.

Le problème est analogue pour le calcul des dates au plus tard.

Nous reviendrons plus longuement sur ces questions au moment de l'informatisation du problème général.

III. LE PERT-COUT OU CALCUL DU COUT D'UN PROJET

A. ENONCE.

On considère un ensemble de tâches liées par des contraintes de succession :

$$t_i \geqslant e_i$$

$$t_j \geqslant t_i + d'_i$$

identiques à celles du PERT classique.

On suppose de plus que la durée d'_j de chaque tâche n'est pas une constante, mais une quantité sur laquelle on peut avoir une certaine action, par exemple en augmentant le personnel ou les moyens matériels mis en oeuvre.

Une telle action permet une diminution de la durée, mais

se traduit par une augmentation du coût.

Pour chaque tâche j on connaît :

- d_j : durée normale (sans augmentation de coût),
- c_j : augmentation de coût par unité de temps gagné,
- θ_j : diminution maximum de la durée.

On aura donc $d'_j = d_j - \tau_j$ avec $0 \leqslant \tau_j \leqslant \theta_j$ et l'accroissement de coût correspondant sera $c_j \tau_j$.

Le PERT-COUT est donc une extension du PERT classique dans laquelle une diminution de la durée minimum totale est possible, en compensation d'une augmentation du coût.

Remarquons cependant que cette diminution est bornée par le paramètre θ_j affecté à chaque tâche et pouvant être nul :

$$d_j - \theta_j \leqslant d'_j \leqslant d_j$$

L'idée générale du procédé est la suivante : on commence un ordonnancement en donnant à chaque tâche sa durée normale d_j. Par l'application de l'algorithme du PERT classique, on obtient alors :

- la durée minimum d'exécution Λ_0,
- la liste des tâches critiques.

On se propose de déterminer les τ_j pour obtenir une durée Λ donnée, $\leqslant \Lambda_0$, et d'organiser le travail (d'ordonnancer les tâches), pour que l'augmentation du coût correspondant à cette réduction de durée soit minimum.

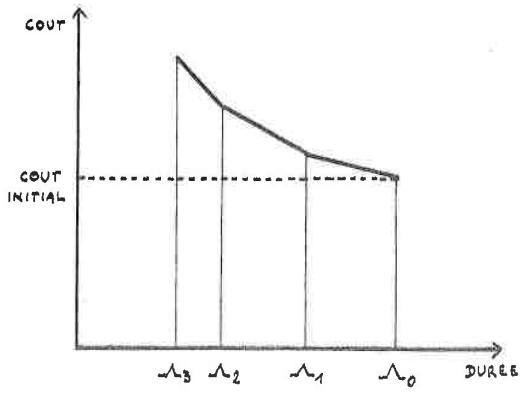
On arrive en fait à deux énoncés voisins :

+ Soit Λ donné, $\leqslant \Lambda_0$, trouver, s'il existe, un ordon-

nancement d'accroissement de coût minimum.

+ Etablir le tableau des coûts minima en fonction de λ et les ordonnancements correspondants.

Ce dernier énoncé se représente graphiquement de la façon suivante :

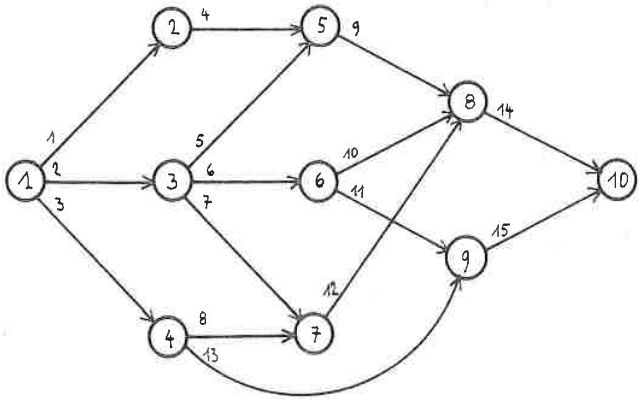


$\lambda_3 = \lambda$ minimal

Nous mettons en abscisse les durées minimales et en ordonnée les coûts correspondants, en commençant avec la durée λ_0 obtenue par l'algorithme du PERT classique et en nous arrêtant dès que nous serons arrivés à la durée λ_i minimale.

B. MATHEMATISATION DU PROBLEME.

1. Formalisation : Pour l'exposé, nous reprenons un PERT dont le graphe est un graphe déjà utilisé et sur lequel les tâches sont représentées par des sommets.



Une contrainte de succession sera :

$$t_k \geqslant t_l + d'_l$$

or $d'_l = d_l - \tau_l$

$$\Rightarrow t_k \geqslant t_l + d_l - \tau_l$$

$$\Rightarrow \boxed{t_k - t_l + \tau_l \geqslant d_l}$$

2. Résolution du problème par la méthode primal-dual, dans le cas général.

Supposons que nous ayons à traiter un problème de PERT-COUT, dont la représentation graphique présente n sommets, m arcs, dont r arcs sont issus de la source.

a. Le problème peut s'énoncer sous sa forme primale.

+ Pour les r sommets 1 liés à la source :

$t_{1} + 1 \geqslant e_{1}$ pour 1 = 1,r

+ Pour les (n-r-1) sommets k restants :

(on exclut la source)

$t_{k} - t_{j} + \tau_{j} \geqslant d_{j}$ pour k = r + 2,n

où j est l'indice d'un sommet prédecesseur du sommet k

contraintes unilatérales

+ Pour la source :

$t_{1} = 0$

+ Pour le puits :

$t_{n} = \Lambda$

contraintes bilatérales

+ Pour les diminutions de durée :

$-\tau_{i} \geqslant -\theta_{i}$ i = 2, n-1

contraintes unilatérales

$\tau_{i} \geqslant 0$ i = 2,n - 1

conditions de signe

---

et on cherche

$$\text{Min} \sum_{i=2,n-1} c_{i}\, \tau_{i}$$

b. Le dual correspondant est alors :

+ Pour les (n-2) sommets i=2,n-1 :

$$\sum_{k=1,x} f_{k} - \sum_{l=1,y} f_{l} = 0$$

avec x arcs arrivant au sommet i

et y arcs partant du sommet i

conditions bilatérales

+ Pour le sommet n :

$$\sum_{k=1,x} f_{k} - \varphi = 0$$

+ Pour les (n-2) sommets i=2,n-1 :

$$\sum_{l=1,y} f_{l} - \gamma_{i} \leqslant c_{i}$$

conditions unilatérales

+ Pour les m arcs j=1,m :

$f_{j} \geqslant 0$

+ Pour les (n-2) sommets i=2,n-1 :

$\gamma_{i} \geqslant 0$

conditions de signe

---

et on cherche :

$$\text{Max} \left[\Lambda\varphi + \sum_{k=2,n-1} d_{k} f_{k+r-1} + \sum_{l=1,r} e_{l} f_{l} - \sum_{i=2,n-1} \theta_{i} \gamma_{i}\right]$$

Dans l'énoncé du problème primal-dual interviennent les paramètres et variables suivants :

Notation	Genre	Dimension	Signification
t	variable primale	n	temps de début d'exécution pour chaque tâche
τ	variable primale	n - 2	diminutions des durées
e	paramètre	r	dates de début des tâches liées à la source
d	paramètre	n - 2	durées des tâches
Λ	paramètre	1	temps total d'exécution
θ	paramètre	n - 2	diminutions maximales des durées
c	paramètre	n - 2	augmentations unitaires de coût
f	variable duale	m	flux
φ	variable duale	1	flot sortant du puits
γ	variable duale	n - 2	flux correcteurs

c. Conditions caractéristiques d'optimalité.

Un optimum est réalisé dans les cas suivants :

+ Soit une contrainte unilatérale du primal est saturée, soit la variable associée du dual est nulle :

$$f_j\,(t_k - t_i + \tau_i - d_i) = 0 \qquad \forall_j$$

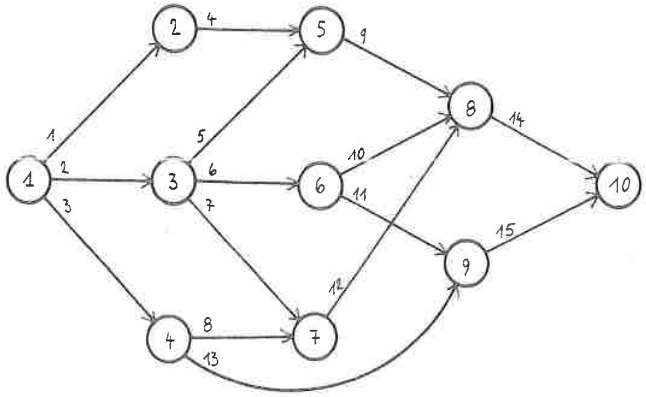
$$\gamma_i\,(\tau_i - \theta_i) = 0 \qquad \forall_i$$

+ Soit une contrainte unilatérale du dual est saturée, soit la variable correspondante du primal est nulle :

$$\tau_i\,\Big(\sum_{l=1,y} f_l - \gamma_i - c_i\Big) = 0 \qquad \forall_i$$

3. Formulation mathématique sur un exemple.

Considérons le graphe de l'exemple suivant :



Ce graphe comporte 10 sommets, la source étant notée 1 et le puits 10 ; et 15 arcs, la notation des arcs n'ayant d'importance que pour les arcs issus de la source.

a. Le primal s'écrit :

VARIABLES DUALES ASSOCIEES	CONTRAINTES	
f_1	$t_2 \geqslant e_1$	contraintes unilatérales
f_2	$t_3 \geqslant e_2$	
f_3	$t_4 \geqslant e_3$	
f_4	$t_5 - t_2 + \tau_2 \geqslant d_2$	
f_5	$t_5 - t_3 + \tau_3 \geqslant d_3$	
f_6	$t_6 - t_3 + \tau_3 \geqslant d_3$	
f_7	$t_7 - t_3 + \tau_3 \geqslant d_3$	
f_8	$t_7 - t_4 + \tau_4 \geqslant d_4$	
f_9	$t_8 - t_5 + \tau_5 \geqslant d_5$	
f_{10}	$t_8 - t_6 + \tau_6 \geqslant d_6$	
f_{11}	$t_9 - t_6 + \tau_6 \geqslant d_6$	
f_{12}	$t_8 - t_7 + \tau_7 \geqslant d_7$	
f_{13}	$t_9 - t_4 + \tau_4 \geqslant d_4$	
f_{14}	$t_{10} - t_8 + \tau_8 \geqslant d_8$	
f_{15}	$t_{10} - t_9 + \tau_9 \geqslant d_9$	
φ	$t_{10} = \Lambda$	contrainte bilatérale
γ_i	$-\tau_i \geqslant -\theta_i \quad \forall i$	contraintes unilatérales
	$\tau_i \geqslant 0$	conditions de signe

La fonction à optimiser est :

$$\text{Min} \left(\sum c_i \tau_i \right)$$

b. Le dual apparaît sous la forme :

VARIABLES PRIMALES CORRESPONDANTES	CONDITIONS	
t_2	$f_1 - f_4 = 0$	contraintes bilatérales
t_3	$f_2 - f_5 - f_6 - f_7 = 0$	
t_4	$f_3 - f_8 - f_{13} = 0$	
t_5	$f_4 + f_5 - f_9 = 0$	
t_6	$f_6 - f_{10} - f_{11} = 0$	
t_7	$f_7 + f_8 - f_{12} = 0$	
t_8	$f_9 + f_{10} + f_{12} - f_{14} = 0$	
t_9	$f_{11} + f_{13} - f_{15} = 0$	
t_{10}	$f_{14} + f_{15} - \varphi = 0$	
τ_2	$f_4 - \gamma_2 \leqslant c_2$	contraintes unilatérales
τ_3	$f_5 + f_6 + f_7 - \gamma_3 \leqslant c_3$	
τ_4	$f_8 + f_{13} - \gamma_4 \leqslant c_4$	
τ_5	$f_9 - \gamma_5 \leqslant c_5$	
τ_6	$f_{10} + f_{11} - \gamma_6 \leqslant c_6$	
τ_7	$f_{12} - \gamma_7 \leqslant c_7$	
τ_8	$f_{14} - \gamma_8 \leqslant c_8$	
τ_9	$f_{15} - \gamma_9 \leqslant c_9$	
	$f_j \geqslant 0$	conditions de signe
	$\gamma_i \geqslant 0$	

La fonction à optimiser est :

$$\text{Max} \left[-\lambda\varphi + \sum d_k f_k + \sum e_l f_l - \sum \theta_i \gamma_i \right]$$

c. Les conditions d'optimalité sont :

(1) $f_j \, (t_k - t_i + \tau_i - d_i) = 0 \qquad \forall_j$

(2) $\gamma_i \, (\tau_i - \theta_i) = 0 \qquad \forall_i$

(3) $\tau_i \, (\sum f_l - \gamma_i - c_i) = 0 \qquad \forall_i$

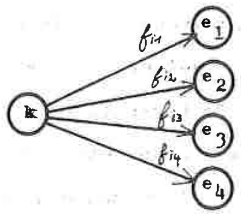
4. Introduction du graphe étendu.

Comme nous l'avons dit auparavant, traiter un PERT-COUT consiste à diminuer la durée de certaines tâches, en augmentant le coût correspondant en conséquence. Résoudre le problème au sens mathématique revient à trouver une solution quelconque. Mais le résoudre au sens de la recherche opérationnelle revient à déterminer la solution optimale. Dans notre cas celà implique que nous diminuerons les durées des tâches, mais en sorte que l'augmentation de coût correspondante soit minimale.

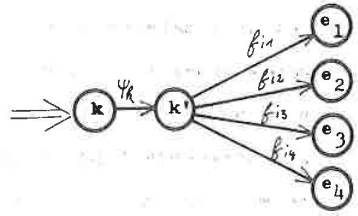
A chaque itération il faudra alors diminuer les tâches dont les augmentations de coût sont les plus petites. La détermination de ces tâches se fait dans la partie "duale" de notre méthode mathématique, par un algorithme du flot maximum. Le flot maximum détermine la coupe minimale qui elle-même contient les tâches dont les durées sont susceptibles d'être diminuées. En effet, si nous assimilons les coûts à des capacités, et si nous appliquons l'algorithme du flot maximum, sur chaque che-

min joignant la source au puits, l'arc saturé sera celui qui aura la capacité la plus petite, et qui correspondra donc à la tâche de moindre augmentation de coût.
Si nous itérons l'algorithme jusqu'à impossibilité de continuer, nous déterminerons un ensemble d'arcs, qui par définition formeront la coupe minimale.
Pour pouvoir appliquer les théorèmes généraux sur les flots dans les graphes, il est commode de modifier le graphe pour se ramener à un problème classique.
Dans sa représentation des tâches par des arcs, Ford-Fulkerson a rajouté des arcs nouveaux pour pouvoir appliquer l'algorithme du flot maximum.
Dans notre représentation des tâches par des sommets, il est intéressant de prolonger les sommets par des arcs. Ces arcs seront affectés de capacités égales aux augmentations unitaires de coût des tâches représentées par les sommets "extrémité initiale" de ces mêmes arcs.
Nous choisirons alors de modifier le graphe de la façon suivante.
A chaque sommet k de G, nous associons un sommet k' et remplaçonsles arcs issus de k par des arcs issus de k', et de mêmes extrémités finales. Nous désignerons par ψ_k le flux dans kk'.

G :



G^+ :



Nous prendrons ψ_k = somme des flux sortant de k dans G ou de k' dans G^+, graphe étendu, et prendrons dans G^+

- la capacité c_k sur tout arc (kk'),
- une capacité infinie sur tout arc issu de k'.

Il est facile de vérifier que :

- les f_i du précédent problème et les

$$\psi_k = \sum_{\substack{\text{arcs} \\ \text{sortant de k'}}} f_l$$

vérifient les conditions de conservation dans G^+;

- sur tout arc (kk')

$$\psi_k \leqslant c_k \Rightarrow \sum_{\substack{\text{arcs} \\ \text{sortants}}} f_l - \gamma_i \leqslant c_i$$

les conditions $f_l \leqslant \infty$ n'intervenant pas.

Le graphe étendu G^+ nous met donc en présence de deux sortes d'arcs :

- Les arcs (k', e_i) qui étaient les arcs (k, e_i)

dans l'ancien graphe G.

Ces arcs matérialisent les liaisons des différents sommets et traduisent fidèlement par rapport au graphe G les successions des tâches. Ils n'interviennent pas quantitativement dans la détermination du flot maximum ; c'est pourquoi on leur associe une capacité infinie.

- Les arcs (k,k') qui sont des arcs supplémentaires, dédoublant les sommets. On leur affecte une capacité égale à l'augmentation unitaire de coût des sommets-tâches dont ils sont issus. Ces arcs-là, contrairement aux précédents, déterminent la valeur du flot maximum.

5. Notion de "tâche bloquée" ou "sommet bloqué".

Une tâche d'indice k, ou le sommet représentatif sera dit "bloqué", si pour cet indice on a $\tau_k = \theta_k$. La durée de la tâche ne peut plus être diminuée.

Deux cas sont possibles :

a. $\theta_k = 0$, $\tau_k = 0$

Les conditions d'optimalité (2) et (3) sont alors vérifiées. Il vient

$$\gamma_k > 0$$
$$\Sigma f_l - \gamma_k - c_k < 0$$

Quels que soient f_l, c_k, on peut trouver $\gamma_k > 0$ qui vérifient ces conditions. En fait cela exprime qu'il n'y a aucune contrainte sur le flux sortant du sommet k. On prendra donc $c_k = \infty$ sur l'arc (k,k') du graphe étendu G^+.

La seule condition d'optimalité qui reste est, puisque $\tau_k = 0$

$$f_j (t_l - t_k - d_k) = 0$$

+ ou $t_l - t_k - d_k = 0$

l'arc est admissible et le flux quelconque peut être différent de 0

+ ou $t_l - t_k - d_k \neq 0$

l'arc n'est pas admissible, mais la variable duale $f_j = 0$.

b. $\theta_k \neq 0$, tel que $\tau_k = \theta_k$

La condition d'optimalité (2) est vérifiée et γ_k est choisi arbitrairement $\geqslant 0$.

La condition du dual associée au τ_l donne

$$\tau_k (\Sigma f_l - c_k - \gamma_k) = 0$$

or $\tau_k \neq 0$

$$\Rightarrow \Sigma f_l = c_k + \gamma_k$$

$$\Rightarrow \gamma_k = \Sigma f_l - c_k \geqslant 0$$

Σf_l peut être arbitrairement grand ; on choisit $c_k = \infty$ sur (k,k').

En résumé, pour une tâche bloquée, il ne reste aucune

limitation supérieure de $\sum f_l$. On associera donc une capacité infinie à l'arc (k,k') correspondant du graphe étendu. A une tâche non bloquée, on associera à l'arc (k,k') la capacité c_k.

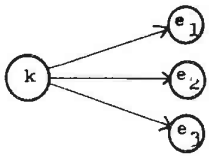
6. Coupe dans le graphe étendu Γ^+

Soit Γ_n (n=indice d'itération) un sous-graphe critique de G. Tout sommet de Γ_n est critique, et tout arc de Γ_n est admissible, car $t_k - t_l + \tau_l - d_l = 0$.

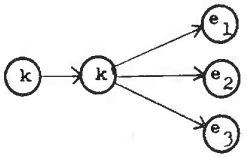
Soit Γ_n^+ le sous-graphe étendu associé, avec les capacités suivantes :

- sur les arcs (k,k')
  - arcs non bloqués : capacité = c_k
  - arcs bloqués : capacité = ∞
- sur les arcs (k',l) capacité = ∞

Γ_n :



Γ_n^+ :



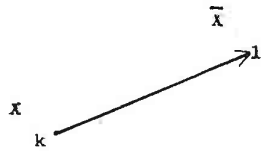
capacité = ∞ pour une tâche bloquée

c_k pour une tâche non bloquée

Cherchons alors par la méthode classique le flot maximal dans Γ_n^+ et mettons en évidence la coupe $X\bar{X}$ correspondante.

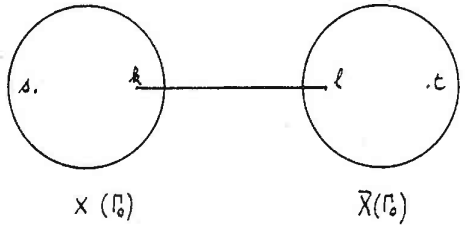
Deux cas sont possibles :

a. Cette coupe existe : Sa valeur est nécessairement finie. Montrons en effet, par l'absurde, que tout arc de $X\bar{X}$ est nécessairement un arc de capacité finie.



Soit (k, l) un tel arc. k étant marqué, supposons que la capacité de (k, l) soit infinie. L'arc n'est alors pas saturé, et on arrivera à marquer l, ce qui est contraire à l'hypothèse puisque $l \in \bar{X}$.
A tout arc de la coupe $X\bar{X}$, qui est un arc (k, k'), correspond une tâche non bloquée puisque l'arc (k,k') a une capacité finie ; et l'augmentation unitaire de coût est c_k = capacité de (k, k').

Définissons une coupe .
Tout sommet k de X est tel qu'il existe au moins un chemin de Γ_0 joignant s à k. Ce chemin sera formé d'arcs admissibles et le calcul du temps t_k se fera d'une façon bien définie. Tout sommet l de X est tel qu'il existe au moins un chemin de Γ_0 joignant l au puits t. Ce chemin sera aussi formé d'arcs admissibles et le calcul de t_l se déduit de façon bien définie de Λ = t puits.
Nous nous trouvons donc dans la situation suivante :



Tout arc (k,l) est un arc issu d'un sommet non bloqué. Ce sommet correspond à une tâche dont la durée peut être diminuée. Il est clair que pour que Λ puisse être diminué, il est suffisant que nous puissions trouver une coupe $X\bar{X}$ de Γ_0 dont tous les arcs ont chacun pour origine une tâche dont la durée peut être diminuée.
La diminution de Λ de une unité nécessite donc la diminution de chaque tâche origine d'un arc de Γ de une unité. Le coût de cette opération sera égal à la somme des augmentations unitaires des coûts relatifs à toutes les tâches origine d'un arc de la coupe. Il est encore égal à la somme des capacités des arcs (k, k') de la coupe de Γ^+.
Le fait de trouver la coupe de Γ^+ de capacité minimale revient à chercher dans Γ^+ les sommets tels que :
- tous les arcs de Γ ayant cette origine fassent partie de $X\bar{X}$,

- l'augmentation unitaire de coût total soit minimale.

b. Cas où la coupe minimale finie n'existe pas.

