MASTER EN SCIENCES INFORMATIQUES

Evaluation et comparaison de systèmes complexes
Chois d'un ordinateur par la méthode CAT

Arnotte, Jean-Philippe

Award date:
1977

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX
NAMUR (BELGIQUE)

INSTITUT D'INFORMATIQUE

EVALUATION ET COMPARAISON DE SYSTEMES COMPLEXES

Choix d'un ordinateur par la méthode CAT

Directeur : M. Ph. VAN BASTELAER

Jean-Philippe ARNOTTE
Mémoire présenté en vue
d'obtenir le grade de
Licencié et Maître en Informatique

Tome 1 : Mémoire

- 1977 -

" Qu ' est - ce que l' homme ? A quoi sert - il ?
Quel est son bien et quel est son mal ?
La durée de sa vie ? Cent ans tout au plus .
Une goutte d ' eau tirée de la mer , un grain de sable ,
telles sont ces quelques années auprès de
l ' Eternité . "

Extrait de la Bible
Eccl. , 1 , 18 , versets 6 à 10

A mon père

Remerciements

Qu'il nous soit permis ici d'exprimer nos plus vifs remerciements à toutes les personnes qui nous ont aidé à mener notre travail à bien par leur direction , leurs conseils et leur aide :

Messieurs J.J. DUJMOVIĆ (Directeur de stage) , Ph. Van BASTELAER (Directeur du mémoire) , J. FICHEFET et J. RAMAEKERS (Professeurs de l'Institut d' Informatique) , T. TOŠIĆ (Directeur de l' Informatique de la Société PROGRES) , D. SPASIĆ (Directeur du développement du Centre PIRC) , ainsi que Madame E. ARNOTTE-OGER qui a dactylographié le présent mémoire .

Citons également , en reconnaissance pour leur excellent accueil Messieurs A. ALEKSIĆ , B. LAZIĆ et D. VELAŠEVIĆ (Professeurs de la Faculté d'Electrotechnique de l'Université de Belgrade, Yougoslavie) les Sociétés PROGRES et PRVI MAJ de Pirot (Yougoslavie) , ainsi que mes nombreux amis de la Faculté d'Electrotechnique et des homes d'étudiants ' Ivo - Lola Ribar ' , ' Patrice Lumumba ' et ' Slobodan Penezić ' .

P R E F A C E

" L'évolution tend à complexifier ".

Cet aphorisme du R.P. Teilhard de Chardin est bien d'actualité en cette fin du 20e siècle : il n'est que citer le prodigieux développement de la science , de la technique et de l'informatique, qui ont mené notre époque à sa fantastique complexité .

Tandis que , des Pharaons à Léonard de Vinci , un érudit pouvait prétendre à l'ensemble des connaissances humaines de son temps , de nos jours , il est devenu pratiquement impossible de dominer l'ensemble d'une seule discipline .

De plus , au sein d'une même discipline , le jugement est parfois rendu malaisé par la complexité de l'objet examiné : ainsi en est-il en informatique du choix d'un ordinateur .

Plus généralement , on se heurte au problème d'évaluer et / ou de comparer des systèmes complexes .

L'objectif du présent mémoire est d'apporter une modeste contribution à la solution de ce type de problèmes .

A cette fin , nous présentons ci-après la méthode CAT et son outil informatique le système SEL .

Namur , juillet 1977 Jean-Philippe Arnotte

C H A P I T R E 1

PLAN

Le présent mémoire comporte deux parties : le mémoire proprement dit et son annexe .

Le mémoire proprement dit comprend l'essentiel du travail tandis que l'annexe comporte la partie technique destinée au praticien.

Tous deux commencent par le plan de leur contenu .

Le plan du mémoire proprement dit se fonde sur la progression suivante : après le plan lui-même , nous présentons le problème et les modèles proposés , puis un modèle particulier , la méthode CAT . Nous passons alors à son outil informatique le système SEL , puis à deux exemples d'utilisation de CAT et de SEL . Enfin , nous comparons CAT aux autres modèles en général , puis point par point à la méthode multicritère Electre 1 . A ce moment , nous présentons notre conclusion générale :

(1) Ce paragraphe comporte un exemple d'utilisation de CAT et Electre 1 pour un même problème .

Note . Chaque source citée dans la bibliographie est précédée d'un numéro qui sert à la référencer ; ce numéro est alors exprimé entre crochets [ ] .

Les principales abréviations utilisées dans ce mémoire sont :

AP	pour	absorption partielle
AT	pour	absorption totale
CAT	pour	criteria aggregation technique
CE	pour	critère(s) élémentaire(s)
CEM	pour	courbe(s) d'efficacité maximale
CNE	pour	critère(s) non-élémentaire(s)
CIS	pour	critère(s) immédiatement subordonné(s)
CPP	pour	critère(s) à plusieurs possibilités
SEL	pour	systems evaluation language
UM	pour	unité(s) monétaire(s)

C H A P I T R E 2

L E P R O B L E M E

2.1 / Le problème

Problème Général

Le présent mémoire s'intéresse au problème de l'évaluation et de la comparaison de systèmes complexes .

Précisons tout d'abord le sens que nous donnons ici aux termes de cette phrase :

Par ' système ' on entend un ensemble d'éléments en relations . Cette définition est volontairement générale ; nous ne nous étendrons pas ici sur les différentes définitions d'un système [41 , 45].

Par ' complexe ' on entend que le système comporte un grand nombre d'éléments et / ou de relations.

Ce nombre doit être tel que l'évaluation et la comparaison - définies ci-dessous - ne peuvent se faire mentalement. D'où la nécessité d'un modèle .

Par ' évaluation ' on exprime le degré de similitude d'un système avec un autre système, dit ' système de référence ' . Par exemple, une mesure - qui se réfère à une unité, ou une cote d'examen - qui se réfère à une cote maximale. Le résultat de l'évaluation est une grandeur .

Par ' comparaison ' on exprime une relation d'infériorité, d'égalité, de supériorité, de préférence ou d'indifférence entre deux ou plusieurs systèmes .
La comparaison ne s'applique directement qu'à des objets de même nature, par exemple des nombres .

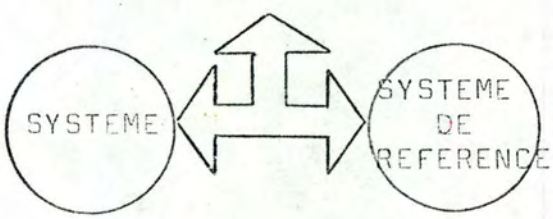
Pour pouvoir comparer deux objets, il est souvent nécessaire de les évaluer d'abord par rapport à un système de référence ; on obtient par exemple deux mesures qui, elles, sont directement comparables . (Fig. 1) .

FIGURE 1

Schémas conceptuels de l'évaluation et de la comparaison .

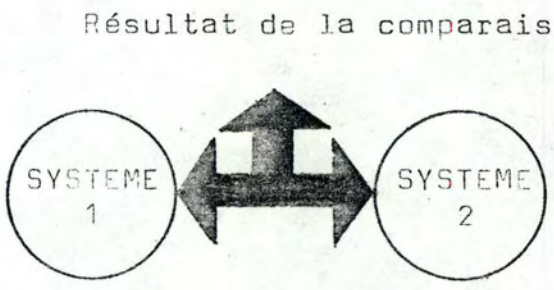
Note : L'opérateur à 3 flèches dénote une confrontation qui amène un résultat . Par souci de clarté, cet opérateur est dessiné en blanc pour une évaluation, et en noir pour une comparaison .

EVALUATION D'UN SYSTEME



Le résultat de l'évaluation est une grandeur (mesure, cote ...)

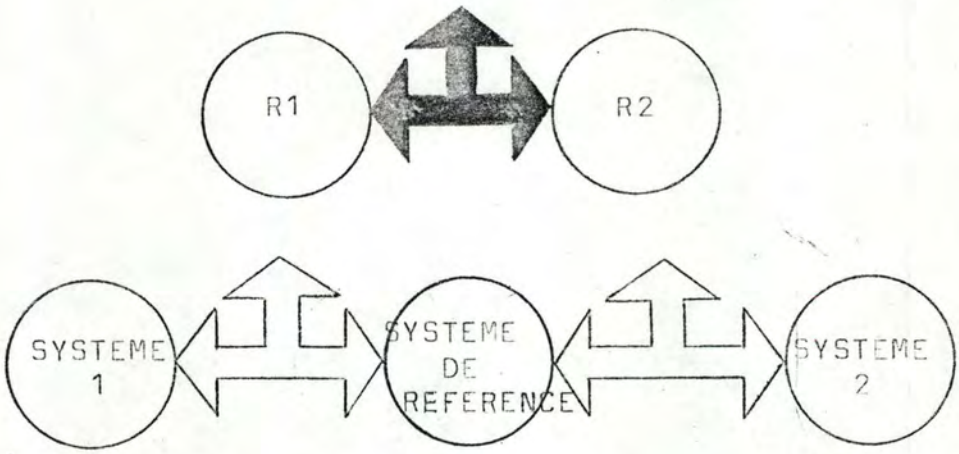
COMPARAISON DIRECTE DE DEUX SYSTEMES



Le résultat de la comparaison est une relation (inférieur, supérieur, égal, préféré, indifférencié ...).

COMPARAISON INDIRECTE DE DEUX SYSTEMES

Résultat de la comparaison indirecte



On évalue d'abord chaque système par rapport à un même système de référence, d'où les résultats R1 et R2 . Puis on compare directement ces deux résultats .

Le problème de l'évaluation et de la comparaison de systèmes complexes se pose par exemple (1) :

- lors du choix d'un ordinateur
- lors de la détermination du degré de développement d'un pays
- lors du choix de matériel aéronautique, etc .

Plusieurs modèles ont été proposés pour résoudre ces problèmes la méthode CAT est l'un deux, qui n'a été utilisé jusqu'à présent que pour le choix d'un ordinateur (2) .

C'est à ce problème particulier que la suite du présent mémoire s'intéresse, par l'intermédiaire de CAT .

(1) On peut aussi comparer sans évaluer : voir le chapitre 7 .

(2) Il s'agit d'un modèle du système et non d'un modèle de son comportement

Problème particulier

L'expression ' choix d'un ordinateur ' appelle comme précision que ce choix implique l'acceptation de toute l'offre d'un vendeur . Celle-ci comprend l'ordinateur lui-même, càd le hardware et le software, des conditions générales, des clauses particulières, d'éventuelles garanties, l'assistance (maintenance et formation), le calendrier des fournitures, etc .

Ces éléments ont leur importance ; ainsi, un bon système hardware peut être mal valorisé par une maintenance faible . Il faut donc tenir compte de tous les éléments de l'offre pour une évaluation correcte . D'autre part, la méthode CAT est surtout un outil d'évaluation et de comparaison, càd l'outil de travail d'un évaluateur ; celui-ci est au service d'un utilisateur et en contact avec le(s) vendeur(s) . Précisons l'acception de ces trois termes dans le cadre de ce mémoire :

- l'utilisateur désigne l'ensemble des entités (personne(s) ou organisation(s)) qui utiliseront les services de l'ordinateur .

- le vendeur désigne l'entité (personne(s) ou organisation(s)) qui soumet une offre à l'utilisateur .

- l'évaluateur désigne l'entité (personne(s) ou organisation(s)) qui choisit l'offre d'un vendeur en se fondant sur les besoins de l'utilisateur, à la demande de ce dernier (1) .
  Il forme un troisième niveau fréquent dans les problèmes complexes, mais non dans les problèmes de la vie courante .

Les problèmes général et particulier étant à présent posés, nous exposons ci-après les différents modèles consacrés à la solution du problème particulier, le choix d'un ordinateur.

(1) L'évaluateur n'a pas à juger si ces besoins sont fondés . Il peut toutefois conseiller l'utilisateur, car ses contacts avec les vendeurs le mettent bien au fait des possibilités du marché .

2.2 / Les modèles

Il existe plusieurs modèles servant d'outils pour le choix d'un ordinateur.

Certains ont été créés spécialement dans ce but, d'autres ont été conçus comme une aide à la décision dans un environnement complexe (1).

En nous limitant aux seuls d'entre eux ayant servi au choix d'un ordinateur, nous distinguons quatre classes de modèles :

- l'approche ad hoc [24, 32]
- les méthodes à coefficients additives [1 à 4, 9 à 19, 21 et 22]
  disjonctive - conjonctive [5 à 8, 51]
- la programmation linéaire [32]
- la méthode cost-value [27, 28, 32].

Citons également la méthode multicritère ELECTRE 1 qui n'a pas encore servi pour le choix d'un ordinateur, mais qui pourrait bientôt s'y prêter (2).

Il est rare qu'un modèle soit appliqué tel quel ; on utilise fréquemment en appoint la technique d'élimination par aspects .

Examinons brièvement la démarche de chaque classe de méthodes ainsi que de la technique d'éliminations par aspects (3) :

L'approche ad hoc est la plus simple ; elle consiste à examiner les offres et à déterminer celle qui paraît le plus adéquate, par un effort de bon sens et éventuellement quelques calculs.
D'usage général dans les problèmes quotidiens, cette approche a l'avantage de la simplicité.
Son principal inconvénient est un manque de fiabilité sensible dans le cas d'objets complexes. Ce défaut est dû à la subjectivité incontrôlée de l'évaluateur, laissant la porte ouverte à certains

(1) Une synthèse approfondie est présentée en [58].
(2) Nous consacrons le chapitre 7 à exposer ELECTRE 1 et à la comparer avec CAT .
(3) mais pas celle d'ELECTRE 1, voir note 2 ci-dessus.

biais d'évaluation tels que l'oubli d'éléments importants, les déformations professionnelles, etc .

Les méthodes à coefficients sont plus évoluées ; leur idée fondamentale est de diviser pour simplifier .

Ainsi un ordinateur peut être subdivisé en classes :

1/ Hardware 2/ Software 3/ Assistance

Chaque classe peut à nouveau être subdivisée, par exemple le Software :

21 / OS 22 / Langages 23 / Utilitaires 24 /Packages

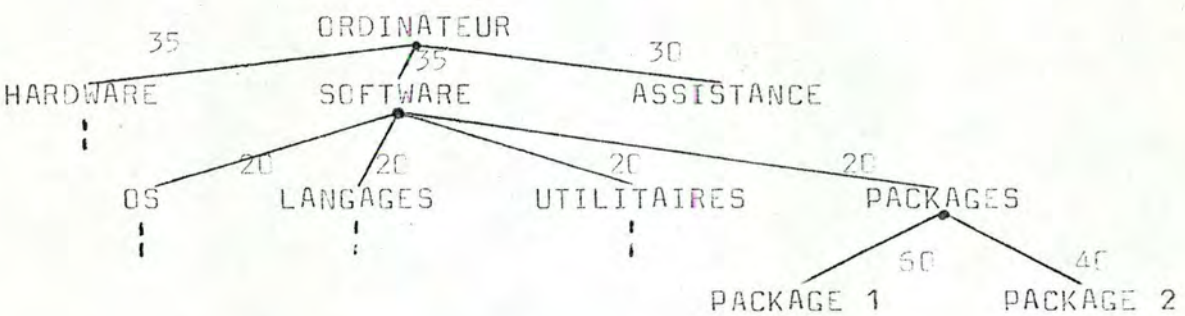
On continue à subdiviser ainsi jusqu'au degré de finesse souhaité et on obtient un arbre (1), à ce détail près que CAT permet la présence de cycles . Nous continuerons toutefois à parler d' 'arbre' .

Pour adapter cet arbre aux besoins de l'utilisateur, on commence par ne retenir que les classes relevantes pour ses besoins on supprimera la classe des utilitaires (2.3) si l'utilisateur ne souhaite pas en disposer .

Ensuite on attribue à chaque sommet - sauf la racine - un poids qui exprime son importance par rapport aux sommets de même niveau, avec lesquels il s'aggrège en un autre sommet de niveau supérieur (Fig. 2).

FIGURE 2

Exemple d'arbre utilisé par les méthodes à coefficients. Les poids sont exprimés en % . Les packages 1 et 2 forment un exemple



(1) Les termes 'arbre', 'cycle', 'sommet', 'racine', 'niveau' et 'sommet pendant' sont ceux de la théorie des Graphes ⌐ 53
L'arbre de notre problème est parfois dit 'arbre de pertinence'.

de classes élémentaires, c à d que l'on ne subdivise plus .

L'évaluateur attribue alors à chaque sommet pendant une cote; celle-ci exprime le degré dans lequel ce sommet possède la caractéristique demandée ; elle est indépendante de son poids .

A ce moment, il faut calculer les cotes des sommets non-pendants de niveau n-1 , puis remonter les niveaux dans l'arbre jusqu'à la racine, pour obtenir ainsi la cote globale de l'ordinateur .

Le processus d'aggrégation c à d la façon dont se calcule la cote (E)d'un sommet en fonction des poids et cotes (p_i et E_i) de ses n sommets immédiatement subordonnés, différencie entre elles les méthodes à coefficients :

- les additives utilisent la moyenne arithmétique $E = \sum_{i=1}^{n} pi\ Ei$ (1)

- les multiplicatives utilisent la moyenne géométrique

$$E = \prod_{i=1}^{n} E_i^{pi} \qquad (2)$$

- la disjonctive - conjonctive (càd CAT) utilise la moyenne généralisée

$$E = \left(\sum_{i=1}^{n} p_i \ E_i^{R} \right)^{1/R}$$

où R est un réel dont le sens et l'utilisation sont exposés au chapitre 3 , pages 26 à 38 (3) .

Lorsqu'une cote globale a été obtenue pour chaque ordinateur, il est aisé de les comparer entre eux, soit par leurs cotes respectives, soit par leur rapport cote / coût respectif, celui-ci traduisant un taux de satisfaction par unité monétaire investie .

Les méthodes à coefficients ont pour avantages principaux :

- la complétude : aucun élément relevant n'est oublié lors de l'évaluation.

(1) Ces formules sous-entendent que $0 < pi < 1$ et $\sum_{i=1}^{n} pi = 1$

(2) citées ici pour mémoire car elles ne sont pas utilisées pour le choix d'un ordinateur [20].

(3) On voit que la moyenne généralisée englobe la moyenne arithmétique (cas où R=1) et la moyenne géométrique (cas où R=0, car $\lim_{R \to 0} \left(\sum_{i=1}^{n} piEi^{R} \right)^{1/R} = \prod_{i=1}^{n} Ei^{pi}$; elle les dépasse donc en puissance [52].

-l'équité : le processus d'évaluation est le même pour tous .

- et enfin, le fait qu'elles délimitent la subjectivité au choix des paramètres de l'arbre et des cotes .

L'inconvénient principal de ces méthodes est qu'elles obligent l'évaluateur à coter un sommet pendant par un nombre ; celui-ci est contestable . Il en est de même pour la pondération .

On remarque de plus, dans le cas additif, l'indépendance implicite des sommets, car la formule d'aggrégation est une moyenne arithmétique ; de même dans le cas multiplicatif, l'aggrégation est toujours la même (moyenne géométrique), quelles que soient les relations entre sommets subordonnés . Ce manque de souplesse peut contredire la réalité ; CAI y échappe par sa formule d'aggrégation de moyenne généralisée (voir chapitre 3) .

La programmation linéaire est la méthode la plus rarement utilisée pour le choix d'un ordinateur ; aucune de nos sources n'expose cette démarche. Sharpe déclare toutefois [64 , page 284] ' (...) an analyst (...) can in somes cases formulate the selection process as an integer linear programming problem. However, even in simple cases it a far from trivial exercice to prepare coefficients that capture all the intricated interrelationships . The prospects for (...) such methods do not appear particularly good ' .

Un point remarquable est que les méthodes à coefficients de type additif sont un cas particulier de programmes linéaires (Voir chapitre 6) .

La méthode Cost-Value se fonde sur l'idée que la valeur d'un objet pour un utilisateur peut être très différente du coût de cet objet .

Dans notre problème, elle consiste à attribuer à chaque offre deux sommes C et V représentant : la première, la somme des coûts des divers éléments de l'offre, la seconde, la somme des valeurs qu'ont ces éléments pour l'utilisateur, exprimées en unités monétaires .

La meilleure offre est alors celle ayant la plus grande différence V - C .

On voit la difficulté de déterminer le coût et d'estimer la valeur de certains éléments tels que la présence d'un compilateur, le temps d'exécution d'une instruction, etc ...

Enfin la technique d'élimination par aspects forme un outil d'appoint pour le choix d'un ordinateur.

Le terme 'appoint' exprime qu'elle ne peut se substituer aux autres méthodes .

En effet, elle a la particularité d'indiquer toujours quelle(s) possibilité(s) rejeter, et non laquelle choisir ; de plus, l'ensemble des possibilités ne se réduit pas toujours à un seul élément . Or c'est là le but de la sélection .

La technique d' élimination par aspects consiste a éliminer les possibilités qui enfreignent certaines exigences, en considérant ces exigences l'une après l'autre . Elles forment donc des conditions sine qua non souvent exprimées lorsque les possibilités sont déjà connues .

C'est ainsi qu'une entreprise sidérurgique belge a éliminé une offre d'ordinateur parce que le siège du vendeur ne se trouvait pas en Belgique .

Nous nous permettons d'attirer l'attention du lecteur sur le point suivant : l'utilisation de la technique d'élimination par aspects constitue une utilisation implicite de conditions sine qua non . Or, la méthode CAT propose une utilisation explicite et claire de celles-ci . C'est pourquoi elle est une méthode auto-suffisante, c à d qui ne nécessite pas la technique d'appoint (1).

(1) Toutefois, s'il existe des possibilités dont on sait, dès avant le traitement par CAT, qu'elles seront rejetées parce qu'elles enfreignent une ou plusieurs conditions sine qua non, on pourra les éliminer sans plus, comme si on utilisait la technique d'appoint.

2.3 / La situation de CAT

La méthode CAT est une des nombreuses méthodes à coefficients . Elle est cependant la seule du type disjonctif -conjonctif .

Cette particularité fait de CAT une innovation intéressante parmi les méthodes servant au choix d'un ordinateur (1) .

Nous lui consacrons le chapitre 3 .

(1) Nous exposons au chapitre 6 ses relations d'inclusion, de compatibilité, d'intersection non-vide ... avec ces dernières.

C H A P I T R E 3

LA METHODE CAT (1)

3.1 / Présentation

La méthode CAT a été créée par le Docteur J.J. DUJMOVIC Professeur à la Faculté d'Electrotechnique de l'Université de Belgrade (Yougoslavie) .

Elle fait l'objet de sa thèse de doctorat ; elle a été présentée en 1974 , en même temps que son outil informatique le système SEL .

Pratiquée depuis lors à Belgrade , ses résultats sont jugés satisfaisants .

L'introduction à Namur de la méthode CAT et du système SEL constitue leur première implantation hors de Yougoslavie .

3.2 / CAT et le choix d'un ordinateur

Le choix d'un ordinateur est un cas particulier d'évaluation et de comparaison de systèmes complexes (3) .

(1) ' CAT ' est le sigle de ' Criteria Aggrégation Technique ' .
Voir page 26.

(2) Les paragraphes 3.1 à 3.3 considèrent le problème indépendamment de tout modèle . L'exposé de CAT à proprement parler commence au paragraphe 3.4 , page 21 .

(3) Sous réserve de la note (1) , page 8 .

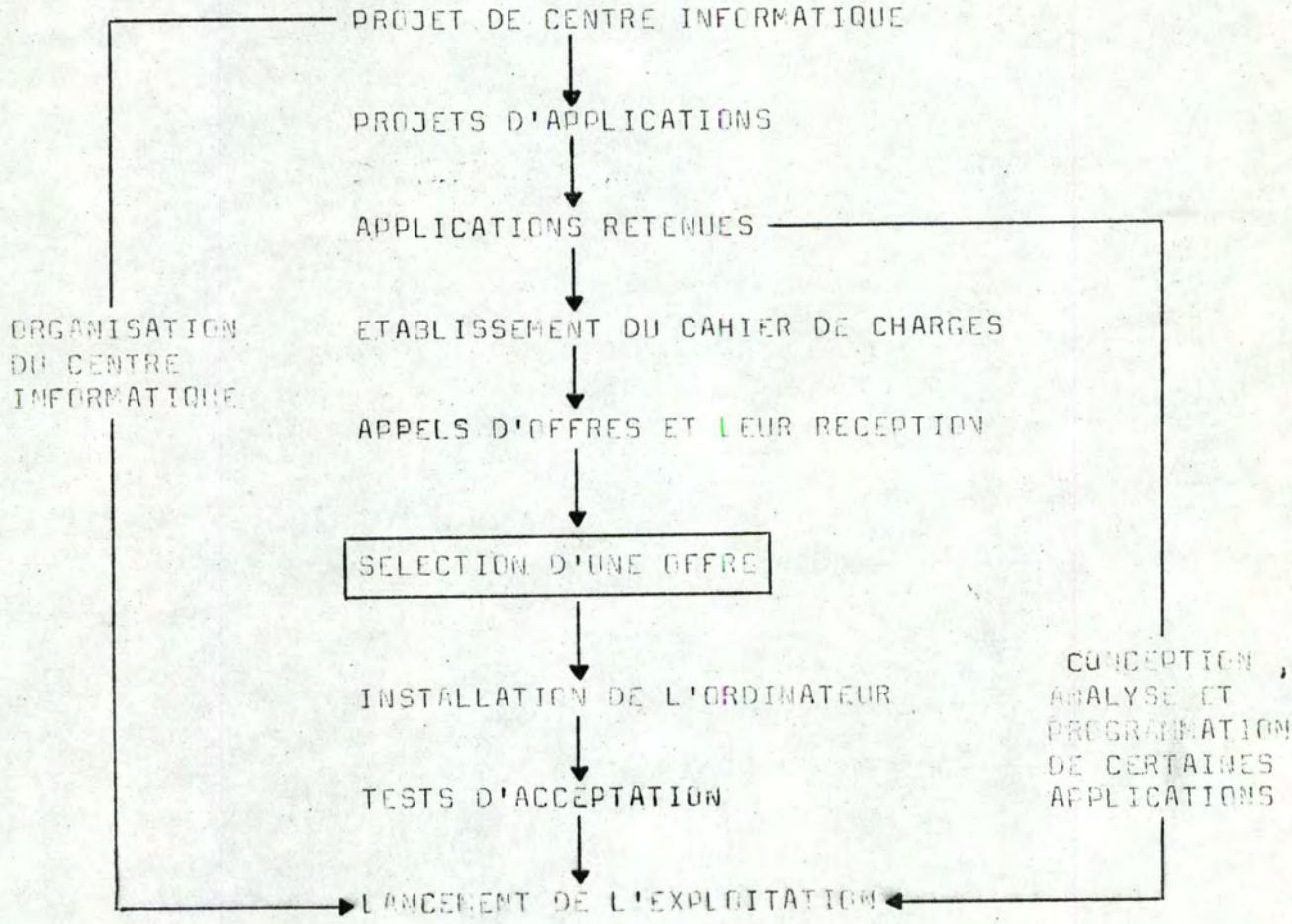
Le choix d'un ordinateur suit une démarche différente, selon qu'il s'agit d'un premier ordinateur ou du remplacement de l'existant .

Les deux démarches sont exposées synoptiquement aux Figures 3 et 4 ci-après .

L'objet de la méthode CAT se limite au processus de sélection d'une offre , qui est encadré dans ces mêmes figures ; il est développé au paragraphe 3.3 ci-après .

FIGURE 3

Démarche simplifiée suivie pour le choix d'un premier ordinateur .



(Le commentaire de cette figure est à la page suivante) .

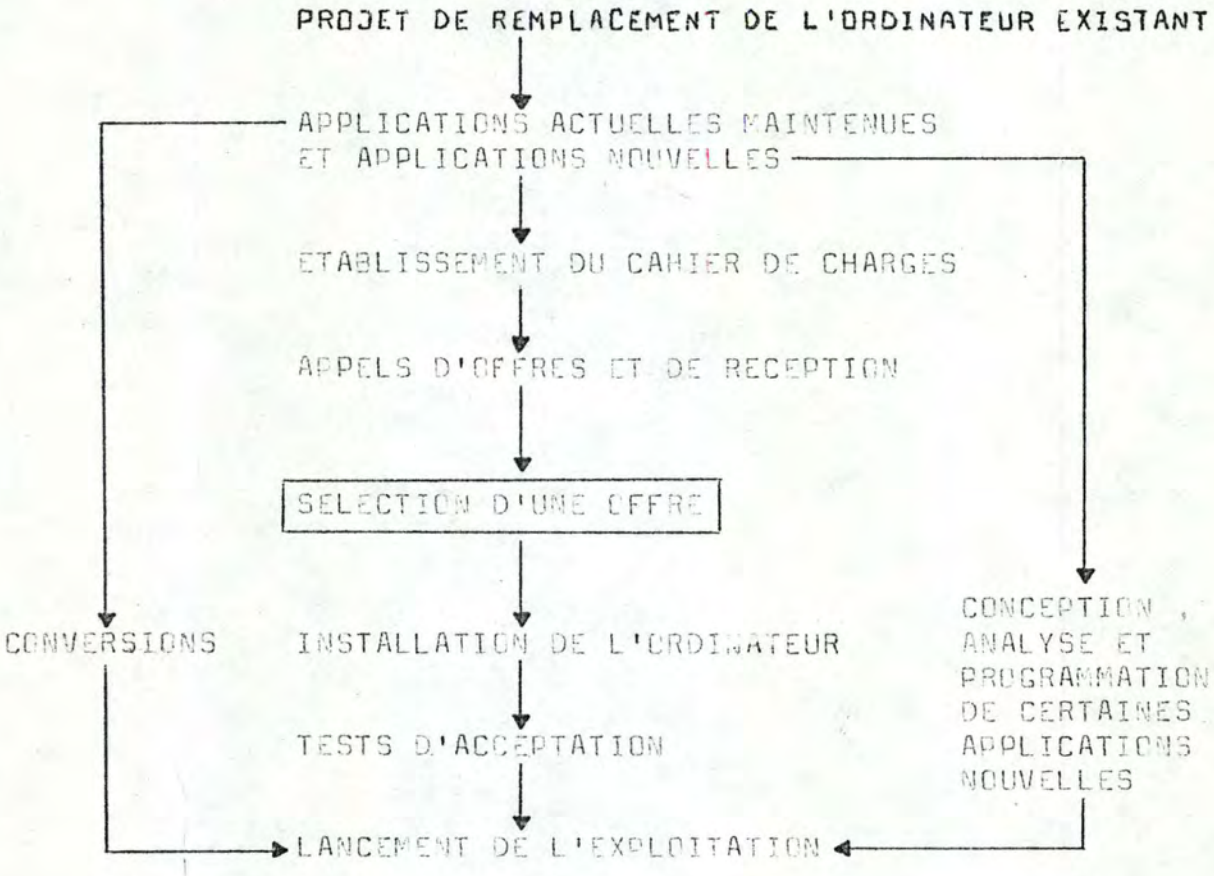
Note . Le choix d'un ordinateur se fonde sur les besoins de l'utilisateur (Voir le paragraphe 3.3. page 19) ; ces besoins permettent d'établir le CAHIER DE CHARGES , qui comprend souvent le profil de l'ordinateur qui serait idéal pour le problème (1) . On lance alors un APPEL D'OFFRES suivi de leur RECEPTION et de la SELECTION d'une d'entre elles Après la signature du contrat ont lieu l'INSTALLATION DE L'ORDINATEUR , les TESTS D'ACCEPTATION et le LANCEMENT DE L'EXPLOITATION .
Simultanément à cette démarche , on peut pourvoir à l'ORGANISATION DU CENTRE et à l'AUTOMATISATION DE CERTAINES APPLICATIONS .

---

(1) Ce profil idéal est dit CONFIGURATION DE REFERENCE en CAT . Exposons succinctement la manière dont le Professeur J.J. DUJMOVIC la calcule : à partir des besoins de l'utilisateur , on calcule approximativement les besoins en entrées/sorties et on les multiplie par un coefficient de sécurité > 1 , ce qui renseigne sur la taille du système nécessaire . On retient alors la vitesse du processeur exigible pour cette taille de système , et on termine en fixant le nombre et le type des périphériques , de la mémoire centrale et des terminaux . Dans le cahier des charges , on joint à la configuration de références les exigences sur l'O.S. , les langages , les structures et les supports de données , etc .

FIGURE 4

Démarche simplifiée suivie pour le remplacement d'un ordinateur . Note . Cette démarche est proche de celle de la Figure 3 . Leur différence essentielle est qu'ici , il existe déjà des applications . Il faut choisir lesquelles maintenir ou pas ; les applications maintenues devront être converties au nouveau système .



3.3 / CAT et le processus de sélection d'une offre

Le choix d'un ordinateur comporte toujours une étape de sélection d'une des offres .

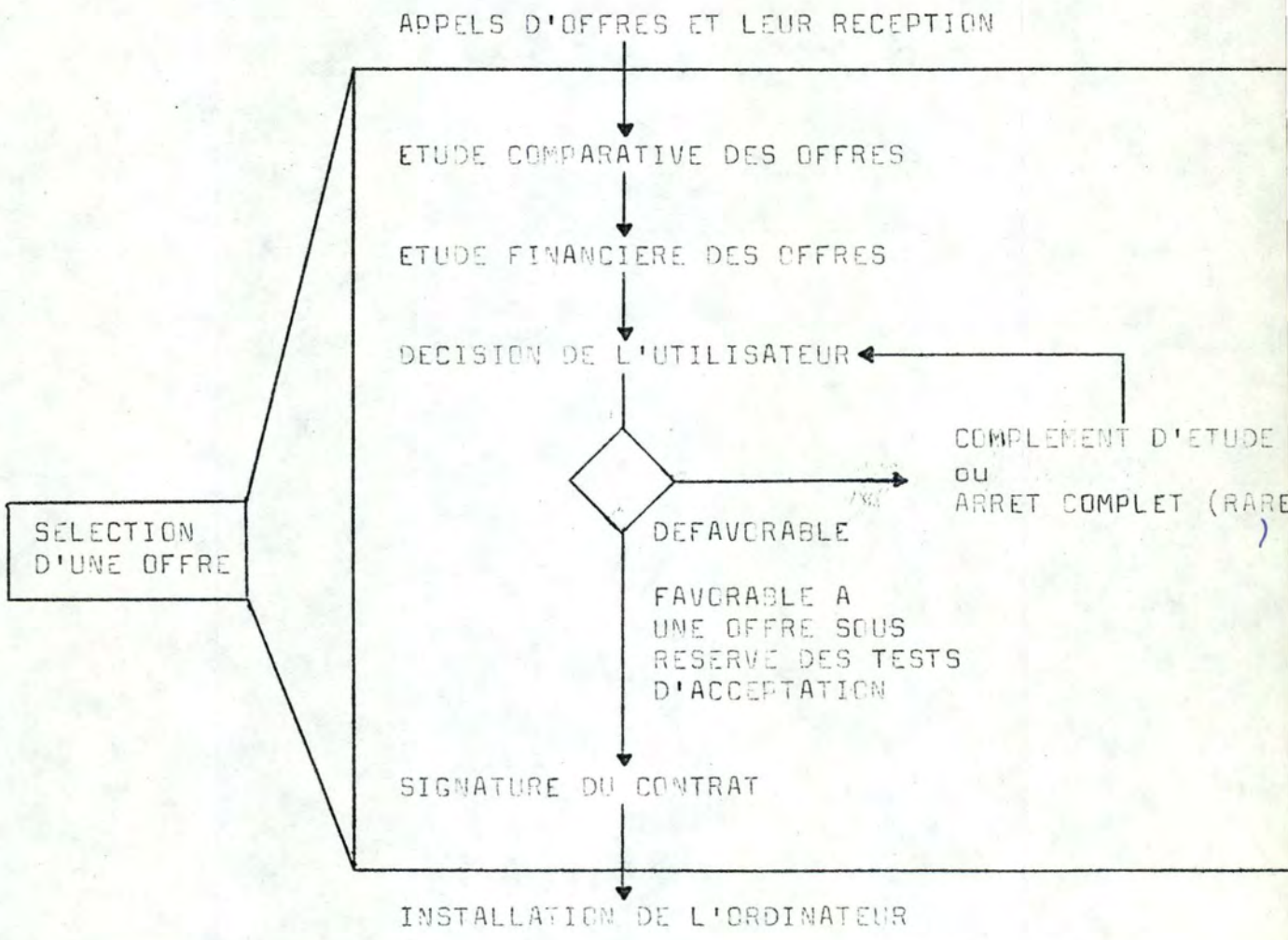
Ce processus de sélection se fonde sur les besoins de l'utilisateur . Par ' besoins ' , on entend :

- les travaux à exécuter (applications , mises au point de programmes , manipulations de fichiers ...)
- divers souhaits et exigences (délais , financement , garanties maintenance , formation , assistance , extensibilité ...) .

Le processus de sélection d'une offre est exposé à la Figure 5 ; il est l'objet de la méthode CAT , qui formalise l'étude comparative des offres .

FIGURE 5

Démarche simplifiée suivie pour la sélection d'une offre .



3.4 / Les fonctions du processus de sélection en CAT

La méthode CAT est une des nombreuses méthodes à coefficients ; leur principe est exposé au chapitre 2 (pages 11 à 13) .

Pour pouvoir exposer CAT en terme des 4 fonctions qu'elle remplit , considérons à nouveau l'arbre de la page 11 .

Précisons tout d'abord la terminologie de CAT , utilisée dans la suite de notre travail :

- les sommets pendants sont dits critères élémentaires (CE)
- les sommets non-pendants sont dits critères non-élémentaires (CNE)
- la racine de l'arbre est dite critère global
- les valeurs que l'offre du vendeur apporte aux CE sont dites les valeurs d'entrée des CE
- enfin , la cote obtenue par un critère est dite son efficacité .

Ensuite , reprenons la façon dont les méthodes à coefficients calculent l'efficacité du critère global à partir des valeurs d'entrée des CE .

On commence par attribuer à chaque CE l'efficacité qui découle de sa valeur d'entrée ; on remonte alors dans l'arbre de CNE en CNE jusqu'au critère global , au moyen d'une formule de moyenne pondérée (1) , ce qui fournit l'efficacité du critère global , càd l'efficacité de l'offre du vendeur .

L'attribution à chaque critère de son efficacité constitue la fonction d'évaluation de CAT (paragraphe 3.4.1) . Elle a comme corollaire l'étude du coût de l'offre et de son rapport Efficacité/Coût , ainsi que la détermination de la liste rangée des offres par préférence décroissante .

Le concept d'évaluation est développé en CAT dans les deux directions suivantes :

(1) Cette formule diffère selon les auteurs (Voir page 12) .

D'une part , il se peut que le vendeur propose plus d'une valeur pour un ou plusieurs critères ; le problème est traité par la fonction d'optimisation de CAT (paragraphe 3. 4. 2) .

D'autre part , la subjectivité inhérente à tout processus d'évaluation oblige à une certaine réserve dans l'utilisation des résultats . Ceux-ci sont influencés par des modifications de l'arbre ; ce problème est traité par la fonction d'analyse de sensibilité de CAT (paragraphe 3. 4. 3) (1) .

Enfin , CAT a adopté l'étude de l'investissement informatique exposée en [61] . Elle en a retenu sa fonction d'actualisation (paragraphe 3. 4. 4) .

Qu'il nous soit permis dans les paragraphes suivants , d'attirer l'attention du lecteur sur le caractère formalisé et systématique de la démarche de CAT .

Notre jugement d'ensemble sur la méthode CAT est présenté dans notre conclusion générale le chapitre 8 (page 94) .

(1) Soulignons le fait que les modifications de l'arbre sont l'objet de l'analyse de sensibilité , tandis que les modifications des valeurs d'entrée de l'arbre sont plutôt l'objet de l'optimisation

3.4.1 L'évaluation

L'évaluation consiste à calculer l'efficacité de chaque critère . Elle est la fonction fondamentale de toutes les méthodes à coefficients , et de CAT en particulier .

L'évaluation est une opération différente selon qu'elle traite des CE ou des CNE .

L'évaluation des CE

Nous avons vu au chapitre 2 (page 12) que la valeur d'entrée du CE est transformée en l'efficacité du CE .

Il s'agit d'une transformation fonctionnelle ; CAT l'explicite par l'utilisation systématique de fonctions d'efficacité .

Une fonction d'efficacité est une relation normative (1) entre les valeurs d'entrée possibles (en abcisse) et leurs efficacités respectives (en ordonnée) .
Elle est la même chose que les fonctions d' Utilité utilisées par exemple en Economie , mais , à cette différence près qu'en CAT toute fonction d'efficacité est normative , tandis qu'en Economie , les fonctions d'Utilité sont généralement descriptives .

L'efficacité est exprimée en % , par un réel compris entre 0 et 100 , ces deux valeurs incluses .

L'efficacité exprime le degré de satisfaction d'un besoin : par exemple , si la mémoire centrale de système proposé satisfait 80 % des besoins de l'utilisateur , son efficacité est de 80 % .

On constate par la pratique que l'expression des fonctions d'efficacité n'est aisée et précise que si les CE sont suffisamment importants (Figure 6) .

Par contre , les fonctions d'efficacité des CE qui ont pour objet de petites caractéristiques telles que le temps d'accès à un disque , la précision d'un calcul , etc , sont plus difficiles à exprimer , et leur précision est plus faible .

(1) Rappelons que ' normatif ' signifie ' valant règle de conduite pour le futur ' (définition à priori) , et que ' descriptif ' signifie ' décrivant une conduite passée (définition a postériori).

Pour ces derniers , CAT propose d'utiliser le concept d'efficacité relative .

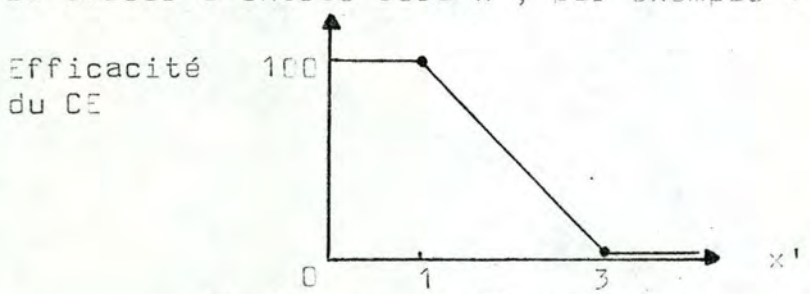
D'abord , signalons que l'efficacité absolue est la situation décrite ci-avant : l'efficacité d'un CE est fonction uniquement de sa valeur d'entrée , en plus bien sûr de la fonction d'efficacité de ce CE .

Par contre , en efficacité relative l'efficacité d'un CE est exprimée comme fonction de la valeur d'entrée de ce CE pour l'offre considérée et des valeurs d'entrée des autres offres .

Soit par exemple le temps d'accès à un disque . L'offre considérée propose la valeur d'entrée x et les autres offres x1 à xn . En efficacité relative on attribue alors à ce critère la valeur d'entrée

$$x' = \frac{x}{\min_i x_i}$$

Il reste alors à définir une fonction d'efficacité dont la valeur d'entrée sera x' ; par exemple :



Ainsi , le meilleur temps d'accès au disque obtient la valeur d'entrée 1 et l'efficacité 100 % , tandis qu'un temps d'accès double de ce dernier obtient la valeur d'entrée 2 et par conséquent l'efficacité 50 % .

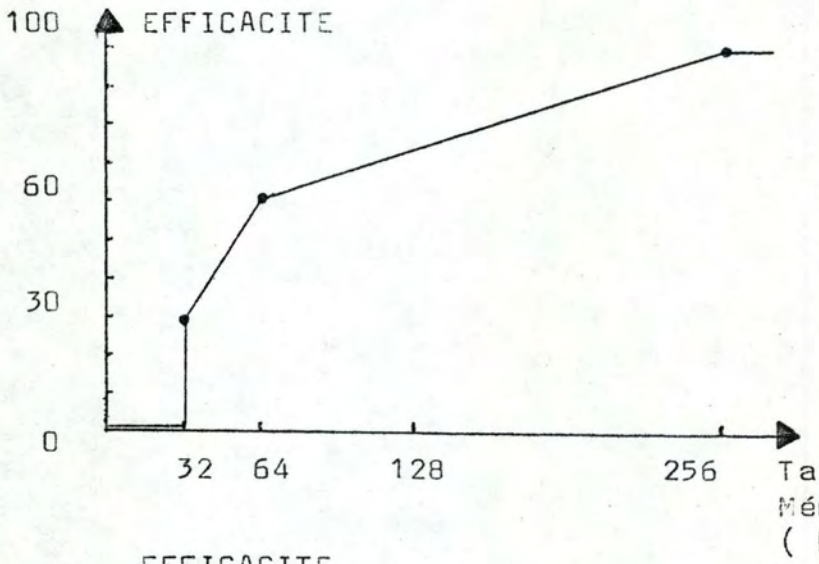
L'utilisation de min xi provient de ce que la meilleure valeur d'entrée est la plus petite . Si elle était la plus grande le dénominateur serait max xi et la fonction d'efficacité serait croissante (1). On peut aussi utiliser indifféremment min xi ou max xi , à condition d'interpréter correctement le x' .

(1) Les fonctions d'efficacité en efficacité relative ont souvent le point critique (X = 1 , E = 100) , càd que la meilleure valeur d'entrée a l'efficacité 100 % .

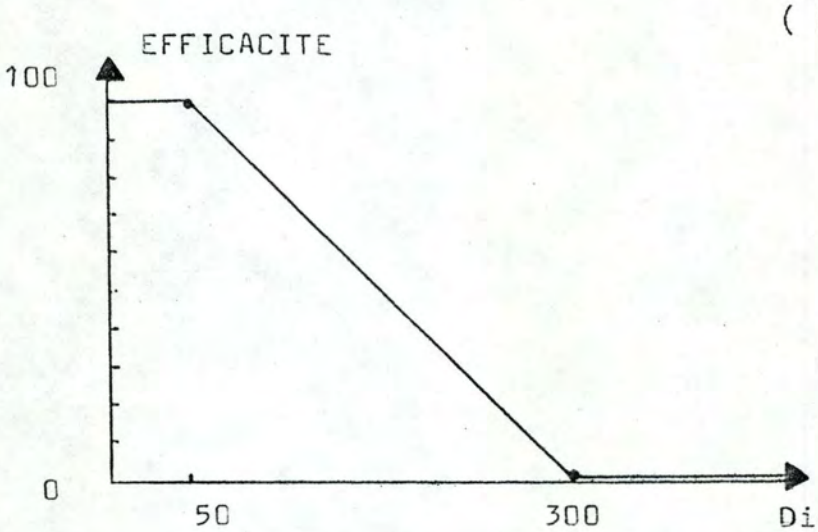
FIGURE 6

Trois exemples de fonctions d'efficacité .
(A gauche des graphes cartésiens , la forme réduite et usuelle des fonctions d'efficacité) .

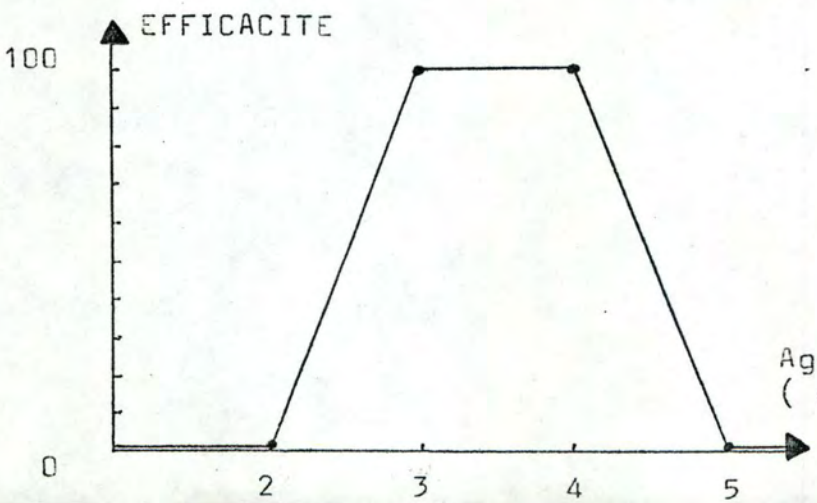
100 - 256
60 - 64
30 - 32
0 - 31



100 - 50
0 - 300



100 - 3 , 4
0 - 2 , 5



L'évaluation des CNE

Nous avons vu au chapitre 2 (page 12) que l'efficacité d'un CNE est calculée par une formule de moyenne pondérée où interviennent les efficacités des critères immédiatement subordonnés (CIS) et leurs poids .

La formule utilisée généralement est la moyenne arithmétique et , dans la méthode DOED , la moyenne géométrique .

En CAT , - et c'est là sa nouveauté fondamentale - la formule utilisée est la moyenne généralisée .

Précisons tout d'abord que nous appelons aggrégation le calcul de l'efficacité d'un CNE à partir des poids et efficacités de ses CIS (1) .

La moyenne généralisée calcule l'efficacité d'un CNE par la formule :

$$E = \left(\sum_{i=1}^{n} Wi \cdot Ei^{R} \right)^{1/R}$$

où Ei sont les efficacités des n CIS
Wi sont les poids des n CIS , avec $0<Wi<1$ et $\sum_{i=1}^{n} Wi = 1$
et R est un coefficient réel dont le sens et l'utilisation sont expliqués ci-dessous .

La moyenne généralisée constitue le coeur de la méthode CAT elle est usuellement appelée FORMULE D'AGGREGATION des critères .

La formule d'aggrégation constitue une généralisation du concept de moyenne . En effet :

- si R = 1 , elle se ramène à la moyenne arithmétique

- si R =-1 , elle se ramène à la moyenne harmonique

(1) En CAT , le coefficient d'aggrégation R intervient aussi (Voir plus loin) .

- si $R \to 0$, elle tend vers la moyenne géométrique
- si $R \to -\infty$, elle se ramène à min Ei
- si $R \to +\infty$, elle se ramène à max Ei

D'où la fonction de variation de E en fonction de R

Valeurs de R	$-\infty$	-1	0	$+1$	$+\infty$
Valeurs de E	min Ei	Moyenne harmonique	Moyenne géométrique	Moyenne arithmétique	max Ei

L'utilisation d'une valeur particulière du réel R permet d'exprimer diverses nuances dans la dépendance mutuelle des CIS au sein du CNE (1) .

Ainsi , avec $R = 1$, on fait la simple moyenne arithmétique des efficacités des CIS , ce qui traduit leur indépendance (2) .

Par contre , avec $R > 1$, la moyenne généralisée se rapproche de max Ei, et d'autant plus que R est plus grand . Se rapprocher de max Ei signifie que la plus grande efficacité attire vers elle l'efficacité E . Intuitivement , cela indique une certaine redondance entre les critères , qui peuvent dans une certaine mesure couvrir mutuellement leurs faiblesses relatives .

On parle de ' disjonction ' , par analogie au OU booléen .

En outre , avec $R < 1$, la moyenne généralisée se rapproche de min Ei , et d'autant plus que R est plus petit . Se rapprocher de min Ei signifie que la plus petite efficacité attire vers elle l'efficacité E . Intuitivement , cela indique que l'ensemble des critères doit être satisfaisant pour que le groupe le soit ; les critères sont nécessaires et non-redondants . Si la nécessité d'un critère est forte , ce critère devient une condition sine qua non ($R < 0$) .

(1) Et non en toute généralité , car les critères sont souvent plus ou moins corrélés ; par exemple la taille de la Mémoire Centrale et le temps d'accès à cette même Mémoire .
(2) Càd qu'il n'y a ni disjonction ni conjonction . Voir plus loin .

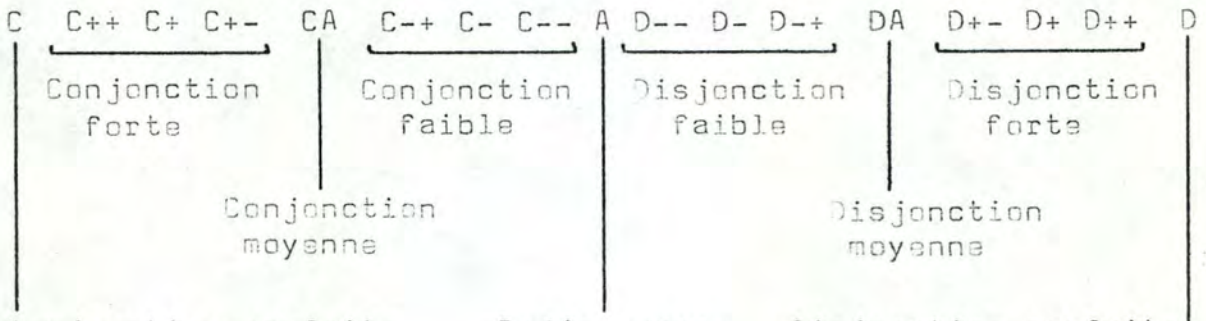
On parle de ' conjonction ', par analogie au ET booléen .

Les modes d'aggrégation sont donc de trois grandes classes disjonction (R > 1) , indépendance (R = 1) et conjonction (R < 1) . En outre , les conjonctions avec R < 0 sont des conditions sine qua non ; en effet , si R est négatif et Ei = 0 , l'expression Ei ** R = + ∞ , et l'expression de la page 26 fournit E = 0 .

Le critère i est donc bien une condition sine qua non : si son efficacité est nulle , l'efficacité de son critère immédiatement ascendant est également nulle .

Par facilité pour l'utilisateur , le choix d'une valeur de R (le coefficient d'aggrégation) est remplacé par le choix d'un opérateur d'aggrégation .

Les opérateurs d'aggrégation sont les suivants :



A chaque opérateur d'aggrégation ainsi défini correspond une valeur du coefficient d'aggrégation R, qui est fonction de l'opérateur d'aggrégation et du nombre des CIS . Le calcul des valeurs de R sort du cadre de notre travail ; sur ce sujet , nous renvoyons à / 47 , 52 /. Nous nous limitons ici à donner les valeurs usuelles de R (Voir le tableau de la page suivante) .

OPERATEURS D'AGGREGATION	COEFFICIENTS D'AGGREGATION			
	N = 2	N = 3	N = 4	N = 5
C		- ∞		
C++	-9.0600	-7.6793	-6.7100	-6.2727
C+	-3.5101	-3.1131	-2.8232	-2.6414
C+-	-1.6540	-1.5493	-1.4519	-1.3627
CA	-0.7200	-0.7314	-0.7145	-0.6897
C-+	-0.1470	-0.2065	-0.2248	-0.2225
C-	0.3619	0.1953	0.1670	0.1707
C--	0.6191	0.5730	0.5473	0.5423
A	1.0000	1.0000	1.0000	1.0000
D--	1.4491	1.5190	1.5639	1.5930
D-	2.0185	2.1873	2.3019	2.3624
D-+	2.7917	3.1011	3.3181	3.4427
DA	3.929	4.4501	4.8250	5.0542
D+-	5.8023	6.6748	7.3171	7.7294
D+	9.5207	11.0948	12.2766	13.0662
D++	20.6303	24.3177	27.1346	29.0810
D		+ ∞		

Tableau donnant ,pour chaque opérateur d'aggrégation , les valeurs possibles du coefficient d'aggrégation en fonction de N .

Le choix d'un opérateur d'aggrégation est assez aisé pour le praticien ; l'algorithme suivi pour ce choix est exposé à la Figure 7

Un exemple d'arbre complet est présenté au chapitre 5 , pages 55 à 73 . Pour sa part , la Figure 8 montre une façon de décrire un arbre .

FIGURE 7

Algorithme pour le choix d'un opérateur d'aggrégation .
Note . Dans les tests , C signifie conjonction , D disjonction ; et A indépendance

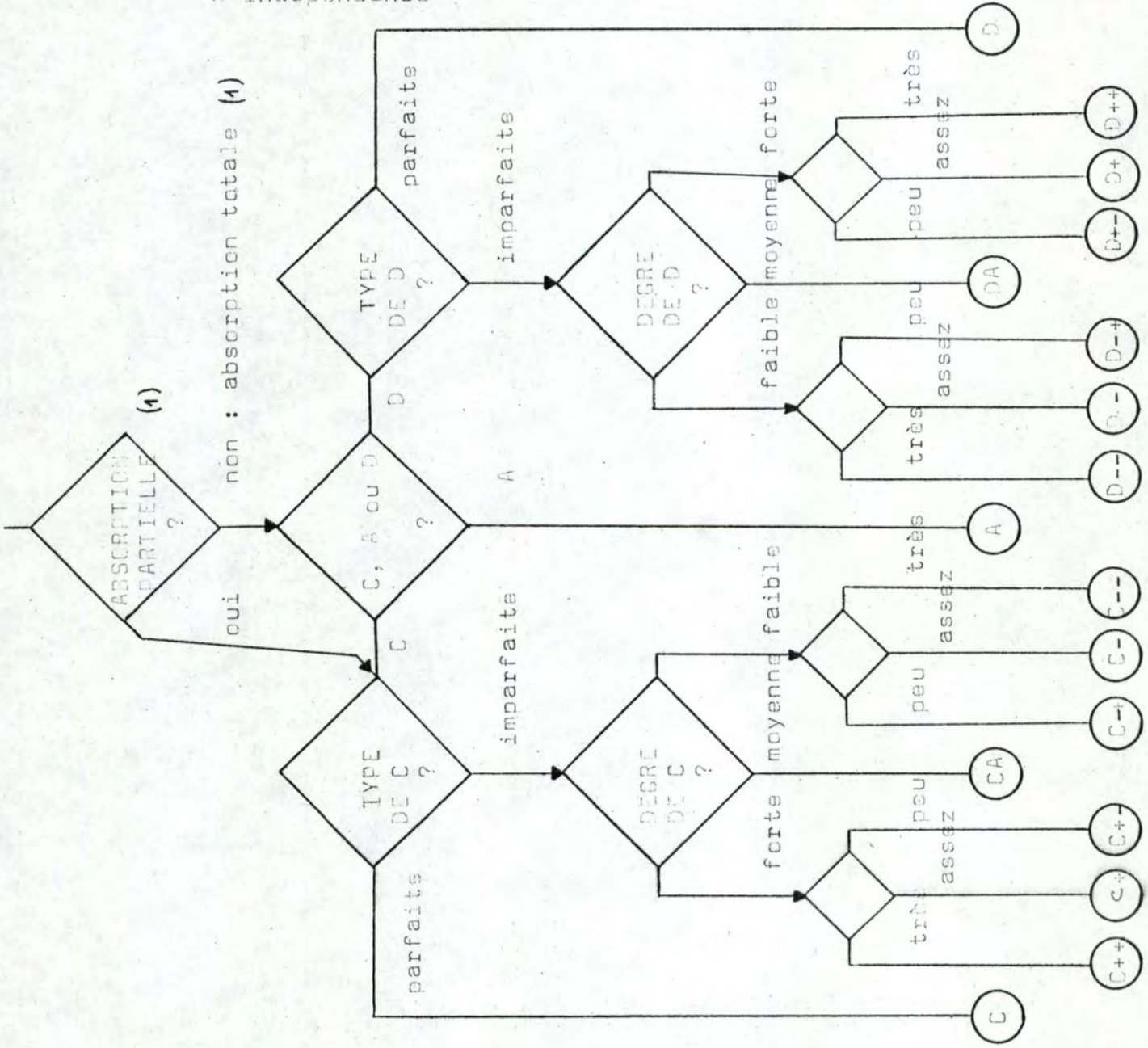


FIGURE 8

Deux formulaires souvent utilisés pour décrire les CE et les CNE . A la page suivante , un exemple de leur utilisation dans le cadre du choix d'un ordinateur hybride [31] .

CAT SYSTEM EVALUATION METHOD — FORM No 1 — ELEMENTARY CRITERIA — DATE / PROJECT / AUTHOR / PAGE

600

Type of CPU-I/O Channels
1 = Selector
2 = Multiplexer
3 = Both

Texte décrivant le CE

Numéro du CE

Fonction d'Efficacité du CE (Forme réduite)

CAT SYSTEM EVALUATION METHOD — FORM No 2 — NON-ELEMENTARY CRITERIA — DATE / PROJECT / AUTHOR / PAGE

No	COMPONENTS FOR EVALUATION-LEVEL 1	MIXED AVERAGING	REL. LEV.2	MIXED AVERAGING	REL. LEV.3	MIXED AVERAGING	REL. LEV.4	MIXED AVERAGING
600	CPU-I/O Chan.	600	External Storage	2450				
610	Disk Capac.	85		25				
610	Disk acc. time	15 C-+						

CNE provenant d'une autre page

Poids des critères et opérateur d'aggrégation

CE utilisé dans une autre page

CNE formé des CE 610 et 620

600 Type of CPU - I/O channels
1 = selector
2 = multiplexer
3 = selector + multiplexer

610 Total disk storage capacity in MBy
3 – 100 – 7
– 90 –
2 – 80 –
– 70 –
– 60 – 2
– 50 –
1 – 40 –
– 30 –
– 20 –
– 10 –
0 – 0 – 0.256

620 Disk storage average access time measured in ms
50 – 100 – 300
– 90 –
100 – 80 –
– 70 –
– 60 –
– 50 –
– 40 –
– 30 –
– 20 –
– 10 –
200 – 0 – 0

630 Card reader speed measured in $\frac{card}{min}$

640 Paper-tape reader speed measured in $\frac{char}{sec}$ (equivalent to card reader speed)
400 – 100 – 600
– 90 –
– 80 – 300
– 70 –
– 60 –
– 50 –
– 40 –
– 30 –
– 20 –
– 10 –
0 – 0 – 0

650 Line printer speed measured in 120-character lines per minute (with 48 different characters minimum)

660 Typewriter
0 = nonexistent
1 = exists
1 – 100 – 1
– 90 –
– 80 –
– 70 –
– 60 –
– 50 –
– 40 –
– 30 –
– 20 –
– 10 –
0 – 0 – 0

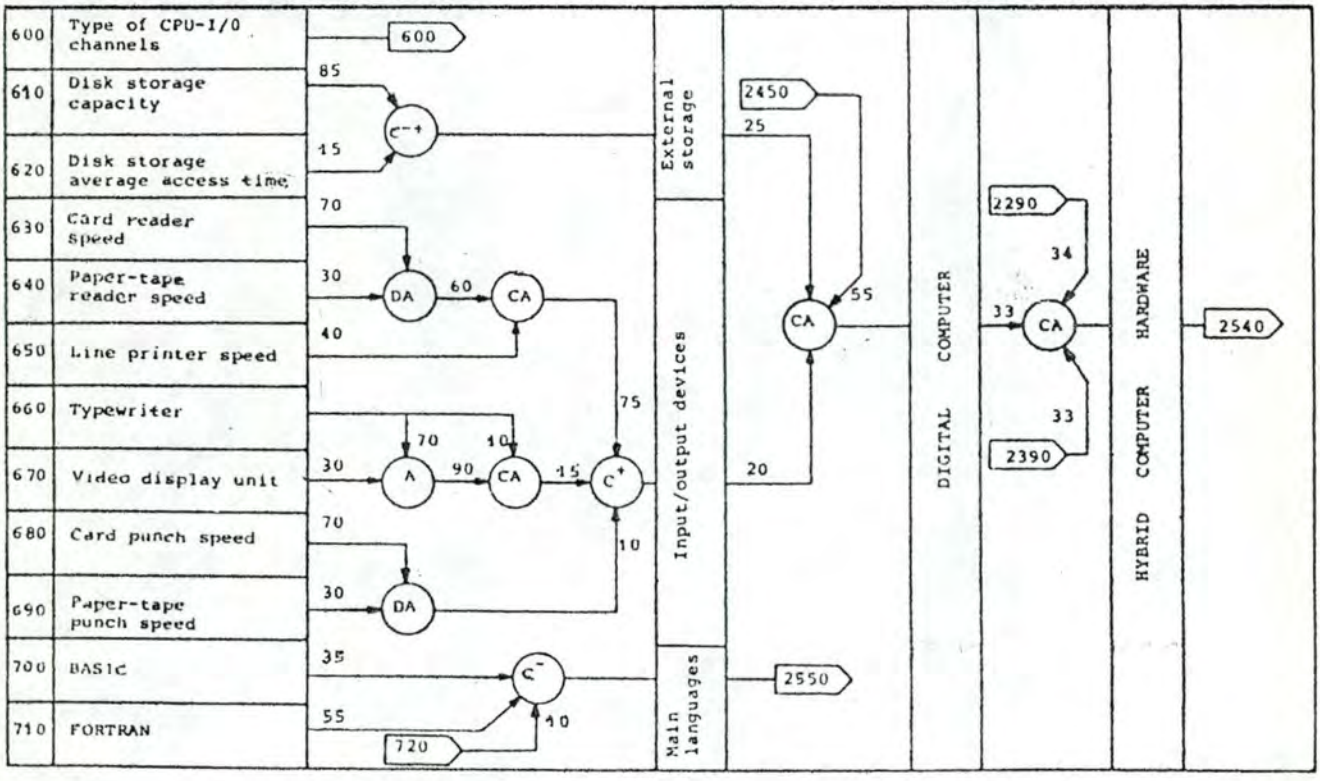
670 Video display unit
0 = nonexistent
1 = exists

680 Card-punch speed measured in 80-column cards per minute
200 – 100 – 266
– 90 –
100 – 80 – 133
– 70 –
– 60 –
– 50 –
– 40 –
– 30 –
– 20 –
– 10 –
0 – 0 – 0

690 Paper-tape punch speed in $\frac{char.}{sec}$ (equivalent to card punch speed)

700 BASIC language
0 = nonexistent
1 = standard BASIC
2 = hybrid BASIC (with possibilities of using assembler instructions)
2 – 100 – 2
– 90 –
– 80 – 1
– 70 –
– 60 –
– 50 –
– 40 –
– 30 –
1 – 20 –
– 10 –
0 – 0 – 0

710 FORTRAN language:
0 = nonexistent
1 = standard FORTRAN
2 = hybrid FORTRAN (with possibilities of using assembler instructions)



(Figure 8 / Le commentaire est à la page précédente) .
On remarque l'absorption partielle entre les CE 660 (Télétype) et 670 (Vidéo) .

La comparaison des offres

L'évaluation d'une offre fournit son efficacité .

Pour pouvoir comparer cette offre aux autres offres il existe trois critères :

- maximiser l'efficacité E (on choisit l'offre la plus efficace)
- minimiser le coût C (on choisit l'offre la moins chère)
- maximiser le rapport $\frac{E}{C}$ (on choisit l'offre la plus efficace par unité monétaire investie) .

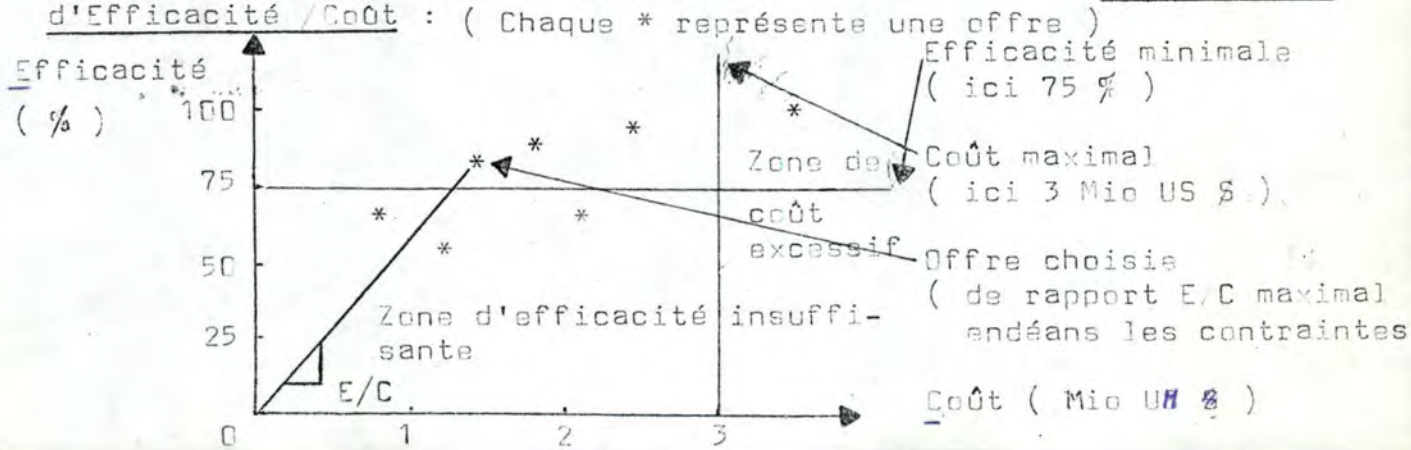
On s'aperçoit cependant que ces critères pris isolément fournissent des résultats peu réalistes :

- si on maximise E , on peut obtenir un coût prohibitif
- si on minimise C on peut obtenir une efficacité insuffisante
- si on maximise le rapport $\frac{E}{C}$, on constate que E est souvent proche de 50 % , ce qui est généralement considéré comme une efficacité insuffisante .

Dans le problème du choix d'un ordinateur , on a toujours utilisé en CAT ce dernier critère , en l'encadrant par deux contraintes :

- une contrainte d'efficacité minimale , en général peu souple
- une contrainte de coût maximale , en général assez souple , car le budget d'acquisition de ressources informatiques est rarement fixé strictement : une certaine extensibilité est habituelle .

On illustre alors les résultats de CAT par le graphique classique de Coût/Performance , que nous appelons ici le graphique d'Efficacité /Coût : (Chaque * représente une offre)

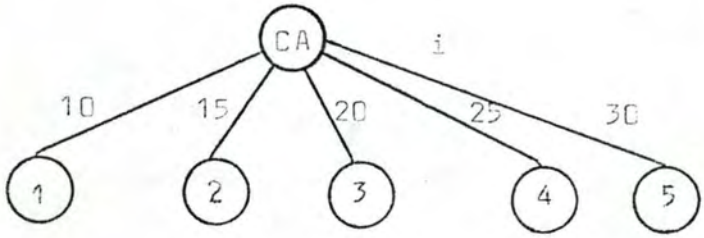


La formule d'aggrégation de CAT comporte deux des développements importants : la propriété d'associativité de cette formule et l'absorption partielle d'un critère .

L'associativité

L'associativité exprime que les CIS d'un CNE peuvent être regroupés en des CNE intermédiaires , sans que l'efficacité du CNE initial en soit jamais modifiée.

Soit un arbre :



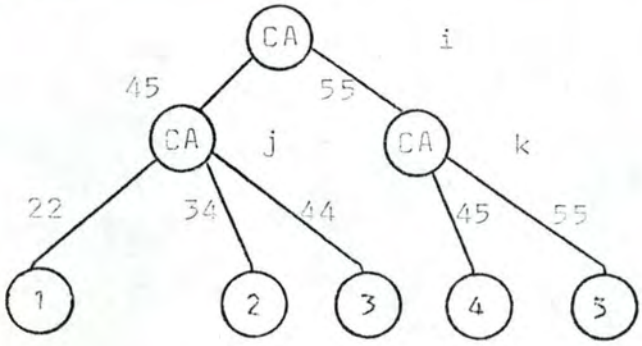
Créons un niveau intermédiaire avec les CNE j et k , dont les CIS seront respectivement les trois premiers CIS du CNE i , et les deux derniers .

Adoptons les règles suivantes :

- les poids de j et k sont la somme des poids que leurs CIS respectifs avaient auparavent (ci-dessous : Wj = 45 = 10 + 15 + 20 et Wk = 55 = 25 + 30)

- le coefficient d'aggrégation de j et k est le même que celui de i (càd le R correspondant à CA avec 5 CIS)

- les poids des CIS de j et k sont les anciens poids normalisés , càd divisés chacun par leur somme à tous , de façon à ce que leur nouvelle somme égale 1 (W1 = 22%= 10%/ 45%, etc) . D'où l'arbre :



Démontrons en toute généralité que les deux arbres ont toujours la même efficacité pour le CNE i .

Pour cela , montrons que la formule de calcul de Ei utilisée dans le second arbre se ramène à la formule du premier arbre .

Dans le second arbre :

$$Ei = (Wj \cdot Ej^R + Wk \cdot Ek^R)^{1/R}$$

où $Wj = W1 + W2 + W3$ et $Wk = W4 + W5$

$$Ei = (Wj \cdot (W1'.E1^R + W2'^R + W3'.E3^R) + Wk \cdot (W4'.E4^R + W5'.E5^R))^{1/R}$$

car $Ej = (W1'.E1^R + W2'.E2^R + W3'.E3^R)^{1/R}$

et $Ek = (W4'.E4^R + W5'.E5^R)^{1/R}$,

où les Wm' sont les poids normalisés des critères 1 à 5 .
càd $Wm' = Wm / Wj \quad m = 1, 2, 3$
$= Wm / Wk \quad m = 4, 5$

$$Ei = (W1.E1^R + W2.E2^R + W3.E3^R + W4.E4^R + W5.E5^R)^{1/R}$$

d'où

$$Ei = (\sum_{1}^{5} Wm \cdot Em^R)^{1/R}$$, càd la formule du premier arbre .

Attention : il faut que R soit constant , càd qu'il corresponde à CA avec 5 CIS , bien qu'il n'en ait plus que trois et deux dans les critères j et k . L'impossibilité d'exprimer cette situation dans le langage SEL rend l'associativité inutilisable sinon manuellement .

L'absorption partielle

L'absorption partielle exprime que les CIS d'un CNE n'ont pas tous la même possibilité de couvrir mutuellement leurs faiblesses relatives .

Observons d'abord que dans un arbre normal, un CNE 'possède' en exclusivité tous ses CIS : on parle d'absorption totale (AT) ; c'est la situation la plus fréquente .

Cependant , la nature du problème peut faire qu'un même critère soit ' possédé ' par plusieurs CNE ; ainsi , le Formware qui intervient dans les deux CNE Hardware et Software . Ce cas est plutôt rare .

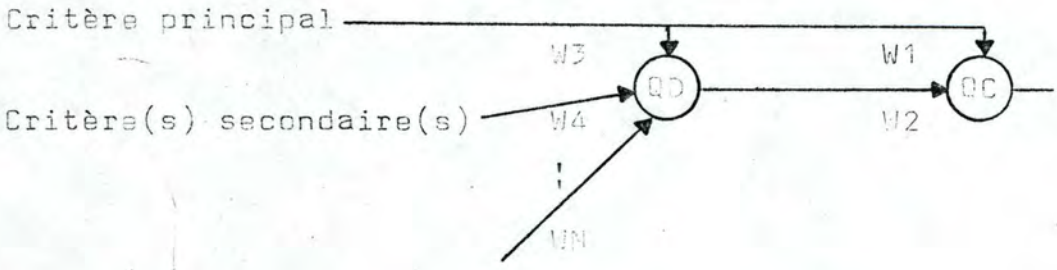
Un cas particulier d'absorption partielle (AP) est introduit par le raisonnement suivant

Considérons deux CIS d'un même CNE , ayant tous deux le même poids . Si leurs efficacités respectives sont x et y , l'efficacité du CNE est z ; elle est inchangée si leurs efficacités avaient été y et x , dans l'hypothèse où tout le reste est constant .

On voit que le critère le plus efficace fait ' remonter ' l'autre (1) , càd que les deux critères couvrent mutuellement leurs faiblesses relatives . Or , la nature du problème peut faire que cette relation existe dans un sens mais que la réciproque soit fausse ou moins vraie (2) .

Par exemple , la vitesse du processeur peut absorber les éventuelles déficiences relatives du langage assembleur . La réciproque est un peu moins vraie .

Cette situation est exprimée par la double aggrégation suivante :

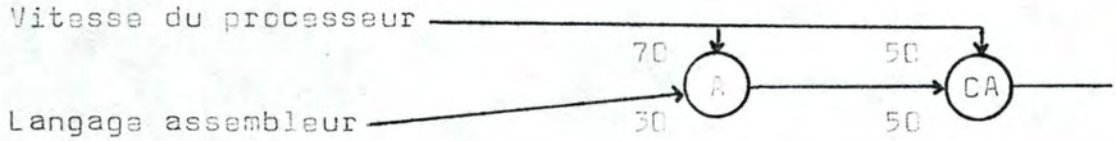


(1) Ce qui est le fondement même du concept de moyenne .

(2) CAT travaille en logique continue ; celle-ci opère sur les valeurs 0 à 1 , alors que la logique booléenne opère sur les valeurs 0 et 1 .

Dans cet arbre , QD et QC dénotent une disjonction et une conjonction quelconques. Comme il y a deux séries de poids et deux opérateurs d'aggrégation à fixer , on préfère souvent fixer QD à ' A ' (1) et W1 = W2 à 50 % ; il reste ainsi à fixer un seul opérateur d'aggrégation et une seule série de poids .

Exemple :



Cette double aggrégation fait que le critère principal est CIS de deux CNE : le CNE intermédiaure avec ' A ' et le CNE avec ' CA ' ; c'est en cela qu'il y a absorption partielle du critère principal par chacun des deux autres .

L'origine de l'absorption partielle est le théorème booléen de l'absorption :

$$X \vee (X \wedge Y) = X$$
$$X \wedge (X \vee Y) = X$$

La transposition de ces expressions en logique continue (2) donne :

où l'efficacité du critère i est celle du critère x .

(1) L'opérateur d'indépendance ' A ' est à la limite entre la conjonction et la disjonction . On peut donc estimer qu'elle fait partie des deux groupes à la fois .

Comme les ordinateurs comportent peu de disjonctions (car peu de redondances) , mais beaucoup de conjonctions , on retient usuellement la seconde forme , en fixant QD à ' A ' ; cela donne des arbres comme celui en haut de la page 37 .

Un exemple réel d'absorption partielle est présenté au chapitre 5 , pages 55 à 73 .

3.4.2. L'optimisation

L'optimisation consiste , lorsque le vendeur propose plus d'une possibilité pour un critère au moins , à déterminer l'efficacité maximale obtenable par l'offre (1) , sous une contrainte de coût maximal .

Plusieurs cas sont possibles :

- d'une part , le critère qui a plusieurs possibilités (CPP) peut être un CE , un CNE ou le critère global
- d'autre part , les différentes possibilités peuvent être assorties ou non d'un coût

Examinons ces cas séparément .

Le CPP est le critère global

Si le CPP est le critère global , cela signifie que le vendeur fait plusieurs offres .

Il suffit de les évaluer séparément , comme si elles provenaient de vendeurs différents .

Leurs coûts interviennent dans leurs rapports Efficacité/Coût respectifs .

Le CPP est un CNE

Si le CPP est un CNE , par exemple l' O.S. dont il existerait plusieurs versions , on ramène ce CNE à un CE .

Pour ce faire , on calcule l'efficacité de chaque possibilité du CNE , via le sous-arbre dont ce CNE est la racine .

(1) Rappelons que l'offre forme le critère global .

On élimine alors ce sous-arbre , et on attribue au CNE devenu CE la fonction d'efficacité ' Identité ' , càd qui a la forme réduite 0-0 , 100-100 (1) .

Les efficacités des différentes possibilités sont alors les valeurs d'entrée ; leurs coûts éventuels sont inchangés .

On traite ce cas comme celui ci-après .

Le CPP est un CE

Si le CPP est un CE , par exemple la taille de la Mémoire Centrale , deux cas se présentent:

- ou les différentes possibilités - nous dirons ' valeurs d'entrée ', car il s'agit d'un CE - n'ont pas d'incidence financière , et on choisit systématiquement la valeur d'entrée de plus grande efficacité

- ou les valeurs d'entrée ont une incidence financière et elles sont toutes assorties d'un coût . Cette situation est la plus fréquente

Avant tout nouveau développement signalons qu'en CAT on entend par ' Configuration ' un ensemble de valeurs d'entrée des CE .

Par exemple , s'il y a 5 CE , l'ensemble (3 , 35 , 10 , 5.6) est une configuration . L'ensemble (4 , 35 , 10 , 5.6) est une autre configuration .

S'il existe des CE qui ont plusieurs valeurs d'entrée possibles , il existe plusieurs configurations ; elles ne diffèrent que par les valeurs de ces CE . L'exemple ci-dessus a deux possibilités pour le 1° CE : 3 et 4 ; cette offre a donc deux configurations .

On voit que si les CE ont tous une seule valeur d'entrée il n'existe qu'une seule configuration .

Si certains CE sont des CPP , le nombre total de configurations possibles est le produit des nombres de valeurs d'entrée possibles de tous les CE .

(1) Voir la page 25 .

Ainsi, une offre ayant 100 CE dont 4 ont respectivement 2 , 3 , 4 et 5 valeurs d'entrée possibles , a 2 * 3 * 4 * 5 * 1 = 120 configurations possibles . 96

La fonction d'optimisation ne formalise que le cas d'une offre où un CE au moins a plusieurs valeurs d'entrée possibles , dont chacune est assortie d'un coût . Cette offre comporte donc plusieurs configurations .

Exposons maintenant cette formalisation .

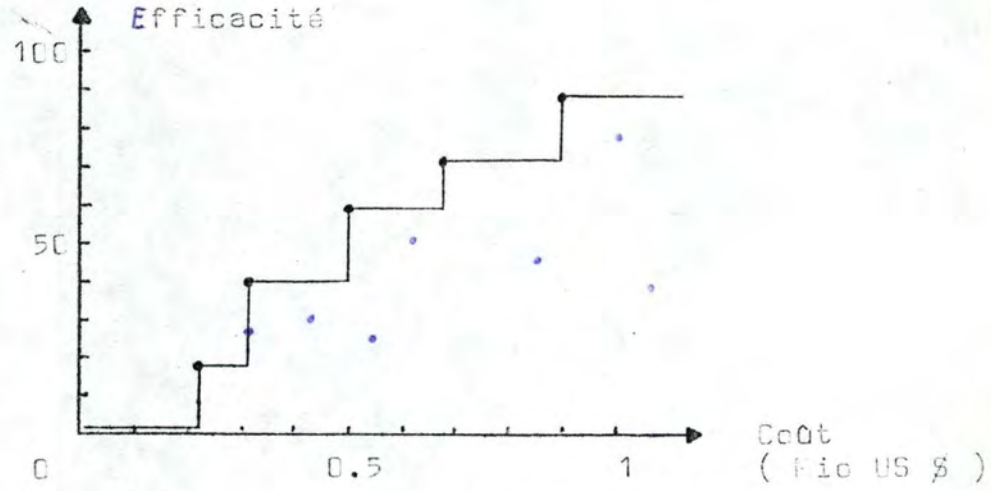
La formalisation de l'optimisation

Dans le cas décrit , l'offre d'un vendeur a pour coût total la somme de deux composantes

- un coût fixe commun à toutes les configurations ; il s'agit du coût total sans les éléments objets d'un CE à plusieurs valeurs d'entrée possibles
- un coût variable qui est la somme des coûts de ces éléments .

On souhaite connaître , pour chaque coût (variable) possible la configuration dont l'efficacité est maximale .

Examinons cette situation sur le graphe cartésien suivant : (chaque point représente une configuation repérée par son efficacité et son coût) :



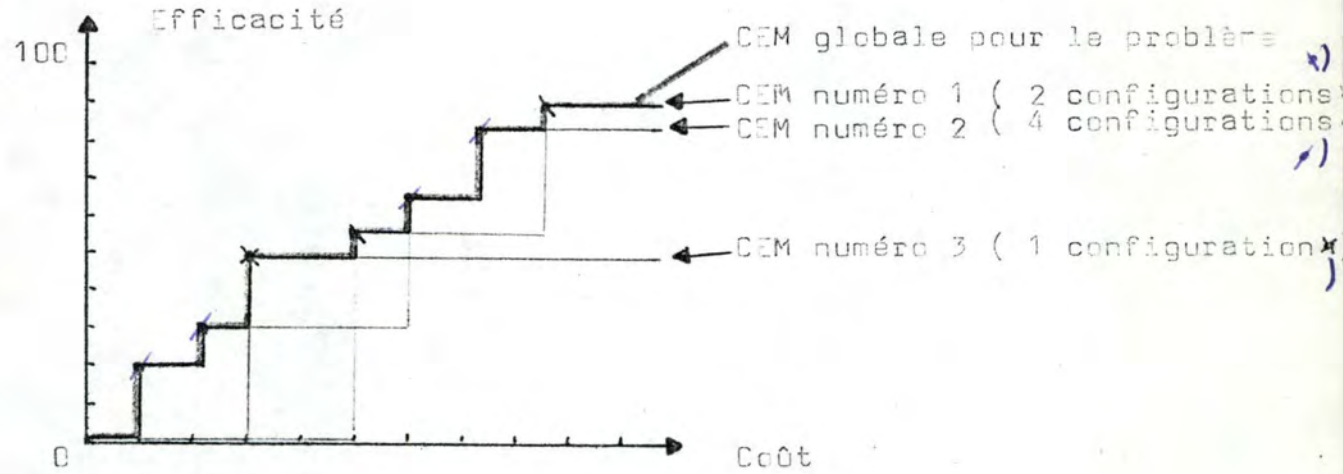
Considérons l'enveloppe supérieure de l'espace des configurations possibles .

Celle-ci fournit , pour un coût donné , la configuration dont l'efficacité est maximale pour ce coût ; ainsi pour C = 0.5 Mio on obtient la configuration dont l'E = 60 % , et pour C = 1 Mio $ on obtient la configuration dont l'E = 90 % , mais dont le C = 0.9 Mio $ seulement .

L'enveloppe supérieure est donc la Courbe d'Efficacité Maximale (CEM) pour cette offre d'un vendeur .

Si l'on superpose les CEM de plusieurs offres , l'enveloppe supérieure de ces courbes donnent la Courbe d'Efficacité Maximale pour notre problème .

Exemple :



Remarquons que chaque critère a une CEM : en effet , pour un CE , le(s) couple(s) (Valeur d'entrée , prix) est (sont) exprimable(s) sous la forme (Efficacité , prix) . Il(s) forme(nt) la CEM du CE .

Exemple de CEM pour un CE

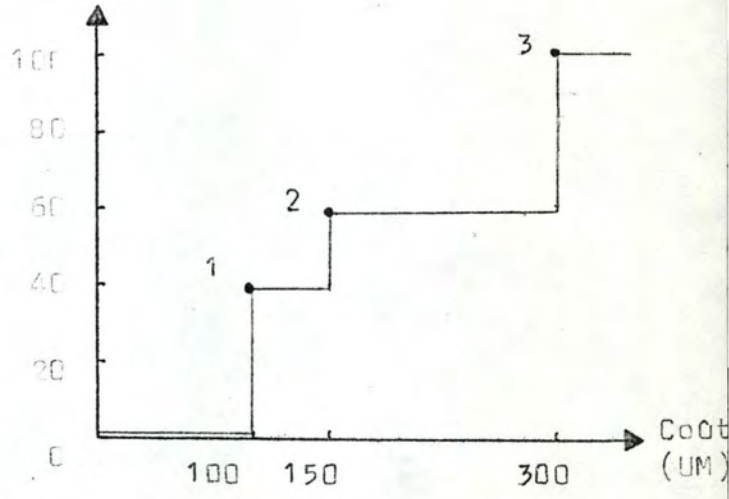
Valeurs d'entrée
(exprimées par leurs
efficacités)

1 / E = 40 % , C = 100 UM

2 / E = 60 % , C = 150 UM

3 / E =100 % , C = 300 UM

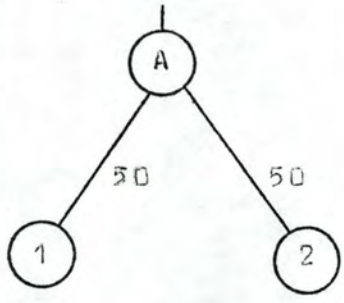
CEM du CE



Les CNE ont également chacun leur CEM , qui découle des CEM de ses CIS .

Exemple de CEM pour un CNE

Schéma de l' aggrégation



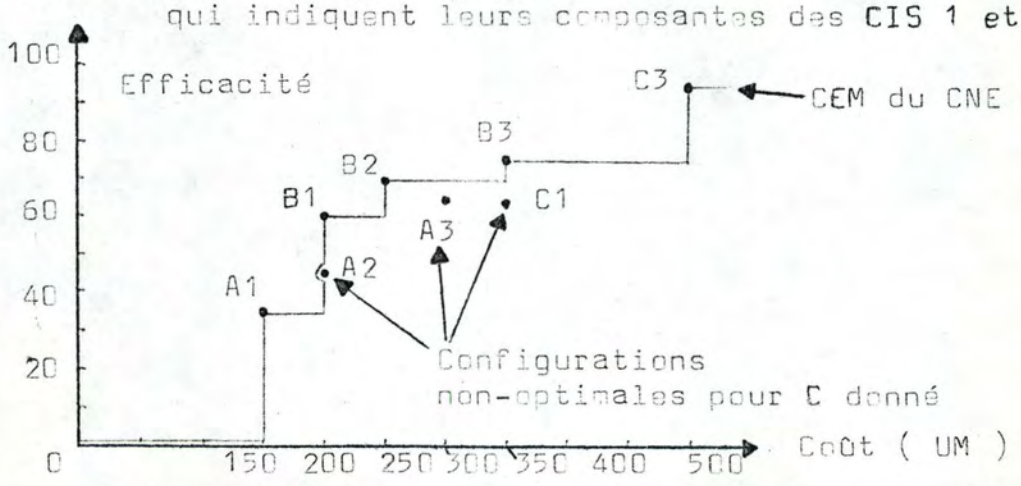
CEM du CIS 1

Identique à celle ci-dessus , même page

CEM du CIS 2

A / E=30 %, C= 50 UM
B / E=80 %, C=100 UM
C / E=90 %, C=200 UM

CEM du CNE (Les configurations sont reprérées par deux caractères qui indiquent leurs composantes des CIS 1 et 2)



En remontant dans l'arbre de proche en proche depuis les CE jusqu'au critère global , on obtient la CEM de l'offre .

Un exemple d'optimisation d'une offre est présenté au chapitre 5 , pages 55 à 73 .

L' algorithme d'optimisation utilisé par le système SEL est présenté dans l'Annexe , chapitre 2 .

Signalons que les vendeurs proposent rarement plus d'une valeur par CE . En outre , ils sont généralement réticents à fournir le détail de leurs prix .

De ce fait , la fonction d'optimisation est peu employée en pratique ; c'est pourtant l'intérêt commun du vendeur et de l'utilisateur d'avoir la meilleure offre possible :

- pour le vendeur , afin d'augmenter sa probabilité de sélection
- et pour l'utilisateur , afin d'avoir toujours pour son problème le mieux de ce que les vendeurs peuvent offrir (1) .

Nous pensons qu'avec le développement de modèles comportant une fonction d'utilisation au sens exposé ici , on arrivera à ce que le vendeur et l'utilisateur discutent plus fréquemment de l'optimisation et de l'offre .

(1) L'auteur T. Scharf déclare quec' est une erreur de croire que le vendeur propose toujours le mieux de ce qu'il a à offrir [16 , 17].

3.4.3 / L'analyse de sensibilité

L'analyse de sensibilité consiste à étudier dans quelle mesure l'efficacité des critères est influencée par des modifications de l'arbre et/ou des valeurs d'entrée (1) .

Les paramètres modifiables dans l'arbre sont :

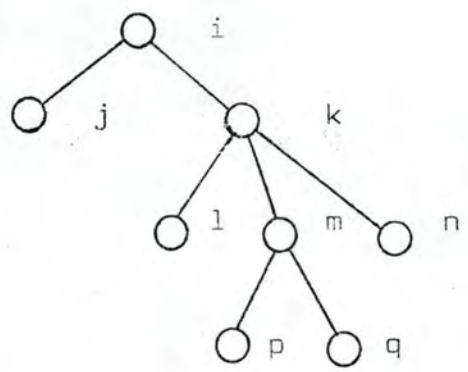
- les valeurs d'entrée
- les fonctions d'efficacité
- les poids
- les opérateurs d'aggrégation
- les critères (en qualité et en quantité)
- enfin , tout groupe d'au moins deux éléments des groupes précédents .

Les développements théoriques de l'analyse de sensibilité mènent à des formules lourdes .

Par conséquent , il est préférable de procéder à plusieurs utilisations de CAT avec différentes valeurs du (des) paramètre(s) litigieux .

La méthode CAT ne formalise donc pas la modification des paramètres de l'arbre; elle se limite à étudier l'efficacité d'un CNE comme fonction de l'efficacité d'un de ses critères subordonnés , dans l'hypothèse où tous les autres paramètres de l'arbre restent constants .

Illustrons cette fonction par un exemple se fondant sur l'arbre suivant :



(1) Sous réserve de la note (1) page 22 .

Etant donné cet arbre , l'efficacité du critère i est :

Ei = f (Ej , Ek)

Ei = f (Ej , El , Em , En)

Ei = f (Ej , El , Ep , Eq , En)

où f est la fonction d'aggrégation décrite à la page
et les variables Ei , Ej , etc, sont les efficacités des critères i , j , etc .

Lors de l'évaluation , les variables Ej , El , Ep , Eq , En , et Ei obtiennent des valeurs que nous notons Ej° , El° , Ep° Eq° , En° et Ei° .

Faisons varier Eq de 0 à 100 en maintenant tous les autres paramètres constants ; on obtient ainsi la fonction de sensibilité de l'efficacité du critère i à l'efficacité du critère q

Ei = f (Ej° , El° , Ep° , Eq , En°)

d'où

Ei = f (Eq) / Ej° à En° et les paramètres donnés de l'arbre .

Le critère q est un CE dans notre exemple , mais il peut tout aussi bien être un CNE , auquel cas le sous-arbre dont il est la racine n'a aucune influence dans la fonction de sensibilité Ei = f (Eq) .

De même, les critères ascendants de i sont sans influence sur cette fonction , tout comme leurs subordonnés autres que i .

Un exemple de deux fonctions de sensibilité est présenté au chapitre 5 , pages 55 à 73 .

3.4.4 / L'actualisation

L'actualisation consiste à calculer la valeur actuelle d'un investissement formé d'un ou de plusieurs cash-flows .

Chaque cash-flow est composé d'un montant financier C - dépenses ou recettes (1)- qui échoit tous les m mois durant la période des mois p à k , comptés de façon à ce que le mois actuel ait le numéro zéro (2) .

Avec le taux d'actualisation annuel r , le taux d'actualisation q pour la période de m mois est :

$$q = r \cdot \frac{m}{12}$$

Ainsi , si le taux d'actualisation annuel est de 10 % , un paiement biennal sera actualisé au taux de 20 % , et un paiement semestriel au taux de 5 % .

La valeur actualisée VA ou cash-flow est alors donnée par la formule classique :

$$VA = C \cdot \frac{q^{k+1} - q^{p}}{q - 1}$$

La valeur actualisée d'un investissement formé de n cash-flows décrits chacun par les valeurs Ci , mi , pi , ki et ri est :

$$VA = \sum_{i}^{n} Ci \cdot \frac{qi^{ki+1} - qi^{pi}}{qi - 1} \qquad \text{où} \quad qi = ri \cdot \frac{mi}{12}$$

La fonction d'actualisation peut être utilisée indépendamment des autres fonctions de CAT , et réciproquement , on utilise souvent ces autres fonctions indépendamment de l'actualisation .

(1) En général , il s'agit de dépenses . On peut toutefois avoir des recettes (remboursement(s) d'acompte(s) , prime(s) , etc) .

(2) p et, ou k peu(ven)t être négatif(s) ou nul(s) si des échéances commencent dans le passé ou actuellement .

La méthode CAT a également adopté l'étude d'un investissement sous l'angle du taux de rentabilité interne , qui consiste simplement à étudier sa valeur actualisée en fonction du taux d'actualisation [44] .

Un exemple d'actualisation est présenté au chapitre 5 , pages 55 à 73 .

C H A P I T R E 4

LE SYSTEME SEL

Le présent chapitre a pour but de présenter le système SEL , càd l'outil informatique de la méthode CAT .

Nous nous limitons ici aux caractéristiques essentielles du système SEL . Pour un exposé détaillé de ce dernier , nous renvoyons à l'Annexe , chapitre 3 .

4.1 / Présentation

Le système SEL constitue l'implémentation de la méthode CAT ; il a été entièrement conçu et programmé par le Professeur J.J. DUJMOVI Docteur de l'Université de Belgrade (Yougoslavie) .

Celui-ci a ensuite présenté le système SEL comme partie intégrante de sa thèse de doctorat (1974) .

Le système SEL est un programme Fortran de plus de 4000 cartes , qui constitue un interpréteur ; celui-ci implante un langage de programmation spécialisé , le langage SEL .

4.2 / Le langage SEL

Le langage SEL permet la programmation rapide de toutes les fonctions de la méthode CAT , à cette limite près que la seule analyse de sensibilité implantée est celle de l'efficacité d'un CNE (1) à l'efficacité d'un critère subordonné .

Ces fonctions , exposées au chapitre 3 sont

- l'évaluation qui calcule les efficacités des critères
- l'optimisation (CEM) qui calcule les courbes d'efficacité maximale des critères

(1) Rappelons que C(N)E signifie ' critère(s) (non-) élémentaire(s) '

- l'analyse de sensibilité de l'efficacité d'un critère à l'efficacité d'un critère subordonné

- et l'actualisation d'un investissement formé d'un ou de plusieurs cash-flows .

Ces quatre fonctions sont l'objet des instructions EFFECTIVENESS , OPTIMIZE , SENSITIVITY et PRESENT VALUE .

On ne peut toutefois les utiliser sans l'aide d'instructions auxiliaires (2) . Ainsi , l'instruction EFFECTIVENESS doit être précédée de l'entrée de l'arbre (instruction INPUT) et de la lecture des valeurs d'entrée pour les CE (instruction READ) .

De même , l'instruction OPTIMIZE doit être précédée de l'entrée des CEM des CE (instruction DATA) , et suivie de l'allocation d'un (de) budget(s) à l'offre (instruction ALLOCATE).

D'autre part , on ne conçoit pas de langage de programmation , même spécialisé , sans la possibilité de faire quelques calculs , des branchements et éventuellement des boucles (instructions ARITHMETIC , BRANCH et REPEAT) .

On souhaite aussi pouvoir imprimer des données et / ou des résultats , ce qui est assuré par les instructions OUTPUT , TITLE , PRINT et DISPLAY .

Enfin , l'instruction STOP termine l'exécution des instructions , tandis que la dernière instruction physique doit être END .

Les 17 instructions du langage SEL sont utilisés par le programmeur pour résoudre son problème d'évaluation .
Le programme-source qui en résulte est géré par les 9 commandes du langage SEL .

Ces commandes sont les suivantes :

- *JOB qui doit toujours être la première commande

- *SEL qui lance la traduction du programme-source en programme INSEL (1)

(1) Le programme INSEL est produit par la traduction du programme-source ; il est différent du code-machine et est contenu dans la mémoire interne de l'interpréteur .

- *EXECUTE qui lance l'exécution du programme INSEL
- *STORE qui sauve le programme INSEL dans un des 10 fichiers de l'utilisateur (Voir le début de la note 1)
- *READ qui lit le programme INSEL dans un des 10 fichiers de l'utilisateur
- *DELETE qui détruit le programme INSEL dans un des 10 fichiers de l'utilisateur
- *PRINT qui imprime le programme INSEL
- *NEW PAGE qui fait passer le listing à la page suivante
- et *EXIT qui doit toujours être la dernière commande .

Les commandes et les instructions du langage SEL sont présentées dans l'Annexe , chapitre 2 .

Deux programmes en langage SEL sont présentés au chapitre 5 ci-après , pages 55 à 73 .

4.3 / La conception du système SEL

Le système SEL est rédigé sous forme modulaire ; il contient 40 modules , dont un programme principal et 39 sous-routines .

Ces sous-routines sont : un moniteur pour l'exécution des commandes , trois traducteurs pour traduire le programme-source , six réalisateurs pour exécuter le programme INSEL et 29 sous-routines auxiliaires qui exécutent des tâches limitées mais assez fréquentes (Figure 9) .

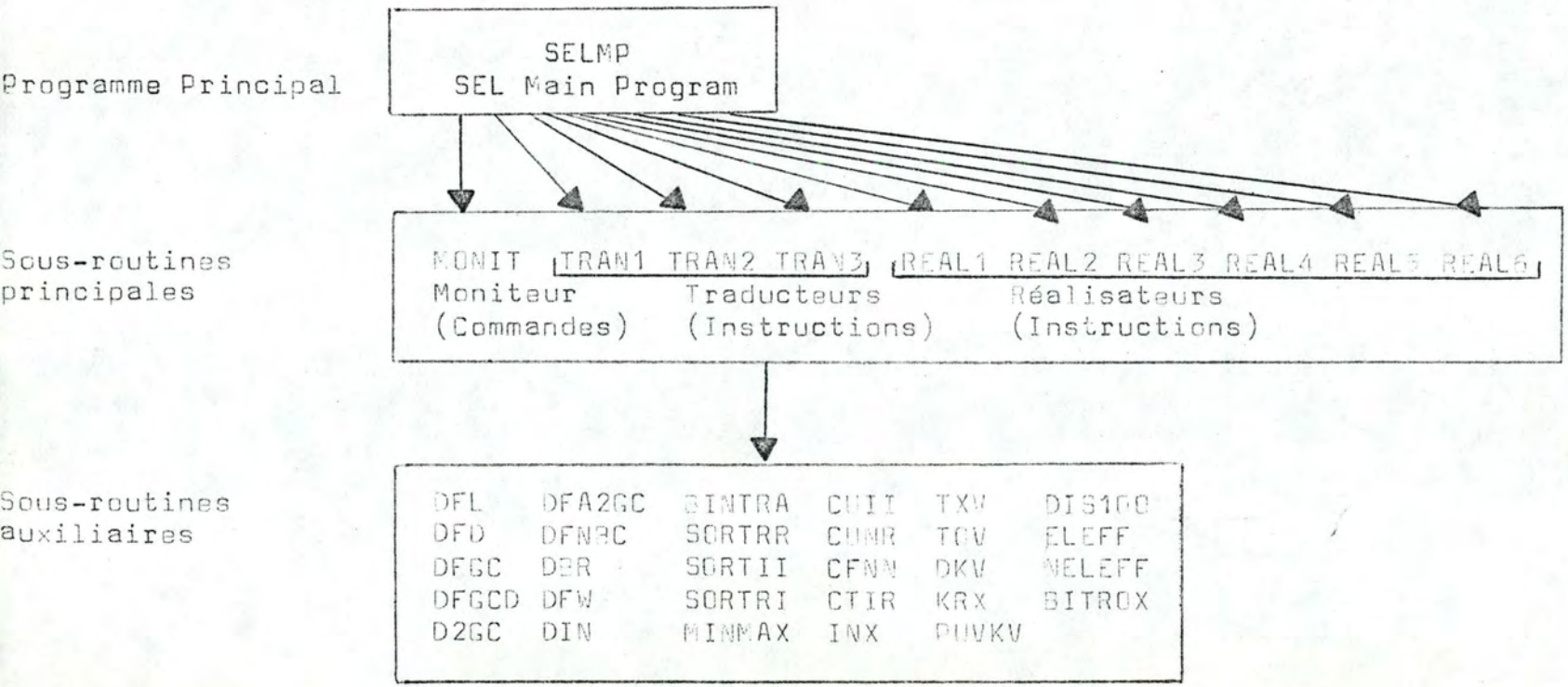
Le programme principal a pour but de créer la zone COMMON , de définir les fichiers de l'interpréteur (1) , d'initialiser les paramètres de l'interpréteur , et de gérer les appels successifs au moniteur , aux traducteurs et aux réalisateurs .

(1) Ces fichiers sont au nombre de 16 . L'utilisateur n'a accès dans son programme qu'aux dix premiers , qui forment la mémoire externe de l'interpréteur ; ils servent à conserver le programme INSEL . Les 6 autres fichiers forment une partie de la mémoire interne de l'interpréteur .

FIGURE 9

La structure du système SEL

Note . Les flèches indiquent les appels Fortran (CALL).
Nous représentons par une seule flèche les nombreux appels
des sous-routines principales aux sous-routines auxiliaires .



Les traducteurs ont pour but de traduire le programme-source en programme INSEL ; celui-ci est exécutable immédiatement (Voir la commande *EXECUTE) . Les traducteurs sont appelés par le programme principal , après que le moniteur ait détecté la commande *SEL . et ait opéré son RETURN au programme principal .

Les réalisateurs ont pour but d'exécuter le programme INSEL . Ils sont appelés par le programme principal , après que le moniteur ait détecté la commande *EXECUTE . et ait opéré son RETURN au programme principal .

Les sous-routines auxiliaires ont pour but de simplifier la programmation et la modification du système SEL . Leur fonction est soit limitée (telle la sous-routine DFL qui détecte la première lettre dans une zone de vecteur) , soit assez évoluée (telle la sous-routine DIS100 qui dessine les graphes cartésiens de plusieurs fonctions) .

L'analyse et la programmation du système SEL sont présentées dans l'Annexe , chapitres 3 et 4 .

Il nous paraît cependant utile de préciser dans le paragraphe ci-dessous les limites auxquelles le programmeur est confronté dans l'utilisation du langage SEL .

4.4 / Les limites du système SEL

Les limites auxquelles le programmeur de langage SEL est confronté sont les suivantes :

- le nombre de critères élémentaires ne peut dépasser 100
- le nombre de critères non-élémentaires ne peut dépasser 90
- le nombre de points critiques d'un CE ne peut dépasser 5 (1) . ni être inférieur à 2 ; de même pourles critères immédiatement subordonnés à un CNE .
- le nombre total de points critiques de tous les CE plus le nombre total des critères immédiatement subordonnés de tous les CNE ne peut dépasser 780 .

(1) Fait sans gravité , grâce à la propriété d'associativité de la formule d'aggrégation (Voir pages 34 à 35) .

En outre , au sein du programme-source :

- le nombre d'instructions ne peut dépasser 99

- le nombre de commandes est illimité

- la longueur totale des titres ne peut dépasser 1037 caractères

- la longueur totale du programme INSEL ne peut excéder 400 entiers . Ce nombre est la somme des longueurs des instructions du programme INSEL ; il est calculé sur la base suivante : 1 pour STOP ; 2 pour INPUT , OUTPUT et PRESENT VALUE ; 3 pour EFFECTIVENESS , DISPLAY et DATA ; 4 pour TITLE , REPEAT , READ , PRINT , SENSITIVITY et OPTIMIZE ; 5 pour ALLOCATE ; 7 pour ARITHMETIC et 9 pour BRANCH .

Accessoirement , notons que le système SEL permet de calculer des moyennes arithmétiques , mais pas géométriques ni harmoniques . En effet , ces trois cas correspondent dans la formule d'aggrégation (page 26) aux valeurs 1 , 0 et -1 du coefficient d'aggrégation R . La première valeur seule correspond à en des opérateurs d'aggrégation définis à la page 28 (A) .

CHAPITRE 5

DEUX EXEMPLES

Le présent chapitre comporte deux exemples d'utilisation de la méthode CAT et du système SEL .

Le premier exemple - réel - est le choix d'un ordinateur pour le Centre PIRC (1) .

Le second exemple - artificiel - est simplement destiné à illustrer les possibilltés de SEL que le premier exemple n'a pas utilisées .

5.1 / Le cas PIRC

Le cas PIRC consiste à créer un Centre d'Informatique à Pirot (Yougoslavie) , selon la démarche exposée au chapitre 3 . Nous ne considérons ici que le processus de sélection d'une des offres.

L'appel d'offres a apporté 5 propositions :

-FACOM 230/38S de FUJITSU
-HONEYWELL 64/20
-IBM S/3-15
-IBM 370/115-2
-et UNIVAC 90/30 .

Cet appel d'offres se fondait sur les caractéristiques exposées au chapitre 5 , pages 58 à 73 .

Les caractéristiques générales des systèmes proposés sont les suivantes (3) :

(1) PIRC - prononcez pirtss - est le sigle de ' PIROTSKI RAČUNSKI CENTAR'(Centre de Calcul de Pirot , Yougoslavie) .
(2) Ces systèmes comportent aussi un ou plusieurs lecteurs de disquettes ou cassettes. Au niveau des terminaux , ils ont tous la possibilité d'avoir des disquettes ou des cassettes .
(3) Par discrétion, nous citons désormais ces systèmes par des numéros anonymes ; ceux-ci ne reflètent pas l'ordre de la liste à l'alinéa précédent .

SYSTEME	MEMOIRE CENTRALE (Kby)	LECTEUR DE CARTES (cartes / min)	LECTEUR DE DISQUETTES OU DE CASSETTES	IMPRIMANTE (lignes / min)	VITESSE DES BANDES MAGNETIQUES (Kby / sec)	CAPACITE DES DISQUES (Gby)	TERMINAUX	
							ECRANS (caract.)	SUPPORTS MAGNETIQUES
1	128	300	1	900	2•43.2	4•70	240	Disqut.
2	128	600	0	800	2•30	3•70	1024	Casset.
3	128	néant	1	1200	2•40	4•70	256	Disqut.
4	128	néant	1	600	2•40	4•20 + 1•5	256	Disqut.
5	128	500	0	500	2•40	4•30	1024	Casset.

Pour les prix de ces systèmes , voir la page 67 .

Notre objet est ici de présenter la démarche concrète de CAT pour choisir l'une de ces offres .

Dans le cadre de ce chapitre , nous nous limitons à un exposé succint des principaux résultats, et nous renvoyons à l'Annexe pour les listings complets .

Nous présentons ci-après l'arbre du cas PIRC avec une inscription sommaire des critères , le programme SEL de ce problème , et les résultats finaux : efficacité globale et taux d'efficacité/coût .