On sait qu'un flot quelconque est toujours inférieur ou égal à une capacité d'une coupe quelconque. Donc l'existence d'une capacité finie entraîne un flot minimal fini.

Prenons un autre procédé que le précédent pour définir une coupe dans Γ^+.

On définit un ensemble X qui comprend :

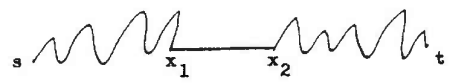
- la source
- tout sommet x tel qu'il existe un chemin reliant la source s à x, et sur lequel tous les arcs ont des capacités infinies.

L'ensemble $\bar{X}$ contiendra tous les autres sommets.

Il y a alors deux cas possibles :

- $t \in X \Rightarrow$ il existe un chemin de s à t dont tous les arcs sont de capacité infinie. Il y passe par conséquent un flot infini.
- $t \in \bar{X}$ Par définition tous les arcs $x\bar{x}$ sont nécessairement de capacité finie.

Soient $x_1 \in X$ et $x_2 \in \bar{X}$



Si la capacité de (x_1, x_2) est infinie, s peut être relié à x_2 et $x_2 \in x$, ce qui est contraire à l'hypothèse.

La capacité de la coupe est donc finie.

On se trouve donc toujours dans l'une des deux situations suivantes :

- soit le flot maximal et la capacité minimale sont tous les deux finis,
- soit le flot maximal et toutes les capacités sont infinis.

Dans le cas b. il existe au moins un chemin de s à t où tous les arcs ont une capacité infinie. Dans Γ toutes les tâches représentées par les sommets de ce chemin sont bloquées.

Comme Λ est supérieur ou égal à la somme des durées des tâches de tout chemin reliant s à t, l'existence d'un chemin de s à t dont toutes les tâches sont bloquées, entraîne que Λ ne peut plus être diminué. C'est de cette façon qu'on aboutit à Λ minimum.

7. Recherche de la solution.

a. Idée générale : Par des traitements alternés du primal et du dual, il s'agit de déterminer les variables primales et duales telles qu'elles vérifient :

- toutes les conditions du primal sauf $t_{final} = \Lambda$
- toutes les conditions du dual,
- les 3 groupes de conditions d'optimalité.

Quand nous obtenons une durée d'exécution égale à Λ nous avons l'ordonnancement de coût minimum pour cette durée et le problème a une solution, tant que Λ sera supérieur à Λ minimum.

b. Marche à suivre :

- On initialise les τ_i à 0 $\forall_i$. Dans le primal, l'application d'un PERT classique nous détermine les sommets critiques, le sous-graphe Γ_1 qui sera formé des arcs de G joignant les sommets critiques. Sur chacun de ces arcs i on aura :

$$t_k - t_l = d_l$$

$$\Rightarrow f_i \neq 0$$

A Γ_1 nous associons Γ_1^+ étendu

avec sur (k, k') | c = infini pour toutes les tâches où $\theta_k = 0$
| $c = c_k$ pour les autres.

- Dans Γ_n^+ étendu, nous cherchons le flot minimal

et surtout la coupe $X\bar{X}$. Si elle n'existe pas, nous arrivons à la fin du problème : la valeur Λ est la plus petite accessible.

Si elle existe elle définit des tâches origine d'arcs de $X\bar{X}$ de durées susceptibles d'être diminuées. On augmente alors les τ_i des sommets concernés et on diminue les durées correspondantes.

La quantité ε dont on diminue les durées des tâches de la coupe est déterminée en fonction de deux critères :

- Premièrement cette quantité doit être inférieure ou égale à toutes les diminutions $\theta_i - \tau_i$, encore possibles des tâches de la coupe.

$$\Rightarrow \varepsilon \leqslant \varepsilon_1 = \min_{\forall i \in \text{coupe}} (\theta_i - \tau_i)$$

Ce critère est d'ailleurs conforme aux contraintes unilatérales du problème primal.

- Deuxièmement nous choisirons la diminution inférieure ou égale à tous les battements du graphe.

$$\Rightarrow \varepsilon \leqslant \varepsilon_2 = \min_{\forall j \in \text{graphe}} (tl_j - t_j)$$

Cette restriction n'est aucunement impo-

sée par la théorie primal-dual même ; elle est purement volontaire et nous évitera dans la résolution sur ordinateur des itérations supplémentaires.

Nous diminuerons donc de $\varepsilon = \min(\varepsilon_1, \varepsilon_2)$.

Pour mieux comprendre l'intérêt de cette deuxième restriction, analysons en détail le déroulement de l'algorithme à partir du moment où la diminution a été faite.

Les durées des tâches de la coupe ayant été diminuées de ε, nous déterminons les nouvelles dates au plus tôt et au plus tard pour chaque sommet. Nous déterminons ainsi le nouveau sous-graphe critique. A ce niveau deux situations peuvent se présenter :

+ Supposons que nous soyons arrivés à la $i^{éme}$ itération. Si le sous-graphe critique de la $i^{ème}$ itération est formé du sous-graphe critique de la $(i-1)^{ème}$ et de certains nouveaux arcs qui sont devenus critiques dans cette dernière itération, les diminutions des durées ont été faites correctement. Nous disposons ainsi d'un nouveau sous-graphe critique auquel nous appliquons une nouvelle fois l'algorithme du flot maximum.
+ Supposons par contre qu'après les diminutions de la $i^{ème}$ itération, le nouveau sous-graphe critique ne contienne plus entièrement le sous-graphe critique de la $(i-1)^{ème}$ itération. Ce phénomène apparaît fréquemment et s'explique ainsi :

supposons qu'il y ait k façons différentes de joindre la source au puits par des chemins critiques. Rappelons que dans la $(i-1)^{ème}$ itération, les durées de la source au puits étaient toutes identiques, quel que soit le chemin critique choisi.

Sur chacun des k chemins existe au moins, par définition un arc appartenant à la coupe. Mais sur certains de ces k chemins, il existe plusieurs arcs appartenant à la coupe. Puisque nous avons diminué la durée de chaque tâche de la coupe d'une même quantité, les durées de ces k chemins ne sont pas égales, ce qui est contraire à notre hypothèse. En effet, quel que soit le chemin critique suivi pour joindre la source au puits, la durée totale doit être la même, par définition même du chemin critique.

Dans une partie du programme que nous appelons "réajustement du chemin critique", ces différentes durées sont rendues égales en rendant critiques les arcs qui ne l'étaient plus ; parallèlement il faut rallonger les durées et diminutions de durée des arcs correspondants, et rediminuer l'augmentation de coût. Alors seulement nous appliquerons à nouveau l'algorithme du flot maximum.

Notons que des exemples traités prouvent que le

"réajustement" est moins fréquent lorsque l'on impose $\varepsilon \leqslant \varepsilon_2$. Il sera en oeuvre sur les graphes des pages 59, 61, 63 et 65.

- Le procédé est arrêté :

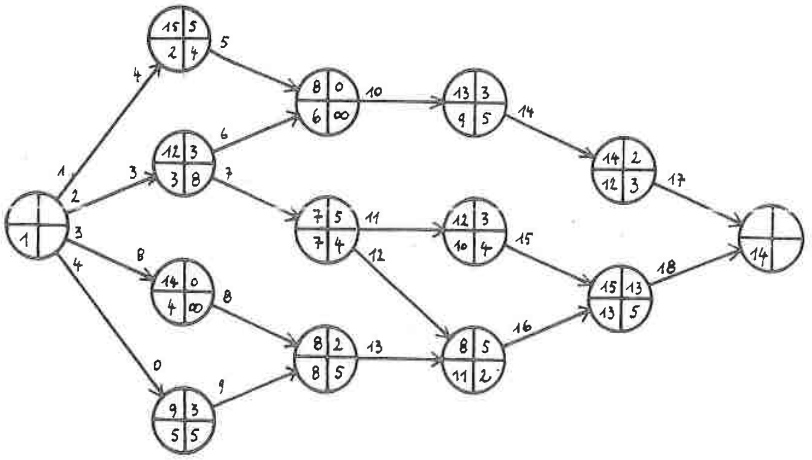
  soit lorsqu'on atteint, et en général dépasse λ donné ;

  soit lorsqu'on arrive à λ minimum du fait qu'on ne trouve plus la coupe $X\bar{X}$.

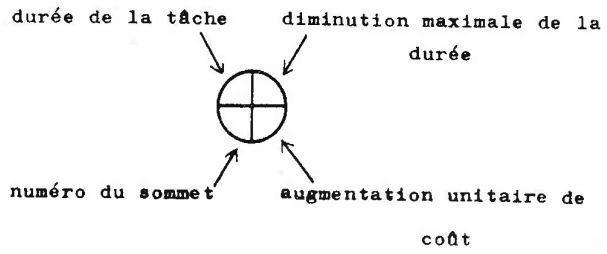
C. DEROULEMENT D'UN EXEMPLE ET EVOLUTION CORRESPONDANTE DU PRIMAL-DUAL.

1. Déroulement graphique : Nous allons résoudre un exemple à la main, en suivant graphiquement l'évolution de la résolution.

   Considérons l'exemple dont la représentation graphique est la suivante :



Ce graphe à 14 (N) sommets et 18 (M) arcs, dont 4 (R) sont issus de la source, est construit ainsi :
chaque sommet représente une tâche et contient les renseignements suivants :



Les arcs portent leur numéro d'identification et définissent les liaisons entre les différentes tâches. Les arcs issus de la source portent en outre les durées entre les tâches succèdant directement à la tâche origine et l'instant 0.

a. Présentation du graphe étendu :

Le graphe étendu G^+ comporte donc :

$$2N - 2 = 26 \text{ sommets}$$

$$M + N - 2 = 30 \text{ arcs}$$

$$R = 4 \text{ arcs issus de la source.}$$

La technique de construction du graphe étendu a été décrite antérieurement. Pourtant nous devons apporter quelques explications supplémentaires, en ce qui

concerne les numérotations :

- les sommets 1 à N et les arcs 1 à M gardent leurs numéros de G dans G^+,
- les sommets et arcs rajoutés sont numérotés respectivement de N + 1 à 2N - 2 et de M + 1 à M + N - 2.

Le graphe étendu se présente de la façon suivante :

+ Les (2N - 2) sommets peuvent être classés en 3 catégories :
  - le sommet initial 1 et le sommet final N qui représentent l'"instant début" et l'"instant fin" des opérations.
  - Les (N - 2) sommets 2, 3, ... N - 1 représentant les différentes tâches et contenant de gauche à droite et de haut en bas les renseignements suivants : . durée normale de la tâche,
    . diminution maximum de la durée,
    . numéro de la tâche,
    . augmentation unitaire de coût.
  - Les (N - 2) sommets supplémentaires N + 1,..., 2N - 2 créés en même temps que les arcs supplémentaires et assurant les liaisons des différentes tâches. Ils ne contiennent que leur numéro.
+ Les (M + N - 2) arcs apparaissent également sous trois formes et portent tous à l'extrémité initiale leur numéro.

  - Les R arcs ayant pour sommet initial la source : ceux-ci sont affectés, de plus, de la durée entre l'instant initial et le début des tâches dont ils joignent les sommets représentatifs. Leur capacité est infinie.
  - Les (M - R) arcs de liaison R + 1, ..., M ayant chacun comme extrémité initiale un sommet supplémentaire, et comme extrémité finale un sommet du graphe initial. Leur capacité est également infinie.
  - Les (N - 2) arcs M + 1, ... M + N - 2 ayant comme extrémité finale un sommet supplémentaire et comme extrémité initiale un sommet du graphe de départ. L'augmentation unitaire de coût des tâches représentées par ces derniers sommets, devient la capacité de ces arcs.

b. Résolution de l'exemple : Au graphe ainsi construit, nous appliquons notre algorithme du PERT-COUT. Remarquons que les dates au plus tôt et au plus tard figureront au-dessus des sommets représentatifs des tâches.

Commenter l'évolution intégrale de l'exemple serait trop long. C'est pourquoi nous limiterons les explications à la quatrième itération ; elle correspond aux opérations qui apparaîtront sur la page 58.

A ce niveau le graphe se présente ainsi :

- les durées des sommets 2, 11 et 12 ont déjà été diminuées au cours des itérations précédentes. Pour le sommet 12 par exemple, la durée de la tâche est passée de 14 à 12, la diminution maximale de 2 à 0. La durée de cette tâche ne pouvant plus être réduite, la capacité de l'arc 29 devient infinie.

+ Au début de cette itération, l'application du PERT classique nous donne :
  - le calendrier des dates au plus tôt et au plus tard ;
  - les arcs critiques qui sont matérialisés par des traits épais et qui forment le sous-graphe critique.

+ L'algorithme du flot maximum appliqué au sous-graphe critique détermine la coupe minimale. Ce sont les arcs 19, 24 et 30.

+ Le minimum des battements étant 13 (sommet 5) et le minimum dont on peut encore diminuer la durée étant 2 (sommet 2), nous diminuons les durées et les diminutions maximales de durée des sommets-tâches 2, 7 et 13 de la quantité 2. Nous passons alors à la page 59 en diminuant les durées.

Pour les sommets 2, 7 et 13, on trouve alors :

- des durées de 10, 5, 13
- des diminutions maximales de 0, 3, 11

$\Rightarrow$ une capacité infinie pour l'arc 19.

L'augmentation de coût passe de 26 à

$$26 + 2\ (4 + 4 + 5) = 52$$

+ Une nouvelle application du PERT classique nous donne :
  - des nouvelles dates au plus tôt et au plus tard,
  - une nouvelle durée totale qui est passée de 49 à 48,
  - un nouveau sous-graphe critique.

Mais ce nouveau sous-graphe critique ne contient plus le précédent.

En effet les arcs 1, 3, 19, 21, 5, 7, 8, 24, 25, 11, 13, 27, 28, 15, 16, 30 et 18, ne sont plus critiques, alors qu'ils l'étaient auparavant.

+ Il s'agit donc de réajuster certains sommets en réaugmentant les durées.

En réajustant le sommet 13, la durée de la tâche 13 passe de 13 à 14 et la diminution maximale de la durée de 11 à 12.

L'augmentation de coût passe de 52 à 52 - 5 = 47.

En réajustant le sommet 7, la durée de la tâche 7 devient égale à 7 et la diminution maximale de

la durée à 5.

L'augmentation de coût passe de 47 à

$$47 - (2 \times 4) = 39.$$

En réajustant le sommet 2, la durée de la tâche 2 est réaugmentée à 11 et la diminution maximale de la durée à 1. Puisque celle-ci est différente de 0, la capacité de l'arc 19 n'est plus infinie et reprend sa valeur initiale 4. Quant à l'augmentation de coût, elle passe de 39 à

$$39 - 4 = 35.$$

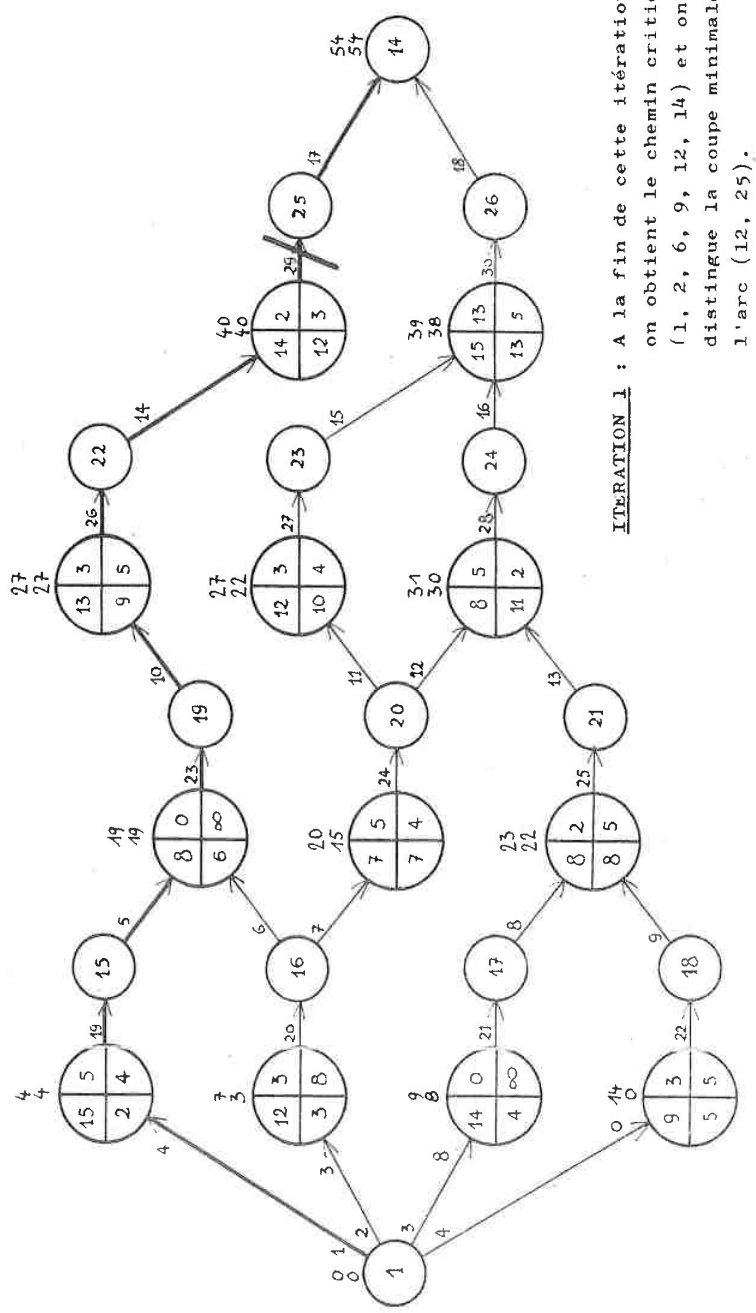
Nous nous rendons compte que les arcs qui doivent l'être, sont à nouveau critiques ; de plus l'arc 6 devient critique. On passe alors à l'itération suivante.

En conclusion, nous constatons que dans cette itération la durée totale passe de 49 à 48, mais qu'en fait seules les tâches 2 et 13 ont subit une diminution de durée de 1 chacune, tandis que l'augmentation de coût passe de 26 à

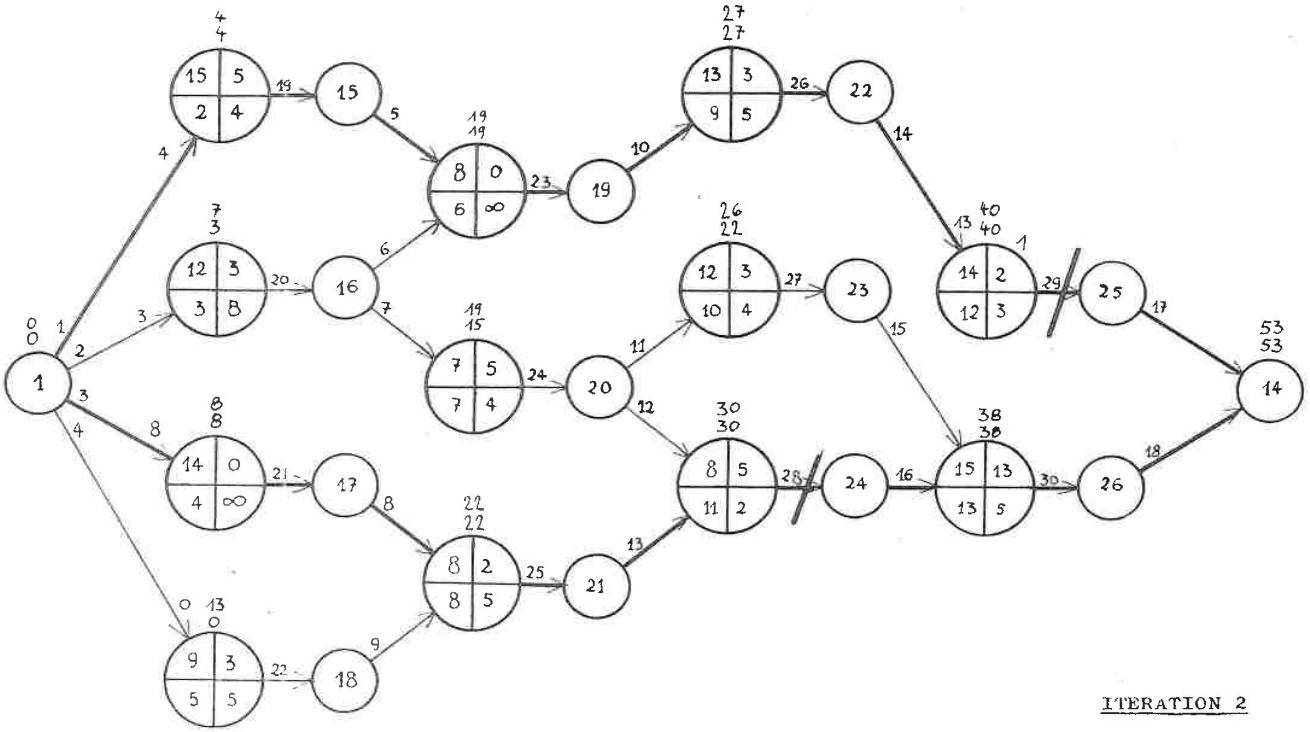
$$26 + (1 \times 4 + 1 \times 5) = 35.$$

De cet exemple, nous devons retenir principalement que les tâches dont on diminue les durées sont bien des tâches-arcs de la coupe, mais pas catégoriquement toutes les tâches-arcs de la coupe.

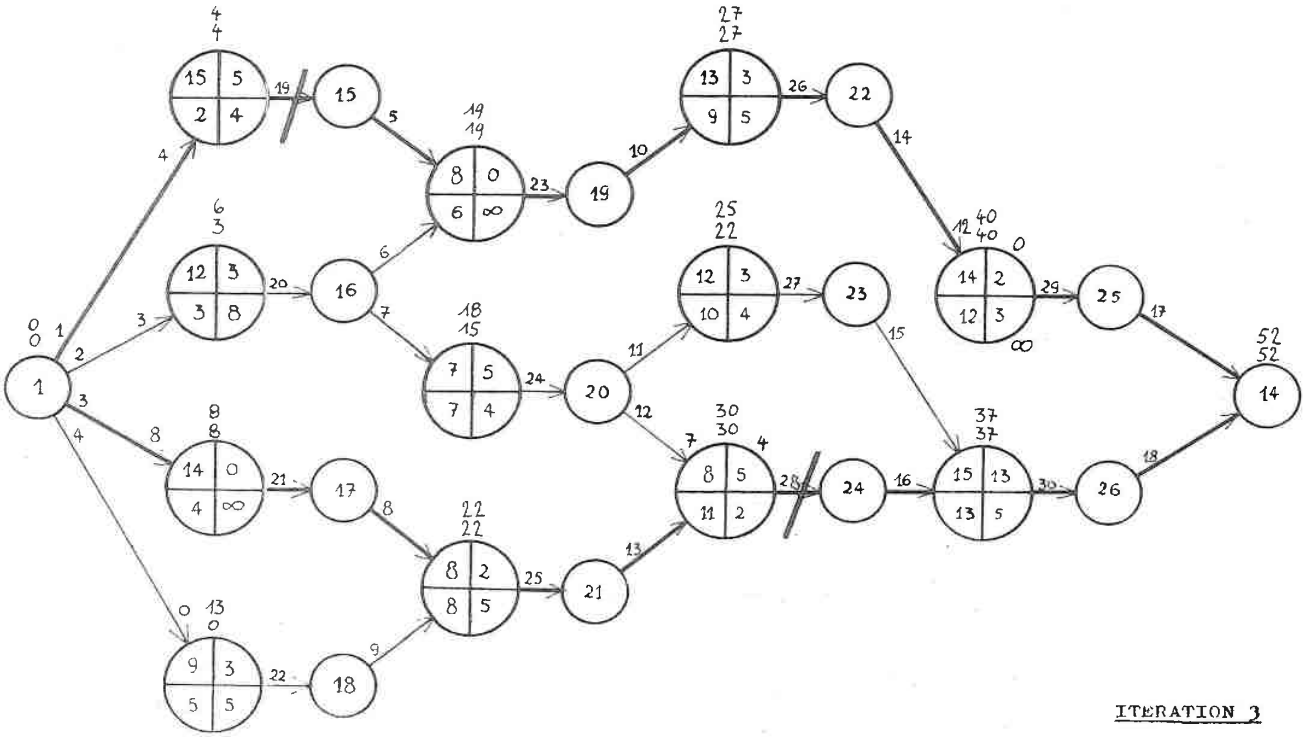
e. Résultats graphiques.



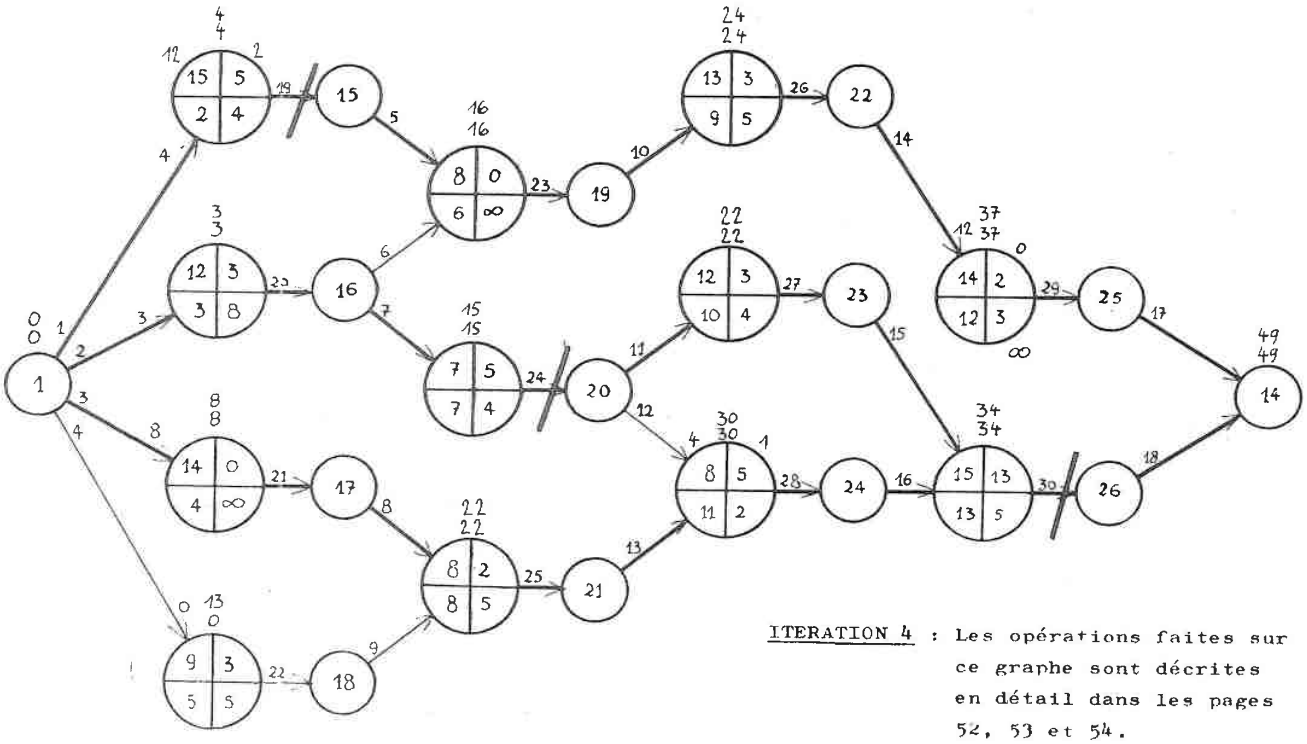
ITERATION 1 : A la fin de cette itération, on obtient le chemin critique (1, 2, 6, 9, 12, 14) et on distingue la coupe minimale, l'arc (12, 25).



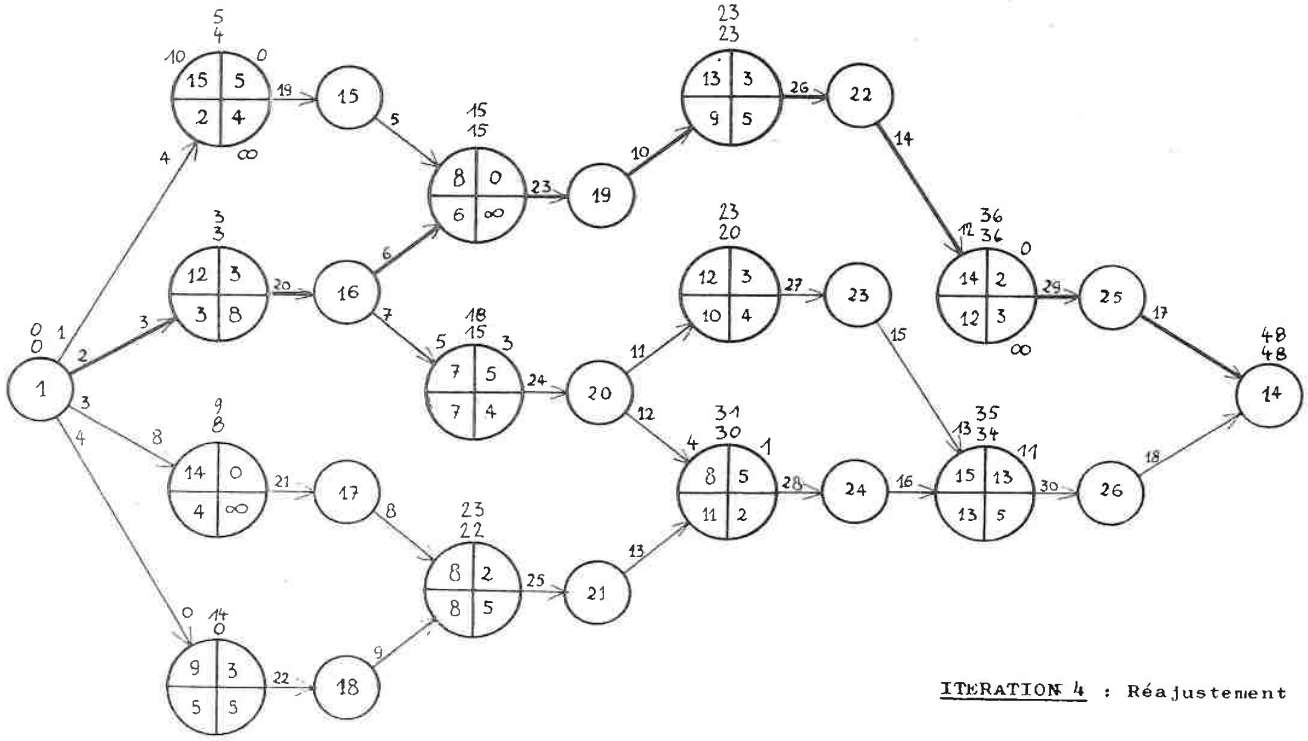
ITERATION 2



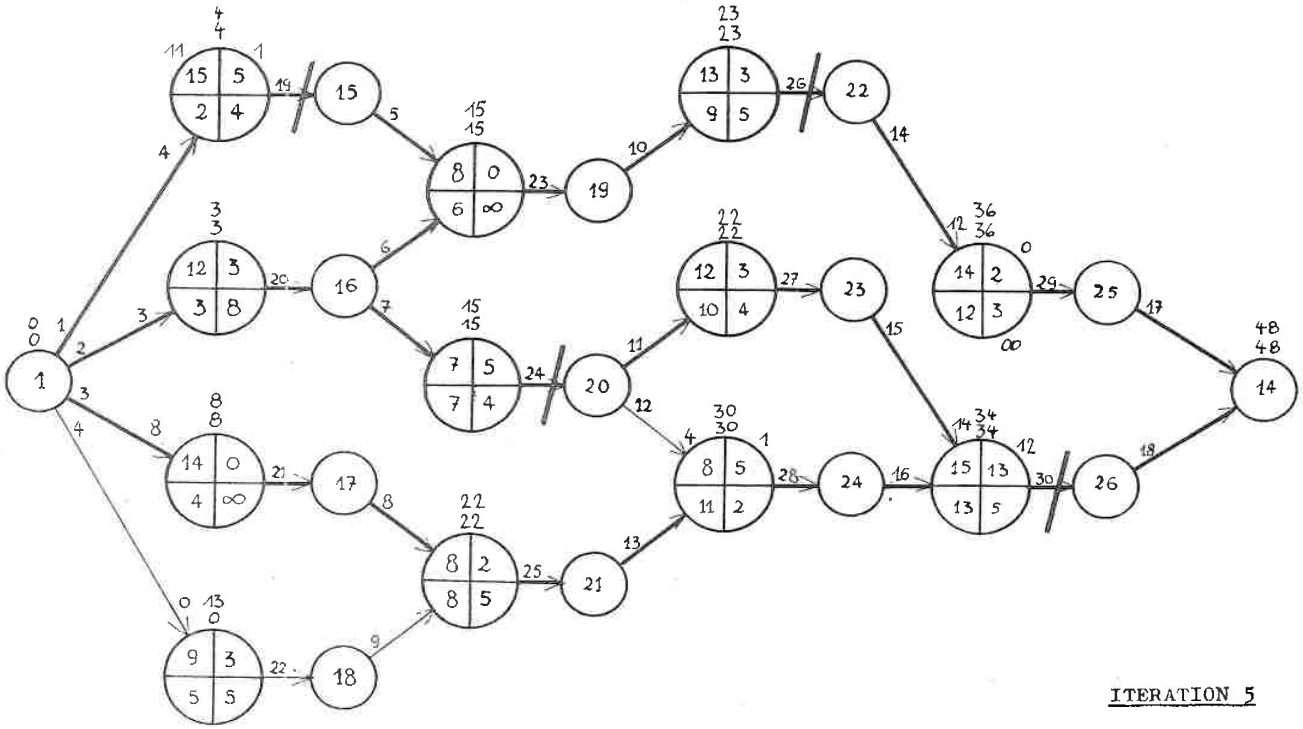
ITERATION 3



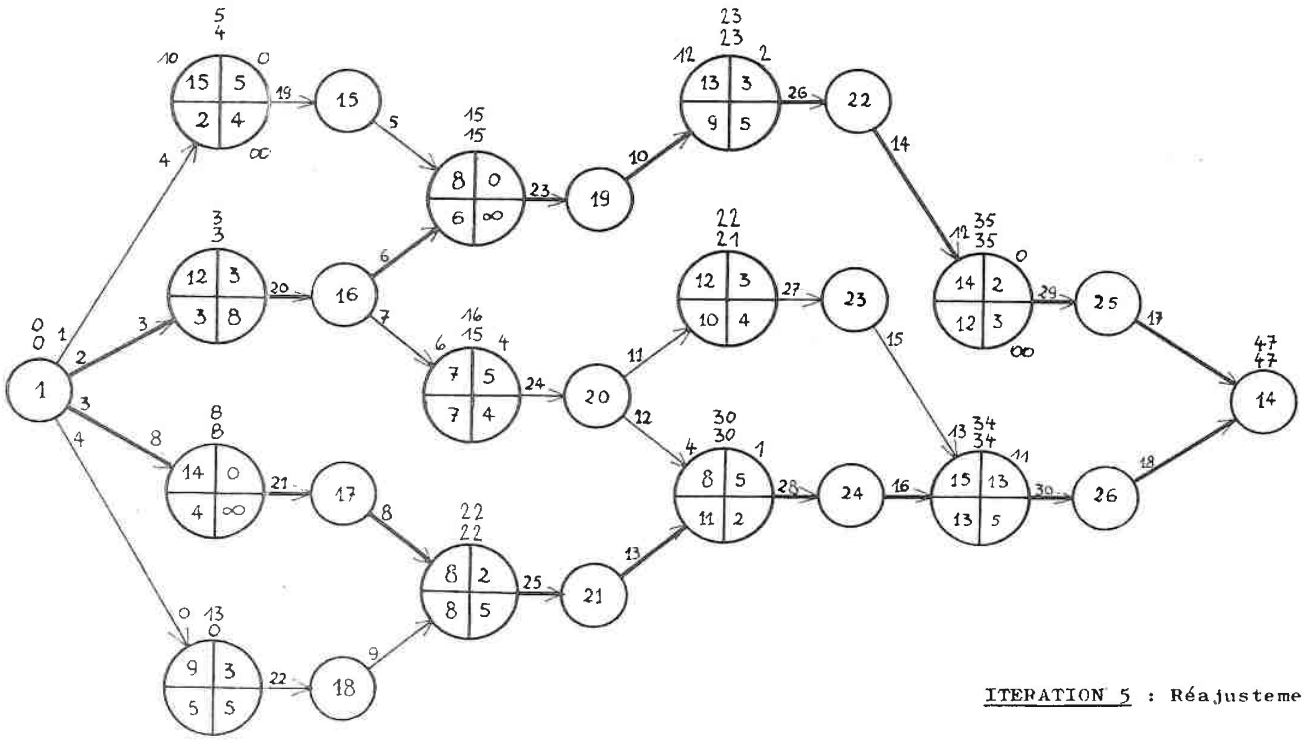
ITERATION 4 : Les opérations faites sur ce graphe sont décrites en détail dans les pages 52, 53 et 54.