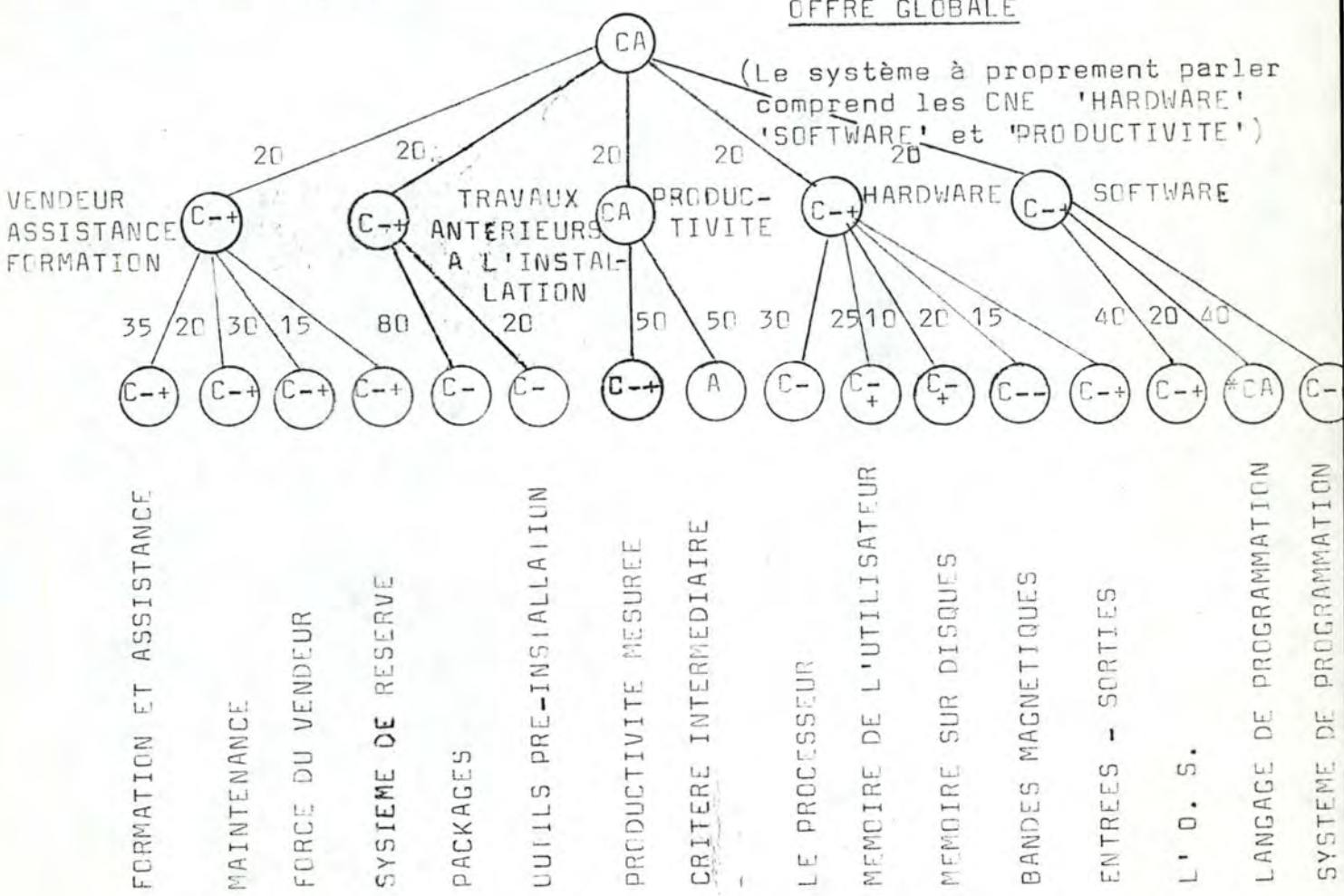
L'arbre utilisé pour PIRC

L'arbre utilisé pour PIRC comporte 83 critères élémentaires et 36 critères élémentaires , soit un total de 129 critères.

Par lisibilité , nous le présentons de manière 'top down', càd en commençant par la racine (Figure 10) .

FIGURE 10

Les niveaux supérieurs de l'arbre utilisé pour PIRC.
Les cercles contiennent l'opérateur d'aggrégation (CNE) ou le numéro du critère (CE) .



Nous présentons ci-après l'arbre du cas PIRC par les sous-arbres qui se rattachent aux sommets pendants de la Figure , page .

Auparavant, illustrons la façon de fixer les poids et les opérateurs d'aggrégation dans les deux cas d'Absorption Totale (AT) et d'Absorption Partielle (AP) .

Examinons à cette fin le CNE PRODUCTIVITE , pour lequel les CE sont :

- le résultat des benchmarks en monoprogrammation (CE n° 39)
- le résultat des benchmarks en multiprogrammation (CE n° 40)
- la vitesse du processeur (CE n° 41)
- la consommation de mémoire par les benchmarks (CE n° 42).

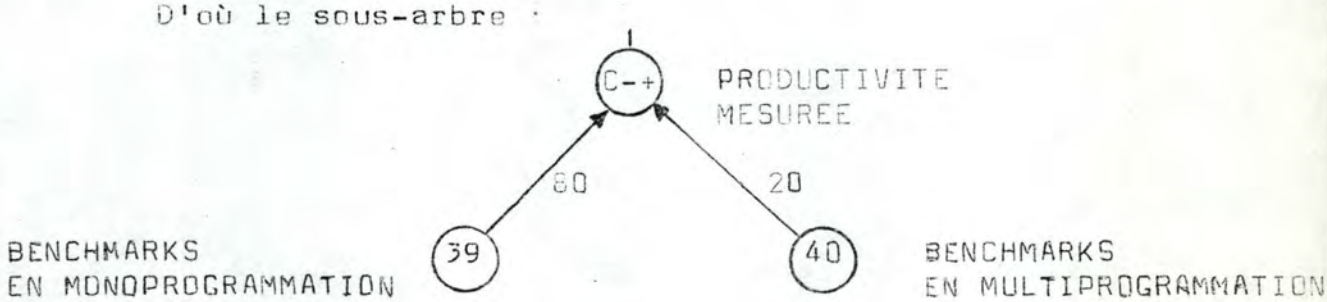
Absorption Totale

D'abord , on décide de créer un CNE PRODUCTIVITE MESUREE avec les CE 39 et 40 .

On estime leur importance relative - pour le cas PIRC - de 4 à 1 , d'où les poids 80 et 20 % .

En outre , les deux modes monoprogrammation et multiprogrammation étant indispensables au problème , on a une conjonction sine qua non ; on l'estime faible, d'où C-+ .

D'où le sous-arbre :



Absorption partielle

A ce moment , on estime la PRODUCTIVITE globale du système comme égale à cette PRODUCTIVITE MESUREE , mais tempérée par les CE 41 et 42 .

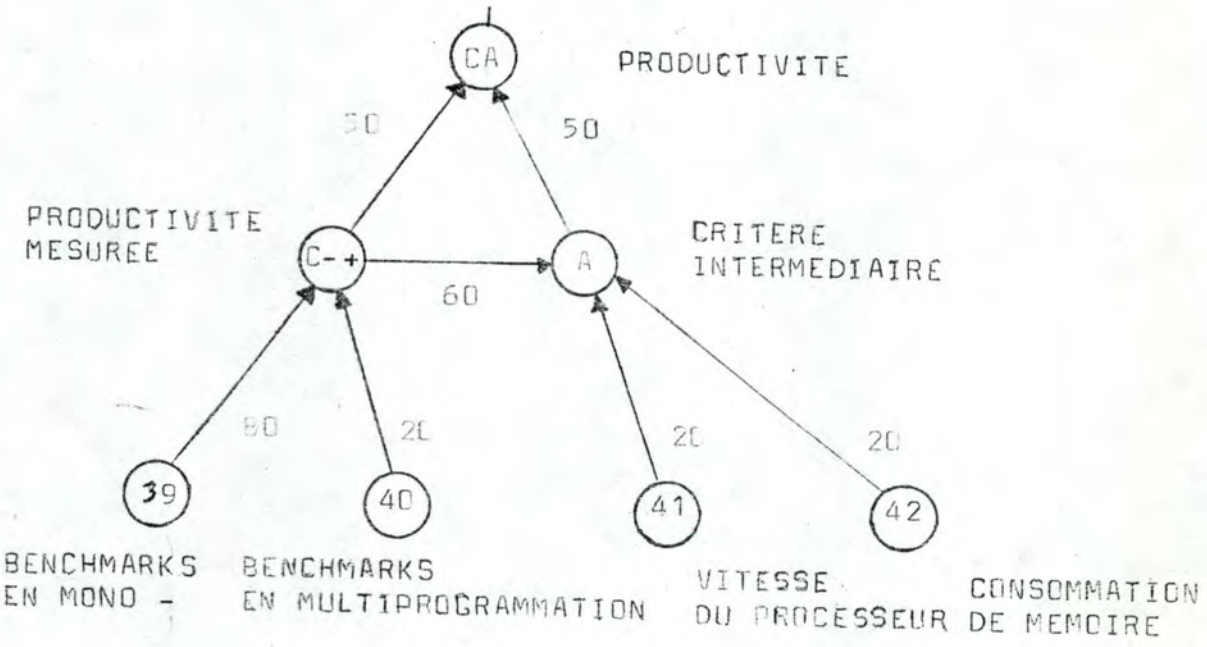
D'où l'absorption partielle à trois entrées , ce qui oblige à la programmer (1) : on aggrège d'abord les trois critères avec l'opérateur d'aggrégation A (?) .

Comme on estime la productivité mesurée plus importante que les deux CI réunis , on la pondère à 60 % . Les deux autres critères se valent en gros , on les pondère chacun à 20 % .

Ensuite , on aggrège la PRODUCTIVITE MESUREE et le CRITERE INTERMEDIAIRE avec les poids 50 et 50 % respectivement .

Ces deux critères - et a travers elles , les trois critères initiaux - sont indispensables au problème ; on a une conjonction sine qua non . que l'on estime moyenne : CA .

D'où le sous-arbre :



Nous présentons ci-après les autres sous-arbres , dans l'ordre croissant des numéros de CI . Par clarté , nous remplaçons les arcs par des arêtes .

(1) Rappelons que seule l' AP à 2 entrées est implantée dans le système SEL .
(2) Rappelons qu'en AP , il existe 2 opérateurs d'aggrégation et 2 séries de poids à fixer . Pour simplifier , on fixe conventionnellement l'opérateur d'aggrégation à ' A ' , et deux poids finaux à 50 et 50 % .

FORMATION ET ASSISTANCE
POSSIBILITES DE FORMATION
FORMATION QUALITATIVE
ASSISTANCE
MAINTENANCE EXIGEE
MAINTENANCE SOUHAITEE
C+
C-
CA
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
20
30
30
20
50,50
50
60
40
75
25
80
10
10
10
60
10
30
80
20
20
40
40
FORMATION DE PROGRAMMEURS
FORMATION D'ANALYSTES DE SYSTEME
FORMATION DE PROGRAMMEURS DE SYSTEME
FORMATION D'OPERATEURS
ORGANISATION DES COURS
MANUELS DE COURS
ASSISTANCE D'ING. DE SYSTEME
REVOCABILITE DES ING. DE SYSTEME
POSSIBILITE DE CHOISIR LA PERIODE D'ASSISTANCE
DELAI D'ARRIVEE SUR APPEL
MAINTENANCE LES JOURS NON-OUVRABLES
DISTANCE DU DEPOT DE PIECES DE RECHANGE
MAINTENANCE POUR LES PIECES ETRANGERES AU VENDEUR
PERIODE DE COUT FIXE POUR LA MAINTENANCE
PERIODE DE MAINTENANCE GARANTIE

DISTANCE INTER-SIEGES VENDEUR-UTILISATEUR

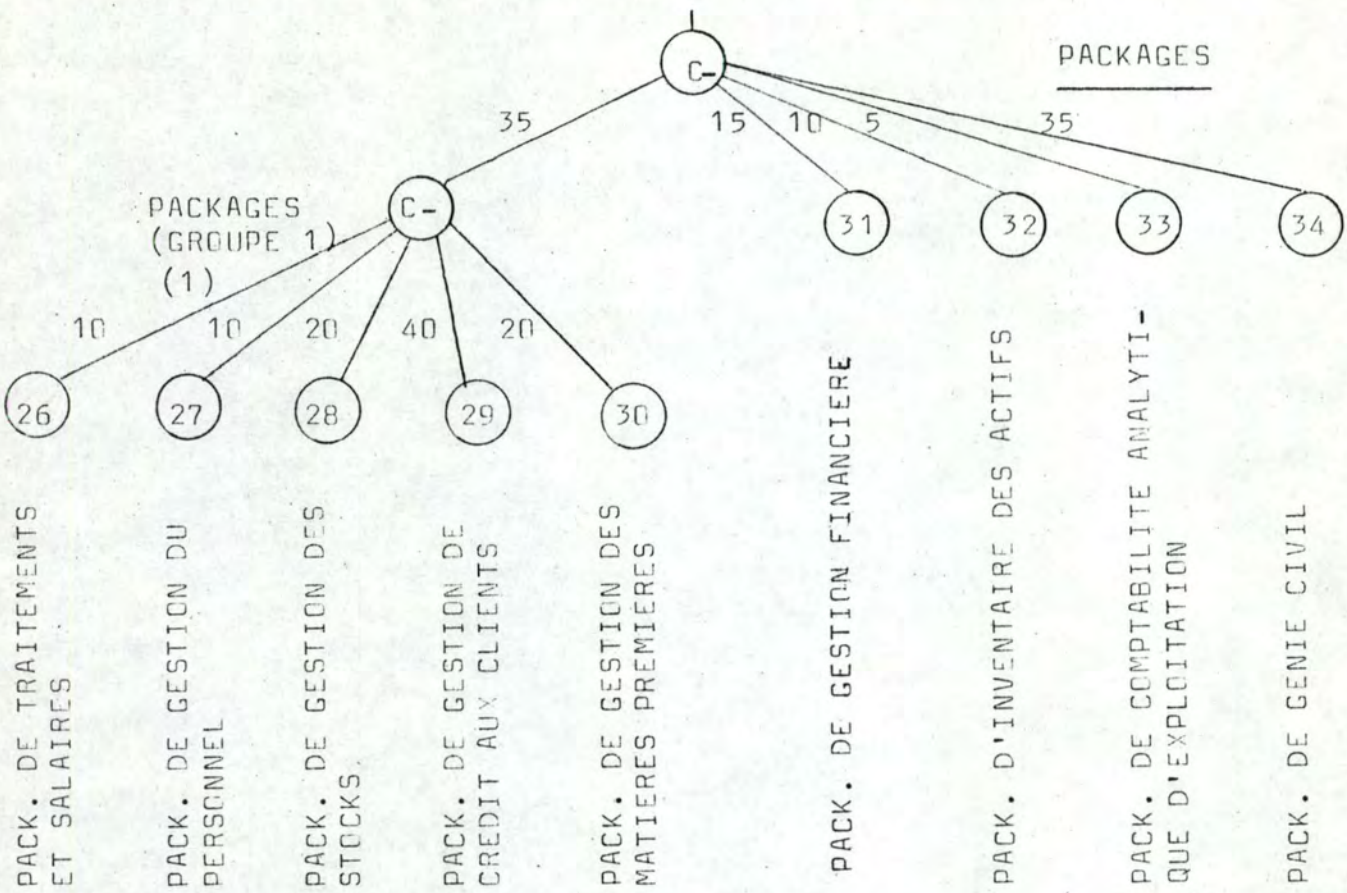
NOMBRE D'ING. DE SYSTEME

NOMBRE DE TECHNICIENS DE MAINTENANCE

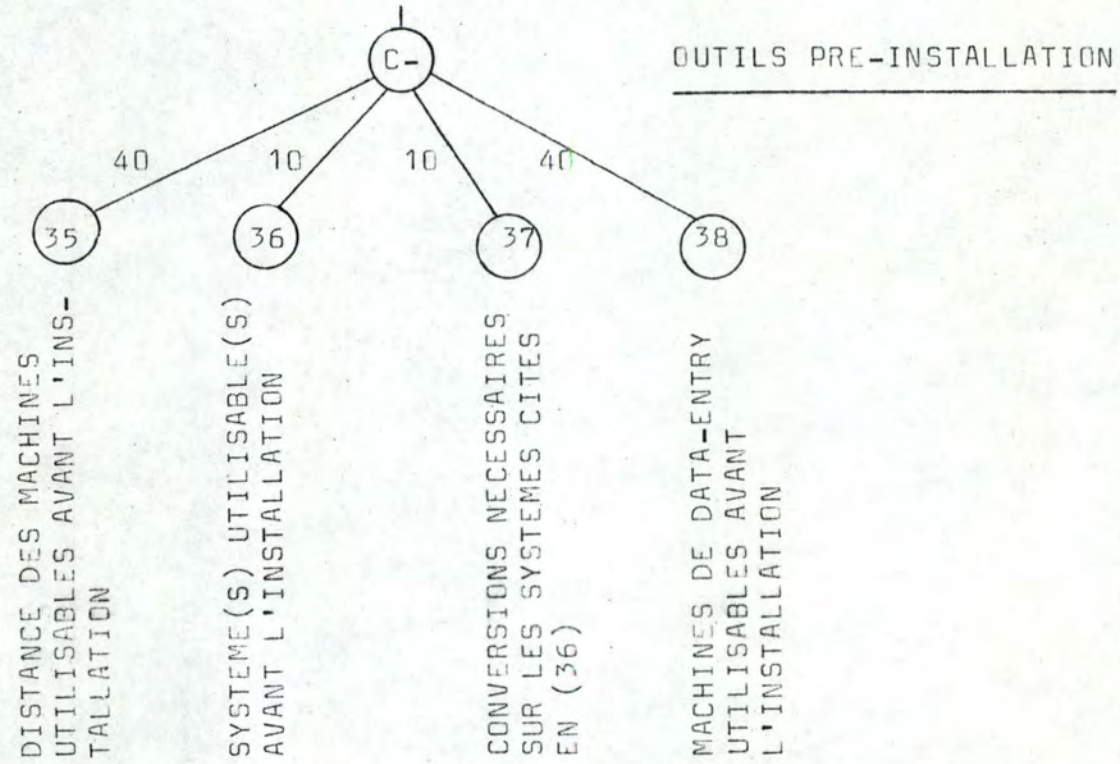
ORGANISATION GENERALE DU VENDEUR

EXECUTION DES BENCHMARKS

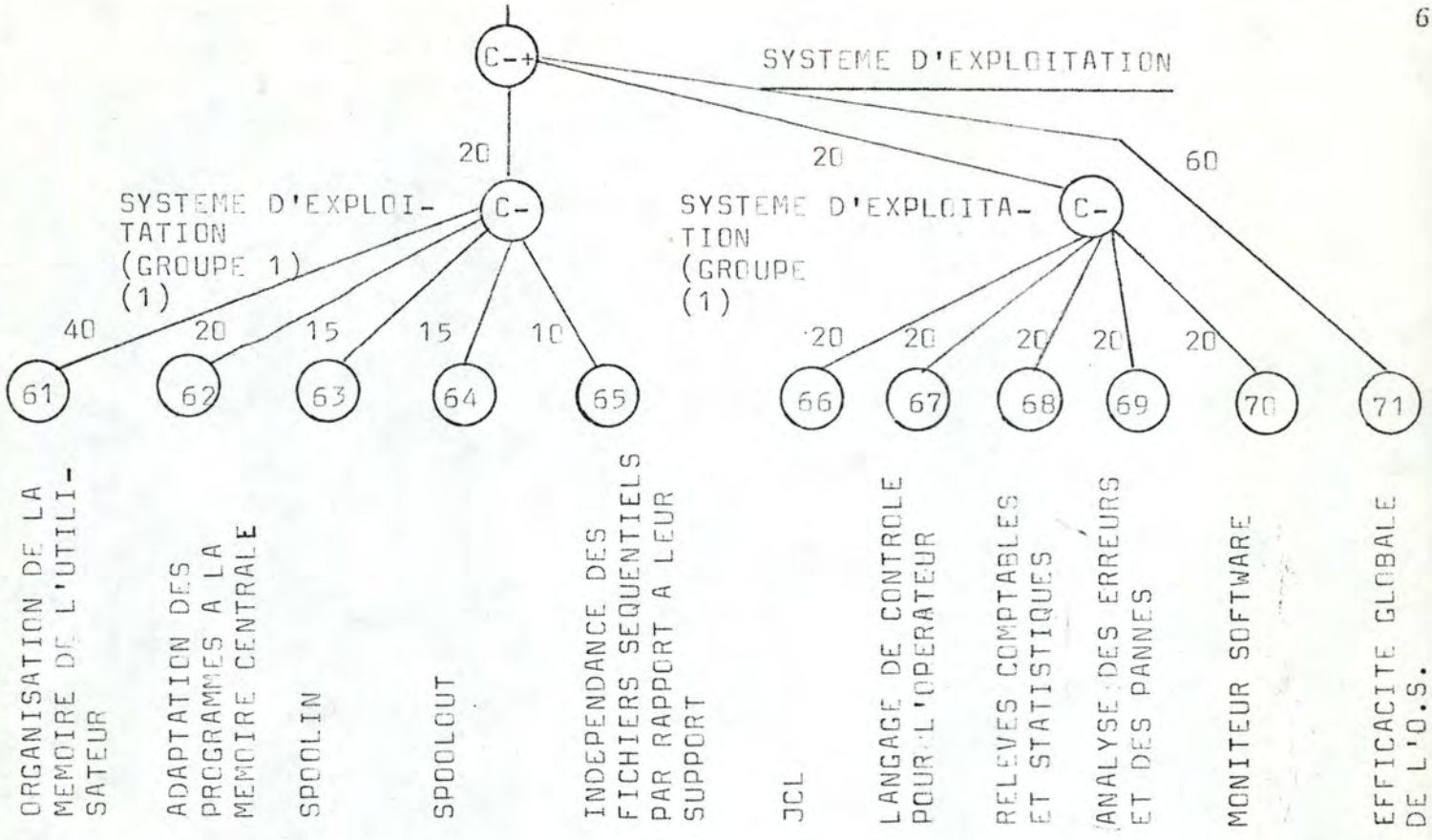
ADOPTION DES BENCHMARKS COMME TEST DE VALIDATION



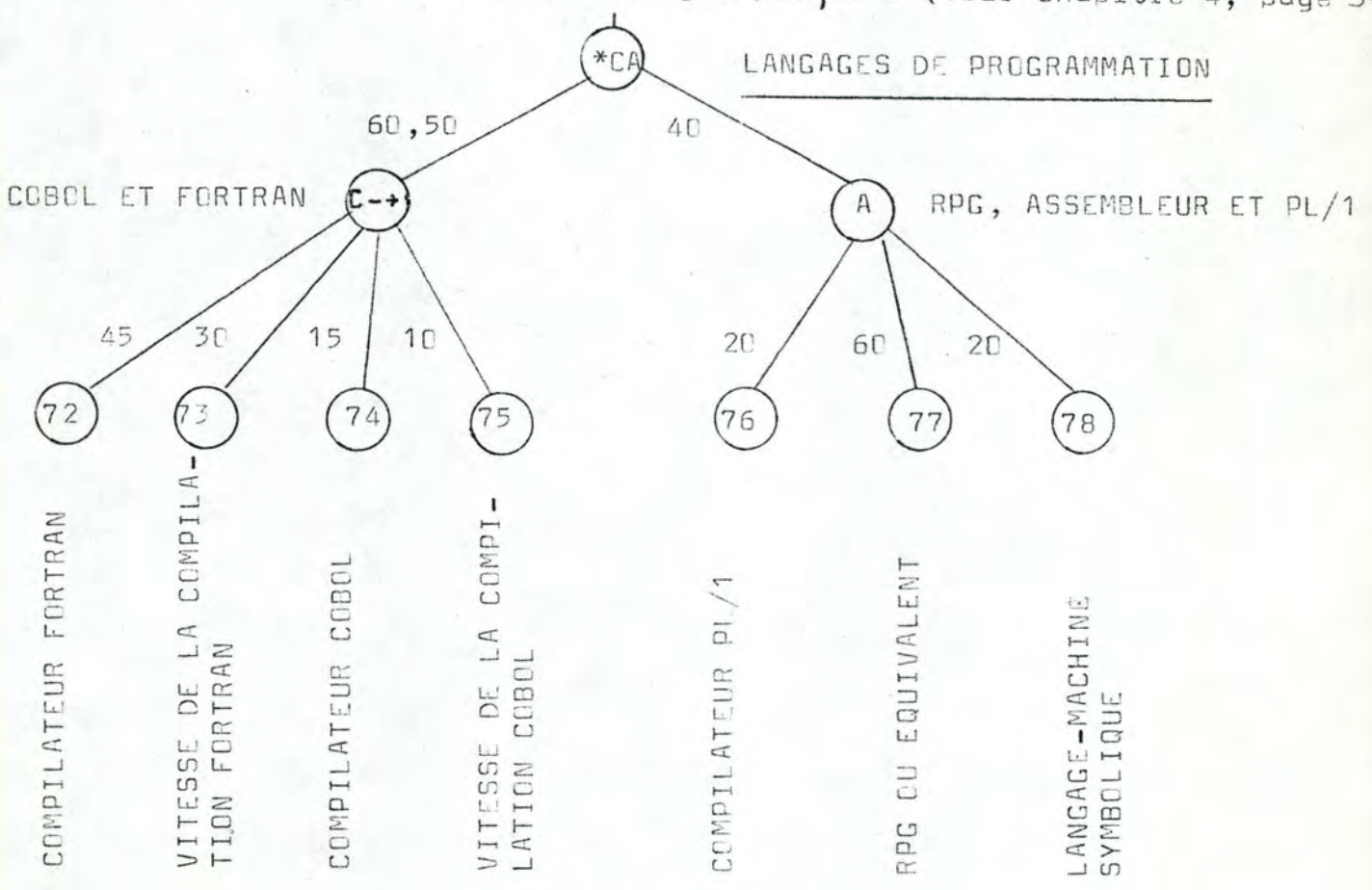
(1) Le systeme SEL ne permet pas d'aggréger simultanément 9 critères ; on les divise donc en deux groupes, de 5 et 4 (voir chapitre 4, pages 34 à 35)

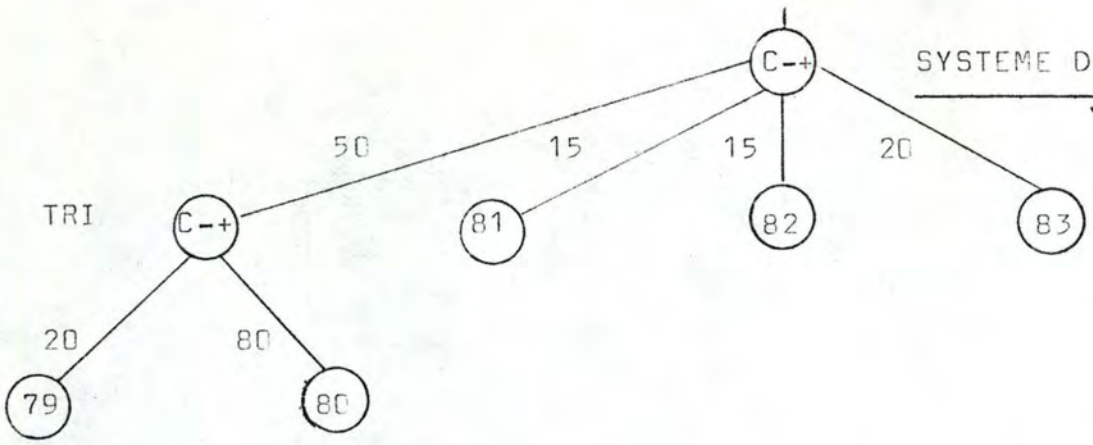


REGISTRES D'INDEX
ACCUMULATEURS
REPERTOIRE DES INSTRUCTIONS-MACHINES
ADRESSES DANS LES INSTRUCTIONS
FORMATS DES INSTRUCTIONS
43
44
45
46
47
15
15
30
20
20
C--
PROCESSEUR
TAILLE DE LA MEMOIRE CENTRALE
TAILLE MAXIMUM DE LA MEMOIRE CENTRALE
48
49
80
20
C-+
MEMOIRE DE L'UTILISATEUR
MECANISMES D'ACCES
CAPACITE AVEC UN ACCES EN PANNE
TAILLE MAXIMUM DE LA MEMOIRE SUR DISQUES
50
51
52
45
45
10
C-+
MEMOIRE SUR DISQUE
VITESSE DE TRANSFERT
DENSITE (BPI)
MONTAGE AUTOMATIQUE
53
54
55
70
20
10
C--
MEMOIRE SUR BANDES MAGNETIQUES
PREPARATION ET ENTREE DES PROGRAMMES
PREPARATION ET ENTREE DES DONNEES
VITESSE D'IMPRESSION
PRINCIPE DE CONSTRUCTION DE L'IMPRIMANTE
SORTIES POUR LE PUPITREUR
56
57
58
59
60
15
20
50
5
10
C-+
ENTREES-SORTIES



(1) Le système SEL ne permet pas d'aggréger simultanément 10 critères on les divise donc en deux groupes de 5 (voir chapitre 4, page 34





PROGRAMMES DE TRI

EFFICACITE DU TRI

STRUCTURES DE DONNEES

SYSTEMES DE BASES DE DONNEES

UTILITAIRES ET PROGRAMMES STANDARTS

L'ensemble de ces composantes forme l'arbre utilisé dans le cas PIRC .

Pour procéder à l'évaluation des offres, il faut entrer cet arbre en mémoire (instructions INPUT) , puis il faut lire le nom du système offert (TITLE) et les valeurs d'entrée des critères élémentaires (READ) ; ensuite , on calcule les efficacités de tous les critères (EFFECTIVENESS) , et on sort les résultats (OUTPUT) . Comme il y a 5 offres, il faut réitérer 4 fois le processus (REPEAT après quoi on a terminé (STOP) .

Ces instructions sont gérées par des commandes : *JOB lance le travail , *SEL, LIST traduit et liste les instructions, *EXECUTE les exécute et *EXIT achève le travail .

Le programme SEL utilisé par le cas PIRC est donc le suivant :

```
*JOB
*SEL , LIST
  INPUT ELC
  INPUT CAS
1 TITLE 2 , 1
  READ ( 1,K99 ) , FORMAT 5
  EFFECTIVENESS
  OUTPUT (E) SORT , HISTOGRAM
  REPEAT 1,4
  STOP
  END
*EXECUTE
          ⎧CE
   data   ⎨CNE
          ⎩Valeurs des CE
*EXIT
```

Nous présentons maintenant les résultats finaux du cas PIRC sous forme d'un tableau (Figure 12) et d' un graphique conséquent (Figure 11) .

FIGURE 12

Graphique reprenant pour chaque offre du cas PIRC (sauf la 4° offre , dont l'efficacité globale est nulle) :

- l'efficacité globale et
- les coûts en achat comptant (.) ainsi qu' en achat à crédit (+) .

On constate que l'achat à crédit amène une augmentation uniforme des prix , ce qui laisse la liste rangée des offres , inchangée .

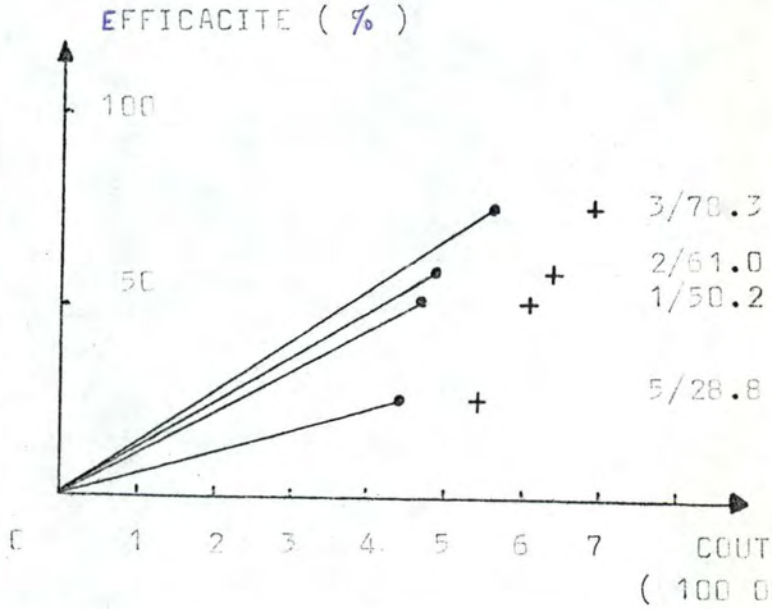


FIGURE 12

RESULTATS FINAUX DE L'EVALUATION DES OFFRES POUR PIRC

L ' efficacité nulle de la productivité en 4 annule les efficacités suivantes , car elles s'aggrègent en CA , càd en conjoncti sine qua non

SYSTEME	EVALUATIONS				COUTS EN COMPTANT , CREDIT , LEASING			TAUX EFF. / COUT	
	PRODUCTIVITE	PRODUCTIVITE , HARDWARE ET SOFTWARE	PRODUCTIVITE , HARDWARE , SOFTWARE , ASSISTANCE ET FORMATION	SYSTEME GLOBAL (COLONNE PRECEDENTE + TRAVAUX PRE-INSTALLATION)	COUT EN ACHAT COMPTANT (ACTUALISE)	COUT EN ACHAT A CREDIT (ACTUALISE)	COUT EN LEASING	EFF. / 10^5US $ COMPTANTS	EFF. / 10^5US $ A CREDIT
	%	%	%	%	US $	US $	US $	% / 10^5 US $	
1	93.0	80.0	65.5	50.2	474 404.0	605 166.9	627 012.8	10.6	8.3
2	67.5	69.9	70.0	61.0	499 739.2	637 485.4	635 137.5	12.2	9.6
3	69.7	73.6	75.7	78.3	569 722.0	695 072.6	Inexistant	13.7	11.2
4	0.0	0.0	0.0	0.0	non-examiné		Inexistant	0.0	0.0
5	71.9	70.6	66.5	28.8	439 771.0	532 466.2	616 392.0	6.5	5.4

5.2 / Exemple complémentaire à PIRC

L'exemple complémentaire à PIRC - nous dirons CPIRC (prononcez cépirtss) - est un cas artificiel destiné à illustrer les commandes et instructions SEL que le premier n'a pas utilisées .

Les deux grandes fonctions de CAT que PIRC a délaissées sont l'analyse de sensibilité et l'optimisation .

Le reste des instructions inutilisées sont auxiliaires , càd ne réalisent aucune fonction de CAT, sauf pour l'instruction PRESENT VALUE (calcul de valeur actualisée) .

Nous construisons donc un programme SEL qui réalise d'abord une évaluation complète ~~(1)~~ puis nous entrons les CEM (2) des critères élémentaires (instruction data dans la première boucle BRANCH) ; à ce moment, une seconde boucle BRANCH réalise l'optimisation (OPTIMIZE) , dont on imprime les résultats (DISPLAY) .

On passe alors à l'analyse de sensibilité du critère global à l' efficacité de chacun des autres (SENSITIVITY) .

Les résultats de l'optimisation , conservés par l'interpréteur SEL , permettent de trouver l'allocation optimale d'un budget de 4500 UM (3) , puis toutes les allocations possibles .

Enfin , un calcul de valeur actualisé est présenté .

Le programme SEL du cas CPIRC est le suivant (on remarquera la présence de toutes les commandes inutilisées par PIRC) :

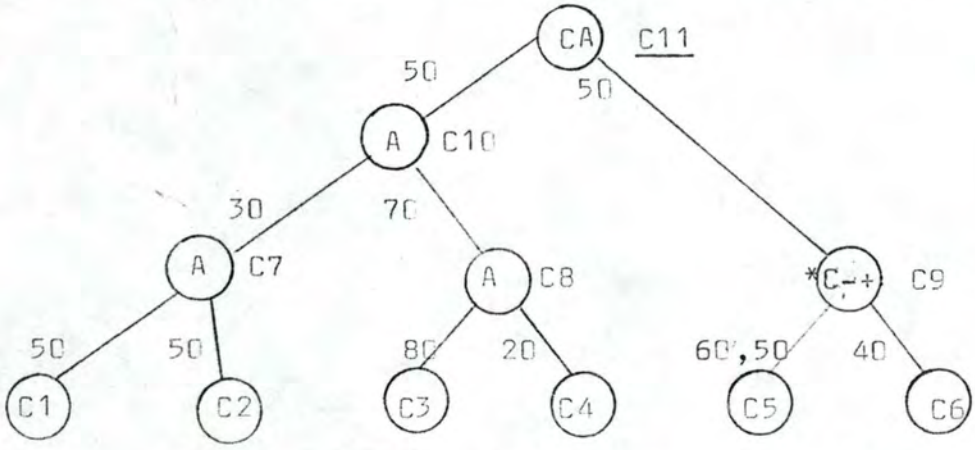
~~(1) Pour permettre à l'analyse de sensibilité et à l'optimisation de se faire~~ .

(2) Rappelons que 'CEM' signifie ' Courbe d'efficacité maximale ' .

(3) 'UM' signifie 'Unité(s) Monétaire(s)' .

```
*JOB
*SEL , LIST
  INPUT ELC
  INPUT CAS
  READ X (1,K99) , FORMAT 5
  EFFECTIVENESS
  OUTPUT (E)
1 K1 = 1
  DATA FOR (K1) PRINT
  BRANCH TO / 1 / K99,1
2 K1 = K99+1
  OPTIMIZE (K1) PRINT
  BRANCH TO / 2 / K100 , 1
  TITLE 3 , 'OPTIMISATION DES CNE'
  DISPLAY (OPTIMISATION 2)
  TITLE 3, 'ANALYSE DE SENSIBILITE'
  SENSITIVITY (ALL) DISPLAY , PRINT
  TITLE 3 , 'ALLOCATION DE 4500 UC'
  ALLOCATE 4500 (SUBSYSTEMZ = 11 , RESULT = PARAMETER)
  TITLE 3 'ALLOCATION GENERALE'
  ALLOCATE (SUBSYSTEMZ = 11 , RESULT = COST)
  PRESENT VALUE = K1
  STOP
  END
* NEW PAGE
* PRINT INSEL PROGRAM
* DELETE FILE 37
* STORE INSEL PROGRAM INTO FILE 37
* READ INSEL PROGRAM FROM FILE 37
* EXECUTE
    données
* EXIT
```

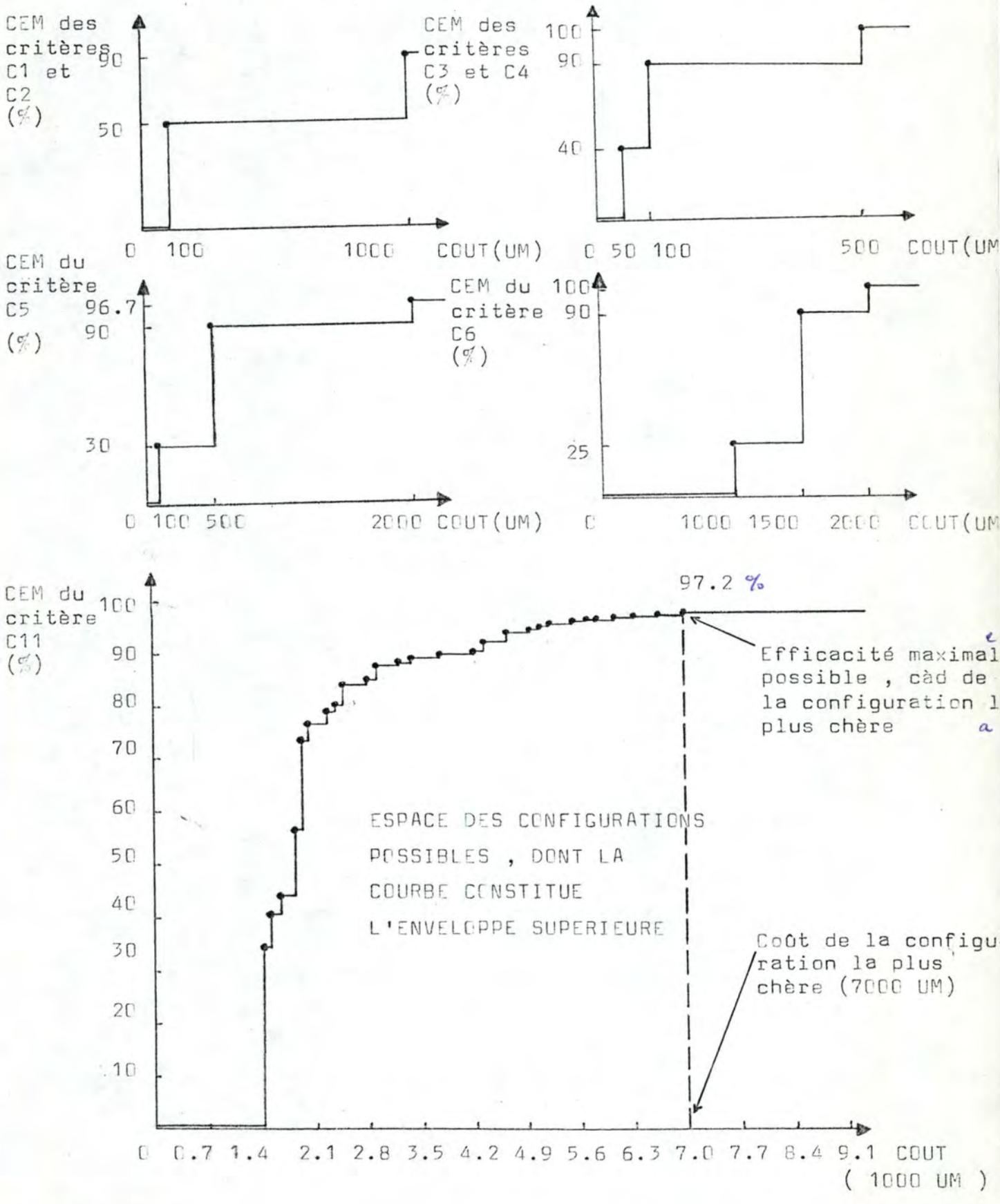
L'arbre utilisé par CPIRC est le suivant :



Cet arbre est binaire , ce qui lui permet de supporter l'optimisation (Voir paragraphe 3. 4. 2 , pages 39 à 44)

FIGURE 13

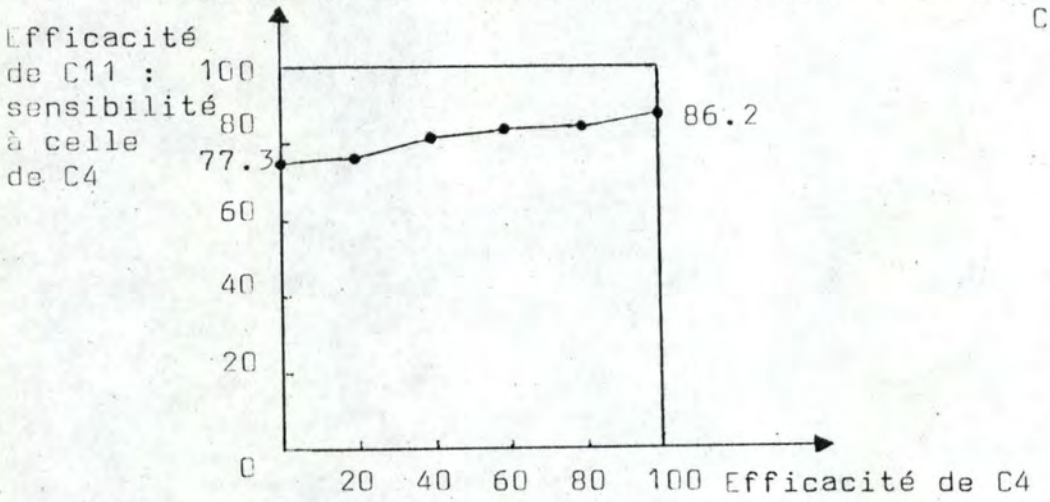
Courbes d'efficacité maximale des critères élémentaires (possibilités offertes) et du critère global (calculée)



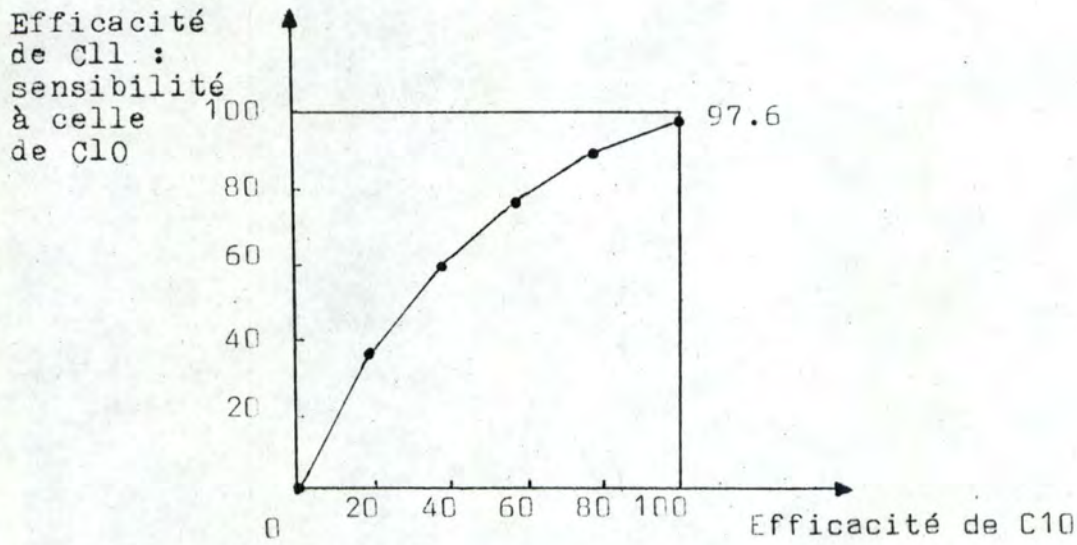
Ensuite , en se fondant sur les résultats de l'évaluation , on calcule la sensibilité du critère global C11 à chacun de ses subordonnés . Nous exposons à la Figure 14 les deux cas de sensibilité la plus faible et la plus forte ; il s'agit de la sensibilité de C11 à C4 et C10 respectivement .

FIGURE 14

Fonctions de sensibilité de l'efficacité de C11 à celles de C4 et C10



Les opérateurs d'aggrégation qui séparent C11 de C4 sont du type 'A' , càd ~~non conjonctif , et a fortiori~~ non conjonctif sine qua non ; cela permet à C4 d'avoir une efficacité nulle sans pour autant annuler l'efficacité de C11 , qui reste à 77.3 % .



L'opérateur d'aggrégation qui sépare C11 de C10 est du type 'CA' , càd conjonctif sine qua non ; si l'efficacité de C10 est nulle celle de C11 le sera donc aussi .

Nous arrivons enfin au calcul de valeur actualisée ; il s'agit d'un cas réel repris dans [51], à l'exception des dates rendues plus ... actuelles .

Le problème est d'actualiser l'ensemble des 5 cash-flows suivants , à un taux de 6 % par an :

- un acompte unique de 93 214.2 US $ pour le système sans le processeur (mars 78)
- un paiement semestriel de 131 836.2 US $ pour le système sans le processeur (septembre 78 à septembre 81)
- un acompte unique de 113 626.0 US $ pour le processeur (mars 80)
- un paiement semestriel de 147 150.8 US $ pour le processeur (septembre 80 à septembre 84) .
- un paiement mensuel de 6 430 US $ pour la maintenance , durant 7 années (septembre 78 à septembre 85) .

Les résultats fournis sont :

- la valeur totale non-actualisée : 3 785 189.5 US $
- la valeur totale actualisée au taux déclaré de 6 % par an : 3 054 372.0 US $
- l'ensemble des valeurs totales actualisées pour les taux d'actualisation de 0 % à 30 % par an , ce qui rend possible l'utilisation de la méthode du taux de rendement interne :

Taux d'actualisation	Valeur actualisée totale
0	3 785 189.5
1	3 647 806.0
2	3 517 040.0
3	3 392 683.5
4	3 274 349.5
5	3 161 686.0
6	3 054 372.0
7	2 952 109.5
8	2 854 615.5
9	2 761 638.5
10	2 672 917.5
11	2 588 233.0
⋮	⋮

28	1 587 549.2
29	1 558 308.2
30	1 520 574.5

C H A P I T R E 6

CAT FACE AUX AUTRES MODELES

Le présent chapitre examine les relations de la méthode CAT avec les autres modèles proposés pour le choix d'un ordinateur .

Ces modèles sont présentés succintement au chapitre 2 .

Dans chacun des paragraphes suivants , nous comparons CAT à l'un des autres modèles : l'approche ad hoc , les méthodes à coefficients , la technique Cost-Value , la méthode Electre 1 et la technique d'éliminations par aspects .

6.1 / CAT et l'approche ad hoc

L'approche ad hoc constitue par définition l'absence de vraie méthode ; dans ce cas , on a généralement une ou plusieurs personnes qui évaluent et/ou comparent les offres de façon mentale .

Les façons de s'y prendre peuvent être très différentes ; on s'accorde toutefois facilement sur les points suivants :

1 - il s'agit d'un problème multicritère
2 - les critères n'ont pas forcément la même importance
3 - les critères ne sont pas forcément indépendants
4 - si l'objet est complexe , l'approche ad hoc est peu fiable ; de ce fait , une formalisation du processus d'évaluation et de formalisation comparaison s'avère nécessaire .

La méthode CAT constitue une de ces formalisations . Elle respecte les 4 points .

Remarquons au passage que les autres modèles ne respectent pas le point 3 , et que la technique d'éliminations par aspects ne respecte pas non plus le point 2 .

6.2 / CAT et les autres méthodes à coefficients

Les méthodes à coefficients autres que CAT sont de type additif et multiplicatif (Chapitre 2 , page 11 , note 3) . Elles n'offrent qu'une partie des possibilités de CAT ; celle-ci les inclut donc , et les dépasse .

Il s'entend cependant que les méthodes a coefficients de type additif et multiplicatif ont le grand avantage de la simplicité .

6.3 / CAT et la programmation linéaire

La programmation linéaire se ramène - pour le choix d'un ordinateur - à une méthode à coefficients de type additif (pour l'évaluation) , et à CAT (pour l'optimisation) .

Montrons d'abord qu'une méthode à coefficients de type additif est un programme linéaire dont les contraintes décrivent l'arbre , càd les fonctions d'aggrégation , les fonctions d'efficacité et la (ou les) valeur(s) d'entrée des critères élémentaires .

Définition des critères élémentaires (CE)

La définition des CE comporte deux éléments :

D'une part , les fonctions d'efficacité . Celles-ci peuvent se décrire par les contraintes définissant des fonctions linéaires par morceaux (1) .

D'autre part , les valeurs d'entrée sont définies par les contraintes (1°) ou (2°) selon qu'un même CE peut avoir une ou plusieurs valeurs d'entrée :

(1°) $X = A$ où X est la variable d'entrée du CE
A est la valeur d'entrée du CE

(2°) $X = \sum_{i=1}^{n} Ai.Yi$ où X est la variable d'entrée du CE
Ai sont les m valeurs d'entrée du CE
Yi sont des variables binaires t.q.

$0 \leqslant Yi \leqslant 1$, Yj entières , $\forall\, i = 1$ à n et $\sum_{i=1}^{n} Yi = 1$

(où n est le nombre des possibilités du CE) .

(1) Si on n'utilise pas les fonctions d'efficacité , la définition des des CE se ramène aux contraintes (1°) et (2°) , où X est remplacé par E , càd l'efficacité du CE .

Définition des critères non-élémentaires (CNE)

La définition des CNE est réalisée par la contrainte :

$$E = \sum_{i=1}^{n} Wi.Ei$$

où E est la variable 'efficacité' du CNE
Ei est la variable 'efficacité' du i° CIS (
Wi est le poids du i° CIS , avec (1)

$$0 < Wi < 1 \ , \ \forall\ i = 1 \text{ à } n \ , \ \sum_{i=1}^{n} Wi = 1 \quad .$$

Pour une simple évaluation (càd sans contrainte de type (2) , page) , on lance alors le P.L. avec la fonction objectif ' maximiser E ' , où E est la variable ' efficacité ' du CNE global , et n'a en fait qu'une seule valeur possible .

Pour une optimisation (càd avec au moins une contrainte de type (2) , page) , on procède de même mais cette fois E a plusieurs valeurs possibles (2) .

Pour calculer la courbe d'efficacité maximale (CEM) d'un critère , il faut itérer avec la fonction objectif ' maximiser E ' , où E est la variable 'efficacité' de ce critère , ainsi que plusieurs valeurs du coût C de ce critère . Ce coût est défini par les contraint es suivantes :

$$C \leqslant K$$

où C est la variable 'coût' de ce critère
K est le coût maximum pour l'itération (3)

(1) Rappelons que 'CIS'signifie 'Critère(s) Immédiatement Subordonné(s l'.

(2) Càd que l'on cherche l'optimum d'une fonction linéaire des efficacités dans l'espace à n dimensions , où n est le nombre des Ce. ;

(3) K a pour valeur initiale le coût de la configuration la plus chère à chaque itération , la nouvelle valeur de K est 0.999 fois le coû de la configuration optimale à l'étape précédente . t

$C = \sum_{i=1}^{n} Ci$ où Ci est la variable 'coût' du i° CE (1)

$Ci = \sum_{j=1}^{m_i} Cij.Yij$ où Cij est le coût de la j° possibilité du i° CE
Yij est une variable binaire t.q.

$0 \leqslant Yij \leqslant 1$, Yij entières , $\forall$ i = 1 à n , $\sum_{j=1}^{m_i} Yij = 1$, $\forall$ j = 1 , m_i
(où m_i est le nombre de possibilités d'entrées pour le i° CE) ,

Inversément , un P.L. fonctionne :

- en évaluation , comme une méthode à coefficience de type additif , pour autant que ces contraintes décrivent un arbre et les valeurs d'entrée des CE (2)

- en optimisation , comme la méthode CAT , avec cette contrainte supplémentaire que deux ou plusieurs valeurs possibles soient offertes pour au moins un CE .

La programmation lineaire se trouve donc - pour le choix d'un ordinateur - au niveau des méthodes coefficients de type additif (pour l'évaluation) , et au niveau de CAT (pour l'optimisation) . CAT l'inclut donc et la dépasse .

Il convient cependant de rappeler que la programmation linéaire est d'un intérêt tout autrement important dans certains domaines .

6.4 / CAT et la technique Cost-Value

La technique Cost-Value diffère fortement de CAT .

Un seul élément de ressemblance apparaît au niveau de leur philosophie fondamentale : toutes deux considèrent le coût de l'offre et de ses composantes comme un aspect particulier , méritant un traitement spécial .

Ce traitement spécial consiste pour la technique Cost-Value à faire du double aspect financier 'valeur' et 'coût' l'unique critère de décision ; et pour CAT , à utiliser le coût comme composante du rapport Efficacité/Coût , ainsi que , facultativement , comme critère .

(1) Subordonné au critère dont on calcule la CEM
(2) On voit le déplacement dans l'utilisation du P.L. : il est destiné à optimiser (et accessoirement à évaluer) , tandis que notre problème est d'abord d'évaluer (et accessoirement d'optimiser) .

6.5 / CAT et la méthode multicritère Electre 1

La méthode multicritère Electre 1 s'écarte de CAT par sa démarche non-quantifiée , sans pour autant se rapprocher de la technique Cost-Value .

Nous consacrons le chapitre 7 ci-après , à la comparer avec CAT .

6.6 / CAT et la technique d'élimination par aspects

La technique d'éliminations par aspects constitue une utilisation de conditions sine qua non , que CAT permet et dépasse : celles-ci sont en effet un cas particulier d'aggrégation et de critères .

6.7 / Conclusion

La comparaison de la méthode CAT avec les autres modèles proposés pour le choix d'un ordinateur , fait apparaître la conclusion suivante :

D'une part , CAT constitue une des formalisations de l'approche ad hoc .

D'autre part , CAT inclut et dépasse les méthodes à coefficients , la programmation linaire et la technique d'éliminations par aspects .

Par contre , CAT est concurrencée par la méthode multicritère Electre 1 et par la technique Cost-Value . Les relations entre CAT et Electre 1 sont présentées au chapitre 7 ci-après . Nous pensons que la conclusion de ce même chapitre s'applique également à la technique Cost-Value , pour une raison analogue , l'abandon d'informations relevantes pour le problème .

La méthode CAT nous semble actuellement le modèle le plus fiable pour le choix d'un ordinateur .

C H A P I T R E 7

COMPARAISON DE CAT ET ELECTRE 1

7.1 / Introduction

L'objectif de ce chapitre est de comparer les méthodes CAT et ELECTRE 1 .

Pour un exposé détaillé de celles-ci, nous renvoyons :

- pour CAT , au chapitre 3 du présent mémoire

- pour ELECTRE 1 , au Mémoire ' Analyse Multicritère ', A. NOTTEBAERT , FNDP NAMUR, 1977 .