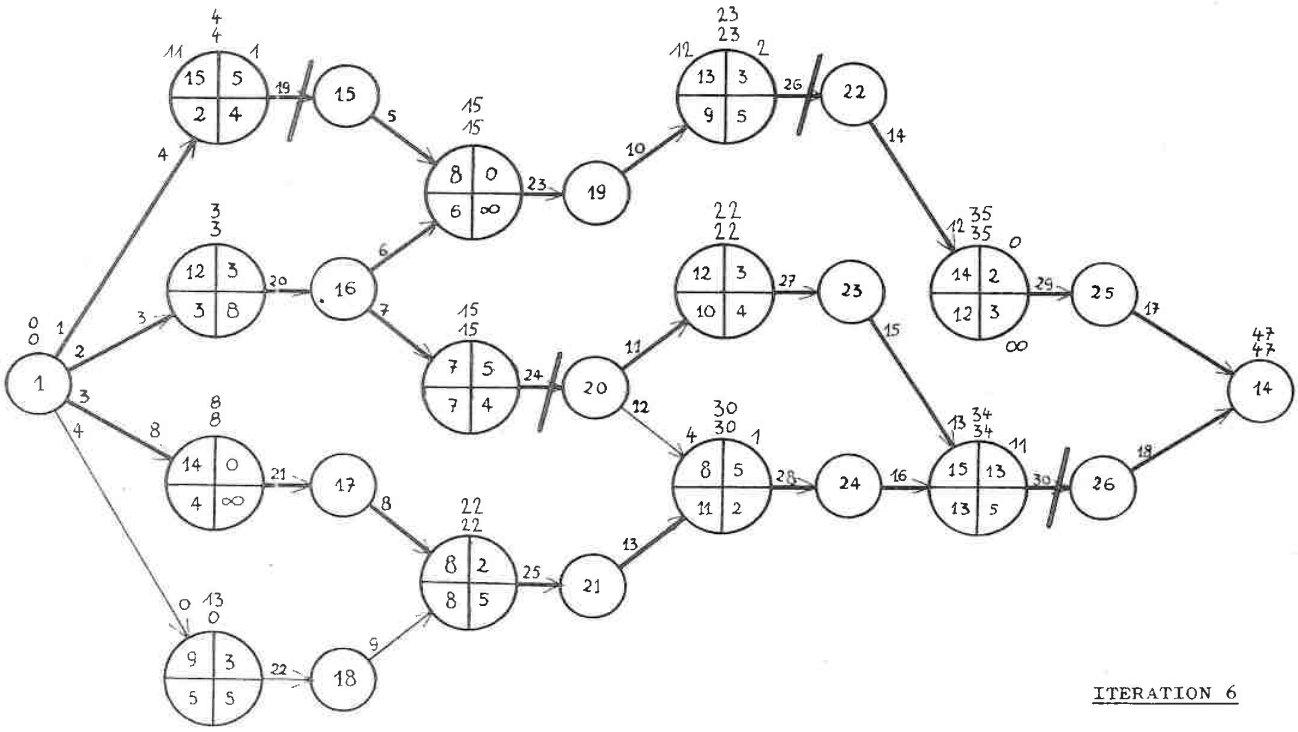
ITERATION 4 : Réajustement



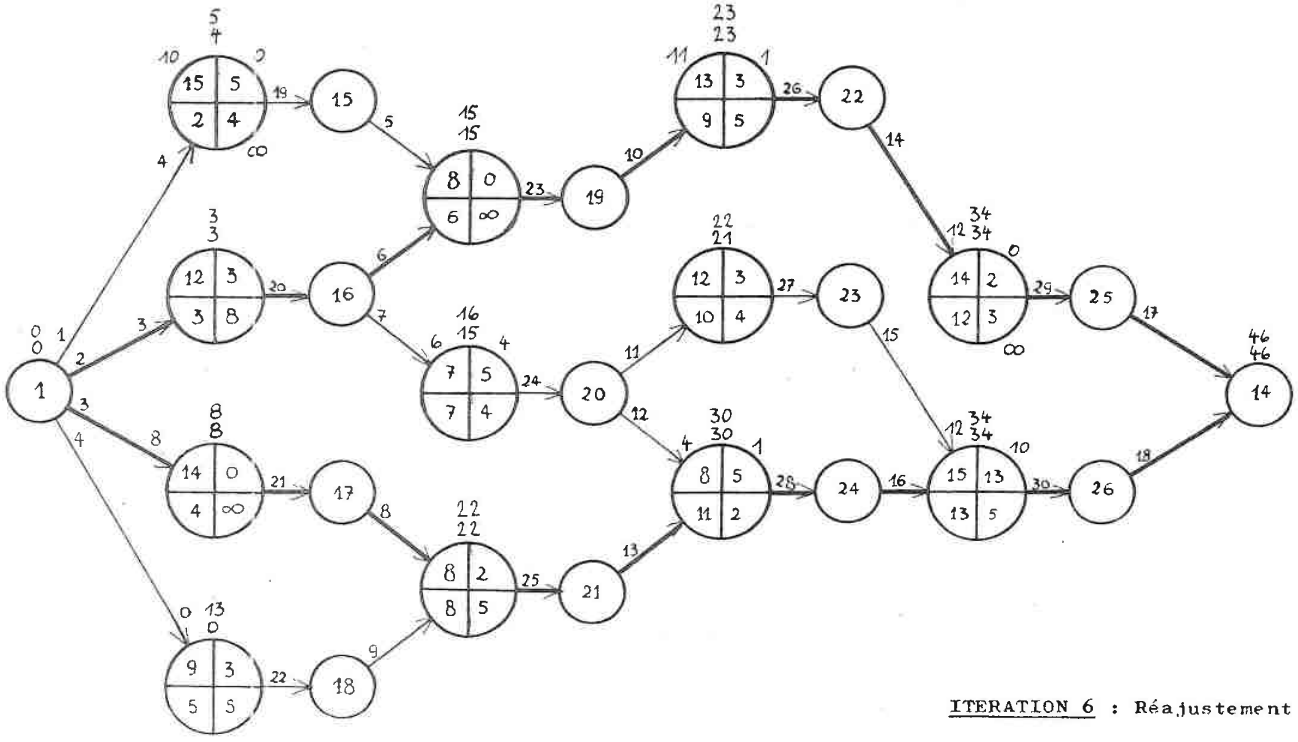
ITERATION 5



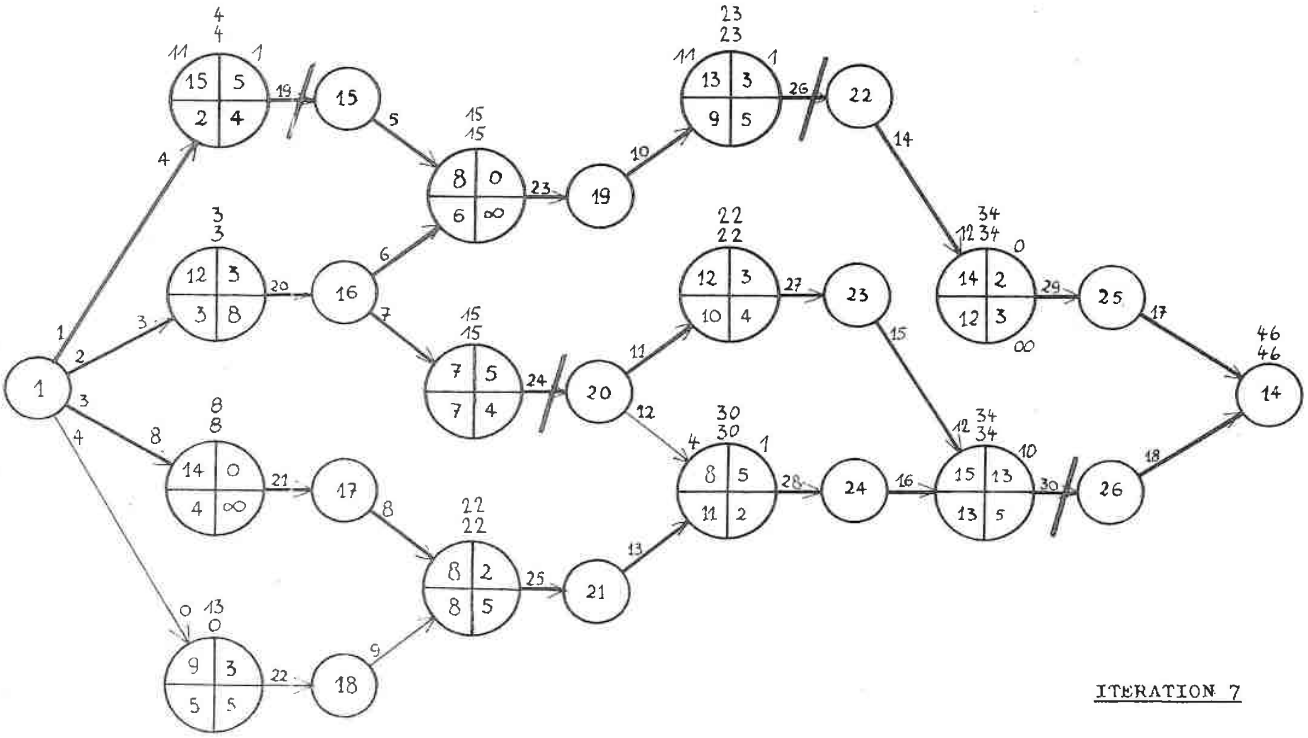
ITERATION 5 : Réajustement



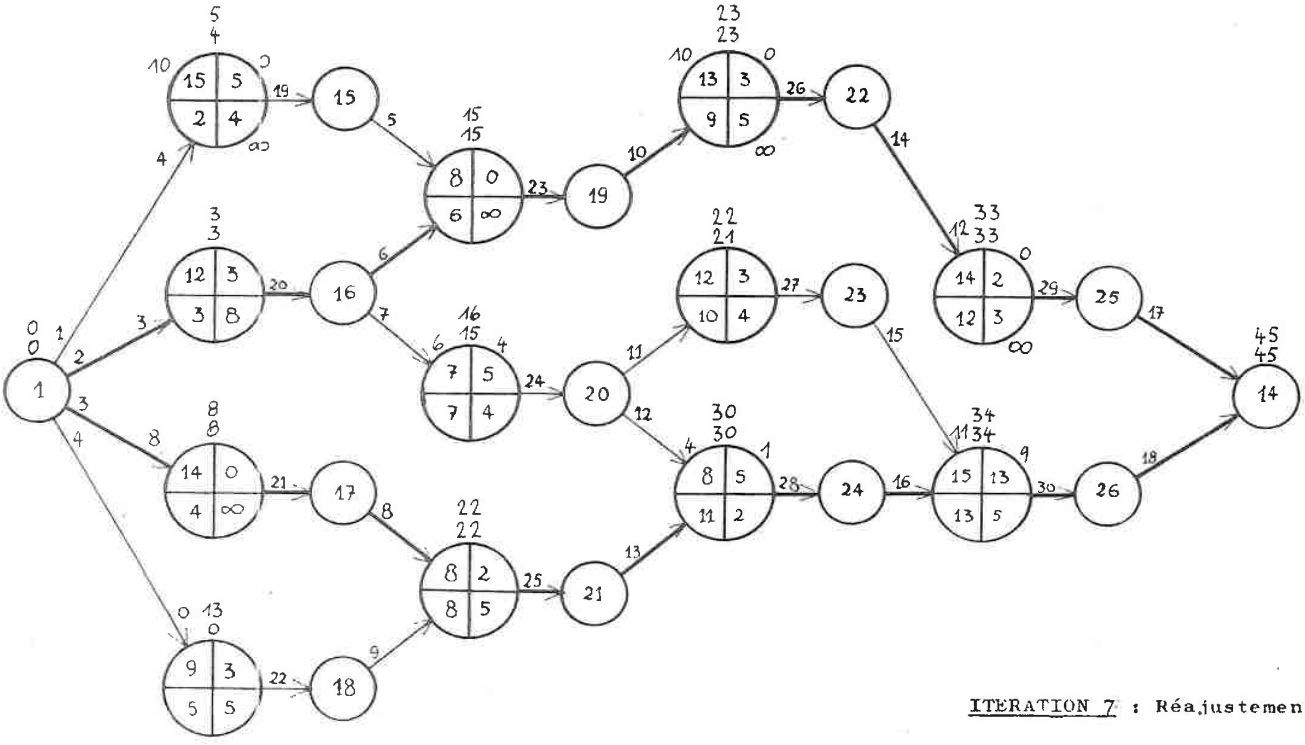
ITERATION 6



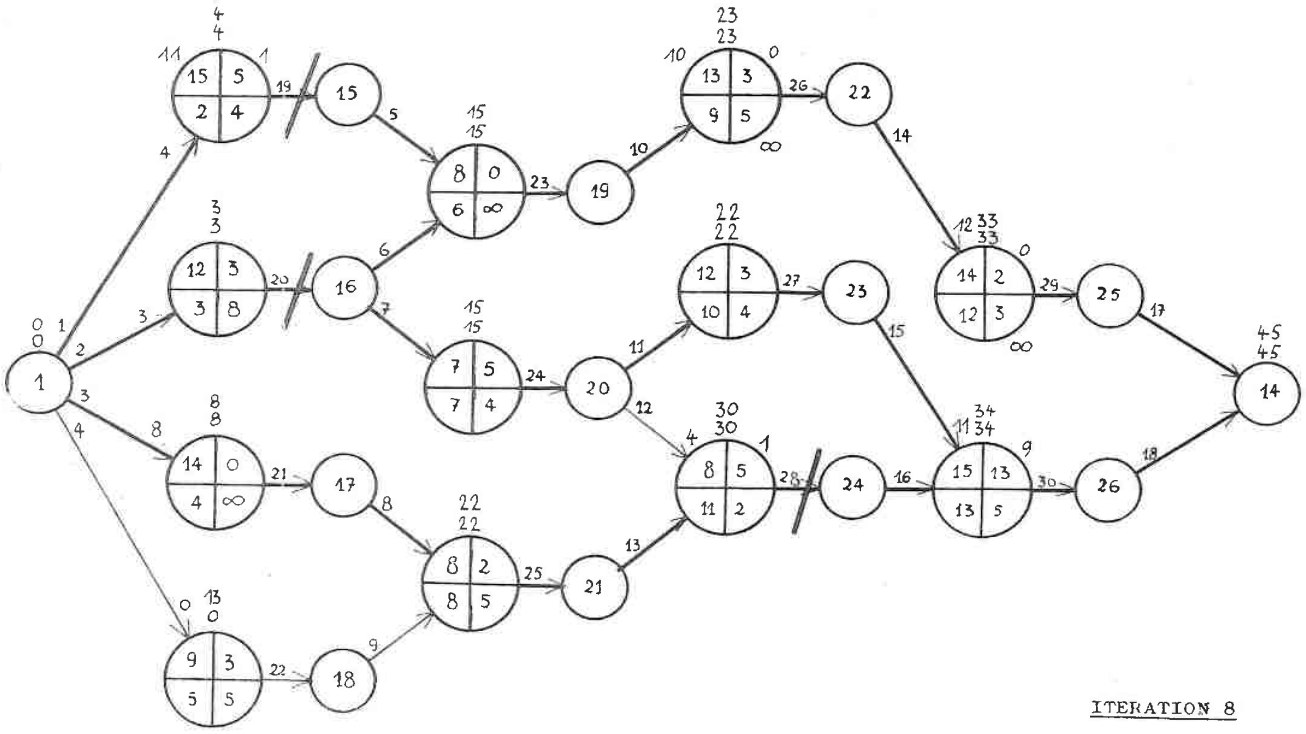
ITERATION 6 : Réajustement



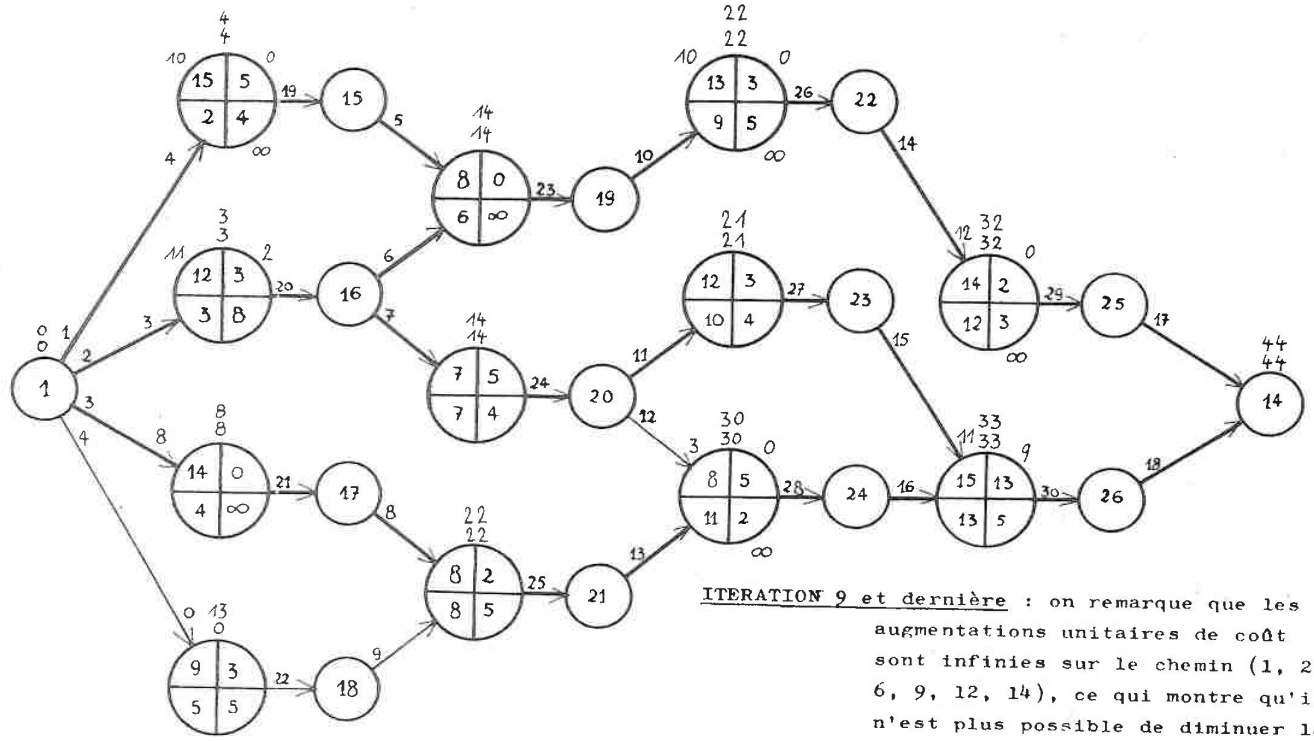
ITERATION 7



ITERATION 7 : Réajustement



ITERATION 8



ITERATION 9 et dernière : on remarque que les augmentations unitaires de coût sont infinies sur le chemin (1, 2, 6, 9, 12, 14), ce qui montre qu'il n'est plus possible de diminuer la durée totale. On a atteint λ minimal = 44.

2. Evolution du PRIMAL-DUAL.

Sur ce même exemple, nous nous proposons maintenant d'étudier le comportement du PRIMAL-DUAL.

a. Ecriture des équations : A cet effet, il est souhaitable d'écrire clairement toutes les équations, de les numéroter, et le cas échéant de leur associer les variables correspondantes.

Il vient :

+ Problème PRIMAL

- Contraintes unilatérales

(1) f_1 $t_2 \geqslant e_1$

(2) f_2 $t_3 \geqslant e_2$

(3) f_3 $t_4 \geqslant e_3$

(4) f_4 $t_5 \geqslant e_4$

(5) f_5 $t_6 - t_2 + \tau_2 \geqslant d_2$

(6) f_6 $t_6 - t_3 + \tau_3 \geqslant d_3$

(7) f_7 $t_7 - t_3 + \tau_3 \geqslant d_3$

(8) f_8 $t_8 - t_4 + \tau_4 \geqslant d_4$

(9) f_9 $t_8 - t_5 + \tau_5 \geqslant d_5$

(10) f_{10} $t_9 - t_6 + \tau_6 \geqslant d_6$

(11) f_{11} $t_{10} - t_7 + \tau_7 \geqslant d_7$

(12) f_{12} $t_{11} - t_7 + \tau_7 \geqslant d_7$

(13) f_{13} $t_{11} - t_8 + \tau_8 \geqslant d_8$

(14) f_{14} $t_{12} - t_9 + \tau_9 \geqslant d_9$

(15) f_{15} $t_{13} - t_{10} + \tau_{10} \geqslant d_{10}$

(16) f_{16} $t_{13} - t_{11} + \tau_{11} \geqslant d_{11}$

(17) f_{17} $t_{14} - t_{12} + \tau_{12} \geqslant d_{12}$

(18) f_{18} $t_{14} - t_{13} + \tau_{13} \geqslant d_{13}$

- Contrainte bilatérale

(19) φ $t_{14} = \Lambda$

- Contraintes unilatérales

(20) γ_2 $-\tau_2 \geqslant -\theta_2$

(21) γ_3 $-\tau_3 \geqslant -\theta_3$

(22) γ_4 $-\tau_4 \geqslant -\theta_4$

(23) γ_5 $-\tau_5 \geqslant -\theta_5$

(24) γ_6 $-\tau_6 \geqslant -\theta_6$

(25) γ_7 $-\tau_7 \geqslant -\theta_7$

(26) γ_8 $-\tau_8 \geqslant -\theta_8$

(27) γ_9 $-\tau_9 \geqslant -\theta_9$

(28) γ_{10} $-\tau_{10} \geqslant -\theta_{10}$

(29) γ_{11} $-\tau_{11} \geqslant -\theta_{11}$

(30) γ_{12} $-\tau_{12} \geqslant -\theta_{12}$

(31) γ_{13} $-\tau_{13} \geqslant -\theta_{13}$

- Conditions de signes

(32) $\tau_2 \geqslant 0$

(33) $\tau_3 \geqslant 0$

(34) $\tau_4 \geqslant 0$

(35) $\tau_5 \geqslant 0$

(36) $\tau_6 \geqslant 0$

(37) $\tau_7 \geqslant 0$

(38) $\tau_8 \geqslant 0$

(39) $\tau_9 \geqslant 0$

(40) $\tau_{10} \geqslant 0$

(41) $\tau_{11} \geqslant 0$

(42) $\tau_{12} \geqslant 0$

(43) $\tau_{13} \geqslant 0$

- Fonction

$$\text{Min} \sum_{i=2,13} c_i \tau_i$$

++ Problème DUAL

- Contraintes bilatérales

(1') t_2 $f_1 - f_5 = 0$

(2') t_3 $f_2 - f_6 - f_7 = 0$

(3') t_4 $f_3 - f_8 = 0$

(4') t_5 $f_4 - f_9 = 0$

(5') t_6 $f_5 + f_6 - f_{10} = 0$

(6') t_7 $f_7 - f_{11} - f_{12} = 0$

(7') t_8 $f_8 + f_9 - f_{13} = 0$

(8') t_9 $f_{10} - f_{14} = 0$

(9') t_{10} $f_{11} - f_{15} = 0$

(10') t_{11} $f_{12} + f_{13} - f_{16} = 0$

(11') t_{12} $f_{14} - f_{17} = 0$

(12') t_{13} $f_{15} + f_{16} - f_{18} = 0$

(13') t_{14} $f_{17} + f_{18} - \varphi = 0$

- Contraintes unilatérales

(14') τ_2 $f_5 - \gamma_2 \leqslant c_2$

(15') τ_3 $f_6 + f_7 - \gamma_3 \leqslant c_3$

(16') τ_4 $f_8 - \gamma_4 \leqslant c_4$

(17') τ_5 $f_9 - \gamma_5 \leqslant c_5$

(18') τ_6 $f_{10} - \gamma_6 \leqslant c_6$

(19') τ_7 $f_{11} + f_{12} - \gamma_7 \leqslant c_7$

(20') τ_8 $f_{13} - \gamma_8 \leqslant c_8$

(21') τ_9 $f_{14} - \gamma_9 \leqslant c_9$

(22') τ_{10} $f_{15} - \gamma_{10} \leqslant c_{10}$

(23') τ_{11} $f_{16} - \gamma_{11} \leqslant c_{11}$

(24') τ_{12} $f_{17} - \gamma_{12} \leqslant c_{12}$

(25') τ_{13} $f_{18} - \gamma_{13} \leqslant c_{13}$

-Conditions de signes

(26') $f_1 \geqslant 0$

(27') $f_2 \geqslant 0$

(28') $f_3 \geqslant 0$

(29') $f_4 \geqslant 0$

(30') $f_5 \geqslant 0$

(31') $f_6 \geqslant 0$

(32') $f_7 \geqslant 0$

(33') $f_8 \geqslant 0$

(34') $f_9 \geqslant 0$

(35') $f_{10} \geqslant 0$

(36') $f_{11} \geqslant 0$

(37') $f_{12} \geqslant 0$

(38') $f_{13} \geqslant 0$

(39') $f_{14} \geqslant 0$

(40') $f_{15} \geqslant 0$

(41') $f_{16} \geqslant 0$

(42') $f_{17} \geqslant 0$

(43') $f_{18} \geqslant 0$

(44') $\gamma_2 \geqslant 0$

(45') $\gamma_3 \geqslant 0$

(46') $\gamma_4 \geqslant 0$

(47') $\gamma_5 \geqslant 0$

(48') $\gamma_6 \geqslant 0$

(49') $\gamma_7 \geqslant 0$

(50') $\gamma_8 \geqslant 0$

(51') $\gamma_9 \geqslant 0$

(52') $\gamma_{10} \geqslant 0$

(53') $\gamma_{11} \geqslant 0$

(54') $\gamma_{12} \geqslant 0$

(55') $\gamma_{13} \geqslant 0$

- Fonction

Max $\left[\lambda \varphi + \sum d_k f_k + \sum e_l f_l - \sum \theta_i \gamma_i \right]$

+++ CONDITIONS D'OPTIMALITE

- Premier groupe

(1) $f_1 \times (t_2 - e_1) = 0$

(2) $f_2 \times (t_3 - e_2) = 0$

(3) $f_3 \times (t_4 - e_3) = 0$

(4) $f_4 \times (t_5 - e_4) = 0$

(5) $f_5 \times (t_6 - t_2 + \tau_2 - d_2) = 0$

(6) $f_6 \times (t_6 - t_3 + \tau_3 - d_3) = 0$

(7) $f_7 \times (t_7 - t_3 + \tau_3 - d_3) = 0$

(8) $f_8 \times (t_8 - t_4 + \tau_4 - d_4) = 0$

(9) $f_9 \times (t_8 - t_5 + \tau_5 - d_5) = 0$

(10) $f_{10} \times (t_9 - t_6 + \tau_6 - d_6) = 0$

(11) $f_{11} \times (t_{10} - t_7 + \tau_7 - d_7) = 0$

(12) $f_{12} \times (t_{11} - t_7 + \tau_7 - d_7) = 0$

(13) $f_{13} \times (t_{11} - t_8 + \tau_8 - d_8) = 0$

(14) $f_{14} \times (t_{12} - t_9 + \tau_9 - d_9) = 0$

(15) $f_{15} \times (t_{13} - t_{10} + \tau_{10} - d_{10}) = 0$

(16) $f_{16} \times (t_{13} - t_{11} + \tau_{11} - d_{11}) = 0$

(17) $f_{17} \times (t_{14} - t_{12} + \tau_{12} - d_{12}) = 0$

(18) $f_{18} \times (t_{14} - t_{13} + \tau_{13} - d_{13}) = 0$

- Deuxième groupe

(2) $\gamma_2 \times (\tau_2 - \theta_2) = 0$

(3) $\gamma_3 \times (\tau_3 - \theta_3) = 0$

(4) $\gamma_4 \times (\tau_4 - \theta_4) = 0$

(5) $\gamma_5 \times (\tau_5 - \theta_5) = 0$

(6) $\gamma_6 \times (\tau_6 - \theta_6) = 0$

(7) $\gamma_7 \times (\tau_7 - \theta_7) = 0$

(8) $\gamma_8 \times (\tau_8 - \theta_8) = 0$

(9) $\gamma_9 \times (\tau_9 - \theta_9) = 0$

(10) $\gamma_{10} \times (\tau_{10} - \theta_{10}) = 0$

(11) $\gamma_{11} \times (\tau_{11} - \theta_{11}) = 0$

(12) $\gamma_{12} \times (\tau_{12} - \theta_{12}) = 0$

(13) $\gamma_{13} \times (\tau_{13} - \theta_{13}) = 0$

- Troisième groupe

(2) $\tau_2 \times (f_5 - \gamma_2 - c_2) = 0$

(3) $\tau_3 \times (f_6 + f_7 - \gamma_3 - c_3) = 0$

(4) $\tau_4 \times (f_8 - \gamma_4 - c_4) = 0$

(5) $\tau_5 \times (f_9 - \gamma_5 - c_5) = 0$

(6) $\tau_6 \times (f_{10} - \gamma_6 - c_6) = 0$

(7) $\tau_7 \times (f_{11} + f_{12} - \gamma_7 - c_7) = 0$

(8) $\tau_8 \times (f_{13} - \gamma_8 - c_8) = 0$

(9) $\tau_9 \times (f_{14} - \gamma_9 - c_9) = 0$

(10) $\tau_{10} \times (f_{15} - \gamma_{10} - c_{10}) = 0$

(11) $\tau_{11} \times (f_{16} - \gamma_{11} - c_{11}) = 0$

(12) $\tau_{12} \times (f_{17} - \gamma_{12} - c_{12}) = 0$

(13) $\tau_{13} \times (f_{18} - \gamma_{13} - c_{13}) = 0$

b. Vérification du comportement du système primal-dual :

C'est une vérification déduite des résultats obtenus graphiquement et qui permet de démontrer que la solution obtenue est bien une solution optimale.