La lecture préalable de ces deux sources est certainement recommandable ; nous nous sommes toutefois efforcés de réduire le nombre des renvois que nous leur faisons .

Dans ce but , nous commençons ce chapitre par un exposé succinct de la démarche d'Electre 1 , suivant le plan :

7.1 / Introduction

7.2 / Présentation et démarche d'Electre 1

7.3 / Le but de CAT et d'Electre 1

7.4 / Détermination des caractéristiques

7.5 / La pondération

7.6 / Evaluation et comparaison

7.7 / Conclusion .

Rappelons enfin que les termes ' arbre ' , ' graphe ' , ' sommet ' , ' sommet pendant ' , ' niveau supérieur ' , et ' niveau inférieur ' sont ceux de la Théorie des Graphes [53].

7.2 / Présentation et démarche d' Electre 1

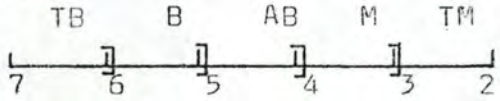
La méthode **Electre** 1 sert à comparer des systèmes complexes (1); elle est l'abréviation de la phrase 'Elimination et Choix traduisant la réalité ' .

Elle a été créée par le Professeur B. ROY , de l'Université de Paris-Dauphine (France) .

La démarche d'Electre 1 se fonde sur l'idée que l'expression d'une préférence doit être simple et inattaquable dans la forme ; pour cela , on ne compare jamais que deux objets à la fois, et une préférence n'est jamais quantifiée .

La préférence est exprimée par $\succ$, en ce sens que ' a $\succ$ b ' signifie ' a est préféré à b '. On exprime l'indifférence entre a et b par ' a $\equiv$ b ' .

On peut aussi souhaiter affiner cette préférence ou indifférence, mais toujours sans la quantifier ; à cette fin ,on utilise une ' échelle de préférence ' . Par exemple , le nombre de places dans une voiture :



où T = Très , B = Bien, A = Assez et M = Mal .

Ainsi, une voiture à quatre places obtient l'appréciation ' Assez-Bien ' .

Nous notons g_i (a) l'appréciation obtenue par l'action a pour la dimension i.

Remarquons aussi que les relations $\succ$ et $\equiv$ sont implicitement présentes dans ces appréciations ; ainsi, g_i (a) = TB et g_i (b) = AB impliquent que pour cette dimension, a $\succ$ b . Les écarts entre appréciations sont dénombrés à partir de l'appréciation TB : ainsi de TB à AB, il y a deux écarts, car TB - AB = 2 ; de même M - B = - 2 écarts.

L'utilisation des échelles de préférence constitue une évaluation ; elle est toutefois facultative, tandis qu'en CAT, elle est indispensable .

(1) Le terme 'complexe ' a ici le sens de 'multicritère ' ce qui est compatible avec notre définition du chapitre 2 , page 6 .

Nous présentons ci-après la démarche d'Electre 1 , puis un exemple traité successivement par elle et par CAT .

Démarche

La démarche d'Electre 1 comporte 7 points :

1 / Choisir les caractéristiques du système, relevantes pour la comparaison ; on les appelera ' dimensions ' .

2 / Pondérer ces dimensions entre elles .

3 / Pour chaque système, dit ' Action ' , rendre connue la valeur de chaque dimension .

4 / Considérer alors les actions 2 à 2 (2) ; puis, pour ces deux actions et chaque dimension successivement, exprimer la préférence ou l'indifférence entre les deux valeurs proposées .

5 / Attribuer à chaque action (a et b) la somme des poids des dimensions i où elle emporte la préférence ($\sum_i p_i$ (a) et $\sum_i p_i$ (b)).

6 / Déclarer que l'action a surclasse l'action b si et seulement si elle satisfait aux deux tests suivants :

1 / Test de concordance

$\boxed{\sum_i p_i \text{ (a)} > C}$, où C est le seuil de concordance et $0 \leq C \leq \sum_i p_i$.

Ce test indique si les dimensions où a l'emporte sur b ont une pondération suffisante pour permettre à a de l'emporter globalement sur b .

2 / Test de discordance

$\boxed{g_i \text{ (b)} - g_i \text{ (a)} < D}$, où D est le seuil de discordance avec $-N \leq D \leq N$, où N est le nombre total d'écarts possibles, càd le nombre total d'appréciations diminué de 1 .

g_i (b) et g_i (a) sont les appréciations des actions b et a pour la dimension .

Ce test vérifie que b ne l'emporte pas trop sur a pour certaines dimensions importantes déterminées à l'avance .

(2) S'il y a n actions, on itérera donc $C_n^2 = \frac{n\,(n-1)}{2}$ fois.

7 / Déclarer que l'action b surclasse l'action a si les mêmes conditions sont satisfaites, mutatis mutandis .

Cette démarche débouche sur un graphe où chaque sommet est une action, et où chaque arc éventuel indique une préférence - on dit ' relation de surclassement '-entre les deux actions qu'il relie

On peut alors déterminer l'action vainqueur, mais seulement si le graphe est décomposable en niveaux , auquel cas l'action vainqueu est la racine de l'arbre (3) .

Si une telle racine n'existe pas , on peut réitérer avec les acti formant le noyau du graphe (1), avec plus ou moins de dimensions, ou encore on peut relâcher ou resserrer C ou D (2).

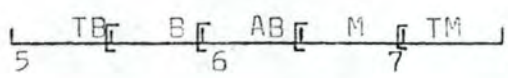
Exemple

Soient trois actions (les voitures V1, V2 et V3), quatre dimensions pondérées entre elles et les valeurs de chaque action pour chaque dimension :

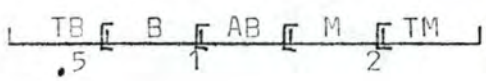
Dimensions D1 à D4 / Actions (voitures)	Consommation (L/100 Km)	Prix (100.000 Fr.)	Capacité du coffre (m3)	Nombre de places
V1	8,5	1	1,5	2
V2	6	1	3	3
V3	9	2	3	4
Poids des dimensions	30 %	30 %	15 %	25 %

Les dimensions 'consommation ' et 'prix ' verront leurs préférences exprimées par les échelles suivantes :

Consommation



P r i x



(1) càd l'ensemble des sommets sans antécédents .

(2) le relâchement consiste à diminuer C et augmenter D, tandis que le resserrement consiste à augmenter C et à diminuer D .

(3) Avec l'hypothèse contestable de transitivité entre les préférence

Les dimensions ' capacité du coffre ' et ' nombres de places ' auront leurs préférences ou indifférences exprimées simplement par $\succ$ ou $\equiv$.

Fixons C = 55 % (la somme totale des poids valant 1), D pour la consommation à 1 écart et pour le prix, à 2 écarts.

Nous itérons alors pour V1-V2, V1-V3 et V2-V3 .

1 / V1-V2

Exprimons les préférences sous forme d'un tableau :

Actions \ Dimensions	D1	D2	D3	D4
V1	8,5-TM	1-AB	1,5	2
	$\prec$	$\equiv$	$\prec$	$\prec$
V2	6-AB	1-AB	3	3

$\sum_i p_i(V1)$ = 0 % , $\sum_i p_i(V2)$ = 70 % .

V2 satisfait le test de concordance car $\sum_i p_i(V2)$ = 70 % > 55 % = C

de même pour le test de discordance :

$g_i(V1) - g_i(V2)$ = TM - AB càd - 2 écarts < 1 (consommation)

AB - AB càd 0 écart < 2 (prix)

V2 surclasse V1; ce résultat est assez intuitif car V2 a toutes ses dimensions meilleures ou aussi bonnes que celles de V1 .

Cette situation est par ailleurs dite ' dominance ' /58/.

2 / V1-V3

On obtient de même :

Actions \ Dimensions	D1	D2	D3	D4
V1	8,5-TM	1-AB	1,5	2
	≡	≻	≺	≺
V3	9-TM	2-TM	3	4

$\sum_i P_i$ (V1) = 30 % , $\sum_i P_i$ (V3) = 40 % .

Ni V1 ni V3 ne satisfont ici le test de concordance . Il n'y a pas de surclassement, quel que soit le résultat du test de discordance .

3 / V2-V3

Actions \ Dimensions	D1	D2	D3	D4
V2	6-AB	1-AB	3	3
	≻	≻	≡	≺
V3	9-TM	2-TM	3	4

$\sum_i P_i$ (V2) = 60 % , $\sum_i P_i$ (V3) = 25 % .

V2 satisfait le test de concordance car $\sum_i p_i$ (V2)= 60% > C=55% ;

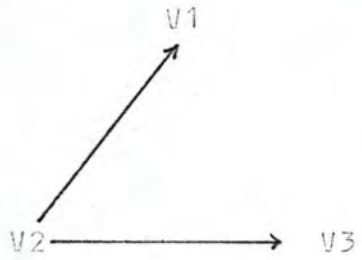
de même pour le test de discordance :

g_i (V3) - g_i (V2)= TM - AB = -2 écarts < 1

pour les deux dimensions D1 et D2 .

Dès lors V2 surclasse V3 .

On obtient le graphe final suivant, qui mène au choix de V2 :

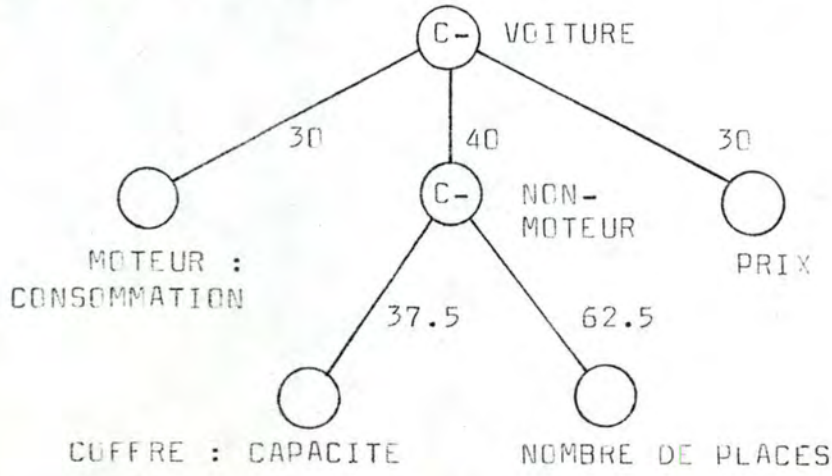


Remarquons cependant que ce résultat est étroitement lié à la pondération des dimensions, aux niveaux des seuils de concordance et discordance, aux échelles de préférences, etc .

En effet, on s'aperçoit par exemple qu'un seuil de concordance C ⩾ 70 % aurait inhibé tout surclassement ; de même qu'un seuil de discordance ⩽ - 2 écarts , etc .

Il paraît donc souhaitable d'utiliser plusieurs fois la méthode Electre 1 en faisant varier certains éléments, de façon à déterminer la sensibilité des résultats à ces variations, et à pouvoir en tirer des conclusions .

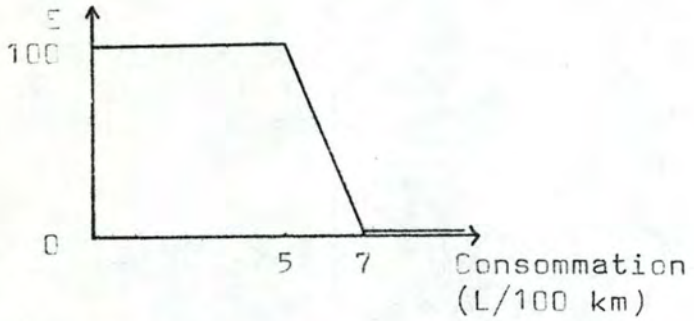
Examinons à présent le même problème, mais en le traitant par CAT . L'arbre utilisé est le suivant :

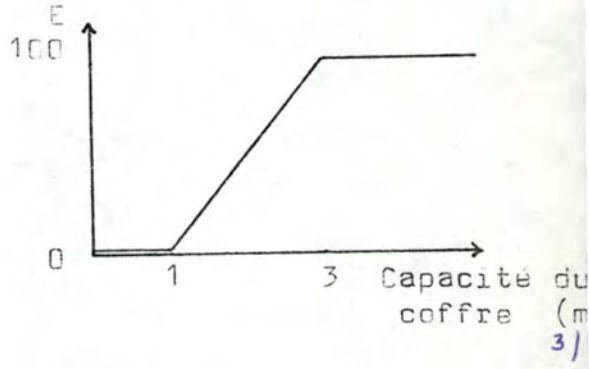


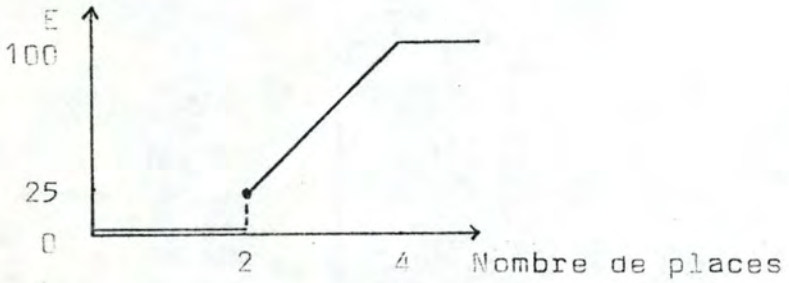
Les poids sont exprimés en % et sont les 'mêmes' que ci-avant, car

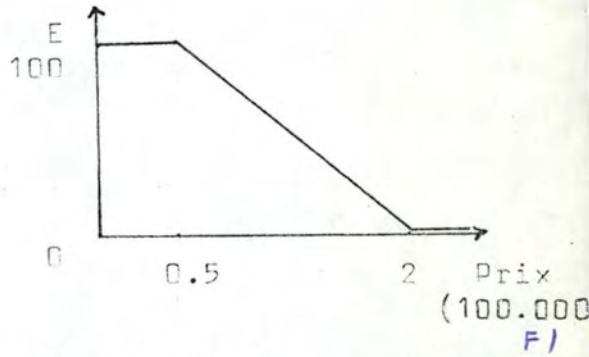
- pour la capacité du coffre , 40 % * 37,5 % = 15 % et
- pour le nombre de places , 40 % * 62,5 % = 25 % .

Les fonctions d'efficacité sont les suivantes :









On remarque que les fonctions d'efficacité pour la consommation et le prix sont très proches de leurs échelles de préférence (1) .

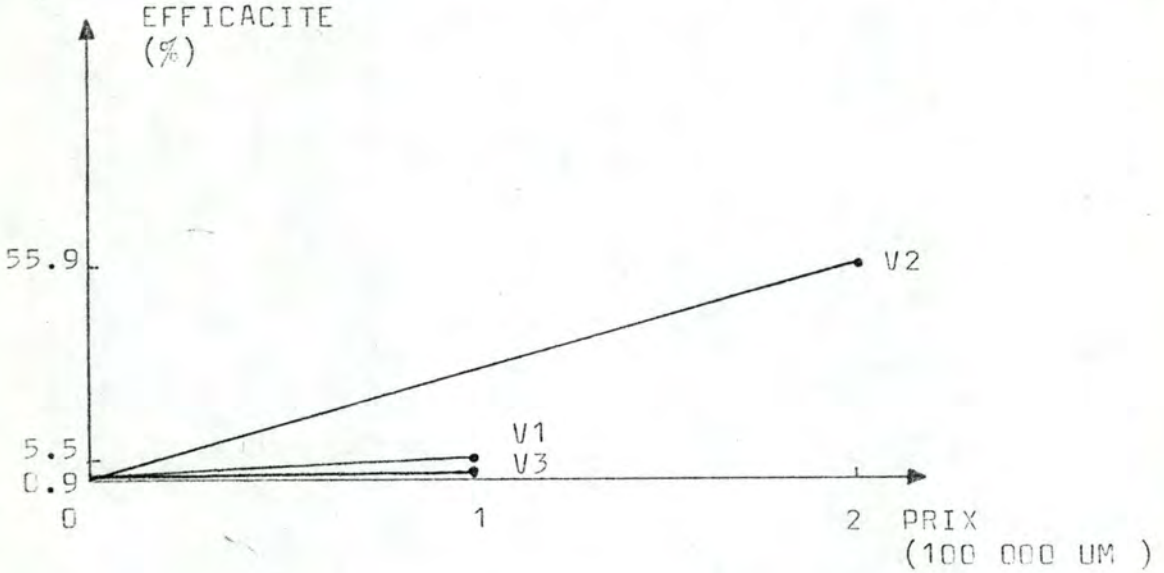
Pour la capacité du coffre et le nombre de places, les fonction d'efficacité sont une information supplémentaire que CAT demande à l'utilisateur .

Les données d'entrée étant les mêmes que celles données à la page 5, on obtient les résultats suivants :

(1) Voir page 80 et 91 .

CRITERES / OFFRES (VOITURES)	ELEMENTAIRES				NON-ELEMENTAIRES	
	MOTEUR : CONSOMMATION	COFFRE : CAPACITE	NOMBRE DE PLACES	PRIX	NON - MOTEUR	CRITERE GLOBAL : VOITURE
V1	0	25	25	50	25	5.5
V2	50	100	50	50	65.8	55.9
V3	0	100	100	0	100	0.9

Les taux d'efficacité/coût sont les suivants (en exprimant le coût en 100 000 UM - ce qui est une pure convention) :
- pour V1 , 5.5 % / 1 = 5.5 % d'efficacité pour 100 000 UM
- pour V2 , ce même résultat est 55.9 % / 1 = 55.9 % , et
- pour V3 , ce même résultat est 0.9 % / 2 = 0.5 % :



D'où la liste rangée : 1°/V2 2°/V1 3°/V3 .
La différence entre V1 et V3 , càd 5.5 % - 0.9 % = 4.6 % doit être considérée comme non-significative , compte tenu de la subjectivité et de l'imprécision inhérente à tout processus d'évaluation .

De ce fait , on débouche ici sur le même résultat V2 avec les deux méthodes CAT et Electre 1, ce qui s'explique par la faible complexité du système 'voiture' considéré ici ; on serait probablement arrivé au même résultat par un traitement mental (voir la conclusion , page 93 .

7.3 / Le but de CAT et ELECTRE 1

CAT a pour but d'évaluer et/ou de comparer des systèmes complexes, tandis qu' Electre 1 se borne à les comparer .

Les systèmes complexes sont nommés ' Actions ' en Electre 1, et 'Offres ' en CAT .

En Electre 1, des évaluations peuvent être faites au moyen des échelles de préférence . Toutefois, ces évaluations locales et facultatives ne se transmettent jamais au système global . Il n'y a donc pas à proprement parler d'évaluation de systèmes complexes .

Pour sa part, le double but de CAT appelle une précision : l'évaluation et la comparaison forment deux processus distincts ; on peut avoir soit une simple évaluation donnant le degré où un système satisfait le but qu'on lui assigne , soit une évaluation suivie d'une comparaison, comme dans le cas du choix d'un ordinateur(1).

Comme Electre 1 se borne à comparer, nous ne retiendrons que cet aspect dans les autres paragraphes du présent chapitre.

Le travail d'évaluation et/ou comparaison est assuré en Electre 1 par un homme d'études , l'évaluateur de CAT ; il est assisté par un demandeur , le délégué de l'utilisateur de CAT ; la décision finale est prise par le décideur , qui est toujours l'utilisateur de CAT ; enfin, cette dernière méthode possède seule explicitement le concept de vendeur .

7.4 / Détermination des caractéristiques et de leur structure

La première étape est commune aux deux méthodes, et consiste à déterminer celles des caractéristiques des systèmes qui sont relevantes pour la comparaison .

Elles sont dites 'Dimensions ' en Electre 1 et ' Critères ' en CAT .

Pour les déterminer, aucune technique n'est proposée de part et d'autre . Cependant :

- CAT utilise implicitement les deux techniques TOP DOWN et BOTTOM UP(2) en général et pour les niveaux inférieurs surtout, la préférence va à la seconde , car elle mène à moins d'oublis de caractéristiques que la première ; celle-ci s'utilise plutôt à posteriori, pour

(1) Voir les chapitres 2 et 3 .

(2) Dans le cadre de ce mémoire, nous adoptons les deux définitions suivantes :
- Technique TOP DOWN : détermination des caractéristiques par

vérifier qu'aucun grand groupe de critères n'a été oublié .

- Electre 1 utilise implicitement la technique top down ; en effet elle travaille en général sur moins de 4 niveaux, et la technique top down a l'avantage d'être plus intuitive et plus simple si le système est bien connu , ce qui est souvent le cas pour les niveaux supérieurs.

La caractéristique particulière 'coût ' intervient comme telle et sans plus en Electre 1, tandis qu'en CAT , elle tient en outre une place plus en rapport avec son importance réelle , dans le cadre du rapport performance/coût .

Une fois déterminées les caractéristiques, Electre 1 se désintéresse de la structure d'aggrégation et considère les sommets pendants comme les seules dimensions, mutuellement indépendantes . Par contre, CAT conserve et développe cette structure par la pondération des critères non-élémentaires (3) et par les opérateurs d'aggrégation qui permettent d'échapper à l'hypothèse d'indépendance entre critères .

Enfin, les deux méthodes pondèrent leurs caractéristiques élémentaires : dimensions en Electre 1 et critères élémentaires en CAT .

La fixation des poids et des opérations d'aggrégation fait l'objet du paragraphe suivant .

décomposition du système en sous-systèmes, eux-mêmes décomposés à leur tour , et ce jusqu'au degré de finesse souhaité . La structure d'aggrégation se dessine durant les décompositions. Rappelons qu'en CAT, celle-ci est plus qu'un arbre (voir chapitre 3) .

- Technique BOTTOM UP : détermination des caractéristiques d'abord , et création de la structure d'aggrégation ensuite.
- La situation la plus fréquente est l'absence de ces deux techniques qui est en fait un mélange des deux, dans lequel l'une ou l'autre tend à s'imposer selon que l'on détermine d'abord les caractéristiques puis la structure (Bottom up) ou le contraire (top down) .

(3) qui n'ont pas d'équivalent en Electre 1 .

7.5 / La fixation des poids

La seconde étape consiste à fixer des dimensions en Electre 1 et des critères en CAT .

Aucune technique n'est proposée à cette fin . Cependant

- En CAT, la pondération se fixe localement, càd entre les subordonnés immédiats d'un critère. Ceux-ci sont suffisamment proches entre eux pour éviter des biais importants . De ce fait, l'absence de méthode est un problème qui n'apparaît pas trop grave .

- En Electre 1, la pondération se fixe globalement , chaque dimension étant sur un pied d'égalité avec les autres ; elles peuvent être très différentes et donc malaisées à pondérer entre elles . De ce fait l'absence de méthode crée un problème assez grave . Et comme en outre le résultat dépend fortement de la pondération, il s'avère indispensable de procéder à plusieurs analyses de sensibilité (1) .

Nous suggérons pour les deux méthodes l'adoption d'une systématique de fixation des poids . Par exemple, la méthode DELPHI [57], la méthode des Jurys d' Assisses (2), ou mieux encore, l'obligation pour un groupe d'experts de se mettre d'accord .

Cette dernière solution a l'avantage de voir explicitées et confrontées les raisons d'une pondération . Il apparaît en effet que la précision d'une pondération repose sur le bien-fondé et la complétude des raisons qui la motivent .

En CAT, la détermination des opérateurs d'aggrégation pose un problème semblable, pour lequel nous suggérons une solution du même genre .

7.6 / Evaluation et Comparaison

La troisième étape consiste à comparer les systèmes .

Elle comporte le fondement même des deux méthodes, et donc leur principal e différence :

- Electre 1 se fonde sur une approche ordinale , càd sans quantification des préférences ; celles-ci exprimée par les opérateurs de préférence , d'indifférence ou encore par une appréciation détermi

(1) Il s'agit bien sûr ici de sensibilité à la pondération .
(2) Les jurés émettent chacun leur opinion ; on élimine alors les deux opinions extrêmes ; on vote à nouveau, mais seulement dans l'intervalle d'opinions ainsi réduit ; on continue jusqu'à l'obtention d'une seule opinion .

au moyen d'une échelle de préférences .

- CAT se fonde sur approche cardinale , càd portant quantification des préférences ; celles-ci sont exprimées par comparaison des cotes obtenues par les systèmes ; ainsi , lorsque deux systèmes a et b obtiennent respectivement les cotes 80 % et 50 % , on infère que a > b .

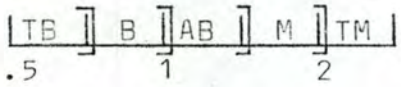
On remarque que les échelles de préférence d'Electre 1 constituent une forme réduite de fonctions d'efficacité (Figure 15).

Figure 15

Echelles de préférence et Fonctions d'efficacité

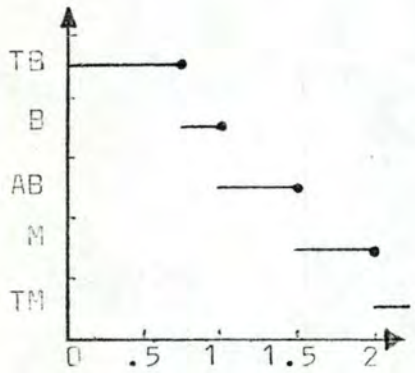
Nous convenons ici de localiser les appréciations TB, B, etc dans les intervalles d'efficacité de 100 à 80 , de 80 à 60 , etc .

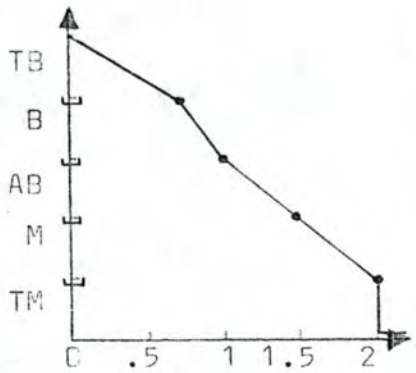
ECHELLES DE PREFERENCE : FORME USUELLE

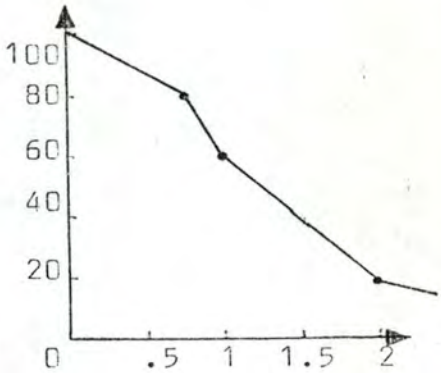


FONCTIONS D'EFFICACITE : FORME USUELLE

100	0
80	0.75
60	1
20	2
0	10







ECHELLES DE PREFERENCE : PREMIERE ET SECONDE FORMES CARTESIENNES

FONCTIONS D'EFFICACITE: FORME CARTESIENNE

Les notions d'approches ' ordinale' et ' cardinale ' mènent à ce que seule CAT tient compte non seulement des préférences, mais encore du degré de préférence .

Ce point est fondamental car il couvre une option entre l'erreur globale et l'erreur locale (1) .

Par ' erreur locale ' , on entend l'erreur ou l'imprécision au niveau d'une caractéristique, càd au niveau de son poids, de son efficacité en CAT ou de l'expression d'une préférence entre deux de ses valeurs en Electre 1 .

Par ' erreur globale ' , on entend l'erreur ou l'imprécision au niveau du résultat global , càd la liste rangée des systèmes par préférence décroissante .

On constate qu'Electre 1 a une erreur locale plus faible que CAT, sauf pour la pondération : en effet , la déclaration de la préférence a $\succ$ b est moins contestable que l'expression numérique des efficacités de a et b ; par contre, CAT l'emporte pour la pondération (2) .

En outre , CAT a une erreur globale plus faible qu'Electre 1 car elle considère l'ensemble des informations relevantes tandis qu'Electre 1 rejette la structure d'aggrégation, ne fait aucune aggrégation (3) et ne va pas au-delà des échelles de préférence pour connaître le degré des préférences .

Enfin, le résultat final est déterminé :

- en CAT , par un critère unique , en ce sens qu'un système est comparé à tous les autres par sa performance ou son rapport performance/coût .

- en Electre 1, par un critère multiple , en ce sens qu'un système est comparé à chacun des autres par une somme de poids propre à ce couple et qui peut changer avec le partenaire . L'approche de CAT assure que l'on aura toujours une offre vainqueur avec d'éventuels ex-aequo, tandis que l'approche d'Electre 2 mènera à un résultat final seulement si le graphe des préférences est décomposable en niveaux .

(1) Ces deux concepts sont propres à ce mémoire .
(2) Nous en exposons les raisons au paragraphe 7.5 , page 90 .
(3) Ce qui rapproche Electre 1 des méthodes à coefficients de type additif .

7.7 / Conclusion

Les méthodes CAT et Electre 1 constituent deux démarches différentes ; elles formalisent toutes deux le processus mental de comparaison , mais elles suivent respectivement des optiques ' cardinale' et ' ordinale' .

Le développement de ces deux optiques mène respectivement CAT et Electre 1 à des erreurs globales assez faible et assez forte et à des erreurs locales assez forte et assez faible .

Dans l'ensemble - et telle est notre conclusion - les deux méthodes apparaîssent recommandables , mais pour des domaines d'applications différents :

- Electre 1 paraît pertinente pour les problèmes où les systèmes à comparer sont assez simples ; par exemple , s'il existe une certaine indépendance entre les dimensions .

- CAT paraît pertinente pour les problèmes où les systèmes à comparer sont complexes et où toute l'information relevante doit être prise en considération .

Nous pensons que le critère principal pour le choix d'une des deux méthodes est la complexité du problème : une grande complexité mènera à CAT , une complexité moyenne à Electre 1 et une complexité faible à un traitement mental .

CHAPITRE 8

CONCLUSION

La conclusion de notre travail comporte trois parties qui ont pour objets : la méthode CAT elle-même , celle-ci face aux autres modèles proposés pour le choix d'un ordinateur , et enfin son développement futur .

La méthode CAT est un nouvel outil d'évaluation et de comparaison de systèmes complexes .

Elle constitue un progrès sensible pour les méthodes à coefficients , et ce , à trois niveaux : l'aggrégation des critères , l'optimisation de l'efficacité des critères , et son outil informatique le système SEL .

En matière de choix d'un ordinateur , la comparaison théorique montre que CAT inclut et dépasse l'approche ad hoc , les autres méthodes à coefficients , la programmation linéaire et la technique d'éliminations par aspects .

Par contre , elle est concurrencée par la méthode multi-critère Electre 1 et par la technique Cost-Value .

Le développement de la méthode CAT nous paraît devoir s'orienter dans deux directions :

- l'utilisation de techniques propres à réduire la subjectivité inhérente aux processus d'évaluation et / ou de comparaison

- et , au niveau du système SEL , la création de nouvelles commandes et / ou instructions qui rempliront des tâches actuellement manuelles

Au terme de notre travail , il nous apparaît que la méthode CAT et le système SEL constituent une contribution réelle à la méthodologie d'approche des problèmes d'évaluation et de comparaison de systèmes complexes .

La méthode CAT nous semble prometteuse en matière de choix d'ordinateurs et , plus généralement , de choix de matériels sophistiqués .

Nous suggérons le développement ultérieur de la méthode CAT et sa comparaison avec les autres modèles .

BIBLIOGRAPHIE

La bibliographie du présent mémoire comporte 66 références Par clarté , nous la présentons en 4 parties :

1 / Choix d'un ordinateur : méthodes à coefficients

2 / Choix d'un ordinateur : autres approches

3 / Analyse de performance (Benchmarks , Simulation) .

4 / Divers (Théorie de l'Utilité , Logique Continue ...) .

1 / CHOIX D'UN ORDINATEUR : METHODES A COEFFICIENTS

1 Baugut G.
Nutzwertmodelle (en langue allemande)
Modelle zur Auswahl von Datenverarbeitungsanlagen , Köln , 1973 , p. 66 - 85

2 Blumenthal Ph. L. Jr.
Applying an OR Technique in Selecting Data Processing Hardware
Journal of Accountancy , aug. 1962 , p. 82 - 83

3 Bromley A. C.
Choosing a set of computers
Datamation , aug. 1965 , p. 37-40

4 Dowkant A. J.
A Methodology for Comparison of Generalized Data Management Systems ; PEGS
US Dept of Commerce , mar. 1967

5 Dujmović J. J.
Criteria Aggregation Technique for Evaluation , Optimization and Selection of Computer Systems
Université de Belgrade , Yougoslavie , 1975 (non publié)

6 Dujmović J. J.
Evaluation , Comparison and Optimization of Hybrid Computers Using the Theory of Complex Criteria
Simulation of Systems , L. Dekker Editor , North-Holland Publishing Company , 1976

7 Dujmović J. J.
System and Computer Evaluation (Sommaire du futur ouvrage homonyme)
Université de Belgrade , 1977 (non publié)

8 Dujmović J. J.
Vrednovanje i Izbor Računara za Pirotski Računski Centar (Evaluation et Sélection d'un ordinateur pour le Centre Informatique de Pirot -- Mémorandum du cas PIRE)
Université de Belgrade , nov. 1976 (non publié)

9 Dworatschek S.
Management Informations-Systeme (en langue allemande)
Idem , De Gruyter , 1971 , p. 169-175

10 Fry J. P. et al.
Data Management Systems Survey
Mitre Corporation Document MTP-329 , jan. 1969

11 Gilbert E.J., Dammerle A. M.
The Value of Increased Performance as applied to System Procurements
Data Management , may 1971

12 Keller R. F., Denham Ch. R.
Computer Selection Procedures
Proceedings of the 1968 ACM National Conference , 1968 , p.679-683

13 Miller W. G.
Selection Criteria for Computer System Adoption
Educational Technology , oct. 1969 , p. 71-75

14 Ollivier R. T.
A Technique for Selecting Small Computers
Datamation , jan. 1970 , p. 141 - 145

15 Rosenthal S.
Analytical Technique for Automatic Data Processing Equipement Acquisition, Spring Joint Computer Conference , 1964 , proceedings , p. 359-366

16 Scharf T.
Weighted Ranking by Levels
IAG Quarterly Journal , 1969 , Vol. 2 , p. 7-23

17 Scharf T.
' Weighted Ranking by Levels ' Computer Evaluation : One Year of Experience
IAG Quarterly Journal , 1970 , Vol. 3

18 Schwartz E. S.
Computer Evaluation and Selection
Data Management , jun. 1968 , p. 56-62

19 Steere R. E. Jr.
How to Select the Right Data Processing Equipement
Administrative Management , aug. 1962 , p. 26-28

20 White D.R. , Scott D.L. , Schulz R. N.
POED - A Method of Evaluating System Performance
IEEE Transactions on Engineering Management , dec. 1963, p. 177-182

21 Williams Q. N. , Perrott R. S. , Weitzman J. , Murray J.A. , Shober J. A.
A Methodology for Computer Selection Studies
Computer and Automation , may 1963 , p. 18-22

22 Zangemeister Ch.
Measurement of Effectiveness of Computerized Information Systems from a Management Point of Vue through Utility Analysis
International Symposium on Economics of Informatics , proceedings , jul. 1974 , vol. 1 , P. 440-451

2 / CHOIX D'UN ORDINATEUR : AUTRES APPROCHES

23 Burch J. G. , Strater F. R.
System Evaluation and Justification
Information Systems : Theory and Practice , Hamilton Publ. Co. Santa Barbara , (California , US), 1974 , 12th chapter

24 Canning R. G.
Data Processing ... Practically Speaking / 3° , Equipment Selection
Data Processing Digest , 1966 , Vol. 12 , N°. 6 , p. 1-9

25 Fein L.
A Figure of Merit for Evaluating a Control Computer System
Automatic Control , may 1960 , p. 39-41

26 Fein L.
Assessing Computing Systems
Data Processing for Management , feb. 1964 , p. 19-21

27 Joslin E. O.
Computer Selection
Addison-Wesley Publishing Company Inc. , 1968

28 Joslin E. O.
Cost-Value Technique for Evaluation of Computer System Proposal
Spring Joint Computer Conference , 1964 , Proceedings ,
p. 367-381

29 Raymont P. G.
The Comparative Evaluation of Computer Systems
(Restricted Distribution by Automation) , june 1971

30 Shirley W. D.
Choosing a Computer (12 parts)
Data and Control , may 1965 - apr. 1966

31 Schneidewind N. F.
The Practice of Computer Selection
Datamation , feb. 1967 , p. 22-25

32 Timmreck E. M.
Computer Selection Methodology
Computing Surveys , dec. 1973 , Vol. 5 , N° 4 , p. 199-222

33 Titus J.
Standardising Computer Selection
Computer and Automation , oct. 1965 , p. 32-34

3 / ANALYSE DE PERFORMANCE (BENCHMARKS , SIMULATION)

34 Canning R. G.
Better Computer Comparisons for You
EDP Analyser , june 1963 , p. 1-9

35 Gosden J. A.
Estimating Computer Performance
Computer Journal , jan. 1963 , p. 276-283

36 Gosden J. A. , Sisson R. L.
Standardized Comparisons of Computer Performance
Joint Computer Conference , 1962 , Proceedings p. 57-61

37 Herman D. J.
The use of a Computer to Evaluate Computers
Joint Computer Conference , 1964 , Proceedings , p. 383-395

38 Joslin E. O. , Aiken J.J.
The Validity of Basing Computer Selections on Benchmark Results
Computer and Automation , jan. 1966 , p. 22-23

39 Miller E.F. Jr
Bibliography on Techniques of Computer Performance Analysis
Computer , sep./oct. 1972 , p. 39-47

40 Statland N.
Methods of Evaluating Computer Systems Performance
Computer and Automation , feb. 1964 , p. 18-23

4 / DIVERS (THEORIE DE L'UTILITE , LOGIQUE CONTINUE ...)

41 Ashby W. R.
Introduction à la Cybernétique
Publié par Dunod , 1958

42 Bassler R. A. , Demoody H. C.
Computer System Evaluation and Selection : An Annotated Bibliography and Keyword index
Publié par College Readings Inc. , Arlington (Virginia , US), 1971

43 Becker G. M. , De Groot M. H. , Marschak J.
Measuring Utility by a Single-Response Sequential Method
Behavioral science , 1964 , Vol. 9 , p. 226-232

44 Godart F.
Cours de Recherche Opérationnelle
FNDP Namur (non publié)

45 Bourgeois M.
Cours de Systèmes Informatiques de Gestion
FNDP Namur , 1977 (non publié)

46 Churchmann C. W. , Ackoff R. L.
An Approximate Measure of Value
Journal of the Operations Research Society of America , Baltimore (Maryland) , may 1954 , p. 172-187

47 Dujmović J. J.
Extended Continuous Logic and the Theory of Complex Criteria
Publié par la Faculté d'Electrotech. de l'Université de Belgrade (Yougoslavie) , N° 498-541 , 1975

48 Dujmović J. J.
Graphic Approach to Weight Conjunctive and Disjunctive Means Calculation
Idem , N° 498-541 , 1975

49 Dujmović J. J.
Numerical Computations of Weighted Power Means
Idem , N° 461-497 , 1974

50 Dujmović J. J.
Two Integrals Related to Means
Idem , N° 412-460 , 1973

51 Dujmović J. J.
Vrednovanje i Izbor Složenih Sistema (Evaluation et Comparaison de Systèmes Compexes -- Thèse de Doctorat)
Université de Belgrade , Faculté d'Electrotechnique , 1974 (non publié)

52 Dujmović J.J.
Weighted Conjunctive and Disjunctive Means and their Application in System Evaluation
Idem qu'en 47 , N° 461-497 , 1974

53 Fichefet J.
Théorie des Graphes et Applications
FNDP Namur , 1973 (non publié)

54 Fichefet J.
Formulation de Modèles en Variables Binaires
FNDP Namur , 1974 , (non publié)

55 Fiock L. R. Jr.
Seven Deadly Dangers in EDP
Hardware Business Review , may/june 1962 , p. 88-96

56 Fishburn P. C.
A dditive Utilities with Finite Sets : Applications in the Management Sciences
Naval Research Logistics Quarterly , 1967 , Vol. 14 , p. 1-13

57 Guigou J.-L.
Analyse des Données et Choix à Critères Multiples
Publié par Dunod , 1974 , p. 188-196

58 MacCrimmon K. R.
An Overview of Multiple Objective Decision Making
University of South Carolina Press , 1973 , p. 18-44

59 MacCrimmon K. R. , Toda M.
The Experimental Determination of Indifference Curves
Review of Economic Studues , 1969 , p. 433-451

60 Riggs J. L.
Economic Decision Models
International Student Edition , 1968 , p. 6-8 & 11-33

61 Sharpe W. F.
The Economics of Computers
Columbia University Press , New York , 1964

62 Shepard R. N. , Bryan G. L. , Shelly M. W.
On Subjectively Optimum Selection Among Multiattribute Alternatives
Human Judgments and Optiamlity , Wiley , 1964 , p. 257-281

63 Terry H.
Comparative Evaluation of Performance Using Multiple Criteria
Management Science , 1963 , Vol. 9 , p. 341-441

64 Tversky A.
Intransitivity of Preferences
Psychological Review , 1969 , p. 31-48

65 Van Horn E. C.
Three Criteria for Designing Computing Systems to Facilitate Debugging
Communications of the ACM , may 1968 , Vol. 11 , N° 5 , p. 360-365

66 Yntema D. S. , Torgerson W. S.
Man-Computer Cooperation in Decisions Requiring Common Sense
IRE Transactions on Human Factor in Electronics , mar. 1961 ,
p. 20-26 .

FIN DU PREMIER TOME

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX
NAMUR (BELGIQUE)

INSTITUT D'INFORMATIQUE

EVALUATION ET COMPARAISON DE SYSTEMES COMPLEXES

Choix d'un ordinateur par la méthode CAT

Directeur : M. Ph. VAN BASTELAER

Jean-Philippe ARNOTTE
Mémoire présenté en vue
d'obtenir le grade de
Licencié et Maître en Informatique

Tome 2 : Annexe

- 1977 -

C H A P I T R E 1

PLAN

Le présent volume constitue l'annexe de notre mémoire ; elle comprend la partie technique et constitue le manuel de l'utilisateur du système SEL .

Le plan de cette annexe est le suivant : après le plan lui-même , nous présentons le langage SEL , puis le système SEL ; nous passons alors aux listings du système SEL , des exemples PIRC et CPIRC , ainsi que de 6 procédures usuelles (1) . Nous terminons alors par quelques suggestions au praticien :

(1) Chaque listing est précédé d'un commentaire explicatif .

C H A P I T R E 2

LE LANGAGE SEL

2.1 / Présentation

Le langage SEL est un langage interprétatif qui permet de programmer aisément une ou plusieurs fonctions de la méthode CAT .

Il est implémenté par le système SEL , qui est présenté au chapitre 3 ci-après , pages 55 à 61 .

2.2 / Les objets du langage SEL

Les principes du langage SEL ont été exposés dans le premier tome , pages 49 à 51 .

Nous les exposons à présent de façon détaillée ; avant de présenter chaque commande et instruction , nous exposons 7 éléments importants , dans les paragraphes 2.2.1 à 2.2.7 ci-après .

2.2.1/ Les constantes du langage SEL

Une constante SEL est la même chose qu'une constante Fortran entière ou réelle sans exposant .

Cette définition est cependant réduite

- par l'instruction READ qui n'accepte des entrées qu'en formats Fortran I5 , I10 , F5.0 et F10.0

- par l'instruction PRINT qui n'utilise que des formats E12.5 F12.4 et F12.7

- enfin et surtout , dans le programme INSEL , l'interpréteur lit les constantes à leurs vraies valeurs , mais les enrégistre dans le programme INSEL sous la forme IIII/E ou IIII est un entier formé des 4 chiffres les plus significatifs du nombre et où E est un exposant qui corrige l'erreur ainsi introduite ; l'exposant est multiplié par 10 pour la conservation (Voir pages 7 et 8) .

Par exemple , si l'interpréteur lit 3.14159 , il enregistre cette constante dans le programme INSEL sous la forme 3142/-30, càd que cette constante interviendra comme 3.142 (l'interpréteur arrondit d'office lorsqu'il tronque) .

2.2.2 / Les variables du langage SEL

L'utilisateur dispose des variables suivantes

- le vecteur entier K(100) , ainsi que

- les vecteurs réels A(20) , X(120) , E(210) et C(120) (1) .

Aucune autre variable ne lui est accessible , hormis bien sûr les variables équivalencées aux précédentes : NBELC avec K99 , NBTOTC avec K100 , et d'autres dont l'utilité est nulle pour l'utilisateur .

(1) La transparence n'étant pas parfaite en SEL , il se fait que le vecteur X(120) est utilisé par l'instruction READ pour entrer les valeurs des CE ; le vecteur E(210) est utilisé par l'instruction EFFECTIVENESS pour contenir les efficacités de tous les critères . et le vecteur C(120) est utilisé par l'instruction DATA pour entrer les CEM des CE .

L'adressage direct d'une variable se fait en accolant le nom du vecteur et la position de la variable ; par exemple , la variable écrite A(22) en Fortran s'écrit A22 en SEL .

L'adressage indirect d'une variable se fait en citant entre parenthèses une variable Ki ; par exemple E(K7) , indique la variable dont la position dans le vecteur E est la valeur de la variable K7 . L'indirection n'est permis qu'avec les variables Ki , car elles sont les seules à être entières .

D'où la synthaxe générale des variables de l'utilisateur

$$\left\{\begin{matrix}\underline{K}\\ \underline{A}\\ \underline{X}\\ \underline{E}\\ \underline{C}\end{matrix}\right\} \quad \left\{\begin{matrix} i \\ (\underline{K}i)\end{matrix}\right\}$$

où i est un entier >0 limité par la taille du vecteur auquel il est accolé (Voir page 4) .

En outre , les instrutions READ et PRINT permettent l'option

$$\left\{\begin{matrix}\underline{K}\\ \underline{A}\\ \underline{X}\\ \underline{E}\\ \underline{C}\end{matrix}\right\} \quad \left(\left\{\begin{matrix} i \\ \underline{K}i\end{matrix}\right\} : \left\{\begin{matrix} j \\ \underline{K}j\end{matrix}\right\}\right) .$$

Cette option indique l'ensemble des variables d'un même vecteur dont les positions sont comprises entre les deux valeurs indiquées , celles-ci incluses ; il s'entend que la première valeur doit être inférieure ou égale à la seconde sous peine d'incident .

Par exemple , l'expression C(6,8) indique les variables C6 , C7 et C8 .

2.2.3 / Les opérandes du langage SEL

Les opérandes du langage SEL sont des variables et des constantes SEL ,telles que décrites aux deux paragraphes précédents .

Un opérande sans signe est un opérande ECRIT sans signe , dont la valeur peut-être négative s'il s'agit d'une variable .

2.2.4 / Les numéros de référence du langage SEL

Chacune des 17 instructions du langage SEL - sauf l'instruction END - peut être précédée d'un numéro de référence qui est un entier >0 et <32768 .

Ce numéro de référence permet de brancher à l'instruction considérée au moyen des instructions BRANCH et REPEAT . Il s'entend qu'un même numéro de référence est unique au sein du programme ; il constitue le numéro de référence externe de l'instruction .

Si l'on compte les instructions en séquence des entiers naturels , le numéro d'une instruction constitue son numéro de référence interne . Par exemple , la troisième instruction a comme numéro de référence interne 3 , et comme numéro de référence externe éventuel , celui qui lui est donné par le programmeur .

2.2.5 / Les numéros des critères

La distinction entre numéros internes et externes des critères est la suivante :

- est externe le numéro que le programmeur donne au critère dans le cadre de l'instruction INPUT (pages 21 à 22)
- est interne le numéro du critère , lorsque les critères sont comptés en séquence des entiers naturels ; il faut alors commencer par compter les CE (de 1 à NBELC) , puis les CNE (de NBELC + 1 à NBTOTC) .

2.2.6 / Les streams de programmes en langage SEL

Tout deck d'instructions doit être précédé d'une commande *SEL , qui en lance la traduction ; la commande *SEL doit être précédée d'au moins une commande *JOB qui la sépare de la précédente commande *SEL ou *READ .

Par exemple , les deux cas PIRC et CPIRC peuvent être exécutés en un seul stream ; pour cela , il suffit d'accoler les deux sources VBA.ARN.PIRC et VBA.ARN.CPIRC , en éliminant la commande *EXIT de la première source .

2.2.7 / Le code INSEL

Les instructions qui forment un programme en langage SEL doivent être traduites en un programme-objet , le programme INSEL (Voir pages 49 à 53 dans le premier tome) .

Celui-ci est formé d'un maximum de 400 entiers , divisés en autant de champs qu'il y a d'instructions .

Ces champs sont de longueurs variables ; leurs longueurs sont données à la page 54 du premier tome .

Pour chaque instruction , le champ commence par le code opératoire OP de cette instruction , suivi de 0 à 8 paramètres .

Les valeurs du code opératoire et des arguments pour chaque instruction sont données au paragraphe 2.4. .

Il est important de noter que dans la rédaction du programme , les constantes et les variables sont rédigées sous une forme arithmétique usuelle ; cette forme est dite code externe en SEL .

Dans le programme INSEL , les constantes et variables sont rédigées en code interne de SEL . Ce code interne utilise deux entiers , de la façon suivante (les entiers sont les arguments dans le code INSEL) :

- une constante entière est codée sous la forme (1 , entier)

- une constante réelle est codée sous la forme (i , j) où i et j sont tels que la constante réelle égale $j * 10^{i/10}$ (1)

- une variable est codée sous la forme (3 , x) où x est un entier de 4 chiffres dont le premier indique un des vecteurs de l'utilisateur (1 , 2 , 3 , 4 ou 5 pour K , A , X , E ou C) , et les trois derniers indiquent la position de la variable dans ce vecteur ; s'il y a indirection , l'ensemble est doté du signe - .

Par exemple , dans l'instruction BRANCH (pour une boucle , page) si l'incrément est l'entier 7 , on aura D5=1 et D6 = 7 ; si l'incrément est le réel 0,5 , on aura D5 = - 10 et D6 = 5 , car $0,5 = 5 * 10^{-10/10}$; si l'incrément est une variable telle que K17 , C(K2) ou K(K3) , on aura respectivement les codes D5 = 3 et D6 = 1017 , D5 = 3 et D6 = -3002 , ainsi que D5 = 3 et D6 = - 1003 .

Rappelons enfin que les instructions SEL servent à programmer les fonctions de la méthode CAT , tandis que les commandes servent à piloter le programme formé par les instructions . Nous présentons ci-après ces mêmes commandes et instructions , en utilisant la syntaxe COBOL ; rappelons qu'on y exprime une option facultative par des crochets [ ], un choix obligatoire par des accolades { }, et les caractères obligatoires en les soulignant ; en SEL , nous soulignons tous les caractères obligatoires , alors qu'en COBOL on ne souligne pas les caractères de ponctuation ; d'autre part , nous ne soulignons pas les options i , j , k et l , car elles représentent des entiers et non les caractères i à l .

(1) L'exposant est destiné à normaliser le nombre ; celui-ci est ramené à un entier formé des 4 chiffres les plus significatifs du nombre réel ; l'exposant est à son tour multiplié par 10 afin de n'avoir jamais les valeurs 1 et 3 , réservées aux codes des entiers et des variables .

2.3 / Les commandes du langage SEL

Les commandes du langage SEL sont au nombre de 9 :

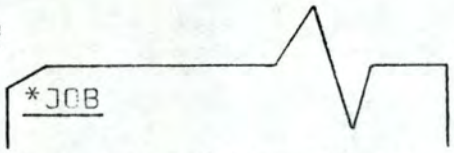
1	/ *JOB	Obligatoirement la première commande	
2	/ *SEL	Traduction	] Acquisition du programme INSEL
3	/ *READ	Lecture	]
4	/ *STORE	Mémorisation	] Conservation du programme INSEL
5	/ *PRINT	Impression	]
6	/ *EXECUTE	Exécution du programme INSEL	
7	/ *DELETE	Destruction du programme INSEL	
8	/ *NEW PAGE	Contrôle du listing	
9	/ *EXIT	Obligatoirement la dernière commande .	

Nous présentons ci-après ces commandes , selon le plan

A / Syntaxe B / Sémantique C / Fonctionnement .

1 /La commande *JOB

A/Syntaxe



Les caractères '*JOB' doivent se trouver dans les positions 1 à 4, et dans cet ordre .

B/Sémantique

Le but de la commande *JOB est d'être la première du deck d'entrée .
Il doit y avoir au moins une commande *JOB ; il peut y en avoir plusieurs (stream de travaux) .

La commande *JOB permet le traitement de toutes les autres commandes , traitement inhibé avant sa première apparition à chacune de celles-ci, elle libère la mémoire-programme de l'interpréteur de son éventuel contenu , le programme INSEL ce qui laisse le champ libre à d'autres travaux du même strea

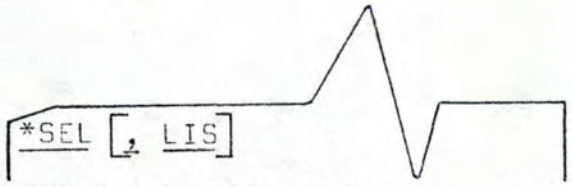
C/Fonctionnement

La commande *JOB a pour effet :
- de mettre à zéro les variables NIN (nombre d'instructions NBELC (nombre de critères élémentaires) et NBTOTC (nombr total de critères)
- de mettre à 1 la variable JOB, ce qui permet le traitemée des autres commandes
- de mettre à 0 le vecteur NF (31), qui indique la taille des courbes d'efficacité maximale des sommets de l'arbre utilisé par l'optimisation (1).

(1) Ce vecteur doit être mis à 0 avant la première instruction DATA. Si cette dernière le faisait, elle devrait le signaler aux autres instructions DATA par un switch . On préfère assurer cette opération à la commande *JOB, ce qui a le même effet .

2 / La commande *SEL

A / Syntaxe



Les caractères *SEL doivent se trouver dans les positions 1 à 4, et dans cet ordre .

B / Sémantique

Le but de la commande *SEL est de lancer la traduction du programme source .

Ces instructions du programme source doivent toutes se trouver immédiatement derrière la commande *SEL , et avant la commande suivante .

La traduction génère le programme INSEL dans sa forme utilisable immédiatement par les autres commandes (1) ; toutefois, si la traduction détecte des erreurs , aucun programme INSEL n'est généré .

L'option LIST provoque le listage des instructions sur le listing . Sans elle, les instructions n'apparaissent pas sur le listing .

C / Fonctionnement

La commande SEL :

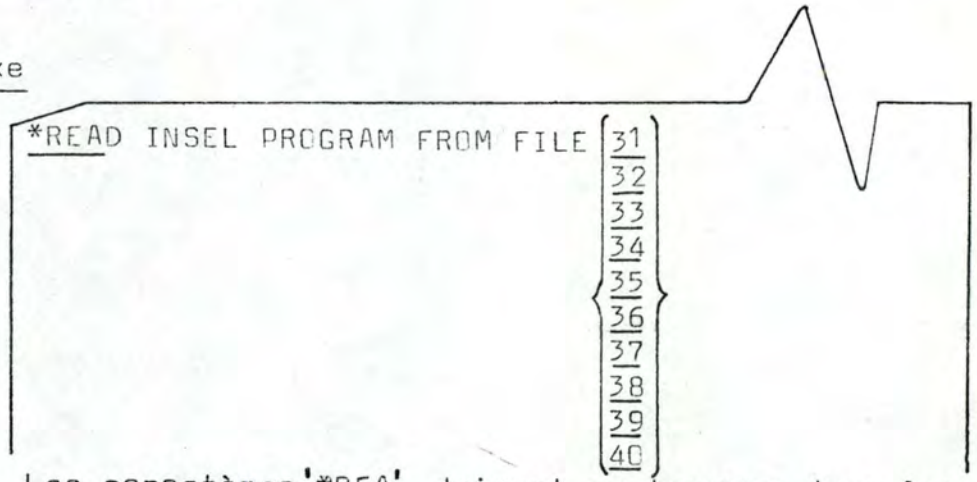
- met à 1 la variable LIST1 si l'option LIST est utilisée , sinon la laisse à 0 , sa valeur preset
- gère toute la traduction des instructions suivant la commande SEL et ce jusqu'à la première instruction END qui signale la fin du programme-source .