Rappelons que toute condition d'optimalité, est le produit d'une variable, soit primale, soit duale, par l'expression d'une contrainte correspondante.

La condition est donc vérifiée, soit lorsque la variable est nulle, soit lorsque la contrainte, qui est donc le second membre de la condition d'optimalité, est saturée.

Nous présenterons les résultats complets sous forme d'un tableau qui sera construit de la façon suivante :

+ Au milieu nous aurons une colonne "indice".
+ A gauche de cette colonne, nous aurons les seconds membres des trois groupes de conditions d'optimalité :
  - le groupe 1 comporté 18 équations. Les seconds membres des 4 premières seront toujours saturés car ils correspondent à des arcs issus de la source ;
  - le groupe 2 contient 12 équations numérotées de 2 à 13. Les cases 4 et 6 du tableau contiennent une croix parce que les tâches 4 et 6 ne peuvent

plus être diminuées ;

- le groupe 3 compte aussi 12 équations numérotées comme préeédemment.
  Au fur et à mesure que les seconds membres des conditions d'optimalité se satureront, nous mettrons un S dans les cases correspondantes.

+ A droite de la colonne "indice" se trouvent les variables et paramètres du PRIMAL-DUAL qui évolueront au cours des itérations.

Considérons la première itération dont les résultats se trouvent dans le tableau de la page 78.

- En appliquant le PERT classique, nous déterminons les dates au plus tôt contenues dans la colonne t.
- Dans les colonnes c, d', $\theta - \tau$ se trouvent à l'initialisation les augmentations unitaires de coût (c), les durées (d) et diminutions maximales des durées (θ), qui sont des données du problème et dont certaines évolueront au cours des itérations.
- Dans le premier groupe de conditions d'optimalité se saturent les seconds membres des équations 5, 10, 14 et 17.

Après cette phase d'initialisation, considérons à présent l'itération 4 de la page 81 qui est plus générale.

- L'algorithme du flot maximum nous permet de remplir la colonne f.
- Il détermine aussi la coupe. Ce sont les tâches 2 et 11, dont on diminuera les durées. A droite sur le tableau, nous marquons la coupe par des flèches ($\Longleftarrow$)

  + τ_2 et τ_{11} sont augmentées de 3,
  + d_2 et $\theta_2 - \tau_2$ ainsi que d_{11} et $\theta_{11} - \tau_{11}$ sont diminuées de 3.
  + A l'aide d'un nouveau PERT classique, nous obtenons les nouveaux temps au plus tôt t.

De plus, dans cette itération se sont saturés les seconds membres des équations 2 du groupe 3, 7, 11 et 15 du groupe 1.

On vérifie aisément que les conditions d'optimalité sont satisfaites.

c. Résultats :

Remarquons au préalable que dans les tableaux qui suivent,

- une flèche barrée indique une tâche de la coupe, mais dont la durée n'a pas été diminuée ;
- un signe ∞ sous un autre nombre a dans une même case des coûts, annonce que le coût de la tâche considérée passe de a à l'∞ dans cette itération ;
- le chemin critique est formé par les arcs dont le flot est différent de 0.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d'	$\theta-\tau$
		—	1							
		—	2		4			4	15	5
		—	3		3			8	12	3
	X	—	4		8			∞	14	0
		S	5		0			5	9	3
	X		6		19			∞	8	0
			7		15			4	7	5
			8		22			5	8	2
			9		27			5	13	3
		S	10		22			4	12	3
			11		30			2	8	5
			12		40			3	14	2
			13		38			5	15	13
		S	14		54					
			15							
			16							
		S	17							
			18							

Résultats de la page 55 et commentés sur les pages 75 et 76.
Pour cette initialisation, on trouve dans les colonnes c, d' et $\theta-\tau$, les données c, d et θ .
Un PERT classique donne les temps t et détermine le chemin critique qui apparaît dans la colonne 1.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d'	$\theta-\tau$
		—	1	3						
		—	2		4			4	15	5
		—	3		3					
	X	—	4		8					
		S	5	3	0					
	X		6		19			∞	8	0
			7		15					
		S	8		22					
			9		27			5	13	3
		S	10	3	22					
			11		30					
S			12		40		1	3	13	1
		S	13		38					
		S	14	3	53					
			15							
		S	16							
		S	17	3						
		S	18							

Résultats des pages 55 et 56.
Nous ne porterons plus désormais sur les tableaux que les renseignements des tâches qui subissent une évolution.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d	$\theta\cdot\tau$	
		—	1	3							
		—	2		4			4	15	5	
		—	3	2	3						
	X	—	4		8			∞	14	0	
		S	5	3	0						
	X		6		19			∞	8	0	
			7		15						
		S	8	2	22			5	8	2	
			9		27			5	13	3	
		S	10	3	22						
S			11		30		1	2	7	4	⇐
S	S		12		40		2	3 ∞	12	0	⇐
		S	13	2	37						
	S		14	3	52						
			15								
	S		16	2							
	S		17	3							
	S		18	2							

Résultats des pages 56 et 57.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d	$\theta\cdot\tau$	
		—	1	4							
S		—	2		4		3	4	12	2	⇐
		—	3	2	3						
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X		6		16			∞	8	0	
		S	7		15						
		S	8	2	22			5	8	2	
			9		24			5	13	3	
		S	10	4	22						
S		S	11		30		4	2	4	1	⇐
S	S		12		37		2	∞	12	0	
		S	13	2	34			5	15	13	
		S	14	4	49						
		S	15								
		S	16	2							
		S	17	4							
		S	18	2							

Résultats des pages 57 et 58, qui sont commentés sur la page 77.

EVOLUTION DU PRIMAL-DUAL

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d	θ-τ	
		—	1	4							
S		—	2	4	4		4	4	11	1	⇐
		—	3	1	3			8	12	3	
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X	S	6		15			∞	8	0	
		S	7	4	15			4	7	5	⇐
		S	8	1	22			5	8	2	
			9		23			5	13	3	
		S	10	4	22			4	12	3	
S		S	11	4	30		4	2	4	1	
S	S		12		36		2	∞	12	0	
S		S	13	1	34		1	5	14	12	⇐
		S	14	4	48						
		S	15	4							
		S	16	1							
		S	17	4							
		S	18	5							

Résultats des pages 58, 59 et 60.

EVOLUTION DU PRIMAL-DUAL

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d	θ-τ	
		—	1	4							
S		—	2	5	4		4	4	11	1	⇐
		—	3	1	3			8	12	3	
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X	S	6	1	15			∞	8	0	
		S	7	4	15			4	7	5	⇐
		S	8	1	22			5	8	2	
S			9		23		1	5	12	2	⇐
		S	10	5	22			4	12	3	
S		S	11	4	30		4	2	4	1	
S	S		12		35		2	∞	12	0	
S		S	13	1	34		2	5	13	11	⇐
		S	14	5	47						
		S	15	4							
		S	16	1							
		S	17	5							
		S	18	5							

Résultats des pages 60, 61 et 62.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d'	$\theta\tau$	
		—	1	4							
S		—	2	5	4		4	4	11	1	⇐=
		—	3	1	3			8	12	3	
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X	S	6	1	15			∞	8	0	
		S	7	4	15			4	7	5	⇐=
		S	8	1	22			5	8	2	
S			9		23		2	5	11	1	⇐
		S	10	5	22			4	12	3	
S		S	11	4	30		4	2	4	1	
S	S		12		34		2	∞	12	0	
S		S	13	1	34		3	5	12	10	⇐
		S	14	5	46						
		S	15	4							
		S	16	1							
		S	17	5							
		S	18	5							

Résultats des pages 62, 63 et 64.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d	$\theta\tau$	
		—	1	4							
S		—	2	5	4		4	4	11	1	⇐=
		—	3	1	3			8	12	3	
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X	S	6	1	15			∞	8	0	
		S	7	4	15			4	7	5	⇐=
		S	8	1	22			5	8	2	
S	S		9		23		3	5/∞	10	0	⇐
		S	10	5	22			4	12	3	
S		S	11	4	30		4	2	4	1	
S	S		12		33		2	∞	12	0	
S		S	13	1	34		4	5	11	9	⇐
		S	14	5	45						
		S	15	4							
		S	16	1							
		S	17	5							
		S	18	5							

Résultats des pages 64, 65 et 66.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d'	θ-τ	
		—	1	4							
S	S	—	2	8	4		5	4 ∞	10	0	⇐
S		—	3	2	3		1	8	11	2	⇐
	X	—	4		8			∞	14	0	
		S	5	4	0						
	X	S	6	8	14			∞	8	0	
		S	7		14			4	7	5	
		S	8	2	22			5	8	2	
S	S		9		22		3	∞	10	0	
		S	10	12	21			4	12	3	
S	S	S	11		30		5	2 ∞	3	0	⇐
S	S		12		32		2	∞	12	0	
S		S	13	2	33		4	5	11	9	
		S	14	12	44						
		S	15								
		S	16	2							
		S	17	12							
		S	18	2							

Résultats des pages 66 et 67.

E V O L U T I O N D U P R I M A L - D U A L

Seconds membres des conditions d'optimalité

Valeurs des variables

3	2	1	INDICE	f	t	γ	τ	c	d'	θ-τ
		—	1	∞						
		—	2					∞	10	0
		—	3					8	11	2
	X	—	4					∞	14	0
			5	∞						
	X		6					∞	8	0
			7					4	7	5
			8					5	8	2
			9					∞	10	0
			10	∞				4	12	3
			11					∞	3	0
			12					∞	12	0
			13					5	11	9
			14	∞						
			15							
			16							
			17	∞						
			18							

Résultats de la page 67.
On constate que le chemin formé par les arcs 1, 5, 10, 14 et 17 a des capacités ~~infinies~~.
Il est donc impossible d'aller plus loin et l'optimum est atteint.

D. INFORMATISATION.

1. Entrée des données : Les données seront lues sur cartes perforées de la manière suivante :

- sur la carte 1 :

  TT, N, M, R en format (4 I 5)

- sur la carte 2 :

  E (I), I = 1,R en format (R I 5)

- sur les cartes 3 à M + 2 :

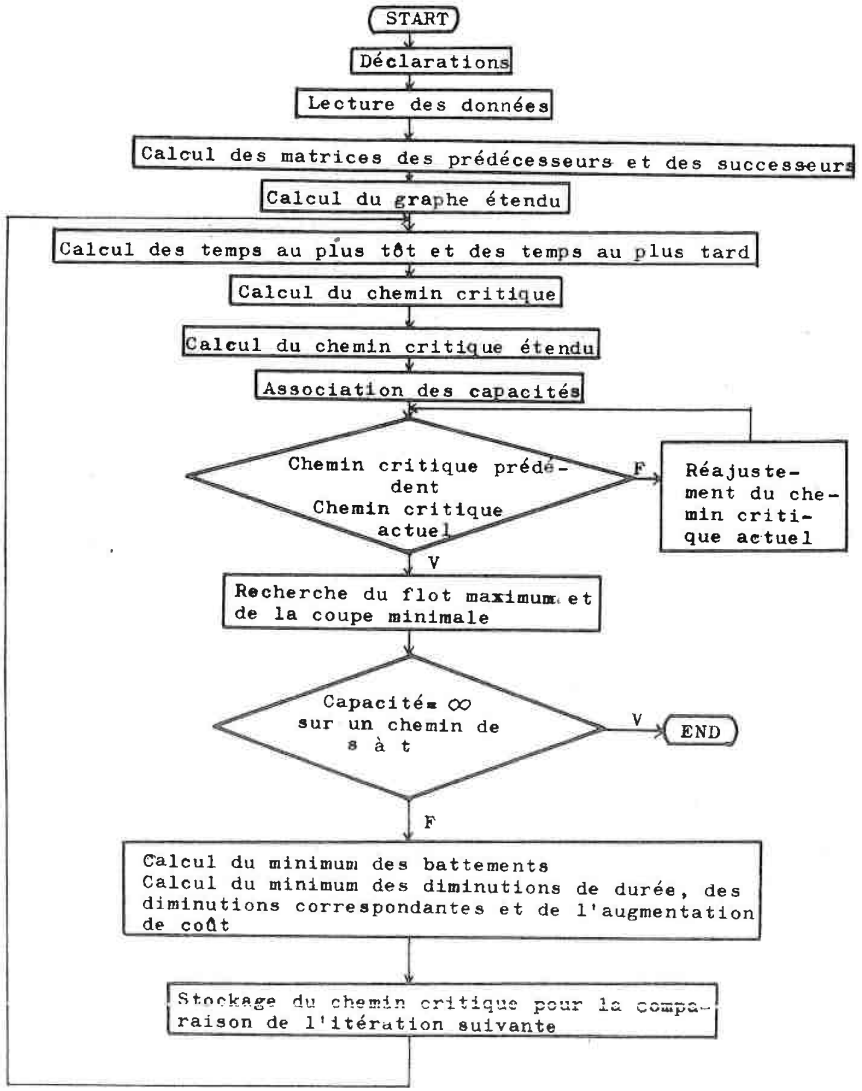
  EXINI (I), EXFIN (I), I = 1,M en format (2 I 5)

- sur les cartes M + 3 à M + N +1 :

  D (I), TETA (I), C (I), I=2,N-1 en format (3 I 5)

L'interprétation des noms de variables sera faite sur la page 90.

2. Organigramme général.



3. Déroulement de l'algorithme.

a. Déclarations

b. Lecture des données : Les uniques données entrées dans l'ordinateur sont les suivantes :

- TT : le temps total donné. C'est le temps total d'exécution que l'on s'impose et qui une fois atteint arrête le programme.
- N : le nombre de sommets du graphe.
- M : le nombre d'arcs du graphe.
- R : le nombre d'arcs ayant pour extrémité initiale la source.
- E (I), I = 1,R : la durée entre une tâche succédant à la tâche 1, et l'instant initial.
- EXINI (I) et EXFIN (I), I = 1,M : l'extrémité initiale et l'extrémité finale de chaque arc du graphe.
- D (I), I = 2,N - 1 : la durée de chaque tâche.
- TETA (I), I = 2,N - 1 : la diminution maximum de la durée de chaque tâche.
- C (I), I = 2,N - 1 : l'augmentation de coût unitaire.

c. Tableaux des successeurs et des prédécesseurs :

Pour les claculs ultérieurs, nous aurons besoin de connaître les successeurs de chaque sommet, de même que les prédécesseurs. C'est le sous-programme PRESU qui effectue, par un transfert de paramètres adéquats, ces deux calculs.

Des priorités de précédence et de succession seront attribuées à chacun des sommets : en effet pour les calculs des dates au plus tôt et des dates au plus tard, la date au plus tôt de la tâche j se calcule en fonction des dates au plus tôt de ses prédécesseurs i_1; i_2, i_3 ...

Les dates au plus tôt des prédécesseurs doivent donc être connues avant celle de j. C'est pourquoi il est utile d'affecter une priorité à chaque sommet. C'est le sommet qui est le plus prioritaire qui aura numériquement la priorité la plus petite. Le problème est similaire pour le calcul des dates au plus tard. C'est le sous-programme ORDO qui affecte ces priorités.

Pour le tableau des prédécesseurs, nous utiliserons les vecteurs suivants :

- PRIOR (I) indique la priorité du sommet I
- SOM (I) contient le numéro du sommet I
- DIM (I) indique le nombre de prédécesseurs du sommet I
- PREDE (I, J), J = 1, DIM (I) contient les J prédécesseurs du sommet I.

Pour le tableau des successeurs, nous utiliserons respectivement :

PRIOR1(I), SOM 1 (I), DIM 1 (I) et SUCCE (I, J).

d. Graphe étendu : Nous avons vu antérieurement que pour appliquer l'algorithme du flot maximum, il était commode de dédoubler les sommets. Le graphe étendu comprendra NBS = 2N - 2 sommets et I3 = M + N - 2 arcs.

Les sommets rajoutés seront numérotés de N + 1 à NBS et les arcs de M + 1 à I3. Dans ce graphe étendu l'arc I aura

- FEXINI (I) comme sommet initial et
- FEXFIN (I) comme sommet final.

C'est sur le graphe étendu que nous appliquerons l'algorithme du flot maximum. A cet effet, nous appliquons aussi les sous-programmes PRESU et ORDO aux sommets du graphe étendu.

Les sous-programmes nous seront retournés avec les renseignements nécessaires rangés dans F PRIOR (I), F SOM (I), F DIM (I), F PREDE (I, J), pour le tableau des prédécesseurs et dans F PRIOR 1 (I), F SOM 1 (I), F DIM 1 (I), F SUCCE (I,J pour la matrice des successeurs.

e. Temps au plus tôt et temps au plus tard : Les temps au plus tôt seront calculés en fonction des temps au plus tôt des prédécesseurs par la formule :

$$t_j = \max (t_{i1} + d_{i1}, t_{i2} + d_{i2}, \ldots)$$

où i1, i2, ... sont les prédécesseurs de j avec $t_1 = 0$, et seront stockés dans

- T (I), I = 1, N

Les temps au plus tard seront connus par

$$t'_j = \min (t'_{j1} - d_i, t'_{j2} - d_i \ldots)$$

où j1, j2 ... sont les successeurs de i avec $t'_N = t_N$. On les gardera dans

- T1 (I), I = 1, N

Pour un sommet donné, le battement

- BAT (I) sera la différence entre le temps au plus tard et le temps au plus tôt.

  A ce niveau, on préservera, à la première itération, le temps total,
- TEMPS qui serait la durée totale de l'ordonnancement si l'on se trouvait dans le cas d'un PERT classique.

f. Chemin critique : Soient E1 l'extrémité initiale et E2 l'extrémité finale de l'arc J.

Si T (E1) = T1 (E1)
si T (E2) = T1 (E2)
si T (E1) + D (E1) = T (E2)
⇒ l'arc J est un arc critique

Il fera partie du chemin critique

- CHECRI (I)

A la première itération on range

CHECRI (I) dans CHECRY (I) I = 1, I1

g. Chemin critique étendu : On vient de déterminer le chemin critique sur le graphe initial. En passant au graphe étendu, il faut aussi agrandir le chemin cri-

tique qui devient le chemin critique étendu. Le chemin critique étendu sera composé d'arcs appartenant au chemin critique et d'arcs rajoutés compris entre deux arcs faisant partie du chemin critique. Le chemin critique étendu est mis dans
- CHE 1 (I), I = 1, I2.

h. Association des capacités : En vue du calcul du flot maximum, il sera affecté une capacité à chaque arc I du graphe étendu, de la façon suivante :

- CAPA (I) = $\begin{cases} 0 \text{ si I} \notin \text{CHE 1} \\ \infty \text{ si I a pour extrémité initiale la source} \\ \infty \text{ si I est de la forme } (k', e_i) \\ C(J) \text{ si I est de la forme } (k, k') \\ \quad \text{avec J = sommet initial de l'arc I} \end{cases}$

i. Comparaison du chemin critique précédent et du chemin critique actuel : Le chemin critique de l'itération précédente se trouve dans CHECRY (I) et l'actuel dans CHECRI (J).
On peut ainsi vérifier si les arcs précedemment critiques le sont encore. Si c'est le cas, on passe à k. Pour la recherche du flot maximum, sinon on continue en séquence pour réajuster les arcs qui ne sont plus critiques.
Les arcs passant de l'état critique à l'état non critique, dans cette itération sont stockés dans le vecteur MANQUE (KK).

j. Réajustement du chemin critique actuel : Cette partie du programme consiste à rendre critique les arcs se trouvant dans MANQUE (KK), selon certains critères et en apportant certaines modifications.
On commence à réajuster l'arc J1 dont le sommet extrémité initiale EXTRIN (J1) a la priorité de succession PRIORI (J1) la plus importante. Pour la tâche matérialisée par ce sommet, on diminue la date au plus tard du battement. L'augmentation de coût AUGCOU est diminuée de la quantité C(S)XBAT(S) tandis que la durée D(S) et la diminution maximale de durée TETA(S) sont augmentées de BAT(S). Ce procédé permet de réajuster un ou plusieurs arcs. On se renvoie à e. pour obtenir un nouveau calendrier et un nouveau vecteur MANQUE (KK).
Aussi longtemps qu'il y aura des éléments dans MANQUE (KK), on réitérera le procédé ; dès l'instant où il sera vide, le programme sera débranché dans i. à k. où l'on déterminera la coupe minimale.

k. Recherche du flot maximum et de la coupe minimale : Cette partie du programme détermine les arcs de la coupe. Ces arcs sont de la forme (k, k') ; la diminution de durée des tâches représentées par les sommets k, entraînera la diminution du temps total avec

une augmentation de coût minimale.

La méthode mise en oeuvre est la méthode de marquage classique.

On utilisera un vecteur PI (I), I = 1,NBS. Ce vecteur contiendra dans la case I la valeur 1 ou 0, selon que le sommet I sera marqué ou non. Le marquage lui-même se fera à l'aide d'une pile PILE (NIV,J), J = 1,4.

PILE (NIV,1) contient le numéro de l'arc dont l'extrémité finale est le sommet marqué ;

PILE (NIV,2) contient le sommet marqué ;

PILE (NIV,3) contient le maximum des flots pouvant passer de la source au puits ;

PILE (NIV,4) contient la valeur 1 ou -1, selon que l'on a à faire à un arc avant, ou à un arc arrière.

α) On commence par initialiser PI (1) = 1 ; PILE(1,1)=0 ; PILE(1;2)=1 ; PILE(1,3)= ∞ ; PILE(1,4)=0

β) On itère en faisant NIV = NIV + 1.

A ce niveau se présentent trois possibilités :

- On arrive à déterminer un successeur J de I = PILE(NIV - 1,2) avec K3 = arc (I,J), tel que CAPA (K3) $\neq$ 0.

  On pose alors DELTA = CAPA (K3) - F (K3), où F (K3) est le flot qui passe dans l'arc K3 au niveau de l'itération précédente.

  Si DELTA $>$ 0 et si PI (J) = 0, on marque J :

PI (J) = 1, et on remplit la pile.

Si J $\neq$ N, on pose I = J et on va à β), sinon on continue en γ).

- On n'arrive pas à marquer de successeur de I.

  On essaie alors de trouver un arc arrière K3 où J est un prédecesseur de I tel que F (K3)$>$0 et PI (J) = 0.