A la fin de la traduction , le traducteur TRAN1 renvoie au moniteur , pour traiter la commande qui suit l'instructio END .

(1) càd *STORE , *READ après un *STORE , *PRINT et *EXECUTE , qui son les seules commandes à utiliser le programme INSEL .

3 / La commande *READ

A / Syntaxe



Les caractères '*REA' doivent se trouver dans les positions 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *READ est de lire le programme INSEL dans le fichier indiqué , et de l'installer dans la mémoire-programme sous forme exécutable (Voir la commande *EXECUTE) .

Le fichier indiqué doit avoir été rempli par une commande *STORE le référençant .

Si le fichier est vide , càd que son premier article contient 0 , le message 'FILE NO.<numéro du fichier>. IS EMPTY ' est listé, et on passe directement à la commande suivante .

C / Fonctionnement

L'effet de la commande *READ est :

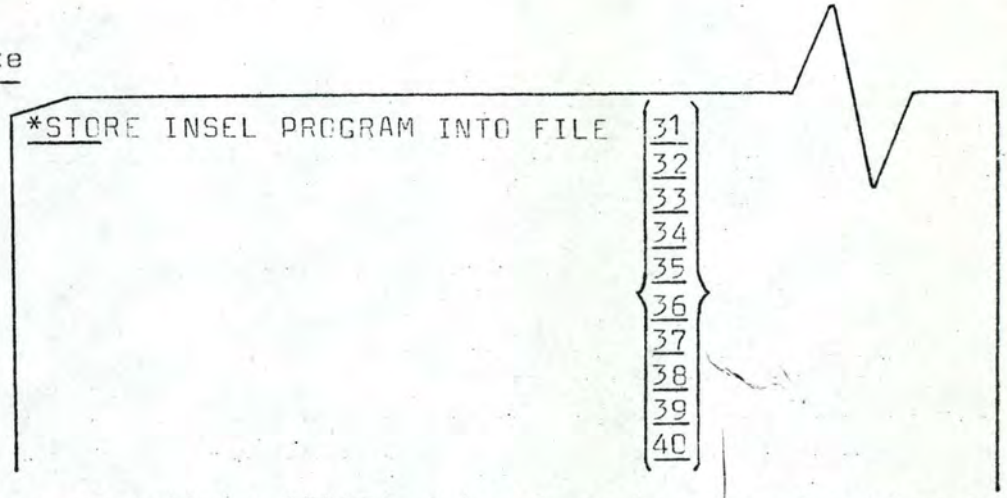
- de lire dans le fichier les valeurs des variables NIN , K (1) à K (NIN+1) , L(1) à L(K(NIN+1)-1) , IND , les éventuels titres que l'on place aussitôt dans le fichier-titres de numéro 42, ainsi que la variable INDLAB et les éventuelles variables LABFIL(J) et IA(J) .

Le sens de ces variables est repris succintement dans le cadre de la commande *STORE , page 13 à 14 .

- de lister le message 'TRANSACTION PERFORMED ' suivi des mêmes renseignements que ceux exposés à la page 14 .

4 /La commande *STORE

A / Syntaxe



Les caractères '*STO' doivent se trouver dans les position 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *STORE est de sauver dans le fichier indiqué le programme INSEL qui se trouve dans la mémoire-programme de l'interpréteur.

Elle sauve toutes les informations propres au programme INSEL ; moyennant une lecture par la commande *READ (page) , ce programme INSEL est à nouveau exécutable , indépendamment de ce qu'entretemps , d'autres commandes et/ou d'autres programmes INSEL aient été traités .

La commande *STORE ne détruit pas la mémoire-programme de l'interpréteur , dont le contenu reste exécutable .

Si le fichier indiqué était déjà occupé , la commande *STORE est sans effet , et le message 'FILENO.(n° du fichier IS OCCUPIED ' est listé ; il signifie qu'une commande *STO a déjà été effectuée sur ce fichier , et qu'il pourra êtr libéré par la commande *DELETE , page .

C / Fonctionnement

L'effet de la commande *STORE est d'écrire sur le fichier indiqué les variables suivantes FORMAT(I10) , sauf pour le titres (FORMAT(A10)) : en

- NIN , le nombre d'instructions
- K(1) à K(NIN+1) , le vecteur qui pointe sur le vecteur L

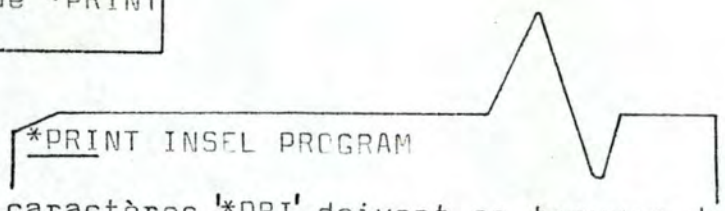
- L(1) à L(K(NIN+1)-1) , le vecteur qui contient le programme INSEL
- IND , le nombres de caractères total des titres (≥ 0)
- les titres s'il y en a , en les lisant dans le fichier-titres de numéro 42
- INDLAB , le nombre de numéros de référence dans le programme (≥ 0
- les couples ((LABFIL(J) , IA(J)) , J=1 , INDLAB) càd pour chaque numéro de référence s'il y en a , ce numéro de référence et le numéro de l'instruction où il apparaît , compté en séquence des entiers naturels .

Ensuite , le message ' TRANSACTION PERFORMED ' est listé ; il donne aussi le nombre total d'instructions NIN , la place totale occupée sur le fichier par l'ensemble , et le nombre total de caractères IN

Il est à noter que la commande *STORE ne conserve dans le fichier indiqué que le programme INSEL , et non les données qu'il a pu lire , telles que l'arbre , les CEM des CE , etc .

5 /La commande *PRINT

A / Syntaxe



Les caractères '*PRI' doivent se trouver dans les positions 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *PRINT est de lister le programme INSEL qui se trouve dans la mémoire-programme de l'interpréteur .

Si celui-ci n'existe pas (1) , le message ' INSEL PROGRAM N EXISTENT ' est listé . On passe alors à la commande suivante .

C / Fonctionnement

L'effet de la commande *PRINT est d'imprimer le tableau intitulé ' INSEL PROGRAM LISTING ' .

Celui-ci comporte l'en-tête

' NO. NAME OP D1 D2 D3 D4 D5 D6 D7 D8 '

qui explique les composantes des lignes suivantes , dont chacune reprend une instruction :

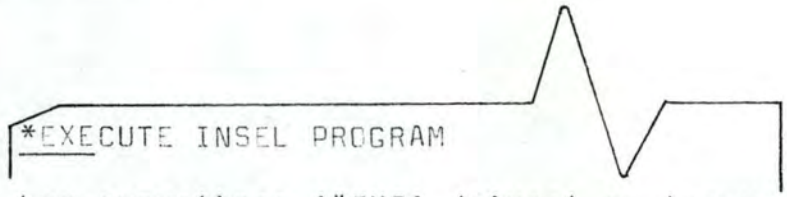
- NO. pour le numéro interne de l'instruction , en séquence des entiers naturels .
- NAME pour le nom abrégé de l'instruction , càd les trois premiers caractères de son nom , qui sont soulignés dans les paragraphes 'Syntaxe ' de ces instructions .
- OP pour le code opératoire de l'instruction, qui est une constante (voir page 7) .
- enfin, les éventuels arguments de l'instruction, D1 à D8 .

Par clarté , le groupe de lignes décrivant les instructions est encadré de deux lignes faites de traits d'union '-' .

(1) Voir note page 16 , deuxième partie .

6 / La commande *EXECUTE

A / Syntaxe



Les caractères '*EXE' doivent se trouver dans les positions 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *EXECUTE est de lancer l'exécutio du programme INSEL qui se trouve dans la mémoire-programme de l'interpréteur .

Il doit y avoir été introduit précédemment :
- par une traduction (Commande *SEL , page 11) ou
- par une lecture (Commande *READ, page 12) .

Si plusieurs commandes de ces deux types précèdent la commande *EXECUTE , le programme INSEL présent est celu introduit par la dernière d'entre elles .

Si le programme INSEL n'existe pas (1) , le message 'Insel Program Inexistent ' est listé .

C / Fonctionnement

L'effet de la commande *EXECUTE est de brancher au réalisateur REAL1 , qui gère toute l'exécution du programme INSEL .

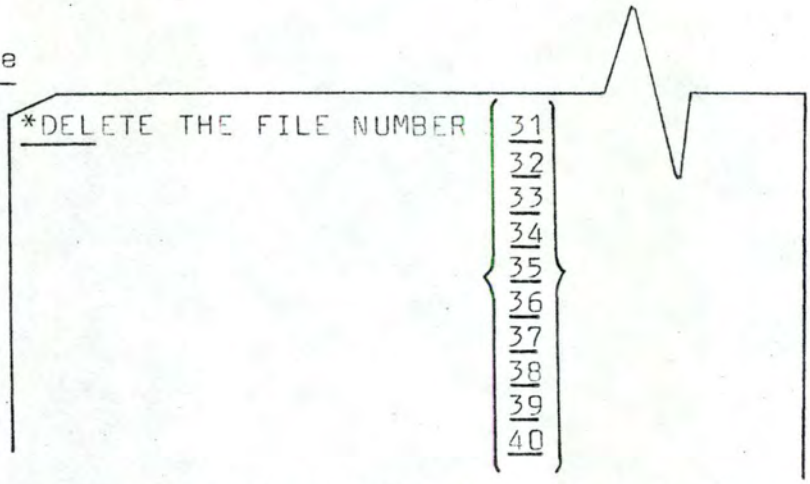
A la fin de cette exécution, le réalisateur REAL1 renvo au moniteur , pour le traitement de la commande qui suit *EXECUTE .

Rappelons que toutes les données nécessaires à l'exécution du programme INSEL doivent se trouver entre la commande *EXECUTE et celle qui la suit immédiatement .

(1) Soit parce qu'aucune commande *SEL ou *READ ne précède *EXECUTE depuis la dernière commande *JOB ,
soit parce qu'une ou plusieurs erreurs ont été détectées lors de la traduction lancée par la commande *SEL .

7 / La commande *DELETE

A / Syntaxe



Les caractères '*DEL' doivent se trouver dans les positions 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *DELETE est de libérer le fichier indiqué afin qu'on puisse l'utiliser dans la commande *STORE (page 13 et 14) .

Si le fichier était déjà libre , la commande *DELETE est sans effet .

Si le fichier indiqué était occupé , son contenu est rendu irrécupérable par la commande *READ .

Dans les deux cas , le message 'FILE NO .<n° du fichier> DELETED ' est listé .

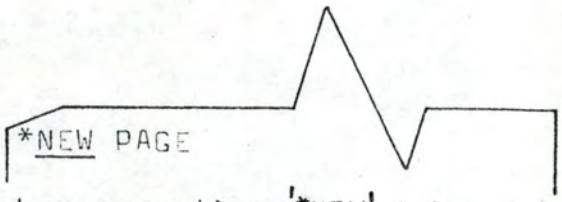
Par prudence, il est recommandé d'utiliser la commande *DELETE avant toute commande *STORE .

C / Fonctionnement

La commande *DELETE consiste simplement à écrire 0 dans le premier article du fichier ; ceci a pour effet de le faire considérer comme vide par le moniteur .

8 /La commande *NEW PAGE

A / Syntaxe



Les caractères '*NEW' doivent se trouver dans les positions 1 à 4 et, dans cet ordre .

B / Sémantique

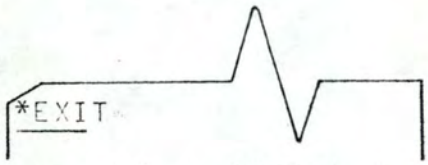
Le but de la commande *NEW PAGE est de faire passer le listing en haut de la page suivante .

C / Fonctionnement

L'effet de la commande*NEW PAGE est d'exécuter une instruction Fortran WRITE, référençant un FORMAT (1H1) .

9 / La commande *EXIT

A / Syntaxe



Les caractères '*EXI' doivent se trouver dans les positions 1 à 4 , et dans cet ordre .

B / Sémantique

Le but de la commande *EXIT est d'être la dernière du deck d'entrée .

Elle arrête le travail de l'interpréteur SEL et renvoie à la commande JCL suivante .

Il s'entend qu'il ne peut y avoir qu'une seule commande *EXIT dans le deck d'entrée , sous peine de voir l'OS essayer de traiter les commandes SEL qui suivraient la première commande EXIT .

C / Fonctionnement

L'effet de la commande *EXIT est d'exécuter une instruction Fortran CALL EXIT .

L'achèvement normal du travail est ainsi réalisé par le moniteur , sans repasser par le programme principal .

2.4 / Les instructions du langage SEL

Les instructions du langage SEL forment le programme qui applique la méthode CAT (les commandes ne servent qu'à piloter ce même programme) .

Les instructions sont au nombre de 17 ; elles se répartissent en :

- instructions d'entrée : INPUT , READ et DATA
- instructions de traitement : EFFECTIVENESS , SENSITIVITY , OPTIMIZE , ALLOCATE et PRESENT VALUE
- instructions de sortie : OUTPUT , TITLE , PRINT et DISPLAY
- et instructions utilitaires : ARITHMETIC , BRANCH , REPEAT , STOP et END .

Elles sont numérotées comme suit

INPUT	1	Entrées
READ	2	
DATA	3	
EFFECTIVENESS	4	Traitement
SENSITIVITY	5	
OPTIMIZE	6	
ALLOCATE	7	
PRESENT VALUE	8	
OUTPUT	9	Sorties
TITLE	10	
PRINT	11	
DISPLAY	12	
ARITHMETIC	13	Utilitaires .
BRANCH	14	
REPEAT	15	
STOP	16	
END	17	

Nous les présentons ci-après selon le plan :

A / Syntaxe B / Sémantique C / Formats D / Code INSEL et E / Algorithme , les paragraphes C et E n'existant que si l'instruction lit des données , imprime des résultats , ou si elle utilise un algorithme qui implante une des fonctions de la méthode CAT .

1 / L'instruction INPUT

A / Syntaxe

$$[n] \quad \underline{\text{INPUT}} \quad \left\{ \begin{matrix} \underline{\text{ELC}} \\ \underline{\text{CAS}} \end{matrix} \right\}$$

Le numéro de référence n est un entier >0 et<32768 .

B / Semantique

Le but de l'instruction INPUT est d'entrer l'arbre du problème , càd de lire les caractéristiques des CE (option ELC) ou des CNE (option CAS) .

Par ' caractéristique ' , on entend

- le numéro (externe) donné par le programmeur
- le libellé descriptif du critère
- ainsi que , pour les CE , la fonction d'efficacité , et pour les CNE , les numéros externes des CIS , leurs poids respectifs et le mode d'aggrégation .

Signalons aussi que ELC est l'abréviation de ELEMENTARY CRITERIA , et que CAS est l'abréviation de CRITERIA AGGREGATION STRUCTURE , càd les CNE .

Les données lues par l'instruction INPUT sont listées si le programmeur le demande (Voir le paragraphe C) .

C / Formats

L'entrée des CE est réalisée par un deck de cartes formé de :

- une carte contenant en colnne 1 un caractère blanc ou pas ; s'il n'est pas blanc , les informations sur chaque CE seront imprimées ; s'il est blanc , ces informations ne seront pas imprimées

- un ensemble de cartes dont chacune décrit un CE , selon le format I4,1X,F4.4,A1,20A2,5(F4.0,I2) , qui correspond aux informations : numéro externe du CE , Efficacité immédiate du CE (si le CE a une efficacité constante) , un caractère indiquant la fin du deck s'il est blanc ou que la carte décrit un CE s'il n'est pas blanc , 40 caractères décrivant sommairement l'objet du CE , et la Fonction d'Efficacité du CE , en 5 couples (Abcisse,Ordonnée) , où l'ordonnée 100 % est écrite 99 % (1) ; enfin ,

- la dernière carte du deck doit comprendre un caractère non-blanc dans la colonne 10 ; dans toutes les autres cartes , la colonne 10 doit être blanche !

En général et par clarté , la première carte porte le mot LIST dans les 4 premières colonnes , et la dernière porte le mot END dans les colonnes 10 à 12 .

L'entrée des CNE est réalisée par un deck de cartes formé de :

- une carte identique à celle utilisée pour les CE (ci-dessus)

- un ensemble de cartes dont chacune décrit un CNE , selon le format I4,1X,A4,A1,20A2,5(I4,F2.2) , qui correspond aux informations numéro externe du CNE , Opérateur d'Aggrégation du CNE , un caractère de même sens et utilisation que ci-dessus , un libellé de même sens que ci-dessus , et 5 couples (Numéro externe du CIS,poids de ce CIS) ; enfin ,

- la dernière carte est semblable à celle ci-dessus ; la même remarque s'applique au deck de cartes décrivant les CNE .

D / Code INSEL

OP = 9

D1 = 0 si ELC , 1 si CAS

(1) L'efficacité ainsi exprimée est rétablie à la valeur de 100 % pour les calculs .

2 / L'instruction READ

A / Syntaxe

$$[n]\ \underline{\text{READ}}\ \left\{\begin{matrix}\underline{K}\\ \underline{A}\\ \underline{X}\\ \underline{E}\\ \underline{C}\end{matrix}\right\}\ \left\{\begin{matrix} i \\ \left(\left\{\begin{matrix} i \\ \underline{K}i\end{matrix}\right\}\ ,\ \left\{\begin{matrix} j \\ \underline{K}j\end{matrix}\right\}\ \right)\end{matrix}\right\}\ \left[,\ \underline{\text{FORMAT}}\left\{\begin{matrix}\underline{5}\\ \underline{10}\end{matrix}\right\}\right]$$

Le numéro de référence n et les options i , j sont des entiers >0 ; n est < 32768 ; pour i et j , voir page 4 . Dans Ki et Kj , i , j $\leqslant 100$ et les valeurs de Ki et Kj doit satisfaire les contraintes ci-avant .

B / Sémantique

Le but de l'instruction READ est de lire les valeurs des variables de l'utilisateur indiquées .

L'option FORMAT permet de choisir entre les formats Fortran I5 et I10 pour le vecteur K , et les formats F5.0 et F10.0 pour les vecteurs réels A , X , E , C . Par défaut , les formats I5 et F5.0 sont utilisés .

En adressage indirect , il appartient au programmeur de vérifier les valeurs des indices Ki et Kj .

REMARQUE IMPORTANTE

L'instruction READ sert notamment à lire les valeurs des CE : READ X(1,K99) , où K99 n'est autre que NBELC , le nombre des CE ; cette variable est mise à jour par l'instruction INPUT ELC .

C / Formats

Les formats utilisés par l'instruction READ sont :

- 16I5 pour le vecteur K et l'option FORMAT 5
- 16F5.0 pour les autres vecteurs et la même option
- 8I10 pour le vecteur K et l'option FORMAT 10
- 8F10.0 pour les autres vecteurs et la même option

D / Code INSEL

OP = 4

D1 = 100 * j + f , où j est le numéro du vecteur (1,2,3,4 ou 5 pour K , A , X , E ou C) , et où f est le format (0 si pas de format , 5 pour l'option FORMAT 5 , et 10 pour l'option FORMAT 10)

D2 = Indice du premier élément à lire dans le vecteur

D3 = Indice du dernier élément à lire dans le vecteur .

Pour D2 et D3 , une valeur négative de l'indice indique l'indirection , càd que la valeur effective de l'indice n'est pas D1 ou D2 , mais K(-D1) ou K(-D2) ; voir page 5 .

3 / L'instruction EFFECTIVENESS

A / Syntaxe

$$[n]\ \underline{\text{EFFECTIVENESS}}\ \left[\left(\begin{array}{c}\underline{\text{ELEMENTARY}}\\ \underline{\text{SUBSYSTEM}}\\ i\ ,\ j\end{array}\right)\right]$$

Le numéro de référence n et les options i , j sont des entiers > 0 ; n est < 32768 ; $i \leqslant j$ et ils sont des numéros externes de critères , càd $i , j \leqslant$ NBTOTC = K100 .

B / Sémantique

Le but de l'instruction EFFECTIVENESS est de calculer l'efficacité :

- des CE (option ELEMENTARY)
- des CNE (option SUBSYSTEM)
- des critères i <u>à</u> j (option i , j) .

Si aucune option n'est choisie , l'instruction EFFECTIVENESS calcule les efficacités de <u>tous</u> les critères .

Il s'entend que les données nécessaires sont présentes , càd que les instructions INPUT ELC , INPUT CAS et READ X(1,K99) ont été exécutées .

D / Code INSEL

OP = 10

D1 et D2 sont égaux à -1 et 2 avec l'option ELEMENTARY ; à -1 et 3 avec l'option SUBSYSTEM ; à -1 et 1 sans aucune option , et à m et n avec l'option i , j , où m et n sont les numéros internes des critères dont i et j sont les numéros externes .

E / Algorithme

Le calcul de l'efficacité d'un critère diffère selon que celui-ci est élémentaire ou pas .

S'il s'agit d'un CNE , il y a encore lieu de distinguer l'absorption totale et l'absorption partielle .

a / Le calcul de l'efficacité d'un CE

L'efficacité d'ún CE se calcule à partir de sa fonction d'efficacité et d'une valeur x proposée .

La fonction d'efficacité comporte des points critiques donnés ; elle est linéaire entre ces points .

L'algorithme est donc évident :

Soient PC1 , PC2 ... PCn les points critiques , et
E(PC1) , E(PC2) ... E(PCn) leurs efficacités , où $n \geqslant 2$ et $\leqslant 5$.

Etant donnée une valeur proposée x :

Si $x \leqslant PC1$, $E = E(PC1)$
Si $x \geqslant PCN$, $E = E(PCn)$
Sinon :
trouver soit PCi tel que $x = PCi$ d'où $E = E(PCi)$
soit PCi et PCj tels que $PCi < x < PCj$ d'où

$$E = E(PCi) + \frac{E(PCj) - E(PCi)}{PCj - PCj} * (x - PCi) ,$$

où la fraction est le taux d'accroissement de l'efficacité entre PCi et PCj , avec $1 < i < n$ et $j = i + 1$.

Le calcul de l'efficacité des CE est:réalisé physiquement par la sous-routine ELEFF (ELEMENTARY EFFECTIVENESSES) du système SEL .

b / Le calcul de l'efficacité d'un CNE

Le calcul de l'efficacité d'un CNE diffère selon que celui-ci opère une absorption partielle ou une absorption totale

Cas d'absorption totale

La formule d'aggrégation est celle exposée à la page 26 du premier tome :

$$E = \left(\sum_{i}^{n} Wi \cdot Ei^{R} \right)^{1/R} \quad \text{avec } 0 < Wi < 1 \text{ et } \sum_{i}^{n} Wi = 1$$

où E est l'efficacité du CNE
Ei est l'efficacité de son i° CIS
Wi est le poids de ce i° CIS
R est le coefficient d'aggrégation , fonction du type d'aggrégation et de n , le nombre de CIS .

Pour les calculs , on utilise cette formule sous la forme :

$$E = \exp \left(\ln \left(\sum_{i}^{n} \frac{Wi \cdot Ei^{R}}{(\min Ei)^{R}} \right) / R + \ln \min Ei \right) \quad \text{si } R \geq 0$$

$$E = \exp \left(\ln \left(\sum_{i}^{n} \frac{Wi \cdot Ei^{R}}{(\max Ei)^{R}} \right) / R + \ln \max Ei \right) \quad \text{si } R < 0$$

Il s'entend que si R vaut -∞ ou + ∞ , càd si l'on a conjonction ou disjonction ou disjonction parfaite , on prendra immédiatement min Ei ou max Ei ou E .

Démontrons à présent que les trois formules ci-dessus sont égales algèbriquement , en prenant le cas où R 0 (la même démonstration vaut pour le cas où R 0 , en changeant 'min' en 'max') :

$$E = \exp \left(\ln \left(\sum_{i}^{n} Wi \cdot Ei^{R} / (\min Ei)^{R} \right) / R + \ln \min Ei \right)$$

$$E = \left(\sum_{i}^{n} Wi \cdot Ei^{R} / (\min Ei)^{1/R} \right) \cdot \min Ei$$

$$E = \left(\sum_{i}^{n} Wi \cdot Ei^{R} \right)^{1/R} \cdot 1/\min Ei \cdot \min Ei ,$$

car min Ei est une constante , d'où finalement :

$$E = \left(\sum_{i}^{n} Wi \cdot Ei^{R} \right)1/R \quad \text{cqfd} .$$

L'emploi de cette formule pour les calculs s'explique ainsi;

Le système calcule les efficacités en utilisant des variables Fortran réelles.

Celles-ci peuvent être le siège d'un sous-passement de capacité , ou d'un dépassement de capacité (UDF ou OVF) .

Or la formule initiale (page précédente) calcule n expressions de la forme Ei^R où $0 \leqslant Ei \leqslant 1$ et $R \in [-\infty , +\infty]$.

Examinons la fonction de variation de la fonction Ei^R :

VALEUR DE R	- ∞	0	1	+ ∞
VALEUR DE Ei^R	+ ∞ à 1	1	Ei	Ei à 0

On voit que si $R > 0$, il faut éviter les UDF , et que si $R < 0$, il faut éviter les OVF .

L'idée est de travailler sur une variable autre que Ei , et qu'elle soit moins sensible que Ei au défaut à éviter . D'où :

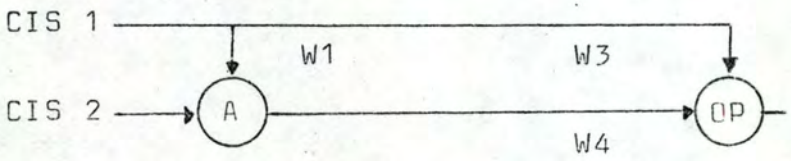
- si $R > 0$, pour éviter les UDF , on prend une variable plus grande que Ei : Ei / min Ei
- si $R < 0$, pour éviter les OVF , on prend une variable plus petite que Ei : Ei / max Ei .

L'amélioration est relative : elle résout le problème des UDF et des OVF pour l'expression Ei^R , mais on introduit du même coup le même problème pour les expressions ln min Ei et ln max Ei . Cette formule n'a toutefois jamais été mise en défaut jusqu'à présent .

Le calcul de l'efficacité d'un CNE - qu'il soit à absorption totale ou partielle - est réalisée par la sous-routine NELEFF (NON-ELEMENTARY EFFECTIVENESSES) du système SEL .

Cas d'absorption partielle

L'absorption partielle est un cas particulier d'aggrégation , dans lequel deux critères s'unissent d'abord en un critère intermédiaire , qui s'unit ensuite lui-même au premier critère (Voir les pages 35 à 38 du premier tome) :



où OP est un opérateur d'aggrégation quelconque

En A , cas où R = 1 , on calcule la simple moyenne arithmétique E4 = W1 . E1 + W2 . E2 .

En OP , on applique l'une des deux formules de calcul , en se ramenant au cas où n = 2 .

4 / L'instruction ARITHMETIC

A / Syntaxe

$$[n] \quad \left\{ \begin{matrix} K \\ A \\ X \\ E \\ C \end{matrix} \right\} \left\{ \begin{matrix} i \\ (Ki) \end{matrix} \right\} = \left\{ \begin{matrix} \text{opérande} \\ \text{opérande} \quad \left\{ \begin{matrix} + \\ - \\ * \\ / \\ \uparrow \end{matrix} \right\} \text{opérande sans signe} \end{matrix} \right\}$$

Le numéro de référence n et l'option i sont des entiers >0 ; n est < 32768 ; pour i , voir page 4.
Dans Ki , $i \leq 100$ et la valeur de Ki doit satisfaire les contrainte ci-avant .

B / Sémantique

Le but de l'instruction ARITHMETIC est de calculer la valeur d'une variable , en évaluant l'expression arithmétique située à droite du signe = .

L'expression arithmétique comporte un ou deux opérandes seulement ; c'est au programmeur à découper des expressions plus complexes .

Nous attirons l'attention du praticien sur la précision limitée des calculs opérés de cette manière (Voir pages 7 à 8) .

Signalons enfin que l'instruction ARITHMETIC sert à initialiser la variable de contrôle de la boucle BRANCH (Voir pages 32 à 34) .

D / Code INSEL

OP = 6

D1 = code interne de la variable à gauche du signe égal (entier formé de quatre chiffres , dont le premier indique le vecteur de l'utilisateur concerné , avec 1 , 2 , 3 , 4 ou 5

pour K , A , X , E , C ; les autres trois chiffres indiquent la position dans ce vecteur ; si l'ensemble est précédé du signe ' - ' , cela indique l'indirection ; voir page 8)

D2 et D3 contiennent le code interne du premier opérande (1)

D4 contient le code de l'opérateur arithmétique (0 si inexistant , et 1 , 2 , 3 , 4 ou 5 si + , - , * , / ou ')

D5 et D6 contiennent le code interne du second opérande (si D4=0 , D5 et D6 sont sans signification) (1) .

(1) Voir pages 7 et 8 .

5 / L'instruction BRANCH

A / Syntaxe

[n] BRANCH TO

- / { j / K{ i / (Ki) } } / — OPTION 1
- /j,k,l/ expr.arithm. — OPTION 2
- /j/val. finale, incrément — OPTION 3

Les numéros de référence n , j , k et l , ainsi que l'option i sont des entiers > 0 ; n est < 32768 ; j , k et l doivent être les numéros de référence d'autres instructions (page 6) ; pour i , voir page 4 . Dans Ki , i est ≤ 100 et la valeur de Ki doit satisfaire les contraintes ci-avant .

Pour la syntaxe de l'expression arithmétique , voir pages 30 à 31.

B / Sémantique

Le but de l'instruction BRANCH est de brancher à une certaine instruction sous certaines conditions .

Le branchement peut être :

- inconditionnel (option 1) , direct avec / j / ou calculé avec / Ki / et / K(Ki) /
- conditionnel (option 2) ; on branche en j , k , l selon que la valeur de l'expression est négative , nulle ou poditive .
- cyclique (option 3) ; dans cette option , l'instruction BRANCH doit être couplée avec une instruction arithmétique qui assigne une valeur initiale à la variable de contrôle de la boucle ; cette instruction est celle qui porte le numéro de référence j .

Exemples :

```
j  K1 = expression arithmétique
        instructions de la boucle
   BRANCH TO /j/ K2 , K3
```

où K2 est la valeur finale de la variable de contrôle , et où K3 est l'incrément de cette variable à chaque passage par l'instruction BRANCH ; lorsque K1 = K2 , on passe à la première instruction qui suit l'instruction BRANCH .

La variable de contrôle est une variable de l'utilisateur ; l'incrément peut être positif , négatif ou nul , car il n'y a aucun contrôle sur ce point ; tant que K1 ≠ K2 , on branche à la première instruction qui suit l'instruction arithmétique d'initialisation .

L'exemple ci-dessus aura le déroulement suivant :

1. K1 = expression arithmétique
2. Instructions de la boucle
3. K1 = K1 + K3
4. Si K3 ⩾ 0 aller en 5 sinon en 6
5. Si K1 ⩾ K2 aller en 7 sinon en 2
6. Si K1 ⩽ K2 aller en 7 sinon en 2
7. Suite du programme (après l'instruction BRANCH) .

Les paramètres K1 , K2 et K3 peuvent être modifiés par la boucle ; c'est au programmeur à assurer la logique de son programme .

D / Code INSEL

OP = 7

* Pour un branchement inconditionnel direct :

D2 = 1 (1)

D3 = numéro de référence interne où sauter

D4 = 6 (code indiquant un branchement inconditionnel)

(1) Voir page 8 .

* Pour un branchement inconditionnel calculé :

D2 = 3 (1)

D3 = Code interne de la variable Ki dont la valeur est celle du numéro de références de l'instruction où sauter

D4 = 6 (Code indiquant un branchement inconditionnel)

* Pour un branchement conditionnel :

D1 = NRI (2) de l'instruction où sauter si l'expression arithmétique est négative

D2 , D3 = Code interne (1) du premier opérande de l'expression arithmétique

D4 = Code de l'opérateur arithmétique (Voir page 31)

D5 , D6 = Code interne du second opérande de l'expression arithmétiqu

D7 = NRI de l'instruction où sauter si l'expression arithmétique est nulle

D8 = Idem D7 , si elle est positive

* Pour une boucle :

D1 = NRI de l'instruction arithmétique d'initialisation

D2 , D3 = Code interne de l'opérande contenant la valeur finale de la variable de contrôle

D4 = 7 (Code indiquant la boucle)

D5 , D6 = Code interne de l'opérande ' incrément ' .

Les arguments inutilisés n'ont aucune influence sur l'instruction

(1) Voir page 8
(2) NRI = numéro de référence interne de l'instruction , càd son numéro si on compte les instructions par la séquence des entiers naturels .

6 / L'instruction REPEAT

A / Syntaxe

[n] REPEAT i , j

Les numéros de référence n et i , ainsi que le paramètre j sont des entiers > 0 et < 32768 ; i est le numéro de référence d'une autre instruction (Voir page 6) . externe

B / Sémantique

Le but de l'instruction REPEAT est de brancher j fois à l'adresse i , puis de continuer en séquence .

Un compteur est initialisé à j lors de la première exécution de l'instruction REPEAT ; il est décrémenté de 1 à chaque exécution ultérieure.

On peut exécuter l'instruction REPEAT moins que j fois car il n'y a pas de contrôle en dehors de son exécution : dans ce cas , elle se ramène à un branchement inconditionnel .

On peut l'exécuter j fois ou plus de j fois : les j premières brancheront à l'instruction i et les suivantes à celle qui suit REPEAT

D / Code INSEL

OP = 3

D1 = NRI de l'instruction où brancher

D2 = j , le nombre de répétitions

D3 = variable interne initialisée à D2 , décrémentée de 1 à chaque passage et restaurée à D2 après avoir atteint la valeur 0

7 / L'instruction OUTPUT

A / Syntaxe

[n] OUTPUT ({ EE / SE / E }) [SORTED / HISTOGRAM / SORTED , HISTOGRAM / HISTOGRAM , SORTED]

Le numéro de référence n est un entier > 0 et < 32768 .

B / Sémantique

Le but de l'instruction OUTPUT est d'imprimer les efficacités obtenues par les CE (option EE) , par les CNE (option SE) , ou par tous les critères (option E) .

Les symboles EE , SE et E sont les abréviations respectives des expressions ELEMENTARY EFFECTIVENESSES , SUBSYSTEM EFFECTIVENESSES et (ALL) EFFECTIVENESSES ; le terme SUBSYSTEM désigne la structure d'aggrégation sans les CE , càd l'ensemble des CNE .

L'impression comporte le libellé de chaque critère et son efficacité ; cette liste est en vrac , mais elle peut être préalablement triée sur les efficacités croissantes (option SORTED) .

On peut aussi demander un histogramme de fréquence des efficacités (option HISTOGRAM) ; les classes de fréquence sont 0 à 10 % , 10 à 20 % , etc ; dans ces classes , la borne supérieure est toujours incluse , et la borne inférieure est toujours exclue , sauf pour la classe 0 à 10 % , qui comprend sa borne inférieure 0 % .

D / Code INSEL

OP = 11

D1 = un entier de quatre chiffres , dont les deux premiers indiquent les résultats souhaités (01 pour SE , 10 pour EE et 11 pour E) , et les deux derniers indiquent le mode d'impression (00 si néant , 01 si HISTOGRAM , 10 si SORTED et 11 si les deux) .

8 / L'instruction TITLE

A / Syntaxe

$$[n]\ \underline{\text{TITLE}}\ \left\{\begin{matrix}1\\2\\3\end{matrix}\right\}, \left\{\begin{matrix}i\\ \text{' suite de caractères '}\end{matrix}\right\}$$

Le numéro de référence n et l'option d'impression i sont des nombres entiers > 0 et < 32768 . Dans l'option ' suite de caractères ' , il doit y avoir au moins un caractère entre les deux quotes ; la suite de caractères peut contenir des quotes .

B / Sémantique

Le but de l'instruction TITLE est de lister un texte sur le listing.

L'option i liste les i premières cartes disponibles du deck de données ; celui-ci peut avoir été déjà entamé par l'exécution préalable d'autres instructions .

L'option 'suite de caractères ' liste cette suite de caractères ; ils sont conservés dans le FICHIER-TITRES (numéro 42) , où le traducteur les a installés .

L'espace séparant le texte listé par l'instruction TITLE du texte listé précédemment est déterminé par un chiffre :

- 1 pour lister à la suite , sans plus
- 2 pour sauter une ligne
- 3 pour sauter en tête de la page suivante .

D / Code INSEL

OP = 2

D1 = 1 , 2 ou 3 (code de saut du papier)

D2 = première position de la zone du fichier-titres où la suite de caractères est enregistrée , où 0 si le texte se trouve sur carte(s)

D3 = dernière position de cette zone , ou le nombre des cartes à lire si le texte se trouve sur carte(s) .

9 / L'instruction PRINT

A / Syntaxe

$$\left[n\right] \quad \underline{\text{PRINT}} \quad \left\{\begin{matrix}\underline{K}\\ \underline{A}\\ \underline{X}\\ \underline{E}\\ \underline{C}\end{matrix}\right\}\left\{\begin{matrix} i \\ \left(\left\{\begin{matrix} i \\ \underline{K}i \end{matrix}\right\} , \left\{\begin{matrix} j \\ \underline{K}j \end{matrix}\right\} \right) \end{matrix}\right\} \quad \left[\underline{\text{COLUMN}}\right]$$

Le numéro de référence n et les options i , j sont des entiers > 0 ; $n < 32768$; pour i et j voir page 4.
Dans Ki et Kj , i , j $\leqslant 100$ et les valeurs de Ki et Kj doivent satisfaire les contraintes ci-avant .

B / Sémantique

Le but de l'instruction PRINT est d'imprimer le contenu de la ou des variable(s) de l'utilisateur spécifiée(s) .

L'impression a lieu par ligne , à raison de 10 valeurs par ligne . L'option COLUMN donne lieu à une impression en colonne .

C / Formats

Les formats d'impression sont :

- E12.5 si la valeur maximale à imprimer est > 999999 ou si la valeur minimale à imprimer est < 0.0001 .
- F12.7 si $0.0001 \leqslant min \leqslant 1$ et $max \leqslant 999999$
- F12.4 si $min > 1$ et $max \leqslant 999999$

Synoptiquement :

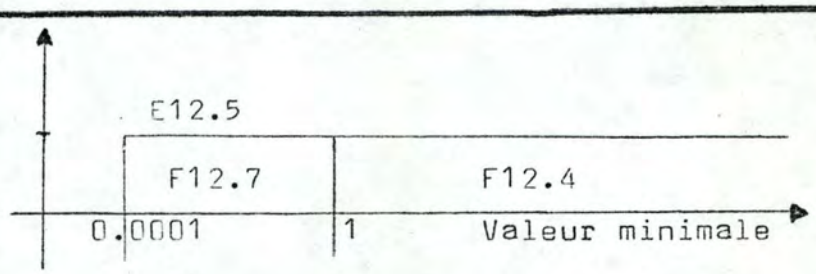


FIGURE 1
Formats d'impression utilisés par l'instruction PRINT , selon les valeurs minimales et maximales à imprimer .

D / Code INSEL

OP = 5

D1 = 100 * j + c où j est la même chose qu'à l'instruction READ (page 24) , et c est le code d'impression (0 si impression par ligne et 1 si impression par colonne)

D2 et D3 sont la même chose qu'à l'instruction READ .

10 / L'instruction DISPLAY

A / Syntaxe

$$[n]\ \text{DISPLAY}\ \left(\ \left\{\begin{array}{l}\underline{\text{ELC}}\\ \underline{\text{SENSITIVITY}}\ \text{TABLE}\ \left\{\begin{array}{l}\underline{1}\\ \underline{2}\end{array}\right\}\\ \underline{\text{OPTIMIZATION}}\ \left\{\begin{array}{l}\underline{1}\\ \underline{2}\end{array}\right\}\end{array}\right\}\ \right)\ \left[\begin{array}{l}i\\ \underline{K}i\end{array}\right]$$

Le numéro de référence n et les oprions i sont des entiers >0 ; n est <32768 ; dans l'option SENSITIVITY , $1 \leqslant i \leqslant 7$; dans l'option ELC , $i \leqslant NBELC = K99$; dans l'option OPTIMIZATION , $K99 = NBELC < i \leqslant NBTOTC = K100$; dans Ki , $i \leqslant 100$, et la valeur de Ki doit respecter les contraintes ci-avant .

B / Sémantique

Le but de l'instruction DISPLAY est de dessiner les fonctions spécifiées càd :

- les fonctions d'efficacité des CE (option ELC)
- les fonctions de sensibilité aux CE (option SENSITIVITY TABLE 1) ou aux CEN (option SENSITIVITY TABLE 2)
- les fonctions d'optimisation enregistrées dans le fichier Fortran 93 (VBA.ARN.93) avec l'option OPTIMIZATION 1 , ou ces mêmes fonctions en terme du rapport Efficacité/Coût , avec l'option OPTIMIZATION 2 .

L'option $\left[\begin{array}{l}i\\ \underline{K}i\end{array}\right]$ a l'effet suivant :

- dans l'option ELC , seule la fonction d'efficacité du CE de numéro interne donné est listée
- dans l'option SENSITIVITY TABLE , la table listée est triée sur les valeurs croissantes d'un des paramètres a , ar , b , br , d+ , d- ou p , selon la valeur de i ou de Ki
- dans l'option OPTIMIZATION , seule la fonction d'optimization du critère de numéro interne donné est listée .

Si cette option est absente , tous les critères ou paramètres sont traités .

C / Formats

Les options ELC et OPTIMIZATION donnent lieu au dessein de leur objet sous la forme d'un graphique cartésien ; celui-ci est réalisé par la sous-routine DIS100 .

Dans l'option SENSITIVITY TABLE , seul un tableau de l'objet est listé , car l'instruction SENSITIVITY assure elle-même l'impression du graphe cartésien de la fonction de sensibilité .

D / Code INSEL

OP = 13

D1 = 1 si ELC , 2 si SENSITIVITY TABLE 1 , 3 si SENSITIVITY TABLE 2 , 4 si OPTIMIZATION 1 et 5 si OPTIMIZATION 2

D2 = spécificateur (0 si vide , i dans l'option i et - i dans l'option Ki) .

11 / L'instruction SENSITIVITY

A / Syntaxe

$$[n]\ \underline{\text{SEN}}\text{SITIVITY}\ \left(\left\{ \begin{matrix} \underline{\text{ALL}} \\ i \\ \underline{K}i \end{matrix} \right\} , \left\{ \begin{matrix} j \\ \underline{K}j \end{matrix} \right\} \right) \left[\begin{matrix} \underline{\text{D}}\text{ISPLAY} \\ \underline{\text{P}}\text{RINT} \\ \underline{\text{D}}\text{ISPLAY}\ ,\ \underline{\text{P}}\text{RINT} \\ \underline{\text{P}}\text{RINT}\ ,\ \underline{\text{D}}\text{ISPLAY} \end{matrix} \right]$$

Le numéro de référence n et les options i , j sont des entiers >0 ; n est < 32768 ; $i \leqslant j$ et $1 \leqslant i , j \leqslant$ NBTOTC ; dans l'option Ki ou Kj , i , $j \leqslant 100$ et les valeurs de Ki et Kj doivent satisfaire les contraintes ci-avant .

B / Sémantique

Le but de l'instruction SENSITIVITY est de calculer la fonction de sensibilité de l'efficacité d'un CNE à l'efficacité d'un de ses critères subordonnés .

Dans l'option ALL , le CNE est le critère global , et on considère tous les autres critères successivement .

Dans l'autre option , le CNE a pour numéro interne j ou Kj , et le critère subordonné , i ou Ki .

Le listage des paramètres d'une fonction de sensibilité est assuré par l'instruction DISPLAY , pages 41 à 42.

D / Code INSEL

OP = 12

D1 = 0 si ALL , i dans l'option i et - i dans l'option Ki

D2 = 0 si ALL , j dans l'option j et - j dans l'option Kj

D3 = 0 si spécificateur vide , 1 si DISPLAY , 10 si PRINT et 11 si PRINT et DISPLAY

E / Algorithme

L'algorithme utilisé par la fonction de sensibilité est exposé dans le 1° tome , pages 45 à 46 .

La fonction de sensibilité obtenue est décrite par les paramètre a , ar , b , br , d+ , d- et p .
Ceux-ci sont définis comme suit : étant donnée une fonction de sensibilité (Figure 2) , on calcule les paramètres selon les formules :

- a = Emax - Emin , càd que a est l'intervalle absolu d'influence de l'efficacité du critère subordonné sur celle du critère ascendant

- ar = a / Emax , càd l'intervelle relatif d'influence

- b = Emax - $\int_0^1$ F . dE où F est la fonction de sensibilité et E est l'efficacité du critère subordonné ;
b est donc la surface comprise entre la fonction de sensibilité et les droites d'aquations x = 0 , x = 1 et y = Emax ; elle est une mesure absolue (1)

- br = b / (Emax - Emin) , càd que br est la mesure b relative à la différence Emax - Emin

- d+ = Emax - E° , càd l'efficacité supplémentaire que le critère ascendant peut acquérir si on porte l'efficacité du critère subordonné de E à 100 % (où E° est l'efficacité obtenue par le critère ascendant lors de l'évaluation qui a précédé obligatoirement l'analyse de sensibilité)

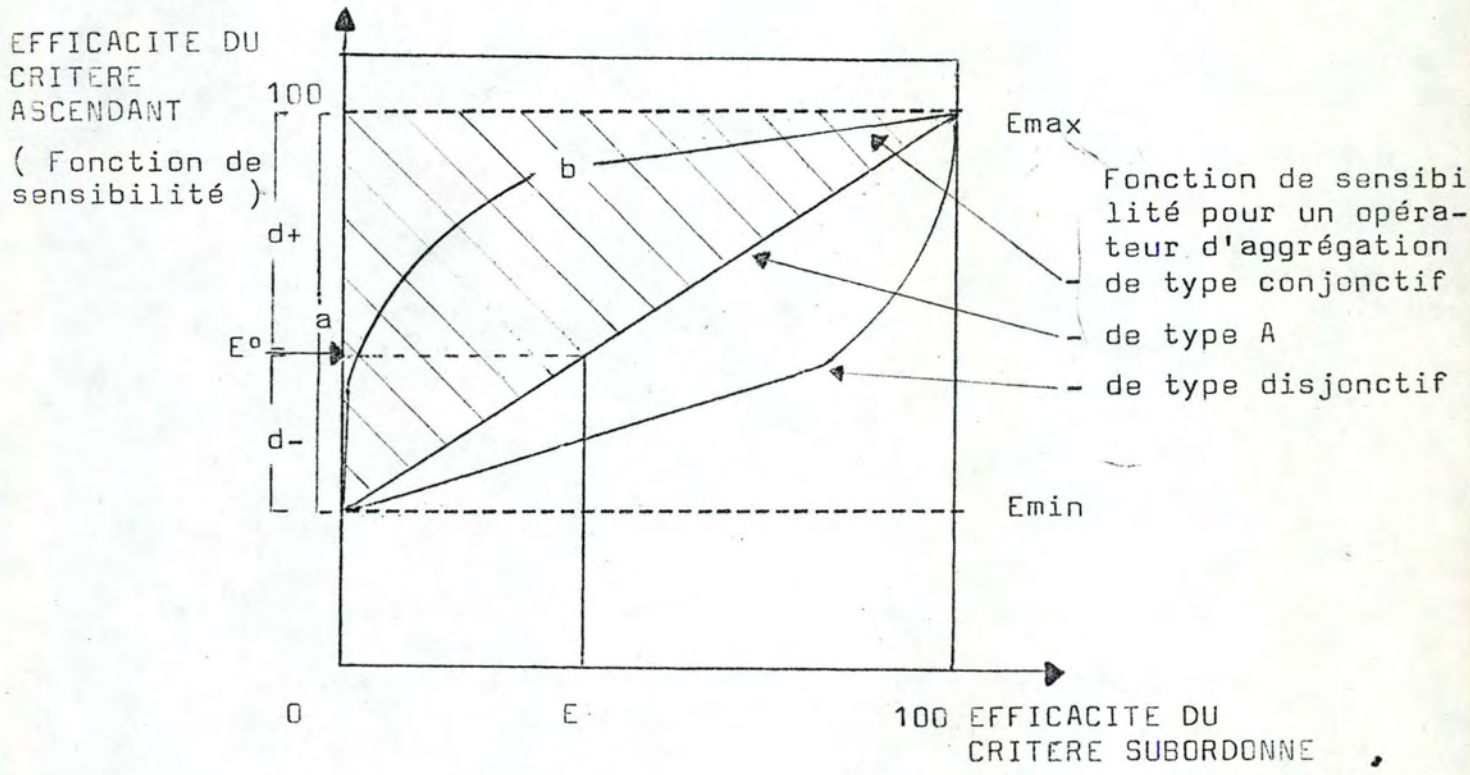
- d- = E° - Emin , càd l'efficacité que perdrait le critère ascendant si on portait l'efficacité du critère subordonné de E à 0 %

- p = (E° - Emin) / (Emax - Emin) , càd la position de E° dans l'intervalle Emin , Emax .

(1) On peut voir sur la Figure 2 que b est une fonction croissante de R ; en d'autres termes , plus l'opérateur d'aggrégation est disjonctif (càd plus le coefficient d'aggrégation R est grand) , plus la surface de mesure b est grande .
Il en est de même pour br .

FIGURE 2

Illustration sur le graphe cartésien des paramètres descriptifs de la Fonction de Sensibilité (sauf les paramètres relatifs ar et br , ainsi que le paramètre positionnel p) .



12 / L'instruction DATA

A / Syntaxe

$$\left[n\right] \quad \underline{\text{DATA}} \quad \text{FOR ELC} \quad \left(\left\{ \begin{matrix} i \\ \underline{K}i \end{matrix} \right\} \right) \quad \left[, \quad \underline{\text{PRINT}} \right]$$

Le numéro de référence n et l'option i sont des entiers >0 ; n est < 32768 ; i est $\leqslant$ NBELC = K99 ; dans Ki , i est $\leqslant 100$, et la valeur de Ki doit respecter les contraintes ci-avant .

B / Sémantique

Le but de l'instruction DATA est d'entrer le CEM du CE indiqué ; la lecture a lieu dans les vecteurs de l'utilisateur X et C .

L'option PRINT commande au listage de ces données .

C / Formats

Les données de l'instruction DATA sont :

- d'abord , une carte lue selon le format I5 ; le nombre lu est le nombre des couples formant la CEM du CE considéré
- ensuite , le nombre de cartes nécessaires pour contenir l'ensemble des valeurs d'entrée possibles , selon le format 8F10.0
- enfin , le nombre de cartes nécessaires pour contenir l'ensemble des coûts correspondants aux valeurs d'entrée possibles , selon le format 8F10.0 .

D / Code INSEL

OP = 14

D1 = i si option i , - i si option Ki

D2 = 1 si PRINT , 0 sinon .

13 / L'instruction OPTIMIZE

A / Syntaxe

[n] OPTIMIZE NELC ({ i / Ki }) [LIMIT ({ j / Kj })] [, PRINT]

Le numéro de référence n et les options i , j sont des entiers > 0 ; n est < 32768 ; K99 = NBELC < i ⩽ NBTOTC = K100 ; j est ⩽ 160 ; dans Ki et Kj , 1 ⩽ Ki , Kj ⩽ 100 et les valeurs de Ki et de Kj doivent respecter les contraintes ci-avant .

B / Sémantique

Le but de l'instruction OPTIMIZE est de calculer la CEM du critère donné .

L'option LIMIT réduit le nombre de points critiques de la fonction d'efficacité calculée au nombre maximum de 160 , qui est aussi la valeur preset .

L'option PRINT liste un tableau qui décrit la CEM calculée du critère .

ATTENTION : L'instruction OPTIMIZE ne concerne qu'un seul critère ! Pour optimiser tout un arbre , le programmeur doit créer une boucle qui fait varier le numéro du critère .

D'autre part , tous les critères subordonnés doivent déjà être optimisés ; pour les critères élémentaires , la fonction d'efficacité maximale est entrée par l'instruction DATA .

D / Code INSEL

OP + 15

D1 = i si option i , -i si option Ki

D2 = 1 si PRINT , 0 sinon

D3 = j si option j , - j si option Kj , 160 par défaut

E / Algorithme

L'algorithme utilisé par la fonction d'optimisation est le suivant :

Soient i = 1 , 2 .., Ki le nombre de composantes de la CEM du premier CIS , où la composante numéro 1 est la moins efficace et donc la moins chère (1)

j = 1 , 2 ... Kj la même chose pour le second CIS

B le budget disponible

Cij le coût du couple (i , j) = C_1 (i)+ C_2 (j)

Eij l'efficacité du couple (i , j) .

En commençant avec comme budget B le coût du couple (i , j) le plus efficace donc le plus cher , le processus est le suivant :

INITIALISATIONS

i = Ki
j = 1
Emax = 0

ITERATIONS

a. si Cij > B goto b
sinon
augmenter j tant que Cij ⩽ B et j ⩽ Kj , puis calculer Eij .
Emax = max (Emax , Eij) , et mémoriser les composantes du couple retenu
j = j + 1
si j > Kj ⟶ FIN

b. i = i - 1
si i < 1 ⟶ FIN
goto a .

On voit que cet algorithme n'opère que sur des arbres binaires ; En SEL , il s'agit du vecteur NF(31) équivalencé au vecteur ISUB , exposé à la page 59 .

Signalons que le résultat de l'optimisation , càd les CEM des critères , sont conservés dans le fichier 93 , décrit aux pages 66 et 67 .

Lorsque ce processus est terminé , on détient le couple d'efficacité maximale pour le budget B . On réitère alors avec B = 0.999*B , et ce jusqu'à ce que le budget B soit si faible qu'aucun couple ne soit plus possible .

L'ensemble des couples (i , j) vainqueurs des différentes itérations de ce processus forment , avec leurs efficacités respectives ,la CEM du critère global .

14 / L'instruction ALLOCATE

A / Syntaxe

$$\left[n\right]\ \underline{\text{ALL}}\text{OCATE}\quad \text{AMOUNT}\ \left[\text{opérande sans signe}\right]\ (\ \underline{\text{SUBSYSTEM}} =$$

$$\left\{\begin{matrix} i \\ \underline{K}i \end{matrix}\right\},\ \text{RESULT} = \left\{\begin{matrix} \underline{\text{C}}\text{OST} \\ \underline{\text{P}}\text{ARAMETER} \end{matrix}\right\}\ )\quad \left[\underline{\text{P}}\text{UNCH}\right]$$

Le numéro de référence n et l'option i sont des entiers >0 ; n est <32768 ; i est $\leq$ NBTOTC , avec NBTOTC ≤ 31 ; dans l'option Ki , i est ≤ 100 et la valeur de Ki doit satisfaire les contraintes ci-avant .

B / Sémantique

Le but de l'instruction ALLOCATE est d'allouer un budget à un critère quelconque , de façon à maximiser son efficacité pour ce budget .