  On peut donc marquer J et remplir la pile.

- S'il est impossible de trouver un arc arrière, on décrémente NIV = NIV - 1.

  Aussi longtemps que NIV sera supérieur à 0, on essaiera de recommencer à marquer par un renvoi à β). Si NIV est égal à 0, c'est l'arrêt de l'algorithme du flot maximum.

γ) Lorsqu'on a atteint le puits, on tranforme les flots de tous les arcs contenus dans PILE (I, 1) pour I = 1,NIV.

Pour K4 = 2,NIV on pose K5 = PILE(K4,1) et F (K5) devient

F(K5) + PILE(NIV,3) x PILE(K4,4).

De plus, pour cette itération, l'arc coupé sera celui auquel aura correspondu le DELTA le plus petit de PILE (I,3), I = 1,NIV.

L'arc de la coupe sera rangé dans COUPE (K) et l'extremité initiale de l'arc dans EXTRIN (K).

L'algorithme continue à se dérouler par un nou-

veau marquage de la source.

l. Calcul du minimum des battements sur tout le graphe

Les sommets dont on diminuera la durée, se trouvent dans EXTRIN (K) K = 1,KA. La quantité EPS dont il faudra diminuer, est choisie $\leqslant$

EPSP 1 = min BAT (J1), J1 = 1,N.

Nous nous imposons cette restriction, non pas parce qu'elle est mathématiquement indispensable, mais parce qu'elle nous évitera une quantité appréciable de réajustements du sous-graphe critique.

m. Calcul du minimum des diminutions de durée sur la coupe :

Il est clair que EPS sera inférieur à EPS 2, qui est le plus petit TETA (L1) - τ(L1), quand L1 prend toutes les valeurs du vecteur EXTRIN. Cette restriction est imposée par les contraintes du problème primal.

n. Calcul des diminutions correspondantes et de l'augmentation de coût :

Pour chaque sommet contenu dans EXTRIN (J1), J1 = 1,KA, nous posons L1 = EXTRIN (J1) ;

nous diminuons les durées

D(L1) = D(L1) - EPS

TETA(L1) = TETA(L1) - EPS

Nous augmentons les

. coût unitaire : COUTU = COUTU + C(L1)

.. coût de l'itération : COUTIT = COUTU xEPS

... augmentation de coût :

AUGCOU =AUGCOU + COUTIT

o. Stockage du chemin critique pour la comparaison de l'itération suivante :

Les éléments se trouvant dans CHECRI (I) vont dans CHECRY (I).

On annule les vecteurs

F (I), PI (I) et PILE (I,J)

et on retourne à e.

p. Arrêt de l'algorithme :

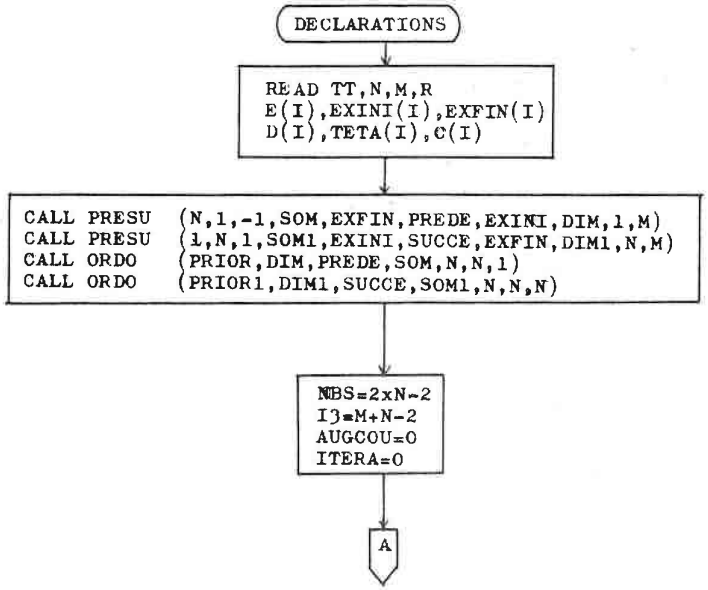
L'algorithme est terminé dans 2 cas :

- Le temps du puits T (N) atteint le temps TT qu'on s'était fixé à priori pour l'exécution totale des travaux ;

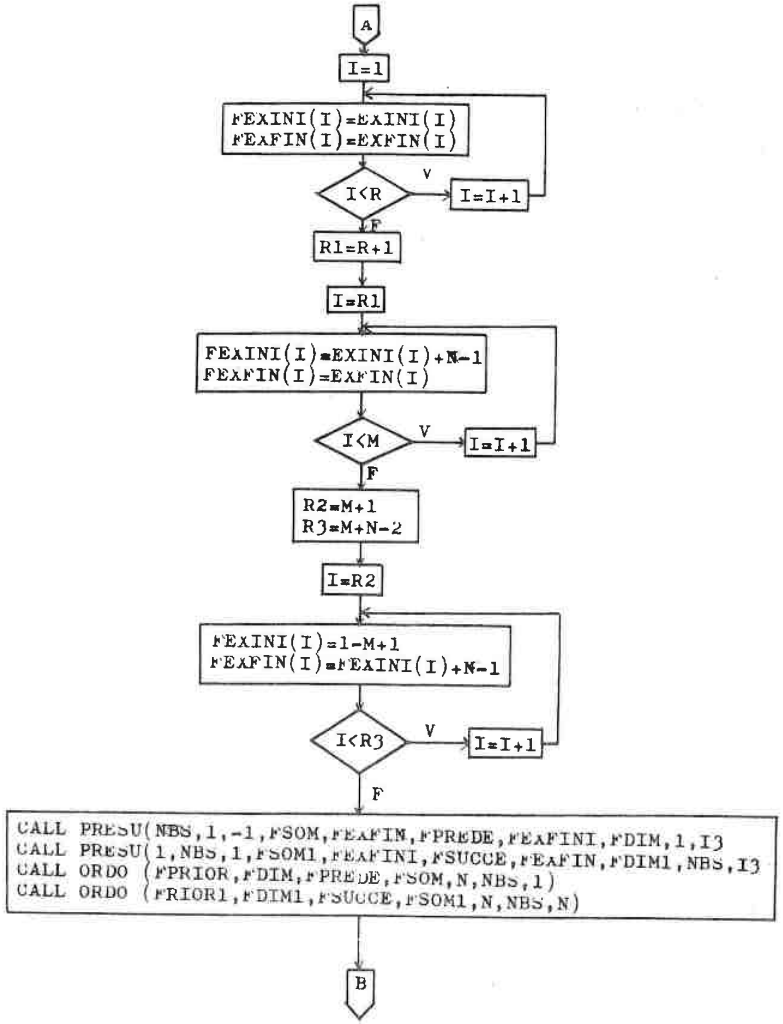
-- Il existe un chemin allant de la source au puits, dont tous les sommets représentent des tâches dont la diminution maximum de durée est nulle, et empêche toute réduction supplémentaire.

4. Organigramme détaillé.

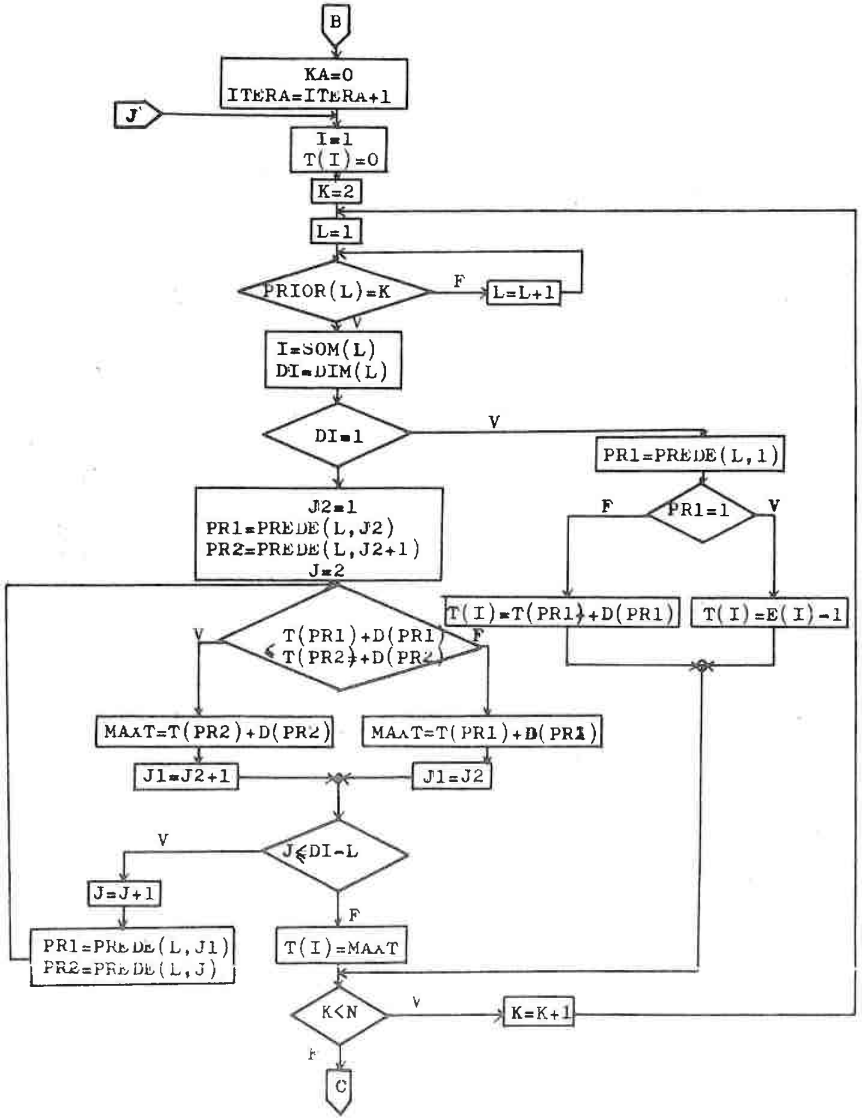
- DECLARATIONS
- LECTURE DES DONNEES
- CALCUL DES MATRICES DES PREDECESSEURS ET DES SUCCESSEURS
- INITIALISATION



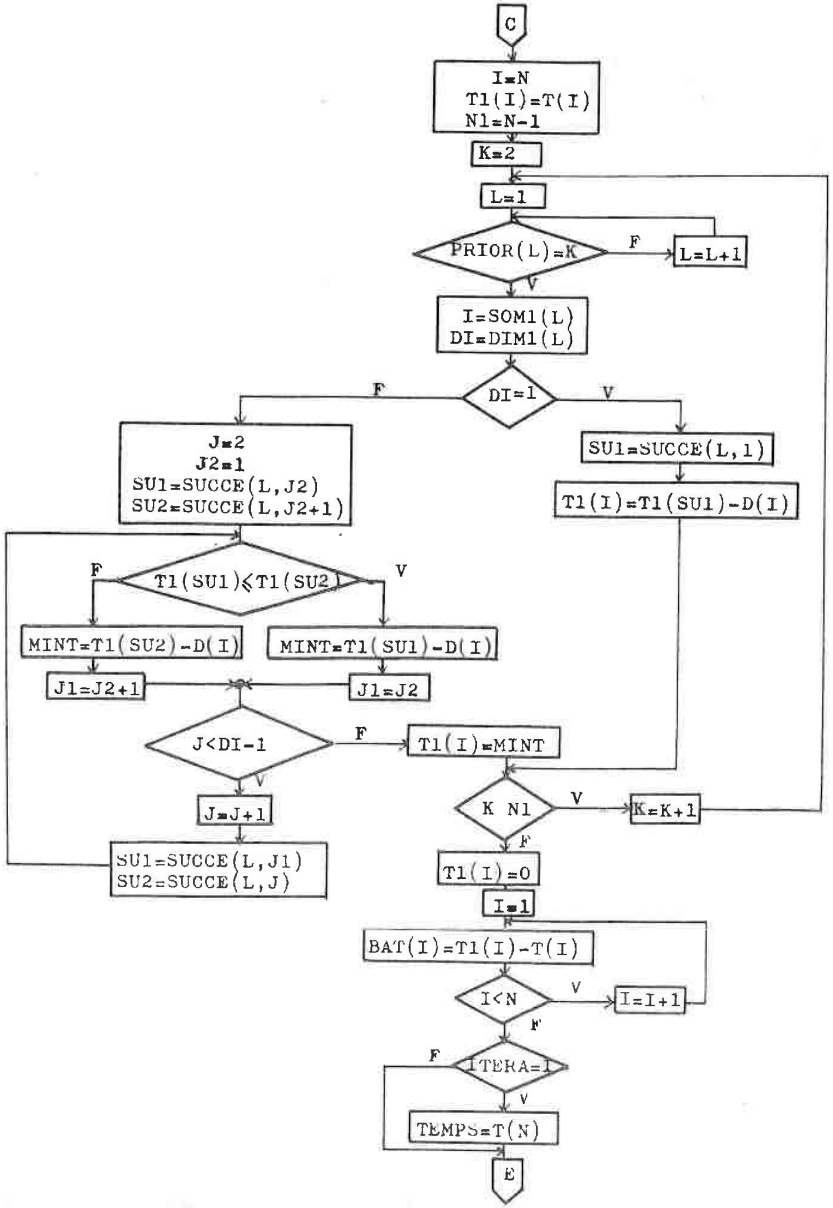
- CALCUL DU GRAPHE ETENDU
- CALCUL DES MATRICES DES PREDECESSEURS ET DES SUCCESSEURS



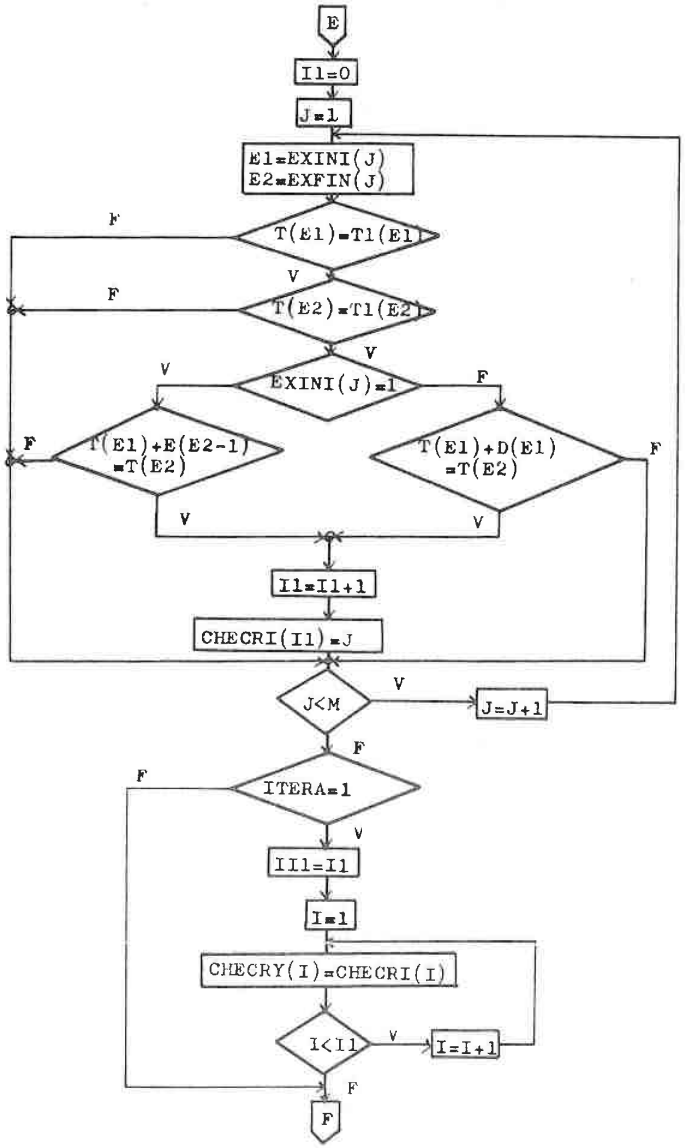
- CALCUL DES DATES AU PLUS TÔT



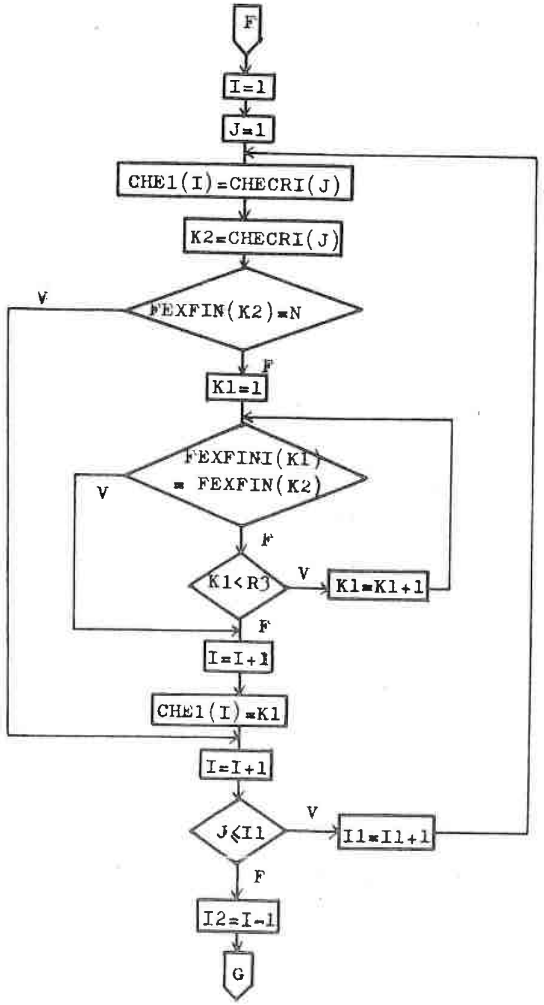
- CALCUL DES DATES AU PLUS TARD ET DES BATTEMENTS