La configuration d'efficacité maximale est déduite de la CEM de ce critère ; cette configuration est décrite soit par les valeurs d'entrée des CE du sous-arbre dont le critère est la racine (option PARAMETER) , ou par les efficacités de ces CE (option COST) .

Si aucun opérande n'est cité , toutes les configurations formant les couples de la CEM sont listées .

L'option PUNCH - inhibée dans la version SEL77 sur le système SIEMENS 4004/151 - permettait la perforation de ces résultats , avec pour but un traitement infographique ultérieur .

ATTENTION : L'instruction ALLOCATE se fonde sur les résultats de l'instruction OPTIMIZE ; ces résultats sont conservés dans le fichier Fortran 93 (nommé VBA.ARN.93 dans le système) .
Il appartient au programmeur de vérifier la logique de son programme .

D / Code INSEL

OP = 17

D1 et D2 contiennent le code interne de l'opérande sans signe , ou 0 et 0 s'il n'est pas spécifié

D3 = i dans l'option i et - i dans l'option Ki

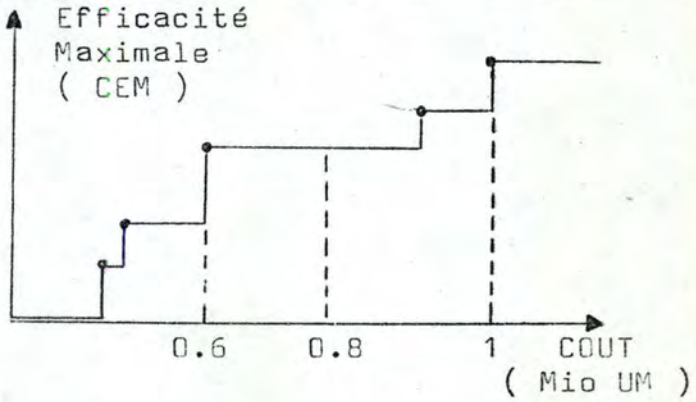
D4 = p . r , où r = 1 dans l'option COST , r = 2 dans l'option PARAMETER , p = 1 dans l'option PUNCH et p = - 1 sans l'option PUNCH

E / Algorithme

Soit une CEM (Figure 3) .

FIGURE 3

Exemple de CEM pour un critère quelconque



Affectons un budget de n UM .

Pour cela , plaçons-nous en l'abcisse x = n UM ; considérons son ordonnée : si celle-ci représente une configuration , elle est la solution .

Sinon , déplaçons-nous de l'abcisse x = n UM vers les abcisses de valeur inférieure : dés que l'on rencontre une abcisse dont l'ordonnée représente une configuration , on a la solution .

Sur la Figure 3 , les deux possibilités de l'algorithme sont illustrées par les budgets de 1 Mio UM et 0.8 Mio UM respectivement .

Concrètement , le système SEL cherche simplement deux configurations de couples (C1 , E1) et (C2 , E2) , qui se suivent immédiatement , et telles que $C1 \leqslant B < C2$; la première configuration est alors la solution .

15 / L'instruction PRESENT VALUE

A / Syntaxe

$$[n]\ \underline{\text{PRESENT}}\ \text{VALUE} = \left\{\begin{matrix}\underline{A}\\ \underline{X}\\ \underline{F}\\ \underline{C}\end{matrix}\right\} \quad \left\{\begin{matrix} i \\ (\underline{K}i)\end{matrix}\right\}$$

pour x, voir page 4 ;

Le numéro de référence n et l'option i sont des entiers > 0 ; n est < 32768 ; dans l'option Ki , $i \leqslant 100$ et la valeur de Ki doit satisfaire les contraintes ci-avant .

Les variables du vecteur K sont absentes , car le résultat est réel .

B / Sémantique

Le but de l'instruction PRESENT VALUE est de calculer la valeur actualisée de l'investissement formé d'un ou de plusieurs cash-flows , décrit en entrée de la manière exposée au paragraphe C .

La fonction d'actualisation et les formules utilisées sont exposées dans le premier tome , pages 47 et 48 .

C / Formats

L'ensemble des cartes qui décrivent l'investissement à actualiser sont les suivantes :

- un nombre quelconque de cartes commentaires , dont le texte commence à la colonne 31 , et dont les colonnes 1 à 30 sont blanches
- un nombre quelconque de cartes dont chacune décrit un cash-flow selon le format F10.0,F5.0,3F5.0,50A1 ; ce format correspond aux grandeurs Ci , ri , mi , pi et ki , suivies d'un commentaire quelconque ; le sens de ces grandeurs est également exposé dans le premier tome , pages 47 et 48
- enfin , une carte entièrement blanche qui indique la fin du deck de données pour l'instruction PRESENT VALUE .

D / Code INSEL

OP = 16

D1 = Code interne de la variable réceptrice ; il s'agit uniquement de la seconde valeur du code interne ; la première n'est pas nécessaire , car seule une variable peut intervenir dans l'instruction PRESENT VALUE .

16 / L'instruction STOP

A / Syntaxe

[n] STOP

Le numéro de référence n est un entier > 0 et < 32768 .

B / Sémantique

Le but de l'instruction STOP est de terminer l'exécution du programme INSEL .

Il peut y avoir plusieurs instructions STOP dans le programme ; il doit y en avoir au moins une ; elle(s) peu(ven)t se trouver n'importe où dans le programme , avant l'instruction END et normalement pas comme première instruction .

L'exécution du programme INSEL est lancée par la commande EXECUTE . Quand elle s'achève par l'instruction STOP , l'interpréteur passe alors à la commande suivante .

D / Code INSEL

OP = 1

Il n'y a pas d'arguments .

17 / L'instruction END

A / Syntaxe

END

L'instruction END n'a pas de numéro de référence .

B / Sémantique

Le but de l'instruction END est d'être la dernière du deck d'instructions dont elle signale la fin au traducteur .

Il ne peut y avoir qu'une seule instruction END dans un deck d'instructions ; elle doit nécessairement en être la dernière .

Dans le cas d'un stream de programmes , il s'entend que chaque programme a sa propre instruction END (Voir le paragraphe 2.2.6 , page 7) .

On voit que l'instruction END provoque la fin de la traduction des instructions , tandis que l'instruction EXIT provoque la fin de leur exécution .

D / Code INSEL

L'instruction END n'a pas de code INSEL ; en effet , elle ne joue aucun rôle dans l'exécution du programme INSEL ; elle n'a donc pas à être traduite en code INSEL .

C H A P I T R E 3

LE SYSTEME SEL

3.1 / Présentation

Le système SEL constitue un interpréteur qui implémente le langage SEL , exposé au chapitre 2 ci-avant (pages 3 à 54) .

Les grandes lignes du système SEL ont été exposées dans le premier tome , pages 49 et 51 à 53 .

Dans les paragraphes qui suivent , nous présentons successivement le fonctionnement du système SEL , la structure de sa mémoire interne et celle de sa mémoire externe .

3.2 / Le fonctionnement du système SEL

Le fonctionnement du système SEL peut se concevoir dans deux optiques :

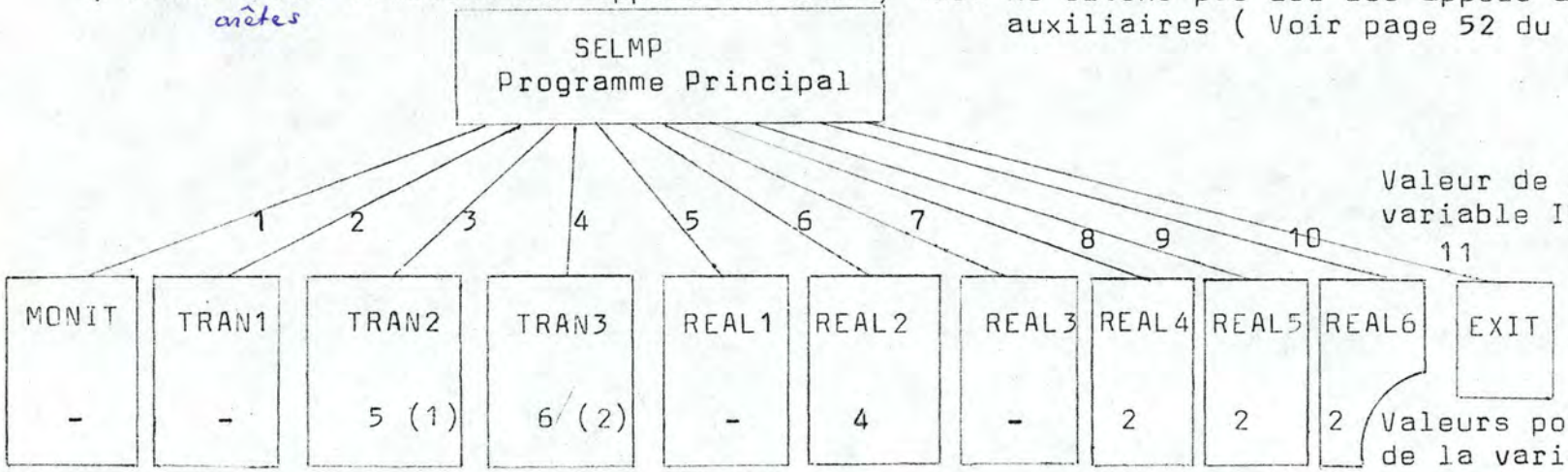
- dans une optique informatique , càd par l'articulation des sous-routines entre elles au moyen d'appels ; ceci est présenté à la Figure 4 , page 56 ; on y voit l'utilisation concrète des variables IPROG et IENT , qui est exposée à la page 56
- dans une optique cybernétique , càd par les trois étapes Entrées - Traitement - Sorties (Figure 5 , page 57) .

Enfin , il nous paraît indispensable de présenter l'architecture interne du système SEL , qui lui permet de mémoriser un arbre sous les formes nécessaires au traitement des instructions (Figures 6 à 8 , pages 58 à 60) .

FIGURE 4

Le fonctionnement interne du système SEL .

(Les ~~flèches~~ arêtes dénotent les appels Fortran ; nous ne citons pas ici les appels aux sous-routines auxiliaires (Voir page 52 du 1° tome)) .



(les tirets '-' indiquent que IPROG n'est pas utilisé au sens strict où nous l'entendons)

(1) La sous-routine TRAN2 traduit 6 instructions , mais en 5 entrées , car READ et PRINT sont traduites ensemble

(2) La sous-routine TRAN3 traduit 7 instructions , mais en 6 entrées , car DATA et OPTIMIZE sont traduites ensemble .

INTERPRETATION ET EXECUTION IMMEDIATE DES COMMANDES

TRADUCTION DES INSTRUCTIONS STOP , TITLE ET REPEAT ; DETECTION DE L'INSTRUCTION END

TRADUCTION DES INSTRUCTIONS READ , PRINT , ARITHMETIC , BRANCH , INPUT ET EFFECTIVENESS

TRADUCTION DES INSTRUCTIONS OUTPUT , SENSITIVITY , DISPLAY , DATA , OPTIMIZE , PRESENT VALUE ET ALLOCATE

EXECUTION DES INSTRUCTIONS STOP , TITLE , REPEAT , READ ET PRINT

EXECUTION DES INSTRUCTIONS ARITHMETIC , BRANCH , SENSITIVITY ET EFFECTIVENESS

EXECUTION DE L'INSTRUCTION INPUT

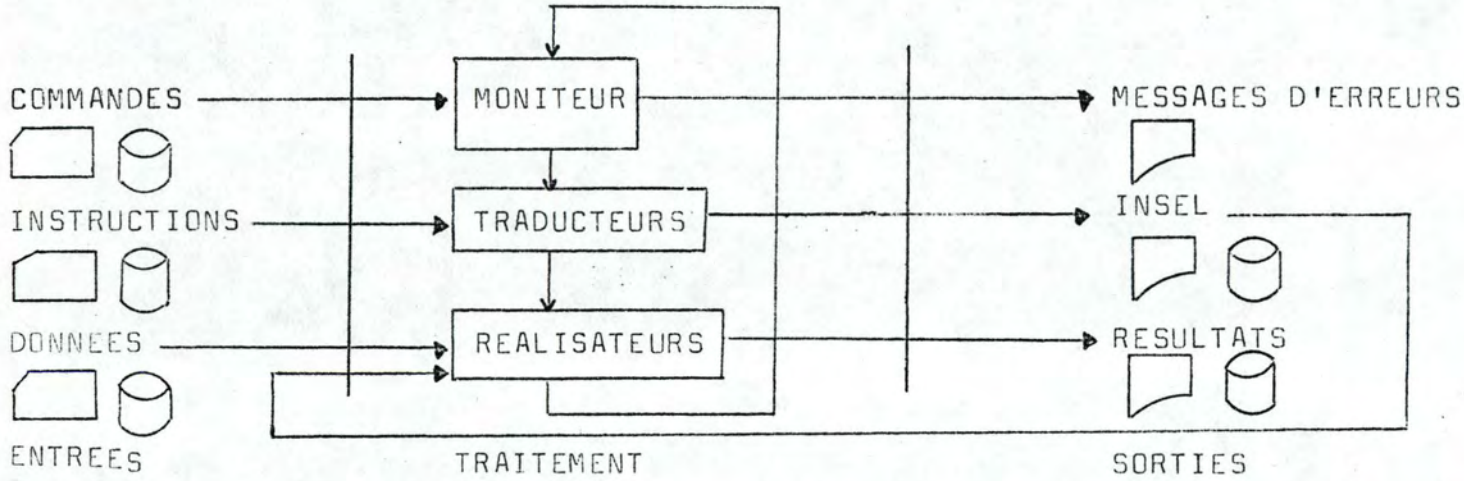
EXECUTION DES INSTRUCTIONS OUTPUT ET DISPLAY

EXECUTION DES INSTRUCTIONS DATA ET OPTIMIZE

EXECUTION DES INSTRUCTIONS PRESENT VALUE ET ALLOCATE

FIGURE 5

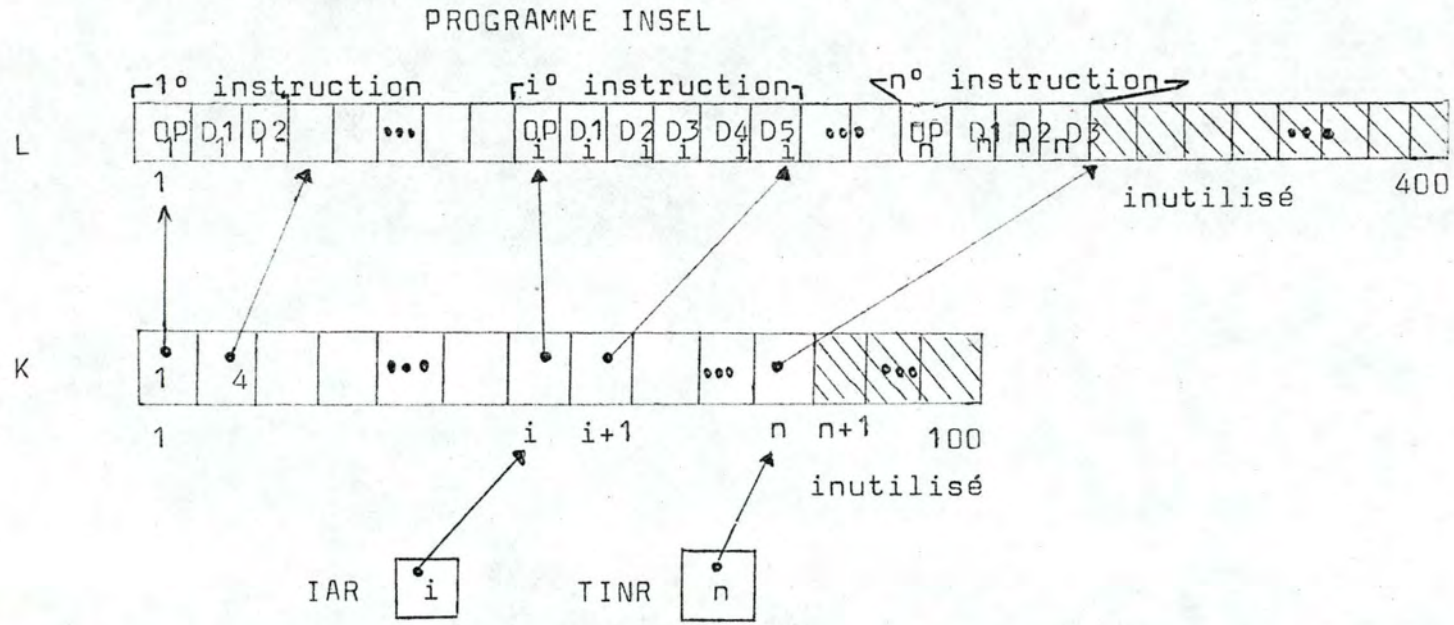
Le fonctionnement du système SEL d'un point de vue cybernétique : Entrées , Traitement et Sorties .



Par clarté , nous avons simplifié les éléments de cette Figure : ainsi , nous avons groupé les commandes , aloes qu'elles sont divisées en deux groupes dans le programme-source , de part et d'autre des instructions . De plus , nous simplifions l'étape de traitement en ne donnant au moyen des arcs , que la succession chronologique des sous-routines .

FIGURE 6

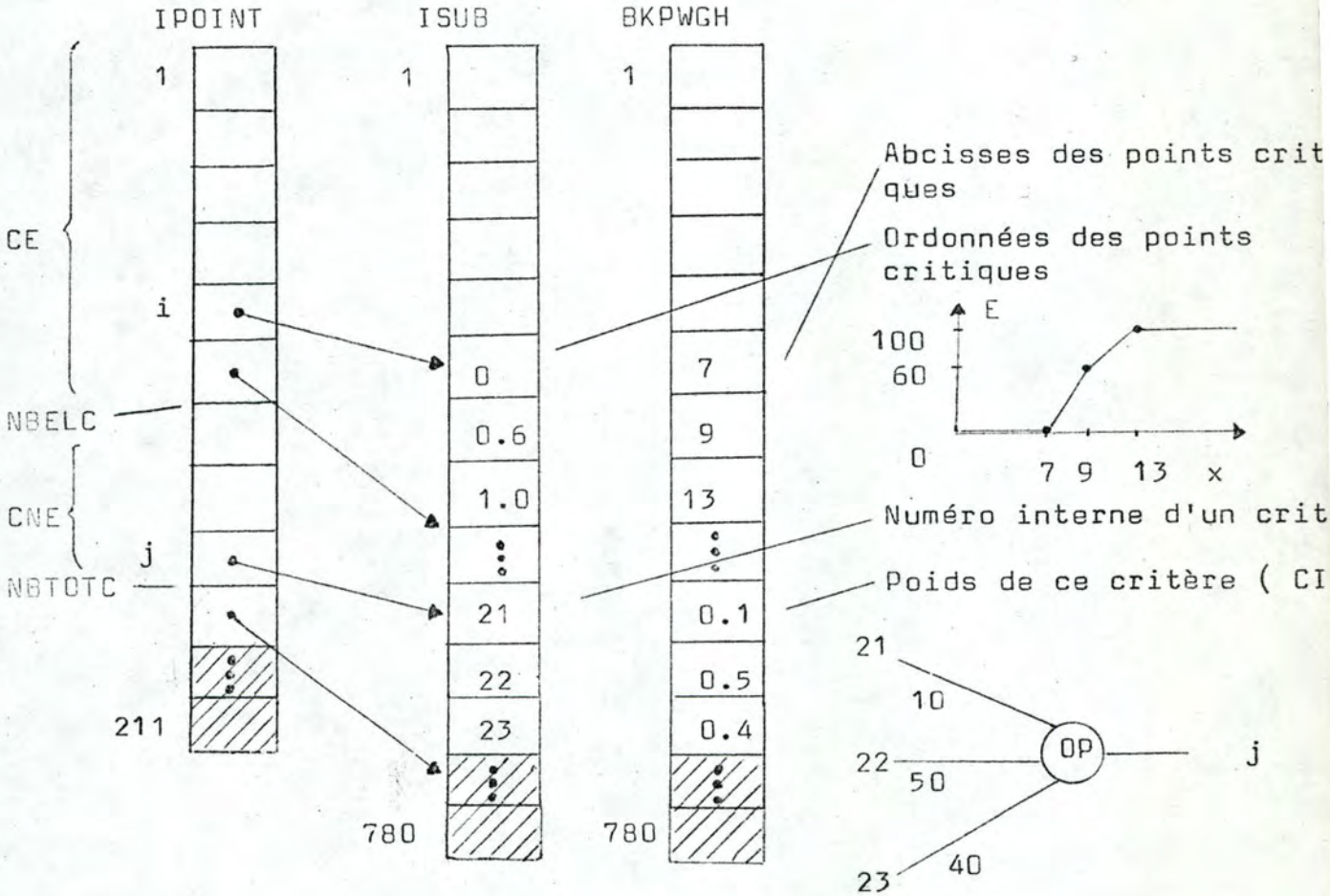
Support physique du programme INSEL .
Note . Le vecteur K pointe sur le vecteur L ; celui-ci contient le programme INSEL ; la i° instruction a pour champ les positions L(K(i)) à L (K(i+1) - 1) .



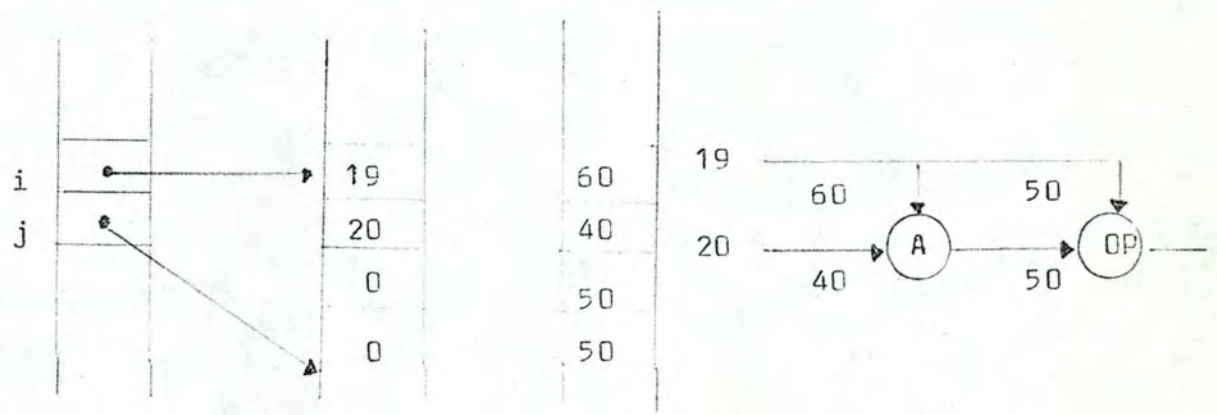
où IAR est le Registre d'Index des Adresses (il contient l'adresse courante) et TINR est le Registre du Nombre Total d'Adresses .

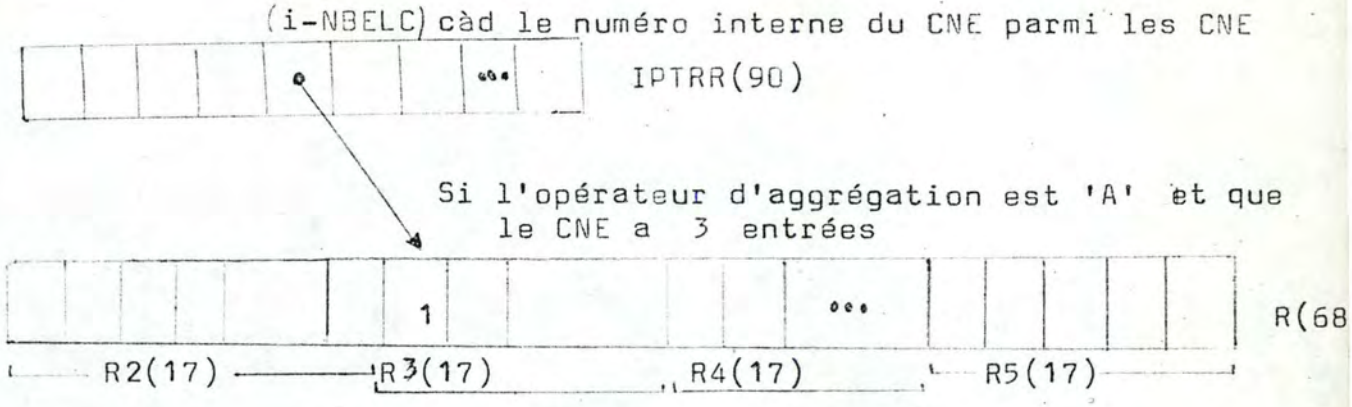
FIGURE 7

La mémorisation interne de la structure de l'arbre : pour les CE , on mémorise les points critiques de la Fonction d'Efficacité ; et pour les CNE , on mémorise le numéro interne et le poids de chaue CIS . Pour chaque CNE , la variable IPTRR(numéro interne du CNE - NBELC) pointe sur le vecteur R (Voir page 64) .



Mémorisation pour les cas d'Absorption Partielle :

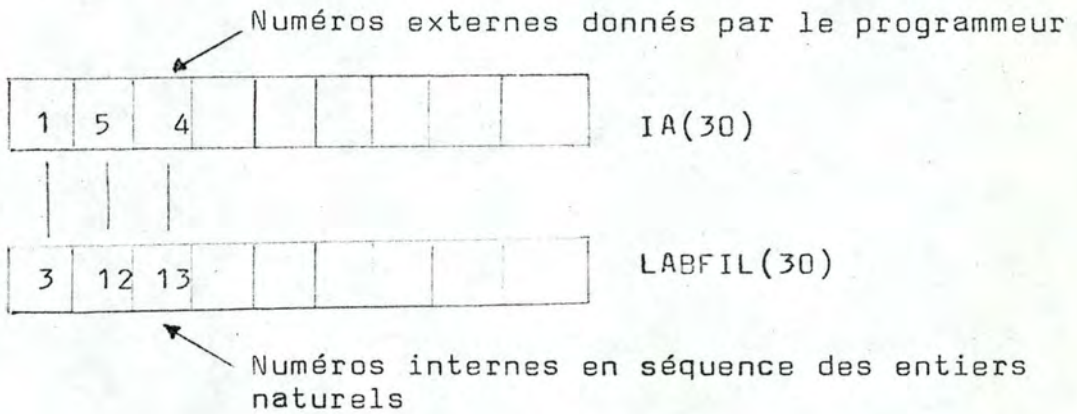




Les vecteurs R2 à R5 sont équivalencés au vecteur R (Voir page 64). Il est à noter que le vecteur NBSUB(211) comporte les numéros externes des critères ; son pendant est le vecteur IPOINT(211), cité à la page 59 .

FIGURE 8

La mémorisation interne des numéros de référence des instructions . Le vecteur IA(30) comprend les numéros externes , et le vecteur LABFIL(30) comprend les numéros internes :



Ainsi , la 3° instruction a comme numéro de référence '1' dans le programme-source SEL ; les 2 premières instructions de ce programme n'ont pas de numéros de référence ,etc .

3.3 / La mémoire interne du système SEL

Par ' mémoire interne ' du système SEL , nous entendons les variables utilisées par le programme Fortran qu'il constitue .

Chaque sous-routine possède ses propres variables de travail ; celles-ci sont accessoires , et il nous paraît peu intéressant de les citer ; pour plus de détails , nous renvoyons aux ordinogrammes (1) et/ou au listing du système SEL (Voir chapitre 4 , pages 68à144) .

Le programme principal pour sa part , définit la zone COMMON , et attribue à certaines variables des valeurs fixes . Il nous paraît intéressant de détailler ces variables :

- le vecteur VIRTUA est destiné à être équivalencé avec le restent de la zone COMMON , à l'exception du vecteur RR(20) , exposé ci-après ; le vecteur VIRTUA sert à sauver la zone COMMON dans le fichier 50 (Voir le paragraphe 3.4 ci-après , page 66)

- le vecteur RR(20) n'est autre que le vecteur de l'utilisateur A(20) (2)

- le vecteur II(100) n'est autre que le vecteur de l'utilisateur K(100)

- les vecteurs K(100) , L(400) , IPOINT(211) , ISUB(780) , NBSUB(211) , IPTRR(90) , IA(30) et LABFIL(30) sont exposés au paragraphe 3.2 , pages 58 à 60

- le vecteur ITREE(210) est destiné à contenir l'image de l'arbre sous la forme d'un graphe ; il sert à l'instruction SENSITIVITY dans les options autres que ALL pour déterminer si le critère subordonné est effectivement un descendant du critère ascendant (3)

- le vecteur IVDIM(5) contient en permanence les dimensions des vecteurs de l'utilisateur , càd 100 , 20 , 120 , 210 et 120 (Voir page 4)

(1) Les ordinogrammes sont en la possession de M. Van Bastelaer .
(2) Les différences entre les noms internes et externes proviennent d'une convention de programmation laissant les noms simples à l'usage du programmeur-système de SEL .
(3) Sur l'instruction SENSITIVITY , voir les pages 43 à 44 bis .

- la variable NBINP contient le nombre maximum de CE pour l'arbre binaire destiné à l'optimisation ; comme le nombre maximum de sommets de cet arbre est actuellement fixé à 31 (Voir pages à 47) , NBINP = 16

- la variable IPROG sert , à chaque retour au programme principal , à déterminer à quelle sous-routine principale brancher ; au sein de cette sous-routine , un traitement particulier peut être paramétré par la variable IENT ; par exemple , IPROG = 10 et IENT = 1 mène à l'exécution de la sous-routine REAL6 , section PRESENT VALUE

- la variable LFIRST contient toujours la valeur de la variable du système K (i) qui pointe sur le vecteur L ; ainsi , si l'on exécute la cinquième instruction , K (5) pointe dans le vecteur L sur le code opératoire de cette instruction (Voir le paragraphe 3.2 , page 58)

- la variable INDEX , ainsi que

- la variable LL ont de multiples usages sans intérêt dans notre exposé

- la variable IENT a été expliquée ci-dessus dans le paragraphe sur la variable IPROG

- la variable NBOPTI contient le nombre de positions possibles pour les vecteurs X , C , E , C1OPT , C2OPT et EREZ destinés à l'optimisation (Voir pages133à138) ; NBOPTI est fixée à 160

- le vecteur IBB(101) sert de buffer pour l'impression des fonctions d'efficacité maximale et de rapport Efficacité/Coût maximal , dans le cadre de la sous-routine auxiliaire DIS100 appelée pzr l'exécution de l'instruction DISPLAY , options OPTIMIZATION 1 et OPTIMIZATION 2 (page 41) .

Enfin , il existe des variables de valeur constante , que le programme principal envoie comme arguments aux sous-routines principales :

- le vecteur INJCL , qui contient les trois premiers caractères du nom de chaque commande , y compris l'option LIST de la commande *SEL , page 11
- le vecteur INSEL qui contient les trois premiers caractères du nom de chaque instruction

- les variables KDIM , LDIM , IXDIM , IEDIM et ICDIM sont équivalencées au vecteur IVDIM(5)

- la variable IDSUBK contient la longueur du vecteur ISUB , càd 780

- la variable INDLAB est destinée à contenir le nombre de numéros de références du programme-source SEL , au cours de l'exécution de l'interpréteur

- la variable NBLAB contient la longueur du vecteur LABFIL , càd 30

- les variables KCR , KLP et KCP sont des paramètres qui contiennent les codes du lecteur de cartes , de l'imprimante et de la perforatrice de cartes ; dans le système SIEMENS 4004/151 , KCR = 5 , KLP = 6 et KCP = 7

- la variable LIST1 est utilisée par la commande *SEL , comme expliqué à la page 11

- la variable NIN est destinée à contenir le nombre des instructions du programme INSEL , y compris l'instruction END ; la variable NIN est limitée en même temps que le programme , par la longueur maximum du programme INSEL (Voir page 54 du 1° tome)

- la variable IAR contient durant l'exécution du programme INSEL le numéro interne de l'instruction courante ; elle est à 0 au début de l'exécution , et est mise à jour par le programme INSEL ; elle constitue le PROGRAM-COUNTER du programme INSEL

- la variable NBFILE contient le nombre de fichiers destinés à sauver le programme INSEL ; ici , elle vaut 10 , par référence aux fichiers 31 à 40

- la variable NBCDS contient le nombre de commandes du langage SEL ; elle vaut 10 , si l'on compte la commande *PUNCH , dont nous avons mis à blanc l'espace dans le vecteur INJCL (cartes 3984 et 3985 du système SEL , page 142)

- la variable NISEL contient le nombre d'inscriptions du langege SEL ; elle vaut 17 , le nombre des instructions existantes actuellement

- la variable IFIL12 est la variable de contrôle du fichier-titre numéro 42 , décrit à la page

- le vecteur ICHAR , qui contient divers caractères

- le vecteur IDIG , qui contient les 10 chiffres

- le vecteur IUSVEC , qui contient les noms des vecteurs de l'utilisateur , càd K , A , X , E et C (Voir page 4)

- le vecteur IAZ qui contient les lettres A et Z

- le vecteur AGGROP , qui contient les opérateurs d'aggrégation

- et les vecteurs R2 , R3 , R4 et R5 , qui contiennent les coefficients d'aggrégation pour N = 2 , 3 , 4 et 5 pour les opérateurs d'aggrégation dans l'ordre du vecteur AGGROP . Les vecteurs R2 à R5 forment le vecteur R , auquel ils sont équivalencés .

Dans ce dernier groupe , les vecteurs sont surtout destinés à l'analyse syntaxique et au calcul des efficacités .

Les variables et vecteurs décrits dans ce chapitre sont programmés en extension dans le système SEL , cartes 3971 à 4007 (pages 141 à 143) .

3.4 / La mémoire externe du système SEL

Par ' mémoire externe ' du système SEL , nous entendons les supports d'informations qui subsistent après l'achèvement de la tâche dans l'ordinateur .

Ces supports sont :

- les cartes perforées en entrée
- les listings
- les fichiers du système .

A Namur , le praticien utilisant le terminal SIEMENS T200 , ce dernier type de support est d'un intérêt particulier ; ceci d'autant plus que le système SEL fait explicitement usage de 16 fichiers du système .

Ces fichiers sont déclarés en Fortran sous les numéros 31 à 34 , 41 à 43 , 50 , 51 et 93 ; ils existent dans le système SIEMENS 4004/151 (1) , sous les noms VBA.ARN.31 , VBA.ARN32 à VBA.ARN.93 .

La structure de ces fichiers est la suivante :

- les fichiers 31 à 40 comportent un maximum de 1600 articles ; chaque article est écrit et donc relu dans les formats I10 ou A10 ;
  leur but est le sauvetage du programme INSEL sur la demande du programmeur ; pour le détail du sauvetage et les manipulations sur ces fichiers , voir les commandes *STORE , *READ et *DELETE , pages 12 à 14 , et 17
- le fichier 41 comporte un maximum de 792 articles ; chaque article est écrit et donc relu dans les formats F10.3 ou I10 ; son but est le sauvetage des résultats de l'analyse de sensibilité sauvetage exécuté automatiquement ; pour chacun des CNE , on sauve 8 éléments : a , ar , b , br , d+ , d- , p (format F10.3) et index (format I10) où index est le numéro interne du critère subordonné ; l'écriture commence à l'article de numéro

(1) Par économie , tous les fichiers du système SEL sont détruits à l'achèvement de la tâche ; voir la procédure VBA.ARN.C.SEL , page 208 .

8 . numéro interne du critère ascendant - 7 ; comme il existe au plus 99 CNE , le nombre maximum d'articles de ce fichier est donc 99 . 8 = 792

- le fichier 42 comporte un maximum de 1037 articles ; chaque article est écrit et donc relu dans le format A10 ;
son but est de conserver le texte des suites de caractères apparues dans les inscriptions TITLE (pages 37 et 38) ; les suites sont écrites les unes à la suite des autres , et le code INSEL de l'instruction TITLE conserve les numéros des première et dernière positions de sa suite dans le fichier ; enfin , le fait d'écrire ce fichier en format A10 permet de le relire à l'Editeur de Fichiers (1)

- le fichier 43 comporte un maximum de 3980 articles ; chaque article est écrit en format A10 et comporte deux caractères ;
son but est de conserver les libellés des critères ; le libellé d'un critère de numéro interne i est écrit à partir de la position 20 . i - 19 ; le fait qu'on utilise que 20 positions pour un libellé de 40 caractères provient de ce que l'instruction INPUT lit ce libellé selon le format 20A2 (carte n° 2844 du système SEL , page 121) ; comme le nombre total de critères est 199 , la taille maximum de ce fichier est 199 . 20 = 3980 articles

- les fichiers 50 et 51 sont simplement destinés à sauver la mémoire de travail de l'interpréteur , lors des nombreux calculs des instructions OUTPUT , DISPLAY , OPTIMIZE et ALLOCATE ; la taille maximum du fichier 50 est de 1230 articles , ce qui est déterminé par le fait qu'il doit conserver le vecteur VIRTUA(1230) ; la taille maximum du fichier 51 est de 780 articles , ce qui est déterminé par le fait qu'il doit conserver le vecteur ISUB(780)

- le fichier 93 comporte un maximum de 14880 articles ; chaque article est écrit en format F10.3 ;
son but est de conserver les CEM des critères : pour les CE les vecteurs X (valeurs d'entrée possibles) , C (Coûts

(1) La taille maximale de 1037 caractères dans l'ensemble des suites provient de ce que les fichiers 31 à 40 doivent contenir l'ensemble des informations décrites aux pages 13 et 14 ; or , leur taille maximale est de 1600 articles ; la longueur maximum pour les autres éléments sauvés est : 1 (NIN) + 100 (K) + 400 (L) + 1 (IND) + 1 (INDLAB) + 60 (LABFIL et IA) = 573 caractères ; la place maximum pour les titres est donc 1600 - 573 = 1037 caractères , ce qui est largement exédentaire pour les besoins courants .

de ces valeurs) , et E (leurs efficacités) ; pour les CNE , les vecteurs C1OPT (CoOts du premier CIS dans le couple optimal) , C2OPT (idem pour le deuxième CIS) et EREZ (Efficacités des couples optimaux formant la CEM) ;
chacun de ces vecteurs comprenant 160 variables , chaque critère possède 3 . 160 = 480 positions ; comme il existe au plus 31 critères dans l'arbre d'optimisation , la taille maximum du fichier est donc bien de 31 . 480 = 14880 articles ; étant donné un critère de numéro interne i , ses 3 vecteurs sont écrits respectivement à partir des positions i . 480 - 479 , i . 480 - 319 et i . 480 - 159 .

C H A P I T R E 4

LISTING DU SYSTEME SEL

Ce chapitre présente le listing du système SEL .

Celui-ci est un fichier catalogué sous le nom VBA.ARN.SEL dans le système SIEMENS 4.004/151 de la RTT à Namur ; il est conservé sur la bande magnétique FUN96A 000153 (1) .

L'ordre des sous-routines qui forme le système SEL est le suivant : d'abord les sous-routines auxiliaires , puis les principales , et enfin le programme principal SELMP . Au sein des deux groupes , l'ordre des sous-routines est celui donné dans le 1° tome , page 52 .

Le format quarto utilisé ici nous oblige à présenter les cartes Fortran du système SEL dans les colonnes 73 à 80 .

Celles-ci sont utilisées pour la numérotation et l'identification des cartes

- les colonnes 73 à 77 contiennent un nombre de 5 chiffres , qui va de 00001 pour la première carte à 04084 pour la dernière , et
- les colonnes 78 à 80 contiennent les caractères ' SEL ' .

Un exemplaire du listing complet et un exemplaire perforé du système SEL sont déposés chez M. Ph. Van Bastelaer , Directeur de notre mémoire (Institut d'Informatique) .

(1) Tous les fichiers de notre travail appartiennent à l'user-id. FUN96A , account-nb. MEMO . Ils sont conservés sur la même bande magnétique FUN96A 000153 .

```
00010000 ------SUBROUTINE DFL(L,LF,LL,INDEX,KOD)------------------------------
00020000       DIMENSION L(1)
00030000       DATA NA,NZ/'A','Z'/
00040000C
00050000C DETECTING THE POSITION OF THE FIRST LETTER IN THE GIVEN VECTOR'S ZONE
00060000C BY SCANNING FROM LEFT TO RIGHT
00070000C L =  GIVEN VECTOR
00080000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
00090000C INDEX = POSITION OF THE FOUND LETTER
00100000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
00110000C AND -1 IF INSUCCESS OR ERROR : LF GREATER THAN LL )
00120000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
00130000C
00140000       IF(LF-LL) 10,10,40
00150000    10 DO 30 INDEX=LF,LL
00160000       IF(FLOAT(L(INDEX))-NA) 30,50,20
00170000    20 IF(FLOAT(L(INDEX))-NZ) 50,50,30
00180000    30 CONTINUE
00190000    40 KOD=-1
00200000       RETURN
00210000    50 KOD=0
00220000       RETURN
00230000       END
00240000C
00250000------SUBROUTINE DFD(L,LF,LL,INDEX,KOD)------------------------------
00260000       DIMENSION L(1)
00270000       DATA NAUGHT,N9/'0','9'/
00280000C
00290000C DETECTING THE POSITION OF THE FIRST DIGIT IN THE GIVEN VECTOR'S ZONE
00300000C BY SCANNING FROM LEFT TO RIGHT
00310000C L =  GIVEN VECTOR
00320000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
00330000C INDEX =  POSITION OF THE FOUND DIGIT
00340000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
00350000C AND -1 IF INSUCCESS OR ERROR : LF GREATER THAN LL )
00360000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
00370000C
00380000       IF(LF-LL) 10,10,40
00390000    10 DO 30 INDEX=LF,LL
00400000       IF(FLOAT(L(INDEX))-NAUGHT) 30,50,20
00410000    20 IF(FLOAT(L(INDEX))-N9) 50,50,30
00420000    30 CONTINUE
00430000    40 KOD=-1
00440000       RETURN
00450000    50 KOD=0
00460000       RETURN
00470000       END
00480000C
00490000------SUBROUTINE DFGC(L,LF,LL,INDEX,KOD,ICHAR)-----------------------
00500000       DIMENSION L(1)
00510000C
00520000C DETECTING THE POSITION OF THE FIRST GIVEN CHARACTER
00530000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FROM LEFT TO RIGHT
00540000C L =  GIVEN VECTOR
```

```
USER I.D.- FUN96A
ACCT. NO.- MEMO
FILENAME - VBA.ARN.SEL
MAILING ADDR.- FACULTE  N.D.
DATE  -  09/09/77
```

```
00550000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
00560000C INDEX = POSITION OF THE FOUND CHARACTER
00570000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
00580000C AND -1 IF INSUCCESS OR ERROR : LF GREATER THAN LL )
00590000C ICHAR = GIVEN CHARACTER
00600000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
00610000C
00620000       IF(LF-LL) 10,10,30
00630000    10 DO 20 INDEX=LF,LL
00640000       IF(FLOAT(L(INDEX)) .EQ. ICHAR) GO TO 40
00650000    20 CONTINUE
00660000    30 KOD=-1
00670000       RETURN
00680000    40 KOD=0
00690000       RETURN
00700000       END
00710000C
00720000------SUBROUTINE DFGCD(L,LF,LL,INDEX,KOD,ICHAR)----------------------
00730000       DIMENSION L(1)
00740000       DATA NAUGHT,N9/'0','9'/
00750000C
00760000C DETECTING THE POSITION OF THE FIRST GIVEN CHARACTER
00770000C OR OF THE FIRST DIGIT ( WHICH OTHER APPEARS THE FIRST )
00780000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FROM LEFT TO RIGHT
00790000C L =  GIVEN VECTOR
00800000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
00810000C INDEX = POSITION OF THE FOUND CHARACTER
00820000C KOD = OUTPUT CODE ( VALUE 1 IF THE CHARACTER HAS BEEN FOUND ,
00830000C 2 IF A DIGIT HAS BEEN FOUND ,
00840000C AND -1 IF INSUCCESS OR ERROR : LF GREATER THAN LL )
00850000C ICHAR = GIVEN CHARACTER
00860000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
00870000C
00880000       IF(LF-LL) 10,10,40
00890000    10 DO 30 INDEX=LF,LL
00900000       IF(L(INDEX) .EQ. ICHAR) GO TO 50
00910000       IF(L(INDEX)-NAUGHT) 30,60,20
00920000    20 IF(L(INDEX)-N9) 60,60,30
00930000    30 CONTINUE
00940000    40 KOD=-1
00950000       RETURN
00960000    50 KOD=1
00970000       RETURN
00980000    60 KOD=2
00990000       RETURN
01000000       END
01010000C
01020000------SUBROUTINE D2GC(L,LF,LL,INDEX1,INDEX2,KOD,ICHAR1,ICHAR2)--------
01030000       DIMENSION L(1)
01040000C
01050000C DETECTING THE POSITION OF THE TWO GIVEN CHARACTERS
01060000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FROM LEFT TO RIGHT,
01070000C BEGINNING BY SEARCHING THAT CHARACTER APPEARING THE FIRST
01080000C IN THE LIST OF ARGUMENTS OF THE SUBROUTINE (ICHAR1)
```

```
01090000C L =  GIVEN VECTOR
01100000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
01110000C INDEX1 , INDEX2 = POSITIONS OF THE TWO FOUND CHARACTERS
01120000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
01130000C AND -1 IF INSUCCESS OR IF ERROR : LF GREATER THAN LL )
01140000C ICHAR1 , ICHAR2 = GIVEN CHARACTERS
01150000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
01160000C
01170000       IF(LF-LL) 10,10,30
01180000    10 DO 20 I=LF,LL
01190000       IF(FLOAT(L(I)) .EQ. ICHAR1) GO TO 40
01200000    20 CONTINUE
01210000    30 KOD=-1
01220000       RETURN
01230000    40 IF(I-LL) 50,30,50
01240000    50 INDEX1=I
01250000       J=I+1
01260000       DO 60 I=J,LL
01270000       IF(FLOAT(L(I)) .EQ. ICHAR2) GO TO 70
01280000    60 CONTINUE
01290000       GO TO 30
01300000    70 INDEX2=I
01310000       KOD=0
01320000       RETURN
01330000       END
01340000C
01350000 ------SUBROUTINE DFA2GC(L,LF,LL,INDEX,KOD,ICHAR1,ICHAR2) ----------
01360000       DIMENSION L(1)
013700000C
01380000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FROM LEFT TO RIGHT
01390000C L = GIVEN VECTOR
01400000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
01410000C INDEX = POSITION OF THE FIRST FOUND OF THE TWO CHARACTERS
01420000C KOD = OUTPUT CODE ( VALUE 1 IF THE FIRST CHARACTER IS FOUND ,
01430000C VALUE 2 IF THE SECOND ONE IS FOUND ,
01440000C AND -1 IF INSUCCESS OR IF ERROR : LF GREATER THAN LL )
01450000C ICHAR1 , ICHAR2 = GIVEN CHARACTERS
01460000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
01470000C
01480000       IF(LF-LL) 10,10,30
01490000    10 DO 20 INDEX=LF,LL
01500000       IF(FLOAT(L(INDEX)) .EQ. ICHAR1) GO TO 40
01510000       IF(FLOAT(L(INDEX)) .EQ. ICHAR2) GO TO 50
01520000    20 CONTINUE
01530000    30 KOD=-1
01540000       RETURN
01550000    40 KOD=1
01560000       RETURN
01570000    50 KOD=2
01580000       RETURN
01590000       END
01600000C
01610000 ------SUBROUTINE DFNBC(L,LF,LL,INDEX,KOD) ----------
01620000       DIMENSION L(1)
```

```
01630000       DATA IBL/' '/
01640000C
01650000C DETECTING THE POSITION OF THE FIRST NON-BLANK CHARACTER
01660000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FORM LEFT TO RIGHT
01670000C L = GIVEN VECTOR
01680000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
01690000C INDEX = POSITION OF THE FIRST NON-BLANK CHARACTER
01700000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
01710000C 1 IF INSUCCESS ( BLANK ZONE ) ,
01720000C AND -1 IF ERROR : LF GREATER THAN LL )
01730000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
01740000C
01750000       IF(LF-LL) 10,10,40
01760000    10 DO 20 INDEX=LF,LL
01770000       IF(FLOAT(L(INDEX)) .NE. IBL) GO TO 30
01780000    20 CONTINUE
01790000       KOD=1
01800000       RETURN
01810000    30 KOD=0
01820000       RETURN
01830000    40 KOD=-1
01840000       RETURN
01850000       END
01860000C
01870000 ------SUBROUTINE DBR(L,LF,LL,LF1,LL1) ----------------------------------
01880000       DIMENSION L(1)
01890000       DATA IBL/' '/
01900000C
01910000C DELIMITING THE BLANK REGIONS ON THE LEFT AND ON THE RIGHT
01920000C INSIDE THE GIVEN VECTOR'S ZONE
01930000C L = GIVEN VECTOR
01940000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
01950000C LF1 , LL1 = LEFT AND RIGHT LIMITS OF THE NON-BLANK REGION
01960000C ( THE RESULTING PAIR LF1=0 AND LL1=-1 INDICATES
01970000C EITHER WHITE VECTOR'S ZONE , OR ERROR : LF GREATER THAN LL )
01980000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
01990000C
02000000       IF(LF-LL) 10,10,30
02010000C OMITTING THE BLANK REGION ON THE LEFT
02020000    10 DO 20 I=LF,LL
02030000       IF(FLOAT(L(I)) .NE. IBL) GO TO 40
02040000    20 CONTINUE
02050000    30 LF1=0
02060000       LL1=-1
02070000       RETURN
02080000    40 LF1=I
02090000C OMITTING THE BLANK REGION ON THE RIGHT
02100000       J=LL+1
02110000       DO 50 I=LF1,LL
02120000       J=J-1
02130000       IF(FLOAT(L(J)) .NE. IBL) GO TO 60
02140000    50 CONTINUE
02150000    60 LL1=J
02160000       RETURN
```

```
02170000       END
02180000C
02190000 ------SUBROUTINE DFW(L,LF,LL,LF1,LL1,KOD)----------------------------
02200000       DIMENSION L(1)
02210000       DATA IBL/' '/
02220000C
02230000C DETECTING THE FIRST STRING OF NON-BLANK CHARACTERS (WORD)
02240000C DELIMITED BY BLANKS
02250000C IN THE GIVEN VECTOR'S ZONE , BY SCANNING FROM LEFT TO RIGHT
02260000C L =  GIVEN VECTOR
02270000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
02280000C LF1 , LL1 = FIRST AND LAST POSITIONS OF THE FOUND WORD
02290000C KOD = OUTPUT CODE ( VALUE 1 IF SUCCESS , 0 IF THERE IS NO BLANK
02300000C CHARACTER ON THE RIGHT OF THE FIRST FOUND CHARACTER ( LL1=LL ) ,
02310000C AND -1 IF BLANK ZONE OR ERROR : LF GREATER THAN LL )
02320000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
02330000C
02340000       IF(LF-LL) 10,10,30
02350000    10 DO 20 I=LF,LL
02360000       IF(FLOAT(L(I)) .NE. IBL) GO TO 40
02370000    20 CONTINUE
02380000    30 KOD=-1
02390000       RETURN
02400000    40 LF1=I
02410000       J=LF1+1
02420000       DO 50 I=J,LL
02430000       IF(FLOAT(L(I)) .EQ. IBL) GO TO 60
02440000    50 CONTINUE
02450000       LL1=LL
02460000       KOD=0
02470000       RETURN
02480000    60 LL1=I-1
02490000       KOD=1
02500000       RETURN
02510000       END
02520000C
02530000------SUBROUTINE DIN(IHEAD,NP,NK,L,LF,INSTR)----------------------------
02540000       DIMENSION IHEAD(1),L(1)
02550000C
02560000C DETECTING WHICH INSTRUCTION IS CONTAINED ON THE CARD
02570000C BY RECOGNIZING THE THREE FIRST CHARACTERS OF THIS INSTRUCTION
02580000C IHEAD = VECTOR CONTAINING THE THREE-CHARACTERS-LONG NAME
02590000C OF EVERY INSTRUCTION
02600000C NP , NK = NUMBERS OF INSTRUCTION IN THE INTERVAL OF WHICH
02610000C THE SEARCH IS MADE
02620000C L = GIVEN VECTOR , THE POSITIONS OF WHICH L(LF),L(LF+1) AND L(LF+2)
02630000C CONTAIN THE NAME OF THE INSTRUCTION TO BE RECOGNIZED
02640000C INSTR = NUMBER OF THE FOUND INSTRUCTION NAME
02650000C ( VALUE INSTR = 0  INDICATES INSUCCESS IN THE SEARCH )
02660000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
02670000C
02680000       DO 30 INSTR=NP,NK
02690000       I=1+3*(INSTR-1)
02700000       IF(IHEAD(I) .NE. L(LF)) GO TO 30
```

```
02710000    10 IF(IHEAD(I+1) .NE. L(LF+1)) GO TO 30
02720000    20 IF(IHEAD(I+2) .EQ. L(LF+2)) GO TO 40
02730000    30 CONTINUE
02740000       INSTR=0
02750000    40 RETURN
02760000       END
02770000C
02780000------SUBROUTINE BINTRA(L,N,INPUT,INDEX)------------------------------
02790000       DIMENSION L(1)
02800000C
02810000C BINARY TRACING OF THE GIVEN ELEMENT IN THE GIVEN SORTED VECTOR
02820000C L = GIVEN VECTOR ( SORTED ON ASCENDING VALUES )
02830000C N = DIMENSION OF VECTOR L
02840000C INPUT = GIVEN ELEMENT
02850000C INDEX = FOUND POSITION OF THE GIVEN ELEMENT IN THE GIVEN VECTOR
02860000C ( VALUE 0 INDICATES INSUCCESS IN THE TRACING ,
02870000C OR ERROR : N LESS OR EQUAL TO ZERO )
02880000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
02890000C
02900000       IL=1
02910000       IR=N
02920000    10 IF(IL-IR)  40,20,70
02930000    20 IF(INPUT-L(IR)) 70,30,70
02940000    30 INDEX=IR
02950000       RETURN
02960000    40 INDEX=(IR+IL)/2
02970000       IF(INPUT-L(INDEX)) 50,80,60
02980000    50 IR=INDEX-1
02990000       GO TO 10
03000000    60 IL=INDEX+1
03010000       GO TO 10
03020000    70 INDEX=0
03030000    80 RETURN
03040000       END
03050000C
03060000------SUBROUTINE SORTRR(X,Y,N)-----------------------------------------
03070000       DIMENSION X(1),Y(1)
03080000C
03090000C SORTING THE FIRST OF THE TWO GIVEN REAL VECTORS ON INCREASING VALUES
03100000C WITH THE SAME REARRANGEMENT OF THE SECOND VECTOR
03110000C X , Y = GIVEN VECTORS
03120000C N = COMMON DIMENSION OF THE TWO VECTORS
03130000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
03140000C
03150000       IDIM=N
03160000    10 INDIC=0
03170000       IDIM=IDIM-1
03180000       DO 30 I=1,IDIM
03190000       IF(X(I+1)-X(I)) 20,30,30
03200000    20 BUFX=X(I)
03210000       BUFY=Y(I)
03220000       X(I)=X(I+1)
03230000       X(I+1)=BUFX
03240000       Y(I)=Y(I+1)
```

```
03250000       Y(I+1)=BUFY
03260000       INDIC=1
03270000    30 CONTINUE
03280000       IF(INDIC) 10,40,10
03290000    40 RETURN
03300000       END
03310000C
03320000 ------SUBROUTINE SORTII(K,L,N)----------------------------------------
03330000       DIMENSION K(1),L(1)
03340000C
03350000C SORTING OF THE FIRST OF THE TWO GIVEN INTEGER VECTORS ON INCR. VALUES
03360000C WITH THE SAME REARRANGEMENT OF THE SECOND VECTOR
03370000C K , L = GIVEN VECTORS
03380000C N = COMMON DIMENSION OF THE TWO VECTORS
03390000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
03400000C
03410000       IDIM=N
03420000    10 INDIC=0
03430000       IDIM=IDIM-1
03440000       DO 30 I=1,IDIM
03450000       IF(K(I+1)-K(I)) 20,30,30
03460000    20 IBUFK=K(I)
03470000       IBUFL=L(I)
03480000       K(I)=K(I+1)
03490000       K(I+1)=IBUFK
03500000       L(I)=L(I+1)
03510000       L(I+1)=IBUFL
03520000       INDIC=1
03530000    30 CONTINUE
03540000       IF(INDIC) 10,40,10
03550000    40 RETURN
03560000       END
03570000C
03580000 ------SUBROUTINE SORTRI(X,L,N)----------------------------------------
03590000       DIMENSION X(1),L(1)
03600000C
03610000C SORTING THE FIRST OF TWO GIVEN VECTORS ON INCREASING VALUES
03620000C WITH THE SAME REARRANGEMENT OF THE SECOND VECTOR
03630000C X , L = GIVEN VECTORS , THE FIRST REAL , THE SECOND INTEGER
03640000C N = COMMON DIMENSION OF THE TWO VECTORS
03650000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
03660000C
03670000       IDIM=N
03680000    10 INDIC=0
03690000       IDIM=IDIM-1
03700000       DO 30 I=1,IDIM
03710000       IF(X(I+1)-X(I)) 20,30,30
03720000    20 BUFX=X(I)
03730000       IBUFL=L(I)
03740000       X(I)=X(I+1)
03750000       X(I+1)=BUFX
03760000       L(I)=L(I+1)
03770000       L(I+1)=IBUFL
03780000       INDIC=1
```

```
03790000    30 CONTINUE
03800000       IF(INDIC) 10,40,10
03810000    40 RETURN
03820000       END
03830000C
03840000 ------SUBROUTINE MINMAX(X,LF,LL,XMIN,XMAX,IAPS,NZERO)----------
03850000       DIMENSION X(1)
03860000C
03870000C DETECTING THE MINIMAL AND MAXIMAL VALUES IN THE GIVEN VECTOR
03880000C X = GIVEN VECTOR
03890000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
03900000C XMIN , XMAX = RESULTING MINIMAL AND MAXIMAL VALUES
03910000C IAPS = INPUT CODE ( IF DIFFERENT FROM ZERO , THE SEARCH IS MADE
03920000C ON THE ABSOLUTE VALUES )
03930000C NZERO = INPUT CODE ( IF EQUAL TO ZERO , THE SEARCH
03940000C DOESN'T CONSIDER THE ZEROS )
03950000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
03960000C
03970000       I=LF
03980000       IF(NZERO) 30,10,30
03990000    10 DO 20 I=LF,LL
04000000       IF(X(I)) 30,20,30
04010000    20 CONTINUE
04020000       XMIN=0.
04030000       XMAX=0.
04040000       RETURN
04050000    30 XMIN=X(I)
04060000       XMAX=XMIN
04070000       IF(XMAX) 40,60,60
04080000    40 IF(IAPS) 50,60,50
04090000    50 XMIN=-XMIN
04100000       XMAX=-XMAX
04110000    60 IF(I-LL) 70,160,160
04120000    70 I=I+1
04130000       DO 150 J=I,LL
04140000       XX=X(J)
04150000       IF(XX) 90,80,110
04160000    80 IF(NZERO) 110,150,110
04170000    90 IF(IAPS) 100,110,100
04180000   100 XX=-XX
04190000   110 IF(XX-XMAX) 130,150,120
04200000   120 XMAX=XX
04210000       GO TO 150
04220000   130 IF(XX-XMIN) 140,150,150
04230000   140 XMIN=XX
04240000   150 CONTINUE
04250000   160 RETURN
04260000       END
04270000C
04280000 ------SUBROUTINE  CUII(L,LF,LL,IDIG,INT)----------
04290000       DIMENSION L(1),IDIG(1)
04300000C
04310000C CONVERTING AN UNSIGNED INTEGER NUMBER FROM FORMAT A (FORTRAN)
04320000C TO AN INTEGER CONSTANT
```

```
04330000C L = VECTOR CONTAINING THE INTEGER NUMBER IN FORMAT A
04340000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
04350000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
04360000C INT = INTEGER VARIABLE DESIGNED TO CONTAIN THE CONVERTED CONSTANT
04370000C ( VALUE INT = -1 INDICATES SYNTAX ERROR IN THE INPUT DATA ,
04380000C OR ERROR : LF GRATER THAN LL )
04390000C NOTE : THERE MAY BE NO BLANKS IN THE ZONE FROM LF TO LL
04400000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
04410000C
04420000       IF(LF-LL) 10,10,40
04430000    10 INT=0
04440000       DO 30 I=LF,LL
04450000       DO 20 J=1,10
04460000       IF(L(I)-IDIG(J)) 20,30,20
04470000    20 CONTINUE
04480000       GO TO 40
04490000    30 INT=10*INT+J-1
04500000       RETURN
04510000    40 INT=-1
04520000       RETURN
04530000       END
04540000C
04550000 ------SUBROUTINE  CUNR(L,LF,LL,IDIG,RAC,KOD)------------------------------
04560000       DIMENSION L(1),IDIG(1)
04570000       DATA IPOINT/'.'/
04580000C
04590000C CONVERTING AN UNSIGNED NUMBER (INTEGER OR REAL) FROM FORMAT A
04600000C (FORTRAN) TO A REAL CONSTANT
04610000C L = VECTOR CONTAINING THE NUMBER IN FORMAT A
04620000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
04630000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
04640000C RAC = REAL VARIABLE DESIGNED TO CONTAIN THE CONVERTED CONSTANT
04650000C KOD = OUTPUT CODE ( VALUE 0 IF OK , AND -1 IF SYNTAX ERROR
04660000C IN THE INPUT , OR ERROR : LF GREATER THAN LL )
04670000C NOTE : THERE MAY BE NO BLANKS IN THE ZONE FROM LF TO LL
04680000C AND THE DECIMAL POINT MAY NOT BE PLACED AT THE END OF THE NUMBER
04690000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
04700000C
04710000       RAC=0
04720000       IEXP=0
04730000       DO 80 I=LF,LL
04740000       IF(IEXP) 10,10,50
04750000    10 DO 20 J=1,10
04760000       IF(L(I)-IDIG(J)) 20,40,20
04770000    20 CONTINUE
04780000       IEXP=1
04790000       IF(L(I)-IPOINT) 30,80,30
04800000    30 KOD=-1
04810000       RETURN
04820000    40 RAC=10.*RAC+J-1.
04830000       GO TO 80
04840000    50 DO 60 J=1,10
04850000       IF(L(I)-IDIG(J)) 60,70,60
04860000    60 CONTINUE
```

```
04870000       GO TO 30
04880000    70 RAC=RAC+(J-1)*0.1**IEXP
04890000       IEXP=IEXP+1
04900000    80 CONTINUE
04910000       KOD=0
04920000       RETURN
04930000       END
04940000C
04950000 ------SUBROUTINE CFNN(L,LF,LL,KOD,IC,INT,RAC,NEXT,IDIG)----------
04960000       DIMENSION L(1),IDIG(1)
04970000       DATA MINUS,IPLUS/'-','+'/
04980000C
04990000C CONVERTING THE FIRST (EVENTUALLY SIGNED) NUMBER IN THE GIVEN VECTOR'S
05000000C ZONE , FROM FORMAT A TO AN INTEGER OR REAL CONSTANT ,
05010000C DEPENDING ON THE INPUT CODE IC (SEE BELOW) ,
05020000C BY SCANNING FROM LEFT TO RIGHT
05030000C L = VECTOR CONTAINING THE NUMBER IN FORMAT A
05040000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
05050000C KOD = OUTPUT CODE ( VALUE 1 IF BLANK ZONE , 0 IF OK ,
05060000C AND VALUE -1 IF SYNTAX ERROR OR LF GREATER THAN LL )
05070000C IC = INPUT CODE ( VALUE 1 FOR INTEGER CONVERSION ,
05080000C AND VALUE 2 FOR REAL CONVERSION )
05090000C INT , RAC = INTEGER AND REAL VARIABLES DESIGNED TO CONTAIN
05100000C THE CONVERTED CONSTANT
05110000C NEXT = VECTOR'S POSITION , FROM WHICH ON
05120000C THE NEXT CONSTANT MAY BE SEARCHED
05130000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
05140000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
05150000C
05160000       IF(LF-LL) 10,10,140
05170000    10 IMINUS=0
05180000       CALL DBR(L,LF,LL,I,KOD)
05190000       IF(KOD) 20,20,30
05200000    20 KOD=1
05210000       NEXT=LL+1
05220000       RETURN
05230000    30 IF(FLOAT(L(I)) .NE. MINUS) GO TO 40
05240000       IMINUS=1
05250000       GO TO 50
05260000    40 IF(FLOAT(L(I)) .EQ. IPLUS) GO TO 50
05270000       I=I-1
05280000    50 CALL DFW(L,I+1,LL,LF1,LL1,KOD)
05290000       IF(KOD) 130,60,60
05300000    60 NEXT=LL1+2
05310000       GO TO (70,90),IC
05320000    70 CALL CUII(L,LF1,LL1,IDIG,INT)
05330000       IF(INT) 140, 80, 80
05340000    80 KOD=0
05350000       RAC=INT
05360000       GO TO 110
05370000    90 CALL CUNR(L,LF1,LL1,IDIG,RAC,KOD)
05380000       IF(KOD) 130,100,140
05390000   100 INT=RAC
05400000   110 IF(IMINUS) 130,130,120
```

```
05410000   120 INT=-INT
05420000       RAC=-RAC
05430000   130 RETURN
05440000   140 KOD=-1
05450000       RETURN
05460000       END
05470000C
05480000 ------SUBROUTINE CTIR(I,R,IC,IFIRST,ILAST,KOD,IDIG) ----------
05490000       DIMENSION I(1),R(1),IDIG(1)
05500000       DIMENSION IB(80)
05510000       COMMON VIRTUA(1230),RR(20)
05520000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
05530000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
05540000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
05550000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
05560000       COMMON LFIRST,INDEX,LL,IENT
05570000       COMMON NBOPTI
05580000       COMMON IBB(101)
05590000       EQUIVALENCE (IBB(21),IB(1))
05600000C
05610000C CONVERTING TOTALLY THE GIVEN VECTOR FROM FORMAT A TO CONSTANTS
05620000C I , R = RESULTING VECTORS , THE FIRST INTEGER , THE SECOND REAL
05630000C IC = INPUT CODE ( VALUE 1 COMMANDS WORKING ON THE INTEGER VECTOR ,
05640000C AND VALUE 2 COMMANDS WORKING ON THE REAL VECTOR )
05650000C IFIRST , ILAST = INDEX OF THE FIRST AND LAST POSITIONS
05660000C OF I OR R TO BE FILLED
05670000C KOD = OUTPUT CODE ( VALUE 0 IF SUCCESS IN THE SEARCH ,
05680000C AND -1 SYNTAX ERROR IN THE INPUT )
05690000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
05700000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
05710000C
05720000       IND=IFIRST
05730000    10 NEXT=1
05740000       READ(KCR,20) IB
05750000    20 FORMAT(80A1)
05760000    30 CALL CFNN(IB,NEXT,80,KOD,IC,INT,RAC,NEXT,IDIG)
05770000       IF(KOD) 40,50,10
05780000    40 RETURN
05790000    50 GO TO (60,70),IC
05800000    60 I(IND)=INT
05810000       GO TO 80
05820000    70 R(IND)=RAC
05830000    80 IF(IND-ILAST) 90,40,40
05840000    90 IND=IND+1
05850000       IF(NEXT-80) 30,30,10
05860000       END
05870000C
05880000 ------SUBROUTINE INX(ICODE,IVAR,VAR) ----------
05890000       DIMENSION X(120),E(240),C(120)
05900000       COMMON VIRTUA(1230),R(20)
05910000       COMMON I(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
05920000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
05930000       EQUIVALENCE (VIRTUA(781),X(1)),(VIRTUA(901),E(1)),
05940000      *(VIRTUA(1111),C(1))
```

```
05950000C
05960000C INSCRIBING AN INTEGER OR REAL CONSTANT IN THE VECTOR'S ELEMENT
05970000C CORRESPONDING TO THE GIVEN CODE 'ICODE' ( SEE SEL ORGANIZATION )
05980000C ICODE = GIVEN CODE POINTING TO A VECTOR'S ELEMENT
05990000C IVAR , VAR = INTEGER AND REAL VARIABLES
06000000C NOTE : THE COMMON ZONE (FORTRAN) MUST CONTAIN THE VECTORS K,A,X,E,C.
06010000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
06020000C
06030000       IND=ICODE
06040000       IF(IND) 10,100,20
06050000    10 IND=-IND
06060000    20 IVEC=IND/1000
06070000       IND=IND-1000*IVEC
06080000       IF(ICODE) 30,30,40
06090000    30 IND=I(IND)
06100000    40 GO TO (50,60,70,80,90),IVEC
06110000    50 I(IND)=IVAR
06120000       RETURN
06130000    60 R(IND)=VAR
06140000       RETURN
06150000    70 X(IND)=VAR
06160000       RETURN
06170000    80 E(IND)=VAR
06180000       RETURN
06190000    90 C(IND)=VAR
06200000   100 RETURN
06210000       END
06220000C
06230000 ------SUBROUTINE TXV(ICODE,INT,RAC,NOC) ----------------------------------
06240000       DIMENSION X(120),E(240),C(120)
06250000       COMMON VIRTUA(1230), R(20)
06260000       COMMON I(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
06270000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
06280000       EQUIVALENCE (VIRTUA(781),X(1)),(VIRTUA(901),E(1)),
06290000      *(VIRTUA(1111),C(1))
06300000C
06310000C TRANSFERRING THE CONTENTS OF THE VECTOR'S ELEMENT
06320000C POINTED TO BY 'ICODE' IN THE TWO VARIABLES INT AND RAC
06330000C ICODE = GIVEN CODE POINTING TO A CELL
06340000C INT , RAC = INTEGER AND REAL VARIABLES
06350000C NOC = OUTPUT CODE ( VALUE 1 IF WORK ON THE INTEGER VARIABLE INT ,
06360000C AND VALUE 2 IF WORK ON THE REAL VARIABLE RAC )
06370000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
06380000C
06390000       IND=ICODE
06400000       IF(IND) 10,110,20
06410000    10 IND=-IND
06420000    20 IVEC=IND/1000
06430000       IND=IND-1000*IVEC
06440000       IF(ICODE) 30,30,40
06450000    30 IND=I(IND)
06460000    40 GO TO (50,60,70,80,90),IVEC
06470000    50 INT=I(IND)
06480000       RAC=INT
```

```
06490000       NOC=1
06500000       RETURN
06510000    60 RAC=R(IND)
06520000       GO TO 100
06530000    70 RAC=X(IND)
06540000       GO TO 100
06550000    80 RAC=E(IND)
06560000       GO TO 100
06570000    90 RAC=C(IND)
06580000   100 NOC=2
06590000       INT=RAC
06600000   110 RETURN
06610000       END
06620000C
06630000 ------SUBROUTINE TOV(L1,L2,INT,RAC,NOC) ----------------------------------
06640000C
06650000C TRANSFERRING THE GIVEN DUBBLE-WORD OPERAND ( SEE SEL ORGANIZATION )
06660000C TO BOTH GIVEN VARIABLES INT AND RAC
06670000C L1 , L2 = INTEGER VARIABLES CONTAINING THE DUBBLE-WORD OPERAND
06680000C INT , RAC = INTEGER AND REAL VARIABLES
06690000C NOC = OUTPUT CODE ( VALUE 1 IF THE OPERAND IS AN INTEGER ,
06700000C AND VALUE 2 IF THE OPERAND IS A REAL )
06710000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
06720000C
06730000       IF(L1-1) 30,10,20
06740000    10 INT=L2
06750000       RAC=INT
06760000       NOC=1
06770000       RETURN
06780000    20 IF(L1-3) 30,40,30
06790000    30 RAC=L2*10.**(L1/10)
06800000       INT=RAC
06810000       NOC=2
06820000       RETURN
06830000    40 CALL TXV(L2,INT,RAC,NOC)
06840000       RETURN
06850000       END
06860000C
06870000 ------SUBROUTINE DKV(L,LF1,LL1,ICODE,IDIG,KOD) ----------------------------
06880000       DIMENSION L(1),NAME(5),IDIG(1)
06890000       COMMON VIRTUA(1230),RR(20)
06900000       COMMON II(100),K(100),LL(400),IPOINT(211),ISUB(780),NBSUB(211),
06910000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
06920000       DATA NBNAME/5/
06930000       DATA NAME/'K','A','X','E','C'/
06940000       DATA ILP,IRP/'(',')'/
06950000C
06960000C DETECTING AND CODIFYING A VARIABLE ( SEE SEL ORGANIZATION )
06970000C TO BE FOUND IN THE GIVEN VECTOR'S ZONE
06980000C L = GIVEN VECTOR
06990000C LF1 , LL1 = BEGINNING AND END OF THE ZONE DELIMITING THE VARIABLE
07000000C ICODE = RESULTING CODE OF THE VARIABLE
07010000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
07020000C KOD = OUTPUT CODE ( VALUE 1 INDICATES ERROR BY INDEX OFF LIMITS ,
```

```
07030000C VALUE 0 MEANS OK , AND -1 SYNTAX ERROR OR LF1 GREATER THAN LL1 )
07040000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
07050000C
07060000       IF(LF1-LL1) 10,30,30
07070000    10 DO 20 IND=1,NBNAME
07080000       IF(L(LF1)-NAME(IND)) 20,40,20
07090000    20 CONTINUE
07100000    30 KOD=-1
07110000       RETURN
07120000    40 ICODE=1000*IND
07130000       IF(FLOAT(L(LF1+1)) .NE. ILP) GO TO 80
07140000       IF(FLOAT(L(LL1)) .NE. IRP) GO TO 30
07150000       IF(L(LF1+2) .NE. NAME(1)) GO TO 30
07160000       LFIRST=LF1+3
07170000       LLAST=LL1-1
07180000       IF(LFIRST-LLAST) 90,90,30
07190000    80 LFIRST=LF1+1
07200000       LLAST=LL1
07210000    90 CALL CUII(L,LFIRST,LLAST,IDIG,INT)
07220000       IF(INT) 30,30,100
07230000   100 ICODE=ICODE+INT
07240000       IF(L(LL1) .NE. IRP) GO TO 120
07250000       ICODE=-ICODE
07260000C TESTING IF INDEX OVERFLOW : INT GREATER THAN IVDIM(IND)
07270000   120 IF(INT-IVDIM(IND)) 140,140,130
07280000   130 KOD=1
07290000       RETURN
07300000   140 KOD=0
07310000       RETURN
07320000       END
07330000C
07340000 ------SUBROUTINE KRX(RAC,L1,L2) ----------------------------------------
07350000C
07360000C CODIFYING A REAL NUMBER IN THE TWO GIVEN INTEGER VECTOR'S ELEMENTS
07370000C RAC = GIVEN REAL NUMBER
07380000C L1 , L2 = GIVEN VECTOR'S ELEMENT ( SUBSCRIPTED IN THE CALL )
07390000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
07400000C
07410000       X=RAC
07420000       IF(X) 20,10,30
07430000    10 L1=0
07440000       L2=0
07450000       RETURN
07460000    20 X=-X
07470000    30 IF(X-10000.) 70,40,40
07480000    40 DO 50 I=1,80
07490000       X=X*0.1
07500000       IF(X-10000.) 60,50,50
07510000    50 CONTINUE
07520000    60 L1=I*10
07530000       GO TO 100
07540000    70 DO 80 I=1,80
07550000       IF(X-1000.) 80,90,90
07560000    80 X=X*10.
```

```
07570000    90 L1=-10*(I-1)
07580000   100 L2=X+0.5
07590000       IF(RAC) 110,120,120
07600000   110 L2=-L2
07610000   120 RETURN
07620000       END
07630000C
07640000 ------SUBROUTINE PUVKV(L,LF,LL,IDIG,L1,L2,KOD) ----------------------
07650000       DIMENSION L(1),IDIG(1)
07660000       DATA NA,NZ,IPOINT/'A','Z','.'/
07670000C
07680000C PACKING OF AN OPERAND ( UNSIGNED VARIABLE OR SIGNED CONSTANT )
07690000C TO BE FOUND IN THE GIVEN VECTOR'S ZONE IN TWO GIVEN VARIABLES
07700000C L = GIVEN VECTOR
07710000C LF,LL = FIRST AND LAST POSITIONS OF THE VECTOR'S ZONE TO BE PROCESSED
07720000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
07730000C L1 , L2 = GIVEN CELLS IN WHICH THE OPERAND HAS TO BE PACKED
07740000C KOD = OUTPUT CODE ( VALUE 1 IF  VALUE OFF LIMITS ,
07750000C VALUE 0 IF OK , AND -1 IF SYNTAX ERROR OR LF GREATER THAN LL )
07760000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
07770000C
07780000       IF(LF-LL) 10,10,20
07790000    10 CALL DBR(L,LF,LL,LF1,LL1)
07800000       IF(LF1) 20,20,30
07810000    20 KOD=-1
07820000       RETURN
07830000C ENQUIRING OF WHETHER VARIABLE OR CONSTANT
07840000    30 IF(FLOAT(L(LF1))-NA) 60,50,40
07850000    40 IF(FLOAT(L(LF1))-NZ) 50,50,60
07860000    50 CALL DKV(L,LF1,LL1,L2,IDIG,KOD)
07870000       L1=3
07880000       RETURN
07890000C ENQUIRING OF WHETHER INTEGER OR REAL CONSTANT
07900000    60 DO 70 I=LF1,LL1
07910000       IF(FLOAT(L(I))-IPOINT) 70,80,70
07920000    70 CONTINUE
07930000       L1=1
07940000       GO TO 90
07950000    80 L1=2
07960000    90 CALL CFNN(L,LF1,LL1,KOD,L1,L2,RAC,NEXT,IDIG)
07970000       IF(KOD) 130,100,20
07980000   100 IF(LL1+2-NEXT) 20,110,20
07990000   110 GO TO (130,120),L1
08000000   120 CALL KRX(RAC,L1,L2)
08010000   130 RETURN
08020000       END
08030000C
08040000 ------SUBROUTINE DIS100(X,Y,LB,NN) ---------------------------------
08050000       DIMENSION X(1),Y(1),LB(101)
08060000       COMMON VIRTUA(1230),RR(20)
08070000       COMMON IK(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
08080000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
08090000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
08100000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
```

```
08110000       COMMON LFIRST,INDEX,LL,IENT
08120000       COMMON NBOPTI
08130000       COMMON IBB(101)
08140000       DATA IBL,ISTAR,II,IPL,IMIN/' ','*','I','+','-'/
08150000C
08160000C DISPLAY OF THE VECTORS X ( ABCISSE ) AND Y ( ORDINATE )
08170000C X , Y = GIVEN VECTORS
08180000C LB = BUFFER FOR PRINTING LINE BY LINE
08190000C NN = COMMON DIMENSION OF THE TWO VECTORS
08200000C IY = ORDINATE VALUE PRINTED ONLY IF MULTIPLE OF 10
08210000C ( IT RUNS DOWN FROM 110 TO 10 BY STEP OF -10 )
08220000C NOTE : XMAX = YMAX = 100 ( X , Y , NN DIFFER FROM WORK TO WORK )
08230000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
08240000C
08250000C BUILDING THE LINES OF THE FIGURE
08260000C
08270000       CALL SORTRR(Y,X,N)
08280000       YGR=101.
08290000       N=NN
08300000       DO 20 I=1,NN
08310000       IF(Y(N)-YGR) 30,30,10
08320000    10 N=N-1
08330000    20 CONTINUE
08340000    30 YGR=99.
08350000       DEUX=2.
08360000       N=NN
08370000       IY=110
08380000       IX=NN
08390000       DO 180 I=1,10
08400000       IY=IY-10
08410000       IFOR=1
08420000       LB(1)=IPL
08430000       DO 170 J=1,5
08440000       DO 40 IX=2,101
08450000    40 LB(IX)=IBL
08460000    50 IF(N) 110,110,60
08470000    60 IF(Y(N)-YGR) 110,110,70
08480000    70 IX=X(N)+0.5
08490000       IF(IX) 100,90,80
08500000    80 IF(IX-100) 90,90,100
08510000    90 LB(IX+1)=ISTAR
08520000   100 N=N-1
08530000       GO TO 50
08540000   110 GO TO(120,140),IFOR
08550000   120 WRITE(KLP,130) IY,LB
08560000   130 FORMAT(I9,1X,101A1)
08570000       GO TO 160
08580000   140 WRITE(KLP,150) LB
08590000   150 FORMAT(10X,101A1)
08600000   160 IFOR=2
08610000       LB(1)=II
08620000   170 YGR=YGR-DEUX
08630000   180 CONTINUE
08640000C
```

```
08650000C BUILDING THE ABCISSE
08660000C
08670000       LB(1)=IPL
08680000       IX=2
08690000       DO 200 J=1,20
08700000       DO 190 I=1,4
08710000       LB(IX)=IMIN
08720000   190 IX=IX+1
08730000       LB(IX)=IPL
08740000   200 IX=IX+1
08750000   210 IF(N) 280,280,220
08760000   220 IF(Y(N)) 270,230,230
08770000   230 IX=X(N)+0.5
08780000       IF(IX) 260,250,240
08790000   240 IF(IX-100) 250,250,260
08800000   250 LB(IX+1)=ISTAR
08810000   260 N=N-1
08820000       GO TO 210
08830000   270 N=0
08840000   280 WRITE(KLP,290) N,LB,(I,I=5,100,5)
08850000   290 FORMAT(I9,1X,101A1/11X,20I5)
08860000       RETURN
08870000       END
08880000C
08890000 ------ SUBROUTINE ELEFF(I,X,E) ------------------------------------------
08900000       DIMENSION EF(5),BKPWGH(780)
08910000       COMMON VIRTUA(1230),RR(20)
08920000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
08930000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
08940000       EQUIVALENCE (VIRTUA(1),BKPWGH(1))
08950000C
08960000C COMPUTING THE EFFECTIVENESS OF THE ELEMENTARY CRITERIA
08970000C I = INDEX OF THE BLOCK FOR WHICH THE EFFECTIVENESS IS COMPUTED
08980000C X = INPUT VALUE OF THE ELEMENTARY CRITERION NUMBER I
08990000C E = VARIABLE DESIGNED TO CONTAIN THE COMPUTED EFFECTIVENESS
09000000C NBKP = NUMBER OF BREAKPOINTS ( BETWEEN 2 AND 5 INCLUSIVELY )
09010000C NOTE : THE COMMON ZONE (FORTRAN) MUST CONTAIN THE VECTORS
09020000C IPOINT , ISUB , AND BKPWGH
09030000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
09040000C
09050000       J1=IPOINT(I)
09060000       JN=IPOINT(I+1)-1
09070000       NBKP=0
09080000       DO 10 J=J1,JN
09090000       NBKP=NBKP+1
09100000    10 EF(NBKP)=ISUB(J)/100.
09110000       IF(X-BKPWGH(J1)) 20,20,30
09120000    20 E=EF(1)
09130000       RETURN
09140000    30 DO 50 J=2,NBKP
09150000       J1=J1+1
09160000       IF(X-BKPWGH(J1)) 40,60,50
09170000    40 E=EF(J-1)+(X-BKPWGH(J1-1))*(EF(J)-EF(J-1))/(BKPWGH(J1)-
09180000      *BKPWGH(J1-1))
```

```
09190000       RETURN
09200000    50 CONTINUE
09210000       E=EF(NBKP)
09220000       RETURN
09230000    60 E=EF(J)
09240000       RETURN
09250000       END
09260000C
09270000 ------SUBROUTINE NELEFF(I,R) ----------------------------------------
09280000       DIMENSION R(1),E(210),BKPWGH(780)
09290000       COMMON VIRTUA(1230),RR(20)
09300000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
09310000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
09320000       EQUIVALENCE (NBELC,II(99))
09330000       EQUIVALENCE (VIRTUA(1),BKPWGH(1)),(VIRTUA(901),E(1))
09340000C
09350000C COMPUTING THE EFFECTIVENESSES OF THE NON-ELEMENTARY CRITERIA
09360000C I = INDEX OF THE BLOCK FOR WHICH THE EFFECTIVENESS IS COMPUTED
09370000C R = VECTOR OF THE AGGREGATION EXPONENTS' VALUES
09380000C NOTE : THE COMMON ZONE (FORTRAN) MUST CONTAIN THE VECTORS
09390000C IPOINT , ISUB , E , AND BKPWGH
09400000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
09410000C
09420000       J1=IPOINT(I)
09430000       J2=J1+1
09440000       J=I-NBELC
09450000       J=IPTRR(J)
09460000       IF(J) 10,10,80
09470000C
09480000C PROCESSING FOR THE CASE OF PARTIAL ABSORPTION
09490000C
09500000    10 J=-J
09510000       RP=R(J)
09520000       IND=ISUB(J1)
09530000       X1=E(IND)
09540000       IND=ISUB(J2)
09550000       X2=E(IND)
09560000C FIRST STEP : THE ARITHMETIC MEAN
09570000       X2=BKPWGH(J1)*X1+BKPWGH(J2)*X2
09580000       W1=BKPWGH(J2+1)
09590000       W2=BKPWGH(J2+2)
09600000       IF(X1) 20,20,50
09610000    20 IF(RP) 30,30,40
09620000    30 E(I)=0.
09630000       RETURN
09640000    40 E(I)=X2*EXP(ALOG(W2)/RP)
09650000       RETURN
09660000    50 IF(RP*(X1-X2)) 70,70,60
09670000    60 E(I)=EXP(RP*ALOG(X1/X2)/RP+ALOG(X2))
09680000       RETURN
09690000    70 E(I)=EXP(RP*ALOG(X2/X1)/RP+ALOG(X1))
09700000       RETURN
09710000C
09720000C PROCESSING FOR THE CASE OF TOTAL ABSORPTION
```

```
09730000C
09740000    80 RP=R(J)
09750000       JN=IPOINT(I+1)-1
09760000       J=ISUB(J1)
09770000       EM=E(J)
09780000       EE=0.
09790000       IF(RP) 90,90,170
09800000C SEARCHING THE MINIMAL EFFECTIVENESS
09810000    90 DO 110 J=J2,JN
09820000       IND=ISUB(J)
09830000       IF(E(IND)-EM) 100,110,110
09840000   100 EM=E(IND)
09850000   110 CONTINUE
09860000C PROCESSING
09870000       IF(RP+1000.) 120,120,130
09880000   120 E(I)=EM
09890000       RETURN
09900000   130 IF(EM) 120,120,140
09910000   140 ELOG=-ALOG(EM)
09920000       DO 160 J=J1,JN
09930000       IND=ISUB(J)
09940000       IF(E(IND)) 160,160,150
09950000   150 EE=EXP((ALOG(E(IND))+ELOG)*RP)*BKPWGH(J)+EE
09960000   160 CONTINUE
09970000       E(I)=EXP(ALOG(EE)/RP-ELOG)
09980000       RETURN
09990000C SEARCHING THE MAXIMAL EFFECTIVENESS
10000000   170 DO 190 J=J2,JN
10010000       IND=ISUB(J)
10020000       IF(E(IND)-EM) 190,190,180
10030000   180 EM=E(IND)
10040000   190 CONTINUE
10050000       IF (RP-1000.) 130,120,120
10060000       END
10070000C
10080000 ------SUBROUTINE BITROX(V,N,VALUE,I) ----------------------------------
10090000       DIMENSION V(1)
10100000C
10110000C BINARY TRACING OF THE POSITION OF A CERTAIN ELEMENT
10120000C IN THE GIVEN VECTOR
10130000C V = GIVEN VECTOR ( SORTED ON INCREASING VALUES )
10140000C N = DIMENSION OF THE GIVEN VECTOR
10150000C VALUE = NUMBER TO BE FOUND OR APPROXIMATED IN THE GIVEN VECTOR
10160000C I = POSITION OF THE FOUND ELEMENT
10170000C NOTE. THIS PROCEDURE RUNS AS FOLLOWS :
10180000C SEARCH AN ELEMENT EQUAL TO VALUE ;
10190000C IF INSUCCESS , SEARCH AN ELEMENT NEXT TO VALUE LESS THAN EPSIL ;
10200000C IF INSUCCESS AGAIN , SEARCH BETWEEN WHICH TWO ELEMENTS VALUE
10210000C IS , AND TAKE THE GREATER ONE .
10220000C  DUJMOVIC / VAN BASTELAER / ARNOTTE   JANUARY 1977
10230000C
10240000       IF(N) 130,130,10
10250000    10 IF(VALUE-V(1)) 20,30,30
10260000    20 I=1
```

```
10270000       GO TO 130
10280000    30 IF(VALUE-V(N)) 50,40,40
10290000    40 I=N
10300000       GO TO 130
10310000    50 EPSIL=V(N)*0.00001
10320000       IL=1
10330000       IR=N
10340000    60 IF(IL-IR) 70,120,120
10350000    70 I=(IL+IR)/2
10360000       DIF=VALUE-V(I)
10370000       IF(DIF) 80,130,100
10380000    80 IF(EPSIL+DIF) 90,130,130
10390000    90 IR=I-1
10400000       GO TO 60
10410000   100 IF(EPSIL-DIF) 110,130,130
10420000   110 IL=I+1
10430000       GO TO 60
10440000   120 I=IL
10450000   130 RETURN
10460000       END
10470000C
10480000 ------SUBROUTINE MONIT(INJCL,INSEL,ICHAR,IDIG,JOB) ----------------------
10490000       DIMENSION INJCL(1),ICHAR(11),IDIG(1),INSEL(1)
10500000       DIMENSION IB(80),NF(31)
10510000       COMMON VIRTUA(1230),RR(20)
10520000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
10530000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
10540000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
10550000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
10560000       COMMON LFIRST,INDEX,LL,IENT
10570000       COMMON NBOPTI
10580000       COMMON IBB(101)
10590000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
10600000       EQUIVALENCE (IBB(21),IB(1))
10610000       EQUIVALENCE(ISUB(780),NF(31))
10620000       DATA NC/'C'/
10630000C
10640000C INTERPRETOR FOR THE MONITOR COMMANDS  *JOB , *SEL , *READ ,
10650000C *PUNCH , *PRINT , *STORE , *EXECUTE , *DELETE , *NEW PAGE , *EXIT
10660000C INJCL = VECTOR OF THE NAMES OF THE MONITOR INSTRUCTIONS
10670000C INSEL = VECTOR OF THE NAMES OF THE SEL INSTRUCTIONS
10680000C ICHAR = VECTOR OF SPECIAL CHARACTERS
10690000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
10700000C JOB = INDICATOR OF CORRECT SEL PROGRAM (VALUE 1 INDICATES THAT
10710000C THE DECK BEGINS BY A *JOB CARD) , OR INCORRECT SEL PROGRAM (VALUE 0)
10720000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
10730000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
10740000C
10750000C----------------------
10760000C INPUT OF SEL COMMANDS
10770000C----------------------
10780000C RECOGNIZING WHICH COMMAND
10790000    10 READ(KCR,100,END=10000) IB
10800000       DO 11 I=1,80
```

```
10810000C ICHAR(1) = BLANK CHARACTER
10820000       IF(FLOAT(IB(I)) .NE. ICHAR(1)) GO TO 90
10830000    11 CONTINUE
10840000       WRITE(KLP,80)
10850000    80 FORMAT (1X,'(BLANK CARD)')
10860000       GO TO 10
10870000    90 WRITE(KLP,101) IB
10880000   100 FORMAT(80A1)
10890000   101 FORMAT(1X,80A1)
10900000C ICHAR(2) = *
10910000       IF(FLOAT(IB(1)) .EQ. ICHAR(2)) GO TO 130
10920000       IF(FLOAT(IB(1)) .EQ. NC) GO TO 10
10930000   110 WRITE(KLP,120)
10940000   120 FORMAT(' ----   INCORRECT SEL COMMAND'/)
10950000       GO TO 10
10960000C-------------------------------------
10970000C DETECTION OF THE TYPE OF THE COMMAND
10980000C-------------------------------------
10990000   130 CALL DIN(INJCL,1,NBCDS,IB,2,INSTR)
11000000       IF(INSTR) 110,110,135
11010000C TESTING THAT THE PROGRAM BEGINS BY THE *JOB COMMAND
11020000   135 IF(JOB) 136,136,140
11030000   136 IF(INSTR-1) 137,1000,137
11040000   137 WRITE(KLP,138)
11050000   138 FORMAT(' ----   THE FIRST COMMAND MUST BE THE *JOB COMMAND'/)
11060000       GO TO 10
11070000   140 GO TO (1000,2000,3000,4000,10,6000,7000,8000,9000,10000),INSTR
11080000C---------------------------
11090000C EXECUTING THE *JOB COMMAND
11100000C---------------------------
11110000C 1000 NIN=0
11120000       JOB=1
11130000       NBELC=0
11140000       NBTOTC=0
11150000       INDEX=2*NBINP-1
11160000       DO 1010 I=1,INDEX
11170000  1010 NF(I)=0
11180000       GO TO 10
11190000C---------------------------
11200000C EXECUTING THE *SEL COMMAND
11210000C---------------------------
11220000C ICHAR(3) = ,
11230000  2000 CALL DFGC(IB,5,77,INDEX,KOD,ICHAR(3))
11240000       IF(KOD) 2010,2030,2010
11250000  2010 LIST1=0
11260000C**************
11270000  2020 IPROG=2
11280000       IENT=1
11290000       RETURN
11300000C**************
11310000  2030 CALL DFL(IB,INDEX+1,78,INDEX,KOD)
11320000       IF(KOD) 110,2040,110
11330000  2040 CALL DIN(INJCL,NBCDS+1,NBCDS+1,IB,INDEX,INSTR)
11340000       IF(INSTR) 110,110,2050
```

```
11350000 2050 LIST1=1
11360000      GO TO 2020
11370000C-------------------------------
11380000C EXECUTING THE COMMAND *EXECUTE
11390000C-------------------------------
11400000 3000 IF(NIN) 3010,3010,3030
11410000 3010 WRITE(KLP,3020)
11420000 3020 FORMAT('----  INSEL PROGRAM NONEXISTENT'/)
11430000      GO TO 10
11440000 3030 WRITE(KLP,3040)
11450000 3040 FORMAT(/)
11460000C**************
11470000      IPROG=5
11480000      IENT=1
11490000      RETURN
11500000C**************
11510000C-----------------------------------------
11520000C EXECUTING THE COMMAND *READ INSEL PROGRAM
11530000C-----------------------------------------
11540000 4000 CALL DFD(IB,5,80,INDEX,KOD)
11550000      IF(KOD) 110,4110,110
11560000 4110 IOUT=1
11570000C
11580000C OPEN SUBROUTINE TESTING IF THE FILE NUMBER IS CORRECT :
11590000C ARRIVAL FROM INSTRUCTIONS 4110 , 6020+1 , AND 7010+1
11600000C
11610000C ICHAR(1) = ' '
11620000 4115 CALL DFGC(IB,INDEX,80,INSTR,KOD,ICHAR(1))
11630000      IF(KOD) 4120,4130,4120
11640000 4120 INSTR=80
11650000      GO TO 4135
11660000 4130 INSTR=INSTR-1
11670000 4135 CALL CUII(IB,INDEX,INSTR,IDIG,NFILE)
11680000      IF(NFILE-30) 4150,4150,4145
11690000 4145 IF(NFILE-30-NBFILE) 4170,4170,4150
11700000 4150 WRITE(KLP,4160)
11710000 4160 FORMAT(' ----   INCORRECT FILE NUMBER'/)
11720000      GO TO 10
11730000 4170 GO TO (4175,6030,7020),IOUT
11740000C
11750000C END OF THE OPEN SUBROUTINE
11760000C
11770000 4175 READ(NFILE'1,1,ERR=2) NIN
11780000      NIN1=NIN+1
11790000      IF(NIN) 4180,4180,4200
11800000 4180 WRITE(KLP,4190) NFILE
11810000 4190 FORMAT(' ----   FILE NO.'I4' IS EMPTY'/)
11820000      GO TO 10
11830000 4200 READ(NFILE'2,1,ERR=2) (K(J),J=1,NIN1)
11840000      KOD=K(NIN1)-1
11850000      READ(NFILE'NIN1+2,1,ERR=2)(L(J),J=1,KOD)
11860000      READ(NFILE'KOD+NIN1+2,1,ERR=2) IND
11870000      IFIL=KOD+NIN1+3
11880000      IF(IND) 4250,4250,4210
```

```
11890000 4210 IOUT=IND/80
11900000      KOD=IND-80*IOUT
11910000      IFIL12=1
11920000      IF(IOUT) 4250,4240,4220
11930000C READING THE TITLES IN FORMAT A10
11940000 4220 DO 4230 INDEX=1,IOUT
11950000      READ(NFILE'IFIL,3,ERR=2) IB
11960000      IFIL=IFIL+80
11970000 4230 WRITE(42'IFIL12,3) IB
11980000      IF(KOD) 4250,4250,4240
11990000 4240 READ(NFILE'IFIL,3,ERR=2) (IB(J),J=1,KOD)
12000000      WRITE(42'IFIL12,3)(IB(J),J=1,KOD)
12010000      IFIL=IFIL+KOD
12020000 4250 READ(NFILE'IFIL,1,ERR=2) INDLAB
12030000      IF(INDLAB) 4255,4255,4260
12040000 4255 KOD=IFIL+1
12050000      GO TO 4290
12060000 4260 READ(NFILE'IFIL+1,1,ERR=2) (LABFIL(J),IA(J),J=1,INDLAB)
12070000      KOD=IFIL+1+2*INDLAB
12080000 4290 WRITE(KLP,4300) NIN,KOD,IND
12090000 4300 FORMAT(6X'TRANSACTION PERFORMED (TOTAL'I4' INSTRUCTION(S),'/
12100000     *6X'FILE SPACE NEEDED FOR INSEL PROGRAM'I5' INTEGER(S),'/
12110000     *6X'TITLE FILE CONTAINS'I4' CHARACTER(S))'/)
12120000      GO TO 10
12130000C-----------------------------------------------
12140000C EXECUTION OF THE COMMAND *STORE INSEL PROGRAM
12150000C-----------------------------------------------
12160000 6000 IF(NIN) 3010,3010,6010
12170000 6010 CALL DFD(IB,5,80,INDEX,KOD)
12180000      IF(KOD) 110,6020,110
12190000 6020 IOUT=2
12200000      GO TO 4115
12210000 6030 READ(NFILE'1,1,ERR=2) J
12220000      IF(J) 6040,6060,6040
12230000 6040 WRITE(KLP,6050) NFILE
12240000 6050 FORMAT(' ----   FILE NO.'I4' IS OCCUPIED'/)
12250000      GO TO 10
12260000 6060 WRITE(NFILE'1,1) NIN
12270000      NIN1=NIN+1
12280000      WRITE(NFILE'2,1)(K(J),J=1,NIN1)
12290000      KOD=K(NIN1)-1
12300000      WRITE(NFILE'NIN1+2,1)(L(J),J=1,KOD)
12310000      IND=IFIL12-1
12320000      WRITE(NFILE'KOD+NIN1+2,1) IND
12330000      IFIL=KOD+NIN1+3
12340000      IF(IND) 6120,6120,6080
12350000 6080 IOUT=IND/80
12360000      KOD=IND-80*IOUT
12370000      IFIL12=1
12380000      IF(IOUT) 6120,6110,6090
12390000C WRITING THE TITLES IN FORMAT A10
12400000 6090 DO 6100 J=1,IOUT
12410000      READ(42'IFIL12,3,ERR=2) IB
12420000      WRITE(NFILE'IFIL,3) IB
```

```
12430000 6100 IFIL=IFIL+80
12440000      IF(KOD) 6120,6120,6110
12450000 6110 READ(42'IFIL12,3,ERR=2) (IB(J),J=1,KOD)
12460000      WRITE(NFILE'IFIL,3)(IB(J),J=1,KOD)
12470000      IFIL=IFIL+KOD
12480000 6120 WRITE(NFILE'IFIL,1) INDLAB
12490000      IF(INDLAB) 6125,6125,6130
12500000 6125 KOD=IFIL+1
12510000      GO TO 4290
12520000 6130 WRITE(NFILE'IFIL+1,1) (LABFIL(J),IA(J),J=1,INDLAB)
12530000      KOD=IFIL+1+2*INDLAB
12540000      GO TO 4290
12550000C-----------------------------------
12560000C EXECUTING THE COMMAND *DELETE FILE
12570000C-----------------------------------
12580000 7000 CALL DFD(IB,5,80,INDEX,KOD)
12590000      IF(KOD) 110,7010,110
12600000 7010 IOUT=3
12610000      GO TO 4115
12620000 7020 IOUT=0
12630000      WRITE(NFILE'1,1) IOUT
12640000      WRITE(KLP,7030) NFILE