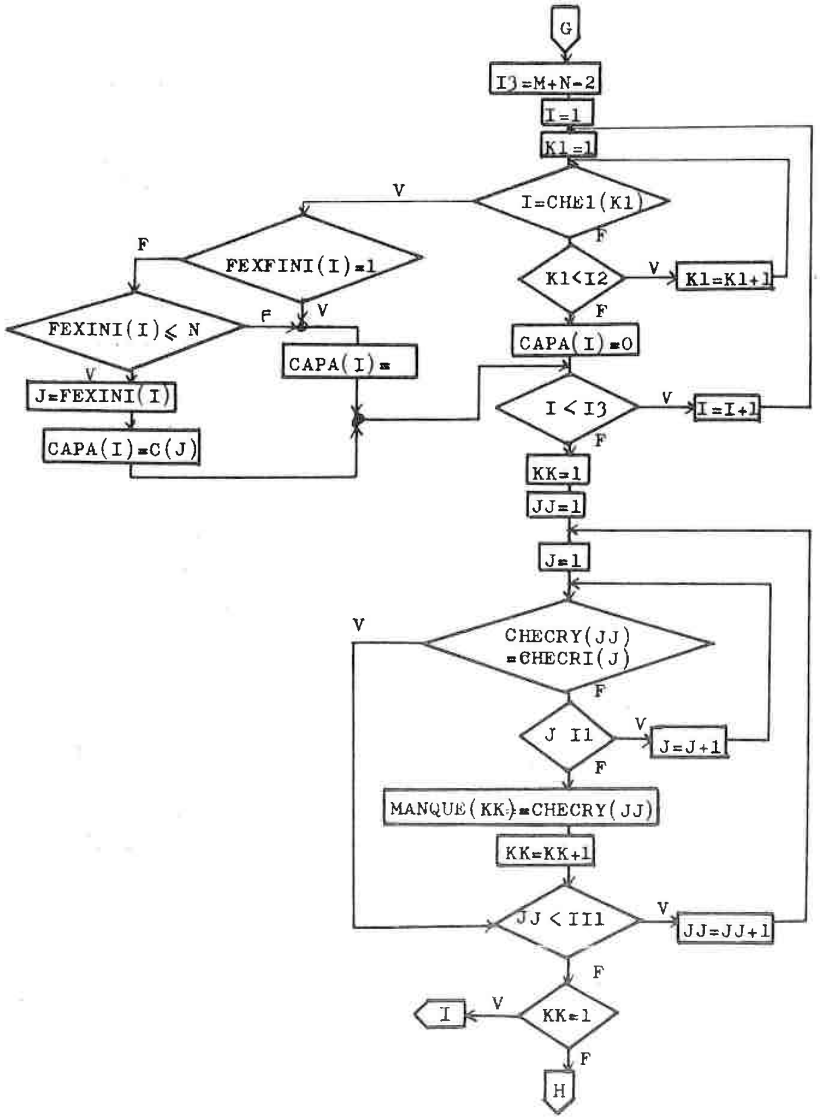
- CALCUL DU CHEMIN CRITIQUE

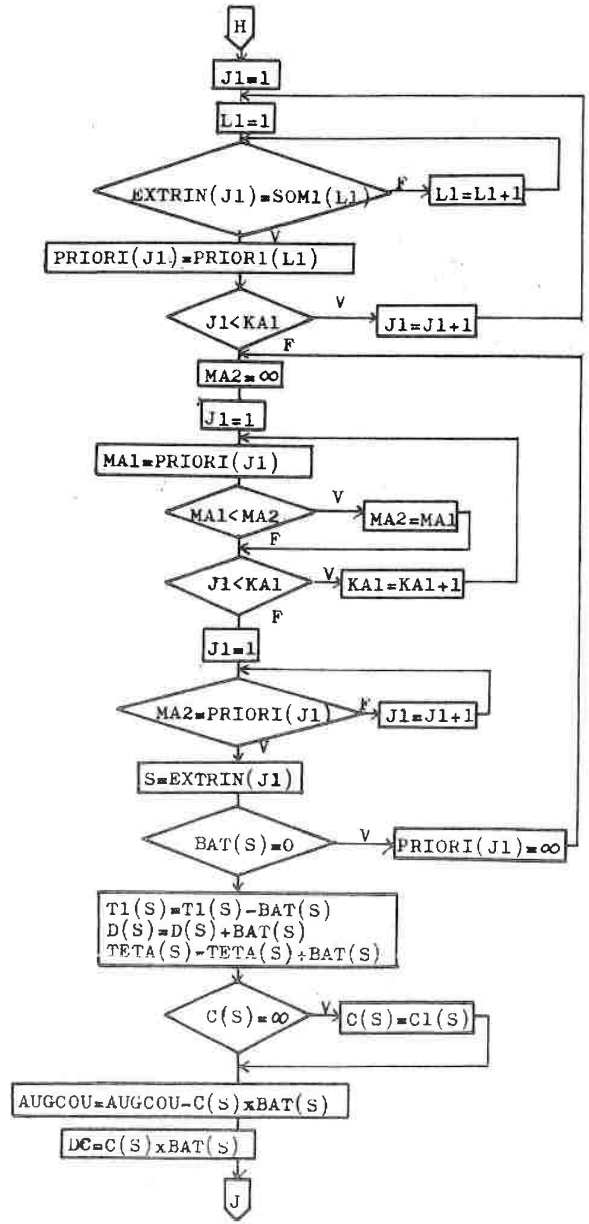


- CALCUL DU CHEMIN CRITIQUE ETENDU

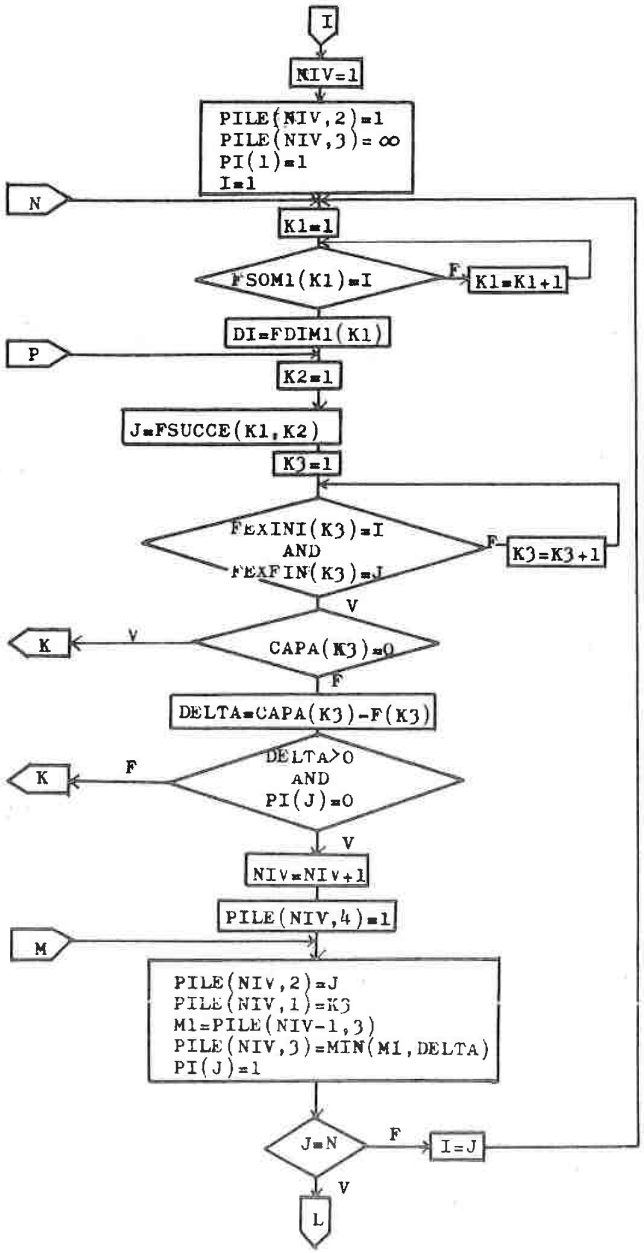


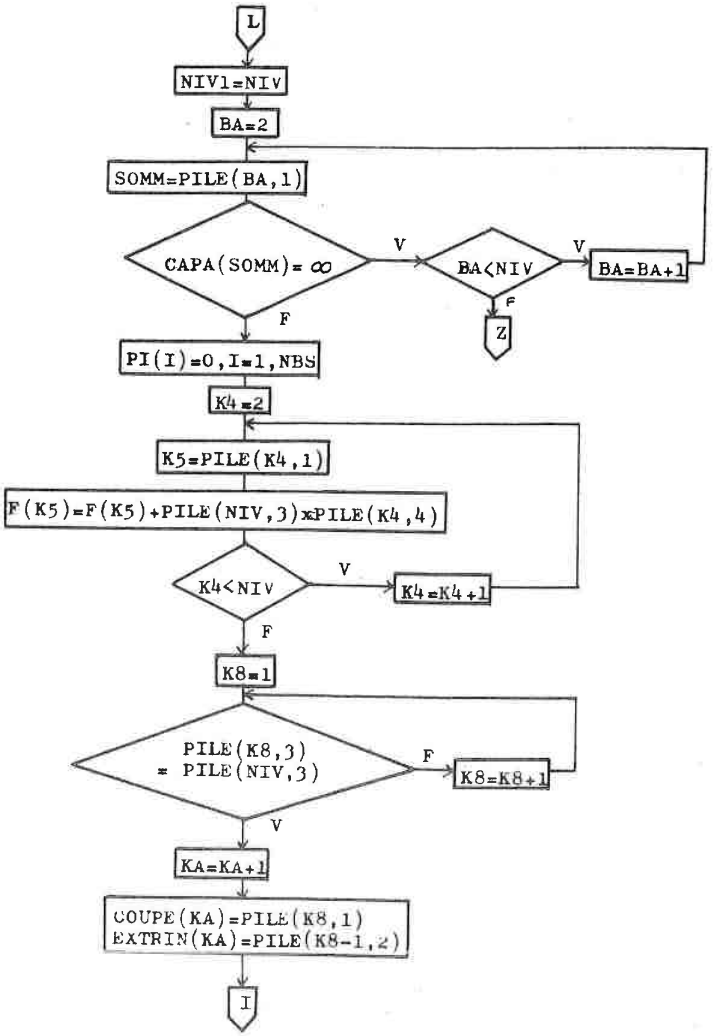
- ASSOCIATION DES CAPACITES
- COMPARAISON DU CHEMIN CRITIQUE PRECEDENT AU CHEMIN CRITIQUE ACTUEL.

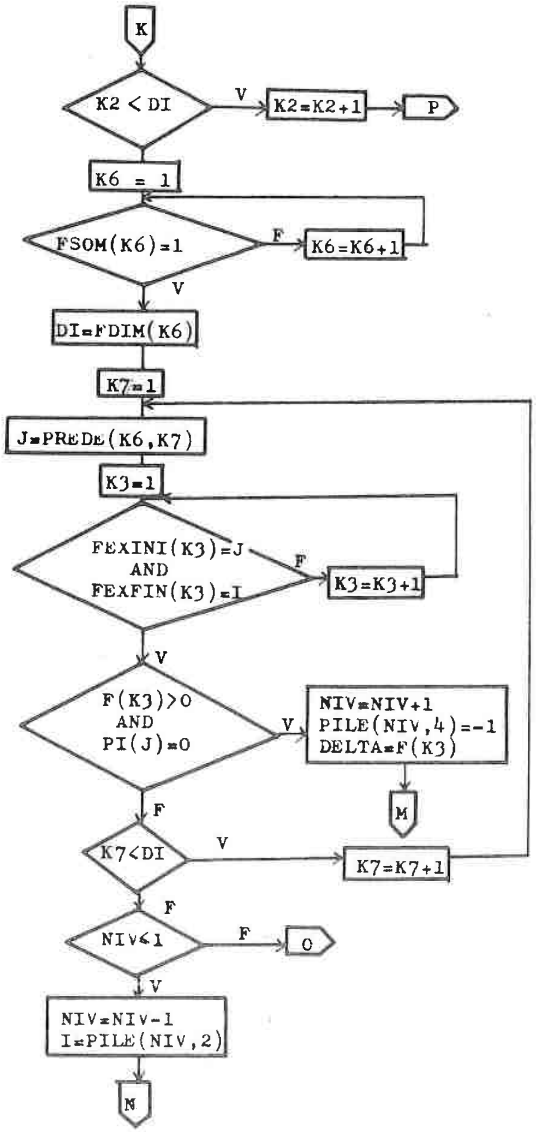




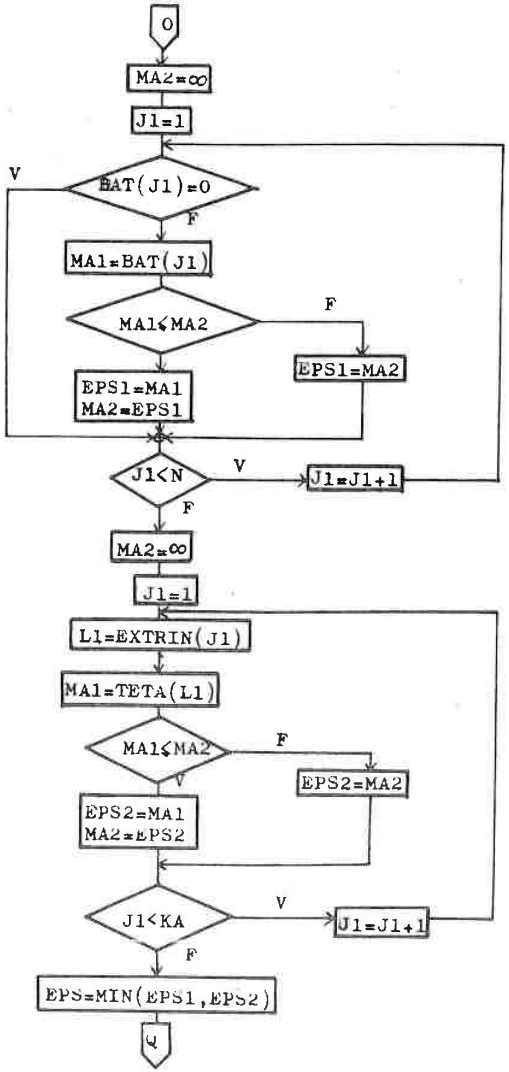
- RECHERCHE DU FLOT MAXIMUM ET DE LA COUPE MINIMALE



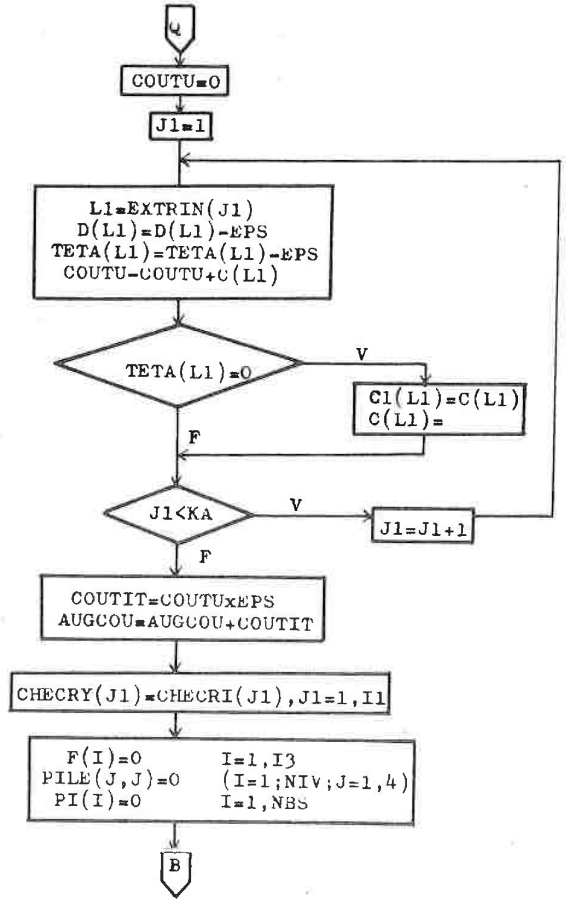

L
NIV1=NIV
BA=2
SOMM=PILE(BA,1)
CAPA(SOMM)= ∞
V
BA<NIV
V
BA=BA+1
F
Z
F
PI(I)=0,I=1,NBS
K4=2
K5=PILE(K4,1)
F(K5)=F(K5)+PILE(NIV,3)xPILE(K4,4)
K4<NIV
V
K4=K4+1
F
K8=1
PILE(K8,3)
= PILE(NIV,3)
F
K8=K8+1
V
KA=KA+1
COUPE(KA)=PILE(K8,1)
EXTRIN(KA)=PILE(K8-1,2)
I


K
K2 < DI
V
K2=K2+1
P
K6 = 1
FSOM(K6)=1
F
K6=K6+1
V
DI=FDIM(K6)
K7=1
J=PREDE(K6,K7)
K3=1
FEXINI(K3)=J
AND
FEXFIN(K3)=I
F
K3=K3+1
V
F(K3)>0
AND
PI(J)=0
V
NIV=NIV+1
PILE(NIV,4)=-1
DELTA=F(K3)
M
F
K7<DI
V
K7=K7+1
F
NIV<1
F
O
V
NIV=NIV-1
I=PILE(NIV,2)
N

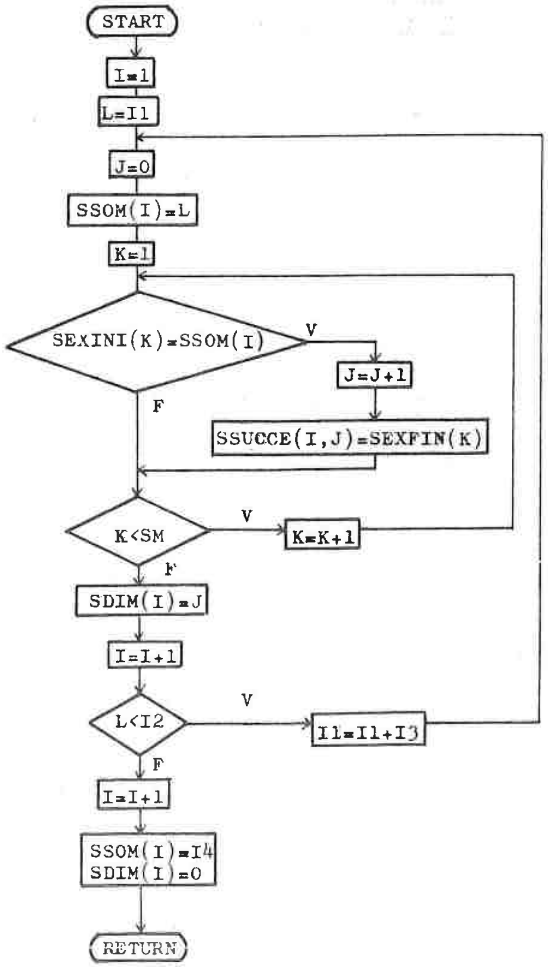
- CALCUL DU MINIMUM DES BATTEMENTS SUR TOUT LE GRAPHE
- CALCUL DU MINIMUM DES DIMINUTIONS DE DUREE SUR LA COUPE



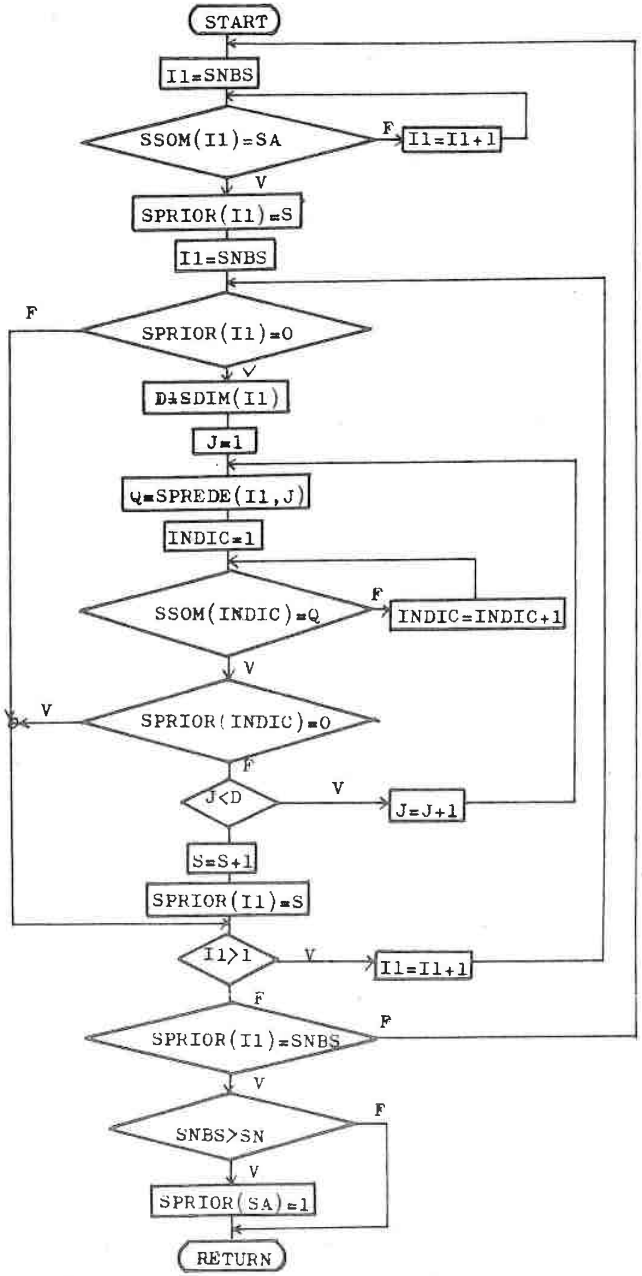
- CALCUL DES DIMINUTIONS CORRESPONDANTES ET DE L'AUGMENTATION DE COUT.
- STOCKAGE DU CHEMIN CRITIQUE POUR LA COMPARAISON DE L'ITERATION SUIVANTE.
- ANNULATION DES VECTEURS.



- SUBROUTINE PRESU(I1,I2,I3, SSOM, SEXINI, SSUCCE, SEXFIN, SDIM, I4, SM)



- SUBROUTINE ORDO (SPRIOR, SDIM, SPREDE, SSOM, SN, SNBS, SA)



5. Listing.


```
@RUN JENN01,S04003,S04003,S30,20

@FOR,ISY  ,CHARL
FOR S11A-10/30/75-16:42:39          CHARL
FORTRAN 11A INTER-PROJETS 1100 (ORSAY)

   MAIN PROGRAM

   STORAGE USED: CODE(1) 002131; DATA(0) 007322; BLANK COMMON(2)
                                                        000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   PRESU
    0004   ORDO
    0005   NINTR$
    0006   NRDU$
    0007   NT02$
    0010   NT01$
    0011   NWDU$
    0012   NSTOP$


   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)


 1*
 2*     C        DECLARATIONS
 3*
 4*           IMPLICIT INTEGER (A-Z)
 5*           DIMENSION PRIOR(50),DIM(50),PREDE(50,6),
 6*          *PRIOR1(50),DIM1(50),SUCCE(50,6),EXINI(50),
 7*          *EXFIN(50),T(50),T1(50),D(50),SOM(50),SOM1
 8*          *(50),CHECRI(100),E(10),TETA(50),C(50),BAT(50)
 9*          *,CAPA(50),C1(50),PILE(50,4),PI(50),F(50),
10*          *COUPE(50),CHECRY(50),MANQUE(50),EXTRIN(50),
11*          *CHE1(50),FEXINI(50),FEXFIN(50),PRIORI(50),
12*          *FSOM(50),FPREDE(50,6),FDIM(50),FSOM1(50),
13*          *FSUCCE(50,6),FDIM1(50),FPRIOR(50),FRIOR1
14*          *(50),DPRIOR(50),DDIM(50),DSOM(50),DPREDE
15*          *(50,6),OGCOU(50),TEMFIN(50)
16*
17*     C        LECTURE DES DONNEES
18*
19*           READ(5,50)TT,N,M,R
20*        50 FORMAT(4I5)
21*           READ(5,51) (E(I),I=1,R)
22*        51 FORMAT(4I5)
23*           DO 2 I=1,M
24*         2 READ(5,52) EXINI(I),EXFIN(I)
25*        52 FORMAT(2I5)
26*           N3=N-1
27*           DO 3 I=2,N3
28*         3 READ(5,53) D(I),TETA(I),C(I)
29*        53 FORMAT(3I5)
30*
31*     C        CALCUL DES MATRICES DES
32*     C        PREDECESSEURS ET DES SUCCESSEURS
33*
34*           CALL PRESU(N,1,-1,SOM,EXFIN,PREDE,EXINI,DIM,1,M)
```

```
      CALL PRESU(1,N,1,SOM1,EXINI,SUCCE,EXFIN,DIM1,N,M)
      CALL ORDO(PRIOR,DIM,PREDE,SOM,N,N,1)
      CALL ORDO(PRIOR1,DIM1,SUCCE,SOM1,N,N,N)
      NBS=2*N-2
      I3=M+N-2
      AUGCOU=0
      ITERA=1

C        CALCUL DU GRAPHE ETENDU

      DO 34 I=1,R
      FEXINI(I)=EXINI(I)
   34 FEXFIN(I)=EXFIN(I)
      R1=R+1
      DO 35 I=R1,M
      FEXINI(I)=EXINI(I)+N-1
   35 FEXFIN(I)=EXFIN(I)
      R2=M+1
      R3=M+N-2
      DO 36 I=R2,R3
      FEXINI(I)=I-M+1
   36 FEXFIN(I)=FEXINI(I)+N-1
      CALL PRESU(NBS,1,-1,FSOM,FEXFIN,FPREDE,FEXINI,FDIM,1,I3)
      CALL PRESU(1,NBS,1,FSOM1,FEXINI,FSUCCE,FEXFIN,FDIM1,NBS,I3)
      NNBS=NBS
      DO 400 B=1,NNBS
      Z=NNBS-B+1
      DPRIOR(B)=FPRIOR(Z)
      DDIM(B)=FDIM(Z)
      DSOM(B)=FSOM(Z)
      DI=DDIM(B)
      DO 301 J=1,DI
  301 DPREDE(B,J)=FPREDE(Z,J)
  400 CONTINUE
      CALL ORDO(DPRIOR,DDIM,DPREDE,DSOM,N,NBS,1)
      DO 302 B=1,NNBS
      Z=NNBS-B+1
      FPRIOR(Z)=DPRIOR(B)
      FDIM(Z)=DDIM(B)
      FSOM(Z)=DSOM(B)
      DI=FDIM(Z)
      DO 303 J=1,DI
  303 FPREDE(Z,J)=DPREDE(B,J)
  302 CONTINUE
      CALL ORDO(FRIOR1,FDIM1,FSUCCE,FSOM1,N,NBS,N)

C        CALCUL DES TEMPS AU PLUS TOT,DES
C        TEMPS AU PLUS TARD ET DES BATTEMENTS

  155 KA=0
  156 I=1
      T(I)=0
      DO 4 K=2,N
      L=1
   90 IF(PRIOR(L)-K)5,6,5
    5 L=L+1
      GOTO 90
```



```
    6 I=SOM(L)
      DI=DIM(L)
      IF(DI-1)7,8,7
    8 PR1=PREDE(L,1)
      IF(PREDE(L,1)-1) 92,91,92
   91 T(I)=E(I-1)
      GO TO 4
   92 T(I)=T(PR1)+D(PR1)
    4 CONTINUE
      GO TO 9
    7 J2=1
      PR1=PREDE(L,J2)
      PR2=PREDE(L,J2+1)
      J=2
   15 IF(T(PR1)+D(PR1)-T(PR2)-D(PR2))10,11,11
   10 MAXT=T(PR2)+D(PR2)
      J1=J2+1
      GO TO 12
   11 MAXT=T(PR1)+D(PR1)
      J1=J2
   12 IF(J-DI+1)13,13,14
   13 J=J+1
      PR1=PREDE(L,J1)
      PR2=PREDE(L,J)
      GO TO 15
   14 T(I)=MAXT
      GOTO 4
    9 I=N
      T1(I)=T(I)
      N1=N-1
      DO 17 K=2,N1
      L=1
   93 IF(PRIOR1(L)-K)18,19,18
   18 L=L+1
      GOTO 93
   19 I=SOM1(L)
      DI=DIM1(L)
      IF(DI-1)20,21,20
   21 SU1=SUCCE(L,1)
      T1(I)=T1(SU1)-D(I)
   17 CONTINUE
      GO TO 22
   20 J=2
      J2=1
      SU1=SUCCE(L,J2)
      SU2=SUCCE(L,J2+1)
   28 IF(T1(SU1)-T1(SU2))23,23,24
   24 MINT=T1(SU2)-D(I)
      J1=J2+1
      GO TO 25
   23 MINT=T1(SU1)-D(I)
      J1=J2
   25 IF(J-DI+1)26,27,27
   26 J=J+1
      SU1=SUCCE(L,J1)
      SU2=SUCCE(L,J)
      GO TO 28
```

```
149*        27 T1(I)=9INT
150*           GOTO 17
151*        22 T1(1)=0
152*           DO 66 I=1,N
153*        66 BAT(I)=T1(I)-T(I)
154*           TEMFIN(ITERA)=T(N)
155*           IF(ITERA.EQ.1) TEMPS=T(N)
156*
157*     C     .  CALCUL DU CHEMIN CRITIQUE
158*
159*           I1=0
160*           DO 30 J=1,M
161*           E1=EXINI(J)
162*           E2=EXFIN(J)
163*           IF(T(E1)-T1(E1))30,31,30
164*        31 IF(T(E2)-T1(E2))30,32,30
165*        32 IF (FXINI(J)-1) 110,109,110
166*       109 IF (T(E1)+E(E2-1)-T(E2)) 30,33,30
167*       110 IF(T(E1)+D(E1)-T(E2))30,33,30
168*        33 I1=I1+1
169*           CHECRI(I1)=J
170*        30 CONTINUE
171*           IF(ITERA-1) 29,16,29
172*        16 II1=I1
173*           DO 1 I=1,I1
174*         1 CHECRY(I)=CHECRI(I)
175*
176*     C        CALCUL DU CHEMIN CRITIQUE ETENDU
177*
178*        29 I=1
179*           DO 40 J=1,I1
180*           CHE1(I)=CHECRI(J)
181*           K2=CHECRI(J)
182*           IF(FEXFIN(K2)-N) 38,39,38
183*        38 DO 450 K1=1,R3
184*           IF(FEXINI(K1)-FEXFIN(K2)) 450,37,450
185*       450 CONTINUE
186*        37 I=I+1
187*           CHE1(I)=K1
188*        39 I=I+1
189*        40 CONTINUE
190*           I2=I-1
191*
192*     C        ASSOCIATIONS DES CAPACITES
193*
194*           I3=M+N-2
195*           DO 97 I=1,I3
196*           K1=1
197*       102 IF(I-CHE1(K1)) 98,99,98
198*        98 IF(K1-I2) 100,101,101
199*       100 K1=K1+1
200*           GOTO 102
201*       101 CAPA(I)=0
202*           GOTO 97
203*        99 IF(FEXINI(I)-1) 103,104,103
204*       104 CAPA(I)=9999
205*           GOTO 97
```



```
206*       103 IF(FEXINI(I)-N) 105,105,104
207*       105 J=FEXINI(I)
208*           CAPA(I)=C(J)
209*        97 CONTINUE
210*
211*     C        COMPARAISON DU CHEMIN CRITIQUE PRECEDENT
212*     C        AU CHEMIN CRITIQUE ACTUEL
213*
214*       168 KK=0
215*           DO 140 JJ=1,II1
216*           DO 141 J=1,I1
217*           IF(CHECRY(JJ)-CHECRI(J)) 141,140,141
218*       141 CONTINUE
219*           KK=KK+1
220*           MANQUE(KK)=CHECRY(JJ)
221*       140 CONTINUE
222*           KK1=KK-1
223*           IF(KK) 143,3000,143
224*
225*     C        REAJUSTEMENT DU CHEMIN CRITIQUE ACTUEL
226*
227*       143 DO 160 J1=1,KA1
228*           L1=1
229*       163 IF(EXTRIN(J1)-SOM1(L1)) 161,162,161
230*       161 L1=L1+1
231*           GOTO 163
232*       162 PRIORI(J1)=PRIOR1(L1)
233*       160 CONTINUE
234*       159 MA2=9999
235*           DO 164 J1=1,KA1
236*           MA1=PRIORI(J1)
237*           IF(MA1.LT.MA2) MA2=MA1
238*       164 CONTINUE
239*           J1=1
240*       167 IF(MA2-PRIORI(J1)) 165,166,165
241*       165 J1=J1+1
242*           GOTO 167
243*       166 S=EXTRIN(J1)
244*           IF(BAT(S)) 157,158,157
245*       158 PRIORI(J1)=9999
246*           GOTO 159
247*       157 T1(S)=T1(S)-BAT(S)
248*           D(S)=D(S)+BAT(S)
249*           TETA(S)=TETA(S)+BAT(S)
250*           IF(C(S).EQ.9999) C(S)=C1(S)
251*           AUGCOU=AUGCOU-C(S)*BAT(S)
252*           OGCOU(ITERA)=AUGCOU
253*           DC=C(S)*BAT(S)
254*           GOTO 156
255*
256*     C        RECHERCHE DU FLOT MAXIMUM ET
257*     C        DE LA COUPE MINIMALE
258*
259*      3000 ITERA=ITERA+1
260*       142 NIV=1
261*           PILE(NIV,2)=1
262*           PILE(NIV,3)=9999
```

```
      PI(1)=1
      I=1
  119 K1=1
  112 IF(FSOM1(K1)-I) 111,139,111
  111 K1=K1+1
      GOTO 112
  139 DI=FDIM1(K1)
      DO 113 K2=1,DI
      J=FSUCCE(K1,K2)
      K3=1
  131 IF(FEXINI(K3).NE.I.OR.FEXFIN(K3).NE.J) GOTO 129
      GOTO 130
  129 K3=K3+1
      GOTO 131
  130 IF(CAPA(K3)) 114,113,114
  113 CONTINUE
      GOTO 115
  114 DELTA=CAPA(K3)-F(K3)
      IF((DELTA.LE.0).OR.(PI(J).NE.0)) GOTO 113
      NIV=NIV+1
      PILE(NIV,4)=1
  137 PILE(NIV,2)=J
      PILE(NIV,1)=K3
      M1=PILE(NIV-1,3)
      PILE(NIV,3)=MIN(M1,DELTA)
      PI(J)=1
      IF(J-N) 117,118,117
  117 I=J
      GOTO 119
  118 NIV1=NIV
      DO 994 BA=2,NIV1
      SOMM=PILE(BA,1)
      IF(CAPA(SOMM).EQ.9999) GOTO 994
      GOTO 993
  994 CONTINUE
      GOTO 1000
  993 DO 191 I=1,NBS
  191 PI(I)=0
      DO 120 K4=2,NIV
      K5=PILE(K4,1)
  120 F(K5)=F(K5)+PILE(NIV,3)*PILE(K4,4)
      K8=1
  123 IF(PILE(K8,3)-PILE(NIV,3)) 121,122,121
  121 K8=K8+1
      GOTO 123
  122 KA=KA+1
      COUPE(KA)=PILE(K8,1)
      EXTRIN(KA)=PILE(K8-1,2)
      GOTO 142
  115 K6=1
  127 IF(FSOM(K6)-I) 125,126,125
  125 K6=K6+1
      GOTO 127
  126 DI=FDIM(K6)
      DO 128 K7=1,DI
      J=FPREDE(K6,K7)
      K3=1
```



```
  134 IF(FEXINI(K3).NE.J.OR.FEXFIN(K3).NE.I) GOTO 132
      GOTO 133
  132 K3=K3+1
      GOTO 134
  133 IF(F(K3).LE.0.OR.PI(J).EQ.1) GOTO 128
      NIV=NIV+1
      PILE(NIV,4)=-1
      DELTA=F(K3)
      GOTO 137
  128 CONTINUE
      IF(NIV-1) 135,135,138
  138 NIV=NIV-1
      I=PILE(NIV,2)
      GOTO 119

C         CALCUL DU MINIMUM DES BATTEMENTS
C         SUR TOUT LE GRAPHE

  135 MA2=9999
      DO 144 J1=1,N
      IF(BAT(J1)) 145,144,145
  145 MA1=BAT(J1)
      IF(MA1-MA2) 148,148,149
  148 EPS1=MA1
      MA2=EPS1
      GOTO 144
  149 EPS1=MA2
  144 CONTINUE

C         CALCUL DU MINIMUM DES DIMINUTIONS
C         DE DUREE SUR LA COUPE

      MA2=9999
      DO 150 J1=1,KA
      L1=EXTRIN(J1)
      MA1=TETA(L1)
      IF(MA1-MA2) 151,151,152
  151 EPS2=MA1
      MA2=EPS2
      GOTO 150
  152 EPS2=MA2
  150 CONTINUE
      EPS=MIN(EPS1,EPS2)

C         CALCUL DES DIMINUTIONS CORRESPONDANTES
C         ET DE L'AUGMENTATION DE COUT

      COUTU=0
      DO 153 J1=1,KA
      L1=EXTRIN(J1)
      D(L1)=D(L1)-EPS
      TETA(L1)=TETA(L1)-EPS
      COUTU=COUTU+C(L1)
      IF(TETA(L1)) 153,300,153
  300 C1(L1)=C(L1)
      C(L1)=9999
  153 CONTINUE
```

```
377*        COUTIT=COUTU*EPS
378*        AUGCOU=AUGCOU+COUTIT
379*        OGCOU(ITERA)=AUGCOU
380*
381* C          STOCKAGE DU CHEMIN CRITIQUE POUR LA
382* C          COMPARAISON DE L'ITERATION SUIVANTE
383*
384*        III=I1
385*        DO 154 J1=1,T1
386*    154 CHECRY(J1)=CHECRI(J1)
387*        KA1=KA
388*        DO 999 I=1,I3
389*    999 F(I)=0
390*        DO 998 I=1,NIV
391*        DO 997 J=1,4
392*    997 PILE(I,J)=0
393*    998 CONTINUE
394*        DO 996 I=1,NBS
395*    996 PI(I)=0
396*        GOTO 155
397*   1000 WRITE(6,2000)
398*   2000 FORMAT(1H1,///,17X,'A',9X,'LA DUREE EST',4X,'POUR UNE AUGMENTATION
399*       *',/,12X,'L''ITERATION',4X,'DIMINUEE A',9X,'DE COUT DE',//)
400*        ITERI=ITERA-1
401*        DO 2001 I=1,ITERI
402*   2001 WRITE(6,2002) I,TEMFIN(I),OGCOU(I)
403*   2002 FORMAT(5X,3I15)
404*        WRITE(6,990) TEMPS,T(N),AUGCOU
405*    990 FORMAT(///,15X,'LA DUREE DES TRAVAUX PASSE DE',I3,'A',I3,'SEMAINES
406*       *',/,15X,'POUR UNE AUGMENTATION DE PRIX DE',I3,'UNITES')
407*        STOP
408*        END



        END OF COMPILATION:        NO  DIAGNOSTICS.
```



```
@FOR,IS CHAR1
FOR S11A-10/30/75-16:43:00 (,0)
FORTRAN 11A INTER-PROJETS 1100 (ORSAY)


   SUBROUTINE PRESU     ENTRY POINT 000131

   STORAGE USED: CODE(1) 000155; DATA(0) 000044; BLANK COMMON(2)
                                                        000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0001   000033 1076       0001   000051 1146       0001   000064 2L
0000 I 000002 J          0000 I 000003 K          0000 I 000001 L


0000 I 000000 I          0000   000004 INJP$
```

```
 1*     SUBROUTINE PRESU(I1,I2,I3,SSOM,SEXINI,SSUCCE,SEXFIN,SDIM,I4,SM)
 2*     INTEGER SSOM,SEXINI,SEXFIN,SSUCCE,SM,I4,SDIM
 3*     DIMENSION SSOM(50),SEXINI(50),SEXFIN(50),SSUCCE(50,6),SDIM(50)
 4*     I=1
 5*     DO 1 L=I1,I2,I3
 6*     J=0
 7*     SSOM(I)=L
 8*     DO 2 K=1,SM
 9*     IF(SEXINI(K)-SSOM(I))2,3,2
10*   3 J=J+1
11*     SSUCCE(I,J)=SEXFIN(K)
12*   2 CONTINUE
13*     SDIM(I)=J
14*     I=I+1
15*   1 CONTINUE
16*     I=I+1
17*     SSOM(I)=I4
18*     SDIM(I)=0
19*     RETURN
20*     END


   END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS CHAR2
FOR S11A-10/30/75-16:43:02 (.0)
FORTRAN 11A INTER-PROJETS 1100 (ORSAY)

    SUBROUTINE ORDO      ENTRY POINT 000162

    STORAGE USED: CODE(1) 000212; DATA(0) 000032; BLANK COMMON(2)
                                                          000000
    EXTERNAL REFERENCES (BLOCK, NAME)

     0003   NERR3$

    STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0001   000031 1L        0001   000016 10L       0001   000051 116G
0001   000117 4L        0001   000104 7L        0001   000073 9L
0000   000006 INJP$     0000 I 000001 I1        0000 I 000003 J

0001   000065 125G      0001   000020 3L
0000 I 000002 D         0000 I 000005 INDIC
0000 I 000004 Q         0000 I 000000 S
```

```
 1*         SUBROUTINE ORDO(SPRIOR,SDIM,SPREDE,SSOM,SN,SNBS,SA)
 2*         IMPLICIT INTEGER (A-Z)
 3*         DIMENSION SPRIOR(50),SDIM(50),SPREDE(50,6),SSOM(50)
 4*         S=1
 5*        10 I1=SNBS
 6*         3 IF(SSOM(I1)-SA) 2,1,2
 7*         2 I1=I1-1
 8*           GOTO 3
 9*         1 SPRIOR(I1)=S
10*           DO 4 I1=SNBS,1,-1
11*           IF(SPRIOR(I1)) 4,5,4
12*         5 D=SDIM(I1)
13*           DO 6 J=1,D
14*           Q=SPREDE(I1,J)
15*           INDIC=1
16*         9 IF(SSOM(INDIC)-Q) 8,7,8
17*         8 INDIC=INDIC+1
18*           GOTO 9
19*         7 IF(SPRIOR(INDIC)) 6,4,6
20*         6 CONTINUE
21*           S=S+1
22*           SPRIOR(I1)=S
23*         4 CONTINUE
24*           IF(SPRIOR(I1).NE.SNBS) GOTO 10
25*           IF(SNBS.GT.SN) SPRIOR(SA)=1
26*           RETURN
27*           END

        END OF COMPILATION:        NO  DIAGNOSTICS.

 @XQT
```



```
MAP 26.2-10/30-16:43

ADDRESS LIMITS     001000 015023      6164 IBANK WORDS DECIMAL
                   040000 054216      6287 DBANK WORDS DECIMAL
STARTING ADDRESS   012673

          SEGMENT $MAIN$          001000 015023      040000 054216

NSWTC$/FOR69              $(1)  001000 001024
NRBLK$/FOR68              $(1)  001025 001047
NRWND$/V1273              $(1)  001050 001134      $(2)  040000 040012
NWEF$/FOR69               $(1)  001135 001340      $(2)  040013 040032
NBDCV$/FOR64              $(1)  001341 001466      $(2)  040033 040075
NFTCH$/FOR69              $(1)  001467 001751      $(2)  040076 040111
NFTV$/FOR                 $(1)  001752 001774
NCNVT$/FOR68              $(1)  001775 002216      $(2)  040112 040206
NCLOS$/V1273              $(1)  002217 002407      $(2)  040207 040237
NWBLK$/FOR68              $(1)  002410 002521
NBSBL$/FOR68              $(1)  002522 002562
NUPDA$/FOR68              $(1)  002563 002616
NBF00$/FOR                                         $(2)  040240 042441
NOTIN$/FOR68              $(1)  002617 003113      $(2)  042442 042445
NOUT$/FOR69               $(1)  003114 004270      $(2)  042446 042504
NIOER$/V1273              $(1)  004271 004476      $(2)  042505 042646
NININ$/FOR68              $(1)  004477 004667      $(2)  042647 042652
NINPT$/FOR69              $(1)  004670 005676      $(2)  042653 042703
NFMT$/FOR69               $(1)  005677 006553      $(2)  042704 042760
NFCHK$/FOR69              $(1)  006554 007541      $(2)  042761 043134
                                                   $(4)  043135 043206
NTAB$/V0973                                        $(2)  043207 043245
CRELAD$/SYS70             $(1)  007542 010356      $(2)  043246 043715
NERR$/V0675               $(1)  010357 010774      $(2)  043716 044114
NSTOP$/FOR69              $(1)  010775 011021      $(2)  044115 044124
NOBUF$/FOR68              $(1)  011022 011062
NIER$/FOR69               $(1)  011063 011244      $(2)  044125 044245
NIBUF$/FOR68              $(1)  011245 011304      $(2)  044246 044246
NINTR$/V0375              $(1)  011305 012303      $(0)  044247 044576
BLANK$COMMON(COMMONBLOCK)
CHAR2                     $(1)  012304 012515      $(0)  044577 044630
                                                   $(2)  BLANK$COMMON
CHAR1                     $(1)  012516 012672      $(0)  044631 044674
                                                   $(2)  BLANK$COMMON
CHARL                     $(1)  012673 015023      $(0)  044675 054216
                                                   $(2)  BLANK$COMMON

SYS$*RLIB$, LEVEL 70-1
END OF COLLECTION - TIME 1.299 SECONDS
```

A L'ITERATION	LA DUREE EST DIMINUEE A	POUR UNE AUGMENTATION DE COUT DE
1	54	0
2	53	3
3	52	8
4	49	26
5	48	35
6	47	45
7	46	55
8	45	65
9	44	79

LA DUREE DES TRAVAUX PASSE DE 54A 44SEMAINES
POUR UNE AUGMENTATION DE PRIX DE 79UNITES

@FIN

```
RUNID: JENN01   ACCT: S04003          PROJECT: S04003
TIME:    TOTAL: 00:00:20.666
         CAU:   00:00:05.167      I/O:  00:00:06.050
         CC/ER: 00:00:09.448      WAIT: 00:00:00.050
SRC:     PS --  6777769           FS -- 1490329
IMAGES  READ:   493       PAGES:  14
START:    16:42:36 OCT 30,1975    FIN:  16:43:22 OCT 30,1975
```


Dans la littérature les problèmes de PERT classiques donnent des éditions extrèmement poussées. Notre tableau des résultats est un tableau de présentation des coûts en fonction des durées.
Il est certain que l'on pourrait éditer les stratégies correspondant à certains choix, mais qu'il paraît préférable de recourir à un problème de bibliothèque pour lequel la partie édition a été beaucoup plus travaillée.

IV. CONCLUSION

Le programme a été mis au point sur un terminal ORDOPROCESSEUR relié à un ordinateur UNIVAC 1110. Sur cet ordinateur, l'utilisateur a accès à 2 x 64 K mots en général, et à 64 K mots lorsque le langage utilisé est le FORTRAN.

Notre programme occupe 12 451 mots dont 6 287 pour les tableaux, quand il est prévu pour un ordonnancement de 50 tâches.
Nous pouvons ainsi prévoir qu'un ordonnancement à 800 tâches ne sature pas encore la mémoire.

En ce qui concerne le temps de calcul, la résolution de l'ordonnancement à 14 tâches qui est l'exemple traité, demande 5 secondes d'unité centrale. Mais ce temps n'étant pas proportionnel au nombre de tâches, nous ne nous prononcerons pas quant au temps de calcul éventuel, dans le cas d'un exemple plus important.

BIBLIOGRAPHIE

- ALGAN M.,SIMMONARD M. : Principes d'une méthode d'exploration de certains domaines, et application à l'ordonnancement de la construction des grands ensembles.
- BALAS E. :Finding a minimaximal path in a disjunctive PERT network. Théorie des graphes (Dunod, 1967).
- BATTERSBY A. : Méthodes modernes d'ordonnancement (Dunod, 1967)
- BERTAUX M. : Expérience d'application des méthodes à chemin critique ORPRO et PERT (Revue du C.N.O.F. 1967).
- FORD L.R., FULKERSON D.R. : Flows in networks (PRINCETON University, 1962).
- FULKERSON D.R. : Expected critical path lengths in PERT netwoks (Operations Research, 1962).
- KAUFMANN A., DESBAZEILLE G. : La méthode du chemin critique (Dunod, 1966).
- LEGRAS J. : Le PERT-COUT (Cours de D.E.A. 1975).
- MODER J.J. , PHILLIPS C.R. : Project management with CPM and PERT (Reinbold Publishing 1964).
- NEMETI L. : Sur le problème d'ordonnancement dans la fabrication en série (R.I.R.O.,1968).
- ROY B. : Contribution de la théorie des graphes à l'étude des problèmes d'ordonnancement.
- ROY B., ROSINSKI J. : Problèmes d'ordonnancement, applications et méthodes.