12650000 7030 FORMAT(' ----    FILE NO.'I4' DELETED'/)
12660000      GO TO 10
12670000C-----------------------------------------
12680000C EXECUTING THE COMMAND *PRINT INSEL PROGRAM
12690000C-----------------------------------------
12700000 8000 IF(NIN) 3010,3010,8010
12710000 8010 WRITE(KLP,8020)(I1,I1=1,8)
12720000 8020 FORMAT('1INSEL PROGRAM LISTING'/' NO.  NAME'5X'OP'8('    D'I1),
12730000     */65('-'))
12740000      DO 8060 N=1,NIN
12750000      I1=K(N)
12760000      I2=K(N+1)-1
12770000      IOUT=L(I1)
12780000      IOUT=1+(IOUT-1)*3
12790000 8050 FORMAT(I4,2X,3A1,2X,9I6)
12800000 8060 WRITE(KLP,8050) N,INSEL(IOUT),INSEL(IOUT+1),INSEL(IOUT+2),
12810000     *(L(I),I=I1,I2)
12820000      WRITE(KLP,8070)
12830000 8070 FORMAT(65('-')/)
12840000      GO TO 10
12850000C-----------------------------------
12860000C TRANSLATING THE COMMAND *NEW PAGE
12870000C-----------------------------------
12880000 9000 WRITE(KLP,9010)
12890000 9010 FORMAT(1H1)
12900000      GO TO 10
12910000C----------------------------
12920000C EXECUTING THE COMMAND *EXIT
12930000C----------------------------
1294000010000 CALL EXIT
12950000      RETURN
12960000C STANDARDS FOR FILE MANAGEMENT
```

```
12970000      1 FORMAT(I10)
12980000      2 CALL EXIT
12990000      3 FORMAT (A10)
13000000        RETURN
13010000        END
13020000C
13030000 ------SUBROUTINE TRAN1(INSEL,ICHAR,IDIG,IAZ,IUSVEC) ----------------------
13040000        DIMENSION INSEL(51),ICHAR(11),IDIG(10),IAZ(2),IEND(3),
13050000       *IFIELD(17),IUSVEC(5)
13060000        DIMENSION IB(80)
13070000        COMMON VIRTUA(1230),RR(20)
13080000        COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
13090000       *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
13100000        COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
13110000       *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
13120000        COMMON LFIRST,INDEX,LL,IENT
13130000        COMMON NBOPTI
13140000        COMMON IBB(101)
13150000        EQUIVALENCE (IBB(21),IB(1))
13160000        DATA IEND/'E','N','D'/
13170000        DATA IFIELD/1,4,4,4,4,7,9,1,2,3,2,4,3,3,4,2,5/
13180000C
13190000C TRANSLATING THE INSTRUCTIONS
13200000C TITLE , REPEAT , EXIT  AND  WAIT  ( IN TWO SCANNINGS )
13210000C NOTE : THE INSTRUCTION 'WAIT' SIMPLY CONTINUES THE JOB ;
13220000C ITS EFFECT WAS ORIGINALLY A FORTRAN 'PAUSE' INSTRUCTION ,
13230000C THAT WAS IMPRACTICABLE ON THE MULTIPROCESSED SIEMENS 4004 .
13240000C IT MAY BE USED NOW AS A COMMENT CARD.
13250000C INSEL = VECTOR OF THE NAMES OF THE SEL INSTRUCTIONS
13260000C ICHAR = VECTOR OF SPECIAL CHARACTERS
13270000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
13280000C IAZ = VECTOR CONTAINING THE LETTERS A AND Z
13290000C IUSVEC = VECTOR CONTAINING THE NAMES OF THE VECTORS
13300000C WHICH MAY BE ADDRESSED BY THE PROGRAMMER
13310000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
13320000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
13330000C
13340000        GO TO(100,7,1111),IENT
13350000C INITIALIZATION OF SEL SOURCE TRANSLATION ( NIN = 0 )
13360000  100 NIN=0
13370000        INDLAB=0
13380000        IFIL12=1
13390000        K(1)=1
13400000        WRITE(KLP,5)
13410000    5 FORMAT(1X)
13420000C GOING ON IN TRANSLATING SEL SOURCE ( NIN = NIN + 1 )
13430000    7 NIN=NIN+1
13440000        IF(NIN .LT. KDIM) GO TO 10
13450000    8 WRITE(KLP,9)
13460000    9 FORMAT(/' ----    INSEL PROGRAM EXCEEDS THE VECTOR L'/
13470000       *6X'TRANSLATION ENDED' )
13480000        NIN=0
13490000C**************
13500000   12 IPROG=1
```

```
13510000       RETURN
13520000C**************
13530000C-------------------------
13540000C  READING THE INSTRUCTION
13550000C-------------------------
13560000    10 READ(KCR,11,END=12) IB
13570000    11 FORMAT(80A1)
13580000C IUSVEC(5) = 'C'    ( TESTING IF COMMENT )
13590000       IF(FLOAT(IB(1)) .EQ. IUSVEC(5)) GO TO 14
13600000       CALL DFL(IB,1,80,INDEX,KOD)
13610000       CALL DFW(IB,1,INDEX-1,LF1,LL1,INT)
13620000       IF(KOD) 1120,17,1120
13630000    14 IF(LIST1) 15,10,15
13640000    15 WRITE(KLP,16) IB
13650000    16 FORMAT(3X,80A1)
13660000       GO TO 10
13670000    17 IF(LIST1) 18,20,18
13680000    18 WRITE(KLP,19) IB,NIN
13690000    19 FORMAT(3X,80A1,I4)
13700000    20 IF(INT) 120,35,35
13710000    35 CALL  CUII(IB,LF1,LL1,IDIG,INT)
13720000       IF(INT) 40,40,46
13730000    40 INSTR=2
13740000       INDEX=120
13750000       GO TO 1000
13760000C THE INSTRUCTION CONTAINS A REFERENCE NUMBER
13770000    46 INDLAB=INDLAB+1
13780000       IF(INDLAB-NBLAB) 47,47,8
13790000    47 LABFIL(INDLAB)=INT
13800000       IA(INDLAB)=NIN
13810000C-----------------------------------------------
13820000C DETECTING AND TRANSLATING THE INSTRUCTION END
13830000C-----------------------------------------------
13840000   120 IF(IB(INDEX) .NE. FLOAT(IEND(1))) GO TO 900
13850000   121 IF(IB(INDEX+1) .NE. FLOAT(IEND(2))) GO TO 900
13860000   122 IF(IB(INDEX+2) .NE. FLOAT(IEND(3))) GO TO 900
13870000   123 NIN=NIN-1
13880000       WRITE(KLP,5)
13890000       IF(NIN) 124,124,128
13900000   124 WRITE(KLP,125)
13910000   125 FORMAT(' INVALID PROGRAM')
13920000   126 WRITE(KLP,127)
13930000   127 FORMAT( ' END OF TRANSLATION'/)
13940000C**************
13950000       IPROG=1
13960000       RETURN
13970000C**************
13980000C ---------------------------------
13990000C SEVERAL CONTROLS OF INSEL PROGRAM
14000000C ---------------------------------
14010000C
14020000C VERIFYING IF EACH INSTRUCTION HAS A VALID OPERATION CODE
14030000C
14040000   128 KOD=0
```

```
14050000       DO 133 INSTR=1,NIN
14060000       LFIRST=K(INSTR)
14070000       IF(L(LFIRST)) 132,132,133
14080000   132 KOD=KOD+1
14090000       L(KOD)=INSTR
14100000   133 CONTINUE
14110000C  IF ALL VALID GO TO 136  ELSE GO TO 173
14120000       IF(KOD) 136,136,173
14130000   134 WRITE(KLP,135)
14140000   135 FORMAT(' NO ERROR IN ABOVE PROGRAM')
14150000       GO TO 126
14160000C
14170000C  CORRECTNESS TEST OF THE INSTRUCTION PRECEDING 'END'
14180000C  IT MAY BE EXIT , BRANCH OR REPEAT
14190000C
14200000   136 INSTR=K(NIN)
14210000       KOD=L(INSTR)
14220000       IF(KOD-1) 139,180,139
14230000   139 IF(KOD-3) 140,180,140
14240000   140 IF(KOD-7) 160,180,160
14250000   160 WRITE(KLP,165)
14260000   165 FORMAT(/' LAST EXECUTABLE STATEMENT BEFORE END'/
14270000      *' STATEMENT IS NOT EXIT, BRANCH OR REPEAT')
14280000       GO TO 180
14290000   173 WRITE(KLP,174) KOD
14300000   174 FORMAT(' FOLLOWING'I4' STATEMENT(S) CONTAIN ERRORS')
14310000       WRITE(KLP,175)(L(INSTR),INSTR=1,KOD)
14320000   175 FORMAT(24I5)
14330000       NIN=0
14340000       GO TO 124
14350000C
14360000C  SECOND SCANNING
14370000C  COMPUTING THE INTERNAL REFERENCE NUMBERS
14380000C  AND TESTING THE CORRECTNESS OF THE JUMP REFERENCES
14390000C
14400000   180 IF(INDLAB-1) 186,186,181
14410000   181 CALL SORTII(LABFIL,IA,INDLAB)
14420000   186 DO 400 INSTR=1,NIN
14430000       LFIRST=K(INSTR)
14440000       IF(L(LFIRST)-3) 210,190,210
14450000C
14460000C  ANALYZING THE INSTRUCTION REPEAT
14470000C
14480000   190 CALL BINTRA(LABFIL,INDLAB,L(LFIRST+1),KOD)
14490000       IF(KOD) 194,194,200
14500000   194 NIN=0
14510000       WRITE(KLP,195) INSTR
14520000   195 FORMAT(' ----   INSTRUCTION NO.'I5' CONTAINS AN INCORRECT REFERENC
14530000      *NUMBER')
14540000       GO TO 400
14550000   200 L(LFIRST+1)=IA(KOD)
14560000       GO TO 400
14570000C  ANALYZING THE INSTRUCTIONS OF THE BRANCH TYPES
14580000   210 IF(L(LFIRST)-7) 400,220,400
```

```
14590000   220 IF(L(LFIRST+4)-6) 230,270,230
14600000C FIRST TYPE : LOOP
14610000   230 CALL BINTRA(LABFIL,INDLAB,L(LFIRST+1),KOD)
14620000       IF(KOD) 194,194,240
14630000   240 L(LFIRST+1)=IA(KOD)
14640000       IF(L(LFIRST+4)-6) 250,244,244
14650000   244 KOD=IA(KOD)
14660000       KOD=K(KOD)
14670000       IF(L(KOD)-6) 194,400,194
14680000C SECOND TYPE : CONDITIONAL BRANCH
14690000   250 DO 260 LL=7,8
14700000       LL1=LFIRST+LL
14710000       CALL BINTRA(LABFIL,INDLAB,L(LL1),KOD)
14720000       IF(KOD) 194,194,260
14730000   260 L(LL1)=IA(KOD)
14740000       GO TO 400
14750000C THIRD TYPE : UNCONDITIONAL AND COMPUTED BRANCHS
14760000   270 IF(L(LFIRST+2)-2) 275,194,400
14770000   275 CALL BINTRA(LABFIL,INDLAB,L(LFIRST+3),KOD)
14780000       IF(KOD) 194,194,280
14790000   280 L(LFIRST+3)=IA(KOD)
14800000   400 CONTINUE
14810000       IF(NIN) 124,124,134
14820000C--------------------------------
14830000C IDENTIFYING THE INSTRUCTION TYPE
14840000C--------------------------------
14850000   900 CALL DIN(INSEL,1,NISEL,IB,INDEX,INSTR)
14860000       IF(INSTR) 910,910,1000
14870000C-------------------------------------
14880000C DETECTING AN ARITHMETIC INSTRUCTION
14890000C-------------------------------------
14900000C ICHAR(8) = '='
14910000   910 CALL DFGC(IB,INDEX+2,79,LL,KOD,ICHAR(8))
14920000       IF(KOD) 40,920,40
14930000   920 INSTR=6
14940000C-----------------------------------------------------
14950000C INITIALIZING THE VECTORS L AND K , FOR ALL INSTRUCTIONS
14960000C-----------------------------------------------------
14970000  1000 LFIRST=K(NIN)
14980000       L(LFIRST)=INSTR
14990000       K(NIN+1)=K(NIN)+IFIELD(INSTR)
15000000       IF(K(NIN+1)-1-LDIM) 1001,1001,8
15010000C-------------------------------------------------
15020000C BRANCH TO THE PARTICULAR INSTRUCTION'S PROCESSING
15030000C-------------------------------------------------
15040000  1001 GO TO(7,1100,1300,1500,1500,1900,2000,7,2200,2400,3000,3300,
15050000      *3500,3800,3800,5000,6000),INSTR
15060000C---------------------------------------
15070000C TRANSLATING THE INSTRUCTION TITLE (OP=2)
15080000C---------------------------------------
15090000  1100 CALL DFD(IB,INDEX+3,80,LF,KOD)
15100000       IF(KOD) 1111,1140,1111
15110000C*********
15120000C
```

```
15130000C OPEN SUBROUTINE FOR THE CASES  :
15140000C - NO INSTRUCTION SPECIFIED AFTER A REFERENCE NUMBER (ARRIV. FROM 11+4)
15150000C - SYNTAX ERROR AT THE INSTRUCTIONS TITLE AND REPEAT ( IN TRAN1 )
15160000C - SYNTAX ERROR IN THE OTHER INSTRUCTIONS PROCESSED BY TRAN2
15170000C   AND TRAN3 , BRANCHING TO TRAN1'S INSTRUCTION 1111
15180000C
15190000C**********
15200000 1111 L(LFIRST)=0
15210000      IF(LIST1) 1120,7,1120
15220000 1120 WRITE(KLP,1130)
15230000 1130 FORMAT(' ----   SYNTAX ERROR AT PREVIOUS STATEMENT')
15240000      GO TO 7
15250000C ICHAR(3) = ,
15260000 1140 CALL DFGC(IB,LF+1,80,LL,KOD,ICHAR(3))
15270000      IF(KOD) 1111,1150,1111
15280000 1150 CALL DBR(IB,LF,LL-1,LF1,LL1)
15290000      IF(LF1) 1111,1111,1160
15300000 1160 IF(LF1-LL1) 1170,1170,1111
15310000 1170 DO 1180 INT=2,4
15320000      IF(IB(LF1) .EQ. IDIG(INT)) GO TO 1190
15330000 1180 CONTINUE
15340000      GO TO 1111
15350000 1190 L(LFIRST+1)=INT-1
15360000      CALL DBR(IB,LL+1,80,LF1,LL1)
15370000      IF(LF1) 1111,1111,1210
15380000C ICHAR(5) = '
15390000 1210 IF(IB(LF1) .NE. ICHAR(5)) GO TO 1250
15400000      IF(IB(LL1) .NE. ICHAR(5)) GO TO 1111
15410000C IS THERE AT LEAST ONE CHARACTER BETWEEN THE TWO QUOTES ?
15420000      IF(LL1-LF1-2) 1111,1240,1240
15430000 1240 LF1=LF1+1
15440000      LL1=LL1-1
15450000      L(LFIRST+2)=IFIL12
15460000      WRITE(42'IFIL12,1245)(IB(INT),INT=LF1,LL1)
15470000 1245 FORMAT(A10)
15480000      IFIL12=L(LFIRST+2)+LL1-LF1+1
15490000  110 L(LFIRST+3)=IFIL12-1
15500000      GO TO 7
15510000 1250 CALL CUII(IB,LF1,LL1,IDIG,INT)
15520000      IF(INT) 1111,1111,1270
15530000 1270 L(LFIRST+2)=0
15540000      L(LFIRST+3)=INT
15550000      GO TO 7
15560000C-----------------------------------------
15570000C TRANSLATING THE INSTRUCTION REPEAT (OP=3)
15580000C-----------------------------------------
15590000 1300 CALL DFD(IB,INDEX+3,80,LF,KOD)
15600000      IF(KOD) 1111,1310,1111
15610000C ICHAR(3) = ','
15620000 1310 CALL DFGC(IB,LF+1,80,LL,KOD,ICHAR(3))
15630000      IF(KOD) 1111,1320,1111
15640000 1320 CALL DBR(IB,LF,LL-1,LF1,LL1)
15650000      IF(LF1) 1111,1111,1330
15660000 1330 CALL CUII(IB,LF1,LL1,IDIG,INT)
```

```
15670000       IF(INT) 1111,1111,1350
15680000 1350 L(LFIRST+1)=INT
15690000       CALL DBR(IB,LL+1,80,LF1,LL1)
15700000       IF(LF1) 1111,1111,1360
15710000 1360 CALL CUII(IB,LF1,LL1,IDIG,INT)
15720000       IF(INT) 1111,1111,1380
15730000 1380 L(LFIRST+2)=INT
15740000       L(LFIRST+3)=INT
15750000       GO TO 7
15760000C
15770000C PREPARING THE EXIT OUT OF THE SUBROUTINE
15780000 1500 IPROG=3
15790000       IENT=1
15800000       RETURN
15810000 1900 IPROG=3
15820000       IENT=2
15830000       RETURN
15840000 2000 IPROG=3
15850000       IENT=3
15860000       RETURN
15870000 2200 IPROG=3
15880000       IENT=4
15890000       RETURN
15900000 2400 IPROG=3
15910000       IENT=5
15920000       RETURN
15930000 3000 IPROG=4
15940000       IENT=1
15950000       RETURN
15960000 3300 IPROG=4
15970000       IENT=2
15980000       RETURN
15990000 3500 IPROG=4
16000000       IENT=3
16010000       RETURN
16020000 3800 IPROG=4
16030000       IENT=4
16040000       RETURN
16050000 5000 IPROG=4
16060000       IENT=5
16070000       RETURN
16080000 6000 IPROG=4
16090000       IENT=6
16100000       RETURN
16110000C STANDARDS FOR FILE MANAGEMENT
16120000    1 FORMAT(I10)
16130000    3 FORMAT(A10)
16140000    2 CALL EXIT
16150000      RETURN
16160000      END
16170000C
16180000 ------SUBROUTINE TRAN2(ICHAR,IDIG,IAZ,IUSVEC) ----------------------------
16190000       DIMENSION ICHAR(11),IDIG(10),IAZ(2),IUSVEC(5),IOPRT(5)
16200000       DIMENSION IELC(3),ICAS(3)
```

```
16210000       DIMENSION IB(80)
16220000       COMMON VIRTUA(1230),RR(20)
16230000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
16240000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
16250000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAP,
16260000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
16270000       COMMON LFIRST,INDEX,LL,IENT
16280000       COMMON NBOPTI
16290000       COMMON IBB(101)
16300000       EQUIVALENCE (IBB(21),IB(1))
16310000       DATA IOPRT/'+','-','*','/','**'/
16320000       DATA IELC,ICAS/'E','L','C','C','A','S'/
16330000       DATA IS/'S'/
16340000C
16350000C TRANSLATING THE INSTRUCTIONS
16360000C READ , PRINT , ARITHMETIC , BRANCH , INPUT ,
16370000C AND EFFECTIVENESS
16380000C ICHAR = VECTOR OF SPECIAL CHARACTERS
16390000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
16400000C IAZ = VECTOR CONTAINING THE LETTERS A AND Z
16410000C IUSVEC = VECTOR CONTAINING THE NAMES OF THE VECTORS
16420000C WHICH MAY BE ADDRESSED BY THE PROGRAMMER
16430000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
16440000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
16450000C
16460000       GO TO(1500,1900,2000,2200,2400),IENT
16470000C-------------------------------------------------------------
16480000C TRANSLATING THE INSTRUCTIONS READ (OP=4) AND PRINT (OP=5)
16490000C-------------------------------------------------------------
16500000 1500 CALL DFGCD(IB,INDEX+3,80,LF,KOD,ICHAR(6))
16510000       DO 1510 INT=1,5
16520000       IF(IB(LF-1) .EQ. IUSVEC(INT)) GO TO 1520
16530000 1510 CONTINUE
16540000       GO TO 1111
16550000 1520 L(LFIRST+1)=100*INT
16560000       LF1=0
16570000       LL1=0
16580000       IF(KOD-1) 1111,1530,1580
16590000C INDIRECTION ( MANY VARIABLES )
16600000 1530 CALL D2GC(IB,LF+1,80,INDEX,LL,KOD,ICHAR(3),ICHAR(7))
16610000       IF(KOD) 1111,1531,1111
16620000 1531 DO 1560 IND=2,3
16630000       KOPRO=LFIRST+IND
16640000       CALL DBR(IB,LF+1,INDEX-1,LF1,LL1)
16650000       IF(IB(LF1) .EQ. IUSVEC(1)) GO TO 1540
16660000 1535 CALL CUII(IB,LF1,LL1,IDIG,INT1)
16670000       IF(INT1) 1111,1111,1537
16680000 1537 IF(FLOAT(INT1)-IVDIM(INT)) 1555,1555,1631
16690000 1540 CALL CUII(IB,LF1+1,LL1,IDIG,INT1)
16700000       IF(INT1) 1111,1111,1545
16710000 1545 IF(FLOAT(INT1)-IVDIM(1)) 1550,1550,1631
16720000 1550 INT1=-INT1
16730000 1555 L(KOPRO)=INT1
16740000       LF=INDEX
```

```
16750000       INDEX=LL
16760000 1560 CONTINUE
16770000       DO 1565 IND=2,3
16780000       KOPRO=LFIRST+IND
16790000       IF(L(KOPRO)) 1680,1680,1565
16800000 1565 CONTINUE
16810000       IF(FLOAT(L(LFIRST+3))-L(LFIRST+2)) 1631,1680,1680
16820000C NO INDIRECTION ( A SINGLE VARIABLE )
16830000 1580 CALL DFA2GC(IB,LF+1,80,LL,KOD,ICHAR(1),ICHAR(3))
16840000       IF(KOD) 1590,1590,1600
16850000 1590 LL=81
16860000 1600 CALL DBR(IB,LF,LL-1,LF1,LL1)
16870000       CALL CUII(IB,LF1,LL1,IDIG,INT1)
16880000       IF(INT1) 1111,1111,1605
16890000 1605 IF(FLOAT(INT1)-IVDIM(INT)) 1610,1610,1631
16900000 1610 L(LFIRST+2)=INT1
16910000       L(LFIRST+3)=INT1
16920000       GO TO 1680
16930000 1631 L(LFIRST)=0
16940000       IF(LIST1) 1632,7,1632
16950000 1632 WRITE(KLP,1633)
16960000 1633 FORMAT(' ----   INCORRECT INDEX,CYCLE OR DOMAIN SPECIFICATION')
16970000       GO TO 7
16980000 1680 IF(L(LFIRST)-5) 1690,1750,1750
16990000C
17000000C TRANSLATING THE FORMAT OF THE INSTRUCTION READ
17010000C
17020000 1690 CALL DFGC(IB,LL,80,INDEX,KOD,ICHAR(3))
17030000       IF(KOD) 7,1700,1700
17040000 1700 CALL DFD(IB,INDEX+1,80,LF,KOD)
17050000       IF(KOD) 1111,1710,1710
17060000 1710 CALL CFNN(IB,LF,80,KOD,1,INT,RAC,LF1,IDIG)
17070000       IF(KOD) 1111,1720,1111
17080000 1720 IF(INT-5) 1730,1740,1730
17090000 1730 IF(INT-10) 1111,1740,1111
17100000 1740 L(LFIRST+1)=L(LFIRST+1)+INT
17110000       GO TO 7
17120000C
17130000C TRANSLATING THE PRINTING OPTION
17140000C
17150000 1750 CALL DFNBC(IB,LL+1,80,INDEX,KOD)
17160000       IF(KOD) 7,1760,7
17170000 1760 IF(IB(INDEX) .NE. IUSVEC(5)) GO TO 1111
17180000 1770 L(LFIRST+1)=L(LFIRST+1)+1
17190000       GO TO 7
17200000C-----------------------------------------
17210000C TRANSLATING THE ARITHMETIC INSTRUCTION (OP=6)
17220000C-----------------------------------------
17230000 1900 CALL DBR(IB,INDEX,LL-1,LF1,LL1)
17240000       CALL DKV(IB,LF1,LL1,KOPRO,IDIG,KOD)
17250000       IF(KOD) 1111,1910,1631
17260000 1910 L(LFIRST+1)=KOPRO
17270000       LF1=LL+1
17280000       LL1=80
```

```
17290000C
17300000C OPEN SUBROUTINE FOR THE INSTRUCTIONS ARITHMETIC AND BRANCH
17310000C ( IN THE LATTER CASE , ARRIVAL FROM 2050+2 )
17320000C
17330000 1919 INDEX=LL1+1
17340000      IF(LF1-INDEX) 1920,1111,1111
17350000 1920 INDEX=INDEX-1
17360000      DO 1930 INT=1,5
17370000      IF(IB(INDEX) .EQ. IOPRT(INT)) GO TO 1942
17380000 1930 CONTINUE
17390000      IF(INDEX-LF1) 1940,1940,1920
17400000C
17410000C UNARY OPERATION
17420000C
17430000 1940 L(LFIRST+4)=0
17440000 1941 CALL PUVKV(IB,LF1,LL1,IDIG,L(LFIRST+2),L(LFIRST+3),KOD)
17450000      IF(KOD) 1111,7,1631
17460000 1942 LL=INDEX
17470000 1943 LL=LL-1
17480000      IF(LL-LF1) 1940,1950,1950
17490000C ICHAR(1) = ' '
17500000 1950 IF(FLOAT(IB(LL)) .EQ. ICHAR(1)) GO TO 1943
17510000C
17520000C BINARY OPERATION
17530000C
17540000 1951 L(LFIRST+4)=INT
17550000C THE VARIABLE 'INDEX' POINTS ON THE ARITHMETIC OPERATOR
17560000      CALL PUVKV(IB,INDEX+1,LL1,IDIG,L(LFIRST+5),L(LFIRST+6),KOD)
17570000      IF(KOD) 1111,1952,1631
17580000 1952 LL1=INDEX-1
17590000      GO TO 1941
17600000C-----------------------------------------
17610000C TRANSLATING THE INSTRUCTION BRANCH (OP=7)
17620000C-----------------------------------------
17630000C ICHAR(11) = '/'
17640000 2000 CALL D2GC(IB,INDEX+3,80,LF1,LL1,KOD,ICHAR(11),ICHAR(11))
17650000      IF(KOD) 1111,2010,1111
17660000C ICHAR(3) = ,
17670000 2010 CALL D2GC(IB,LF1+1,LL1-1,LF,LL,KOD,ICHAR(3),ICHAR(3))
17680000      IF(KOD) 2060,2020,1111
17690000C
17700000C CASE OF BRANCH (TO J,K OR L DEPENDING ON THE ARITHMETIC EXPRESSION)
17710000C
17720000 2020 KOD=LFIRST+1
17730000      IND=0
17740000      DO 2045 INT=7,9
17750000      IND=IND+1
17760000      CALL DBR(IB,LF1+1,LF-1,INDEX,KOPRO)
17770000      CALL CUII(IB,INDEX,KOPRO,IDIG,L(KOD))
17780000      IF(L(KOD)) 1111,1111,2030
17790000 2030 GO TO(2035,2040,2050),IND
17800000 2035 LF1=LF
17810000      LF=LL
17820000      GO TO 2045
```

```
17830000 2040 LF1=LL
17840000      LF=LL1
17850000 2045 KOD=LFIRST+INT
17860000 2050 LF1=LL1+1
17870000      LL1=80
17880000      GO TO 1919
17890000 2060 CALL DFNBC(IB,LL1+1,80,INDEX,KOD)
17900000      IF(KOD) 1111,2080,2070
17910000C
17920000C CASE OF UNCONDITIONAL AND COMPUTED BRANCHS ( TO I , KI OR K(KI) )
17930000C
17940000 2070 L(LFIRST+4)=6
17950000      LF1=LF1+1
17960000      LL1=LL1-1
17970000 2071 CALL DFNBC(IB,LF1,LL1,INDEX,KOD)
17980000      IF(KOD) 1111,2072,1111
17990000 2072 IF(IB(INDEX) .EQ. IUSVEC(1)) GO TO 1941
18000000 2073 IF(IB(INDEX)-FLOAT(IDIG(1))) 1111,1941,2074
18010000 2074 IF(IB(INDEX)-FLOAT(IDIG(10))) 1941,1941,1111
18020000C
18030000C CASE OF LOOP ( N , KEND , DELTA )
18040000C
18050000 2080 CALL DBR(IB,LF1+1,LL1-1,INT,IND)
18060000      CALL CUII(IB,INT,IND,IDIG,L(LFIRST+1))
18070000      IF(L(LFIRST+1)) 1111,1111,2090
18080000 2090 LF1=INDEX
18090000      LL1=80
18100000      INT=7
18110000C ICHAR(3) = ','
18120000      CALL DFGC(IB,LF1+1,79,INDEX,KOD,ICHAR(3))
18130000      IF(KOD) 1111,1951,1111
18140000C-----------------------------------------
18150000C TRANSLATING THE INSTRUCTION INPUT (OP=9)
18160000C-----------------------------------------
18170000 2200 CALL DBR(IB,INDEX,80,LF1,LL1)
18180000      LL1=LL1-3
18190000      DO 2210 KOD=1,3
18200000      LL=LL1+KOD
18210000      IF(IB(LL) .NE. IELC(KOD)) GO TO 2220
18220000 2210 CONTINUE
18230000      L(LFIRST+1)=0
18240000      GO TO 7
18250000 2220 DO 2230 KOD=1,3
18260000      LL=LL1+KOD
18270000      IF(IB(LL) .NE. ICAS(KOD)) GO TO 1111
18280000 2230 CONTINUE
18290000      L(LFIRST+1)=1
18300000      GO TO 7
18310000C-----------------------------------------------
18320000C TRANSLATING THE INSTRUCTION EFFECTIVENESS (OP=10)
18330000C-----------------------------------------------
18340000C ICHAR(6) = '('
18350000C ICHAR(7) = ')'
18360000 2400 CALL DFGC(IB,INDEX+3,80,LF,KOD,ICHAR(6))
```

```
18370000       CALL DFGC(IB,INDEX+3,80,LL,KOPRO,ICHAR(7))
18380000       IF(KOD+KOPRO+1) 2410,1111,2420
18390000 2410 L(LFIRST+1)=-1
18400000       L(LFIRST+2)=1
18410000       GO TO 7
18420000 2420 IF(LF-LL) 2425,1111,1111
18430000C ICHAR(3) = ',
18440000 2425 CALL DFGC(IB,LF+1,LL-1,INDEX,KOD,ICHAR(3))
18450000       IF(KOD) 2450,2430,2430
18460000 2430 CALL DBR(IB,LF+1,INDEX-1,LF1,LL1)
18470000       CALL CUII(IB,LF1,LL1,IDIG,INT)
18480000       IF(INT) 1111,1111,2435
18490000 2435 CALL DBR(IB,INDEX+1,LL-1,LF1,LL1)
18500000       CALL CUII(IB,LF1,LL1,IDIG,IND)
18510000       IF(INT-IND) 2440,2440,1111
18520000 2440 L(LFIRST+1)=INT
18530000       L(LFIRST+2)=IND
18540000       GO TO 7
18550000 2450 CALL DFNBC(IB,LF+1,LL-1,INDEX,KOD)
18560000       IF(KOD) 1111,2455,1111
18570000 2455 IF(IB(INDEX) .NE. IELC(1)) GO TO 2470
18580000 2460 L(LFIRST+1)=-1
18590000       L(LFIRST+2)=2
18600000       GO TO 7
18610000 2470 IF(IB(INDEX) .NE. IS) GO TO 1111
18620000 2480 L(LFIRST+1)=-1
18630000       L(LFIRST+2)=3
18640000C PREPARING THE EXIT OUT OF THE SUBROUTINE
18650000    7 IPROG=2
18660000       IENT=2
18670000       RETURN
18680000 1111 IPROG=2
18690000       IENT=3
18700000       RETURN
18710000C STANDARDS FOR FILE MANAGEMENT
18720000    1 FORMAT(I10)
18730000    2 CALL EXIT
18740000       RETURN
18750000       END
18760000C
18770000 ------SUBROUTINE TRAN3(ICHAR,IDIG,IUSVEC)----------------------------
18780000       DIMENSION ICHAR(11),IDIG(10),IUSVEC(5)
18790000       DIMENSION IPRI(3)
18800000       DIMENSION IB(80)
18810000       COMMON VIRTUA(1230),RR(20)
18820000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
18830000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
18840000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
18850000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
18860000       COMMON LFIRST,INDEX,LL,IENT
18870000       COMMON NBOPTI
18880000       COMMON IBB(401)
18890000       EQUIVALENCE (IBB(21),IB(1))
18900000       DATA IS/'S'/
```

```
18910000       DATA IH/'H'/
18920000       DATA IE/'E'/
18930000       DATA IP/'P'/
18940000       DATA ID/'D'/
18950000       DATA IAA/'A'/
18960000       DATA IL/'L'/
18970000       DATA IPRI/'P','R','I'/
18980000       DATA IO/'O'/
18990000C
19000000C TRANSLATING THE INSTRUCTIONS
19010000C OUTPUT , SENSITIVITY , DISPLAY ,DATA ,
19020000C OPTIMIZE AND ALLOCATE
19030000C ICHAR = VECTOR OF SPECIAL CHARACTERS
19040000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
19050000C IUSVEC = VECTOR CONTAINING THE NAMES OF THE VECTORS
19060000C WHICH MAY BE ADDRESSED TO BY THE PROGRAMMER
19070000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
19080000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
19090000C
19100000       GO TO(3000,3300,3500,3800,5000,6000),IENT
19110000C-------------------------------------------
19120000C TRANSLATING THE INSTRUCTION OUTPUT (OP=11)
19130000C-------------------------------------------
19140000 3000 CALL D2GC(IB,INDEX+3,80,LF1,LL1,KOD,ICHAR(6),ICHAR(7))
19150000       IF(KOD) 1111,3010,1111
19160000 3010 INT=0
19170000       CALL DBR(IB,LF1+1,LL1-1,LF,LL)
19180000       KOD=LL-LF
19190000       IF(KOD) 1111,3020,3040
19200000 3020 IF(IB(LF) .NE. IE) GO TO 1111
19210000       INT=INT+1100
19220000       GO TO 3110
19230000 3040 IF(KOD-1) 3050,3050,1111
19240000 3050 IF(IB(LF) .NE. IE) GO TO 3080
19250000       IF(IB(LL) .NE. IE) GO TO 1111
19260000       INT=INT+1000
19270000       GO TO 3110
19280000 3080 IF(IB(LF) .NE. IS) GO TO 1111
19290000       IF(IB(LL) .NE. IE) GO TO 1111
19300000       INT=INT+100
19310000 3110 CALL DFNBC(IB,LL1+1,80,INDEX,KOD)
19320000       IF(KOD) 3195,3120,3195
19330000 3120 LF1=0
19340000       LL1=0
19350000C DETERMINING THE OPTIONS' LIST
19360000       DO 3180 LF=1,2
19370000       IF(IB(INDEX) .NE. IS) GO TO 3140
19380000       LF1=10
19390000       GO TO 3160
19400000 3140 IF(IB(INDEX) .NE. IH) GO TO 1111
19410000 3150 LL1=1
19420000 3160 GO TO(3165,3190),LF
19430000 3165 CALL DFGC(IB,INDEX+1,80,LL,KOD,ICHAR(3))
19440000       IF(KOD) 3190,3170,3190
```

```
19450000 3170 CALL DFNBC(IB,LL+1,80,INDEX,KOD)
19460000      IF(KOD) 1111,3180,1111
19470000 3180 CONTINUE
19480000 3190 INT=INT+LF1+LL1
19490000 3195 L(LFIRST+1)=INT
19500000      GO TO 7
19510000C-----------------------------------------------
19520000C TRANSLATING THE INSTRUCTION SENSITIVITY (OP=12)
19530000C-----------------------------------------------
19540000 3300 CALL D2GC(IB,INDEX+3,80,LF,LL,KOD,ICHAR(6),ICHAR(7))
19550000      IF(KOD) 1111,3301,1111
19560000C CASE OF COMPUTING THE SENSITIVITY FOR ALL NODES
19570000 3301 CALL DBR(IB,LF+1,LL-1,LF1,LL1)
19580000      IF(LL1-LF1-2) 3310,3302,3310
19590000 3302 IF(IB(LF1) .NE. IAA) GO TO 3310
19600000      IF(IB(LF1+1) .NE. IL) GO TO 1111
19610000      IF(IB(LL1) .NE. IL) GO TO 1111
19620000      L(LFIRST+1)=0
19630000      L(LFIRST+2)=0
19640000      GO TO 3370
19650000C CASE OF COMPUTING THE SENSITIVITY FOR THE SPECIFIED NODES
19660000 3310 CALL DFGC(IB,LF+1,LL-1,INDEX,KOD,ICHAR(3))
19670000      IF(KOD) 1111,3331,1111
19680000 3331 DO 3360 IND=1,2
19690000      KOPRO=LFIRST+IND
19700000      CALL DBR(IB,LF+1,INDEX-1,LF1,LL1)
19710000      IF(IB(LF1) .EQ. IUSVEC(1)) GO TO 3340
19720000 3335 CALL CUII(IB,LF1,LL1,IDIG,INT)
19730000      IF(INT) 1111,1111,3355
19740000 3340 CALL CUII(IB,LF1+1,LL1,IDIG,INT)
19750000      IF(INT) 1111,1111,3345
19760000 3345 IF(FLOAT(INT)-IVDIM(1)) 3350,3350,1631
19770000 3350 INT=-INT
19780000 3355 L(KOPRO)=INT
19790000      LF=INDEX
19800000      INDEX=LL
19810000 3360 CONTINUE
19820000      DO 3365 IND=1,2
19830000      KOPRO=LFIRST+IND
19840000      IF(L(KOPRO)) 3370,3370,3365
19850000 3365 CONTINUE
19860000      IF(L(LFIRST+2) .LT. L(LFIRST+1)) GO TO 1631
19870000C DETECTING THE OPTION
19880000 3370 LF1=0
19890000      LL1=0
19900000      CALL DFNBC(IB,LL+1,80,INDEX,KOD)
19910000      IF(KOD) 3490,3390,3490
19920000 3390 DO 3450 LF=1,2
19930000      IF(IB(INDEX) .NE. IP) GO TO 3410
19940000      LF1=10
19950000      GO TO 3430
19960000 3410 IF(IB(INDEX) .NE. ID) GO TO 1111
19970000      LL1=1
19980000 3430 GO TO(3435,3490),LF
```

```
19990000 3435 CALL DFGC(IB,INDEX+1,80,LL,KOD,ICHAR(3))
20000000      IF(KOD) 3490,3440,3490
20010000 3440 CALL DFNBC(IB,LL+1,80,INDEX,KOD)
20020000      IF(KOD) 1111,3450,1111
20030000 3450 CONTINUE
20040000 3490 L(LFIRST+3)=LF1+LL1
20050000      GO TO 7
20060000C-----------------------------------------
20070000C TRANSLATING THE INSTRUCTION DISPLAY (OP=13)
20080000C-----------------------------------------
20090000 3500 CALL D2GC(IB,INDEX+3,80,LF,LL,KOD,ICHAR(6),ICHAR(7))
20100000      IF(KOD) 1111,3510,1111
20110000 3510 CALL DBR(IB,LF+1,LL-1,LF1,LL1)
20120000      IF(LF1) 1111,1111,3520
20130000 3520 IF(IB(LF1) .NE. IE) GO TO 3530
20140000 3525 L(LFIRST+1)=1
20150000      GO TO 3550
20160000 3530 IF(IB(LF1) .NE. IS) GO TO 3542
20170000 3535 L(LFIRST+1)=KOD+2
20180000      IF(IB(LL1)-FLOAT(IDIG(2))) 1111,3550,3540
20190000 3540 IF(IB(LL1)-FLOAT(IDIG(3))) 3541,3541,1111
20200000 3541 L(LFIRST+1)=KOD+3
20210000      GO TO 3550
20220000 3542 IF(IB(LF1) .NE. IO) GO TO 1111
20230000 3543 KOD=2
20240000      GO TO 3535
20250000 3550 CALL DBR(IB,LL+1,80,LF1,LL1)
20260000      IF(LF1) 3560,3555,3560
20270000 3555 L(LFIRST+2)=0
20280000      GO TO 7
20290000 3560 IF(IB(LF1)-IUSVEC(1)) 3585,3570,3585
20300000 3570 CALL CUII(IB,LF1+1,LL1,IDIG,INT)
20310000      IF(INT) 1111,1111,3575
20320000 3575 IF(INT-IVDIM(1)) 3580,3580,1111
20330000 3580 L(LFIRST+2)=-INT
20340000      GO TO 7
20350000 3585 CALL CUII(IB,LF1,LL1,IDIG,INT)
20360000      IF(INT) 1111,1111,3588
20370000 3588 IF(L(LFIRST+1)-1) 3592,3592,3589
20380000 3589 IF(L(LFIRST+1)-3) 3590,3590,3592
20390000 3590 IF(INT-7) 3592,3592,1631
20400000 3592 L(LFIRST+2)=INT
20410000      GO TO 7
20420000C-------------------------------------------------------------
20430000C TRANSLATING THE INSTRUCTION DATA (OP=14) AND OPTIMIZE (OP=15)
20440000C-------------------------------------------------------------
20450000 3800 KOPRO=LFIRST+1
20460000C TRANSLATING THE NUMBERS OF THE BLOCKS FOR BOTH INSTRUCTIONS
20470000      CALL D2GC(IB,INDEX+1,80,LF,LL,KOD,ICHAR(6),ICHAR(7))
20480000      IF(KOD) 1111,3810,3810
20490000 3810 CALL DBR(IB,LF+1,LL-1,LF1,LL1)
20500000      IF(LF1) 1111,1111,3811
20510000 3811 IF(IB(LF1) .EQ. IUSVEC(1)) GO TO 3830
20520000 3820 CALL CUII(IB,LF1,LL1,IDIG,INT)
```

```
20530000       IF(INT) 1111,1111,3825
20540000 3825 IF(L(LFIRST)-14) 3826,3826,3827
20550000 3826 IF(INT-NBINP) 3845,3845,1631
20560000 3827 IF(INT-2*NBINP+1) 3845,3845,1631
20570000 3830 CALL CUII(IB,LF1+1,LL1,IDIG,INT)
20580000       IF(INT) 1111,1111,3835
20590000 3835 IF(INT .GT. IVDIM(1)) GO TO 1631
20600000 3840 INT=-INT
20610000 3845 L(KOPRO)=INT
20620000       IF(L(LFIRST)-14) 3854,3854,3846
20630000C TRANSLATING THE LIMIT (OPTION 'LIMIT')
20640000C FOR THE INSTRUCTION OPTIMIZE ONLY
20650000 3846 CALL D2GC(IB,LL+1,80,LF1,LL1,KOD,ICHAR(6),ICHAR(7))
20660000       IF(KOD) 3850,3847,3847
20670000 3847 LL=LL1
20680000       LF=LF1
20690000       CALL DBR(IB,LF+1,LL-1,LF1,LL1)
20700000       IF(LF1) 1111,1111,3841
20710000 3841 IF(IB(LF1) .EQ. IUSVEC(1)) GO TO 3852
20720000 3848 CALL CUII(IB,LF1,LL1,IDIG,INT)
20730000       IF(INT) 1111,1111,3849
20740000 3849 IF(INT-NBOPTI) 3851,3851,3850
20750000 3850 INT=NBOPTI
20760000 3851 L(LFIRST+3)=INT
20770000       GO TO 3854
20780000 3852 CALL CUII(IB,LF1+1,LL1,IDIG,INT)
20790000       IF(INT) 1111,1111,3853
20800000 3853 IF(INT .GT. IVDIM(1)) GO TO 1631
20810000 3856 INT=-INT
20820000       GO TO 3851
20830000C TRANSLATING THE OPTION 'PRINT' FOR BOTH INSTRUCTIONS
20840000 3854 CALL DFGC(IB,LL+1,80,LF,KOD,ICHAR(3))
20850000       IF(KOD) 3855,3860,3860
20860000 3855 L(LFIRST+2)=0
20870000       GO TO 7
20880000 3860 CALL DFNBC(IB,LF+1,78,INDEX,KOD)
20890000       IF(KOD) 1111,3870,1111
20900000 3870 DO 3880 IND=1,3
20910000       LF=INDEX+IND-1
20920000       IF(IB(LF) .NE. IPRI(IND)) GO TO 1111
20930000 3880 CONTINUE
20940000       L(LFIRST+2)=1
20950000       GO TO 7
20960000C-------------------------------------------------
20970000C TRANSLATING THE INSTRUCTION PRESENT VALUE (OP=16)
20980000C-------------------------------------------------
20990000 5000 CALL DFGC(IB,INDEX+3,78,LF,KOD,ICHAR(8))
21000000       IF(KOD) 1111,5010,1111
21010000 5010 CALL DFW(IB,LF+1,80,LF1,LL1,KOD)
21020000       IF(KOD) 1111,5020,5020
21030000 5020 CALL DKV(IB,LF1,LL1,L(LFIRST+1),IDIG,KOD)
21040000       IF(KOD) 1111,7,1631
21050000C-----------------------------------------------
21060000C TRANSLATING THE INSTRUCTION ALLOCATE (OP=17)
```

```
21070000C-------------------------------------------------
21080000 6000 CALL D2GC(IB,INDEX+3,80,LF,LL,KOD,ICHAR(6),ICHAR(7))
21090000      IF(KOD) 1111,6010,1111
21100000 6010 CALL D2GC(IB,LL+1,80,LF1,LL1,KOD,ICHAR(6),ICHAR(7))
21110000      IF(KOD)6020,6060,6020
21120000 6020 LF1=LF
21130000      LL1=LL
21140000      LL=LF1-1
21150000      CALL DFD(IB,INDEX+3,LL,LF,KOD)
21160000      IF(KOD) 6030,6040,6030
21170000C CASE IN WHICH NO INPUT RESOURCES ARE SPECIFIED
21180000 6030 L(LFIRST+1)=0
21190000      L(LFIRST+2)=0
21200000      GO TO 6100
21210000C TESTING EITHER VARIABLE OR CONSTANT
21220000 6040 DO 6050 KOD=1,5
21230000      IF(IB(LF-1) .EQ. IUSVEC(KOD)) GO TO 6060
21240000 6050 CONTINUE
21250000C CASE OF CONSTANT
21260000      GO TO 6090
21270000C CASE OF VARIABLE
21280000 6060 LF=LF-1
21290000C PACKING
21300000 6090 CALL PUVKV(IB,LF,LL,IDIG,L(LFIRST+1),L(LFIRST+2),KOD)
21310000      IF(KOD)1111,6100,1111
21320000C DETERMINING THE SUBSYSTEM
21330000 6100 CALL D2GC(IB,LF1+1,LL1-1,LF,LL,KOD,ICHAR(8),ICHAR(3))
21340000      IF(KOD)1111,6110,1111
21350000 6110 CALL DBR(IB,LF+1,LL-1,LF1,LL1)
21360000      IF(LF1) 1111,1111,6120
21370000 6120 IF(IB(LF1) .NE. IUSVEC(1)) GO TO 6140
21380000 6130 LF1=LF1+1
21390000      KOD=1
21400000 6140 CALL CUII(IB,LF1,LL1,IDIG,INDEX)
21410000      IF(INDEX)1111,1111,6150
21420000 6150 IF(KOD)6170,6170,6160
21430000 6160 INDEX=-INDEX
21440000 6170 L(LFIRST+3)=INDEX
21450000C DETERMINING THE RESULT
21460000      CALL D2GC(IB,LL+1,80,LF1,LL1,KOD,ICHAR(8),ICHAR(7))
21470000      IF(KOD) 1111,6180,1111
21480000 6180 CALL DBR(IB,LF1+1,LL1-1,LF,LL)
21490000      IF(LF)1111,1111,6190
21500000 6190 IF(IB(LF) .NE. IUSVEC(5)) GO TO 6210
21510000C CASE OF 'COST' OPTION
21520000 6200 INDEX=1
21530000      GO TO 6222
21540000 6210 IF(IB(LF) .NE. IP) GO TO 1111
21550000C CASE OF 'PARAMETER' OPTION
21560000 6220 INDEX=2
21570000C TESTING IF THE 'PUNCH' OPTION IS USED
21580000 6222 CALL DFNBC(IB,LL1+1,80,LF,KOD)
21590000      IF(KOD) 1111,6230,6240
21600000 6230 IF(IB(LF) .NE. IP) GO TO 1111
```

```
21610000 6235 L(LFIRST+4)=-INDEX
21620000      GO TO 7
21630000 6240 L(LFIRST+4)=INDEX
21640000C PREPARING THE EXIT OUT OF THE SUBROUTINE
21650000    7 IPROG=2
21660000      IENT=2
21670000      RETURN
21680000 1111 IPROG=2
21690000      IENT=3
21700000      RETURN
21710000 1631 L(LFIRST)=0
21720000      IF(LIST1) 1632,7,1632
21730000 1632 WRITE(KLP,1633)
21740000 1633 FORMAT(' ----   INCORRECT INDEX,CYCLE OR DOMAIN SPECIFICATION')
21750000      GO TO 7
21760000C STANDARDS FOR FILE MANAGEMENT
21770000    1 FORMAT(I10)
21780000    2 CALL EXIT
21790000      RETURN
21800000      END
21810000C
21820000      SUBROUTINE REAL1(IDIG,IUSVEC)
21830000      DIMENSION IDIG(1),IUSVEC(5)
21840000      DIMENSION X(120),E(210),C(120)
21850000      DIMENSION IB(80)
21860000      COMMON VIRTUA(1230),RR(20)
21870000      COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
21880000     *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
21890000      COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
21900000     *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
21910000      COMMON LFIRST,INDEX,LL,IENT
21920000      COMMON NBOPTI
21930000      COMMON IBB(101)
21940000      EQUIVALENCE (VIRTUA(781),X(1)),(VIRTUA(901),E(1))
21950000      EQUIVALENCE (VIRTUA(1111),C(1))
21960000      EQUIVALENCE (IBB(21),IB(1))
21970000      DATA IBL/' '/
21980000C
21990000C EXECUTING THE INSTRUCTIONS
22000000C EXIT ,TITLE , REPEAT , READ AND PRINT
22010000C IDIG = VECTOR CONTAINING THE TEN DIGITS : 0 , 1 , ... , 9
22020000C IUSVEC = VECTOR CONTAINING THE NAMES OF THE VECTORS
22030000C WHICH MAY BE ADDRESSED TO BY THE PROGRAMMER
22040000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
22050000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
22060000C
22070000      GO TO(100,10,20),IENT
22080000  100 IAR=0
22090000   10 IAR=IAR+1
22100000   20 IF(IAR-1) 30,70,25
22110000   25 IF(IAR-NIN) 70,70,30
22120000   30 WRITE(KLP,40)
22130000   40 FORMAT(' ----   TERMINATION BY JUMP OUT OF INSEL PROGRAM')
22140000C**************
```

```
22150000       IPROG=1
22160000       RETURN
22170000C**************
22180000    70 LFIRST=K(IAR)
22190000       KOP=L(LFIRST)
22200000C BRANCH TO THE PARTICULAR INSTRUCTION'S TRANSLATIONS
22210000       GO TO(200,400,500,530,530,1500,1700,2000,2100,2700,3000,3300,
22220000      *3500,3800,4000,5000,6000),KOP
22230000C-------------------------------
22240000C EXECUTING THE INSTRUCTION STOP
22250000C-------------------------------
22260000   200 WRITE(KLP,210)
22270000   210 FORMAT(/' END OF SEL PROGRAM EXECUTION'/)
22280000C**************
22290000       IPROG=1
22300000       RETURN
22310000C**************
22320000C-------------------------------
22330000C EXECUTING THE INSTRUCTION TITLE
22340000C-------------------------------
22350000   400 KOD=L(LFIRST+1)
22360000       LF1=L(LFIRST+2)
22370000       LL1=L(LFIRST+3)
22380000       IF(LF1) 450,410,450
22390000   410 READ(KCR,411) IB
22400000   411 FORMAT(80A1)
22410000   412 GO TO (441,420,430),KOD
22420000   420 WRITE(KLP,422) IB
22430000   422 FORMAT(1H0,80A1)
22440000       GO TO 432
22450000   430 WRITE(KLP,431) IB
22460000   431 FORMAT(1H1,80A1/)
22470000   432 LL1=LL1-1
22480000       IF(LL1) 10,10,440
22490000   440 READ(KCR,411) IB
22500000   441 WRITE(KLP,445) IB
22510000   445 FORMAT(1X,80A1)
22520000       GO TO 432
22530000   450 LL1=LL1-LF1+1
22540000       READ(42'LF1,3,ERR=2)(IB(IND),IND=1,LL1)
22550000       LF1=LL1+1
22560000       DO 460 IND=LF1,80
22570000   460 IB(IND)=IBL
22580000       LL1=1
22590000       GO TO 412
22600000C-------------------------------
22610000C EXECUTING THE INSTRUCTION REPEAT
22620000C-------------------------------
22630000   500 IF(L(LFIRST+3)) 510,510,520
22640000   510 L(LFIRST+3)=L(LFIRST+2)
22650000       GO TO 10
22660000   520 L(LFIRST+3)=L(LFIRST+3)-1
22670000       IAR=L(LFIRST+1)
22680000       GO TO 20
```

```
22690000C-----------------------------------------
22700000C EXECUTING THE INSTRUCTIONS READ AND PRINT
22710000C-----------------------------------------
22720000   530 KOD=L(LFIRST+1)/100
22730000       LF1=L(LFIRST+2)
22740000       LL1=L(LFIRST+3)
22750000       KOP=L(LFIRST+1)-100*KOD
22760000       IF(LF1) 532,533,533
22770000   532 LF1=-LF1
22780000       LF1=II(LF1)
22790000       IF(LF1) 538,538,533
22800000   533 IF(LL1) 535,536,536
22810000   535 LL1=-LL1
22820000       LL1=II(LL1)
22830000   536 IF(LL1-LF1) 538,540,540
22840000   538 WRITE(KLP,539)
22850000   539 FORMAT(' ----   INCORRECT READ/PRINT'/
22860000      *6X,'EXECUTION SUPPRESSED'/)
22870000C**********
22880000       IPROG=1
22890000       RETURN
22900000C**********
22910000   540 IF(LL1-IVDIM(KOD)) 542,542,538
22920000   542 IF(L(LFIRST)-4) 550,550,1000
22930000C-------------------------------
22940000C EXECUTING THE INSTRUCTION READ
22950000C-------------------------------
22960000   550 GO TO (560,613,640,660,680),KOD
22970000   560 IF(KOP-5) 603,606,610
22980000   603 CALL CTIR(II,RR,1,LF1,LL1,IND,IDIG)
22990000       IF(IND) 604,10,604
23000000   604 WRITE(KLP,605) IUSVEC(KOD)
23010000   605 FORMAT(' ----   INCORRECT INPUT DATA FOR VECTOR *A1/)
23020000C**********
23030000       IPROG=1
23040000       RETURN
23050000C**********
23060000   606 READ(KCR,607)(II(IND),IND=LF1,LL1)
23070000   607 FORMAT(16I5)
23080000       GO TO 10
23090000   610 READ(KCR,611)(II(IND),IND=LF1,LL1)
23100000   611 FORMAT(8I10)
23110000       GO TO 10
23120000C
23130000   613 IF(KOP-5) 620,631,633
23140000   620 CALL CTIR(II,RR,2,LF1,LL1,IND,IDIG)
23150000       IF(IND) 604,10,604
23160000   621 FORMAT(16F5.0)
23170000   623 FORMAT(8F10.0)
23180000   631 READ(KCR,621)(RR(IND),IND=LF1,LL1)
23190000       GO TO 10
23200000   633 READ(KCR,623)(RR(IND),IND=LF1,LL1)
23210000       GO TO 10
23220000C
```

```
23230000   640 IF(KOP-5) 650,651,653
23240000   650 CALL CTIR(II,X,2,LF1,LL1,IND,IDIG)
23250000       IF(IND) 604,10,604
23260000   651 READ(KCR,621)(X(IND),IND=LF1,LL1)
23270000       GO TO 10
23280000   653 READ(KCR,623)(X(IND),IND=LF1,LL1)
23290000       GO TO 10
23300000C
23310000   660 IF(KOP-5) 670,671,673
23320000   670 CALL CTIR(II,E,2,LF1,LL1,IND,IDIG)
23330000       IF(IND) 604,10,604
23340000   671 READ(KCR,621)(E(IND),IND=LF1,LL1)
23350000       GO TO 10
23360000   673 READ(KCR,623)(E(IND),IND=LF1,LL1)
23370000       GO TO 10
23380000C
23390000   680 IF(KOP-5) 690,691,693
23400000   690 CALL CTIR(II,C,2,LF1,LL1,IND,IDIG)
23410000       IF(IND) 604,10,604
23420000   691 READ(KCR,621)(C(IND),IND=LF1,LL1)
23430000       GO TO 10
23440000   693 READ(KCR,623)(C(IND),IND=LF1,LL1)
23450000       GO TO 10
23460000C---------------------------------
23470000C EXECUTING THE INSTRUCTION PRINT
23480000C---------------------------------
23490000C DETERMINING THE FORMAT
23500000  1000 INT=0
23510000       GO TO (1011,1001,1002,1003,1004),KOD
23520000  1001 CALL MINMAX(RR,LF1,LL1,RAC1,RAC2,1,0)
23530000       GO TO 1005
23540000  1002 CALL MINMAX( X,LF1,LL1,RAC1,RAC2,1,0)
23550000       GO TO 1005
23560000  1003 CALL MINMAX( E,LF1,LL1,RAC1,RAC2,1,0)
23570000       GO TO 1005
23580000  1004 CALL MINMAX( C,LF1,LL1,RAC1,RAC2,1,0)
23590000  1005 IF(RAC1-0.1) 1008,1006,1006
23600000  1006 IF(RAC2-99999.9) 1007,1007,1011
23610000  1007 INT=-1
23620000       GO TO 1011
23630000  1008 IF(RAC1-0.0001) 1011,1009,1009
23640000  1009 IF(RAC2-999.999) 1010,1010,1011
23650000  1010 INT=1
23660000  1011 IF(KOP) 1012,1260,1012
23670000C PRINTING THE VECTOR AS A COLUMN
23680000  1012 WRITE(KLP,1020) IUSVEC(KOD)
23690000  1020 FORMAT(/' INDEX'12X,A1/25('-'))
23700000       GO TO(1030,1070,1140,1180,1220),KOD
23710000  1030 WRITE(KLP,1040) (IND,II(IND),IND=LF1,LL1)
23720000  1040 FORMAT(I5,3X,I12)
23730000  1050 WRITE(KLP,1060)
23740000  1060 FORMAT(25('-')/)
23750000       GO TO 10
23760000  1070 IF(INT) 1080,1100,1120
```

```
23770000 1080 WRITE(KLP,1090)(IND,RR(IND),IND=LF1,LL1)
23780000 1090 FORMAT(I5,6X,F12.4)
23790000       GO TO 1050
23800000 1100 WRITE(KLP,1110)(IND,RR(IND),IND=LF1,LL1)
23810000 1110 FORMAT(I5,6X,E12.5)
23820000       GO TO 1050
23830000 1120 WRITE(KLP,1130)(IND,RR(IND),IND=LF1,LL1)
23840000 1130 FORMAT(I5,6X,F12.7)
23850000       GO TO 1050
23860000 1140 IF(INT) 1150,1160,1170
23870000 1150 WRITE(KLP,1090)(IND,X(IND),IND=LF1,LL1)
23880000       GO TO 1050
23890000 1160 WRITE(KLP,1110)(IND,X(IND),IND=LF1,LL1)
23900000       GO TO 1050
23910000 1170 WRITE(KLP,1130)(IND,X(IND),IND=LF1,LL1)
23920000       GO TO 1050
23930000 1180 IF(INT) 1190,1200,1210
23940000 1190 WRITE(KLP,1090)(IND,E(IND),IND=LF1,LL1)
23950000       GO TO 1050
23960000 1200 WRITE(KLP,1110)(IND,E(IND),IND=LF1,LL1)
23970000       GO TO 1050
23980000 1210 WRITE(KLP,1130)(IND,E(IND),IND=LF1,LL1)
23990000       GO TO 1050
24000000 1220 IF(INT) 1230,1240,1250
24010000 1230 WRITE(KLP,1090)(IND,C(IND),IND=LF1,LL1)
24020000       GO TO 1050
24030000 1240 WRITE(KLP,1110)(IND,C(IND),IND=LF1,LL1)
24040000       GO TO 1050
24050000 1250 WRITE(KLP,1130)(IND,C(IND),IND=LF1,LL1)
24060000       GO TO 1050
24070000C PRINTING THE VECTOR ON LINES
24080000 1260 GO TO(1270,1290,1360,1400,1440),KOD
24090000 1270 WRITE(KLP,1280)(II(IND),IND=LF1,LL1)
24100000 1280 FORMAT(1X,10I12)
24110000       GO TO 10
24120000 1290 IF(INT) 1300,1320,1340
24130000 1300 WRITE(KLP,1310)(RR(IND),IND=LF1,LL1)
24140000 1310 FORMAT(1X,10F12.4)
24150000       GO TO 10
24160000 1320 WRITE(KLP,1330)(RR(IND),IND=LF1,LL1)
24170000 1330 FORMAT(1X,10E12.5)
24180000       GO TO 10
24190000 1340 WRITE(KLP,1350)(RR(IND),IND=LF1,LL1)
24200000 1350 FORMAT(1X,10F12.7)
24210000       GO TO 10
24220000 1360 IF(INT) 1370,1380,1390
24230000 1370 WRITE(KLP,1310)( X(IND),IND=LF1,LL1)
24240000       GO TO 10
24250000 1380 WRITE(KLP,1330)( X(IND),IND=LF1,LL1)
24260000       GO TO 10
24270000 1390 WRITE(KLP,1350)( X(IND),IND=LF1,LL1)
24280000       GO TO 10
24290000 1400 IF(INT) 1410,1420,1430
24300000 1410 WRITE(KLP,1310)( E(IND),IND=LF1,LL1)
```

```
24310000       GO TO 10
24320000 1420 WRITE(KLP,1330)( E(IND),IND=LF1,LL1)
24330000       GO TO 10
24340000 1430 WRITE(KLP,1350)( E(IND),IND=LF1,LL1)
24350000       GO TO 10
24360000 1440 IF(INT) 1450,1460,1470
24370000 1450 WRITE(KLP,1310)( C(IND),IND=LF1,LL1)
24380000       GO TO 10
24390000 1460 WRITE(KLP,1330)( C(IND),IND=LF1,LL1)
24400000       GO TO 10
24410000 1470 WRITE(KLP,1350)( C(IND),IND=LF1,LL1)
24420000       GO TO 10
24430000C PREPARING THE EXIT OUT OF THE PROGRAM
24440000 1500 IPROG=6
24450000       IENT=1
24460000       RETURN
24470000 1700 IPROG=6
24480000       IENT=2
24490000       RETURN
24500000 2000 IPROG=6
24510000       IENT=3
24520000       RETURN
24530000 2100 IPROG=7
24540000       RETURN
24550000 2700 IPROG=6
24560000       IENT=5
24570000       RETURN
24580000 3000 IPROG=8
24590000       IENT=1
24600000       RETURN
24610000 3300 IPROG=6
24620000       IENT=4
24630000       RETURN
24640000 3500 IPROG=8
24650000       IENT=2
24660000       RETURN
24670000 3800 IPROG=9
24680000       IENT=1
24690000       RETURN
24700000 4000 IPROG=9
24710000       IENT=2
24720000       RETURN
24730000 5000 IPROG=10
24740000       IENT=1
24750000       RETURN
24760000 6000 IPROG=10
24770000       IENT=2
24780000       RETURN
24790000C STANDARDS FOR FILE MANAGEMENT
24800000    1 FORMAT(I10)
24810000    3 FORMAT(A10)
24820000    2 CALL EXIT
24830000       RETURN
24840000       END
```

```
24850000C
24860000 ------ SUBROUTINE REAL2(R) ----------------------------------------
24870000       DIMENSION IB(80)
24880000       DIMENSION R(1),X(120),E(210)
24890000       DIMENSION EXLINE(21),EYLINE(21)
24900000       DIMENSION XLINE(21),YLINE(21)
24910000       COMMON VIRTUA(1230),RR(20)
24920000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
24930000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
24940000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
24950000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
24960000       COMMON LFIRST,INDEX,LL,IENT
24970000       COMMON NBOPTI
24980000       COMMON IBB(101)
24990000       EQUIVALENCE (IBB(21),IB(1))
25000000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
25010000       EQUIVALENCE (VIRTUA(781),X(1)),(VIRTUA(901),E(1))
25020000       DATA EXLINE/0.,0.01,0.02,0.03,0.04,0.05,0.07,0.09,0.12,0.15,
25030000      *0.2,0.25,0.30,0.35,0.4,0.5,0.6,0.7,0.8,0.9,1.0/
25040000       DATA EPSIL/0.0001/
25050000C
25060000C EXECUTING THE INSTRUCTIONS
25070000C ARITHMETIC INSTRUCTION , BRANCH , WAIT ,
25080000C SENSITIVITY AND EFFECTIVENESS
25090000C R = VECTOR OF THE AGGREGATION EXPONENTS' VALUES
25100000C NOTE : THE COMMON ZONE ( FORTRAN ) IS THE SAME AS IN THE MAIN PROGRAM
25110000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
25120000C
25130000       GO TO(1500,1700,2000,3300,2700),IENT
25140000C-------------------------------------
25150000C EXECUTING THE ARITHMETIC INSTRUCTION
25160000C-------------------------------------
25170000 1500 LF1=1
25180000 1505 KOD=L(LFIRST+4)
25190000       CALL TOV(L(LFIRST+2),L(LFIRST+3),INT1,RAC1,IR1)
25200000       IF(KOD) 1670,1670,1510
25210000 1510 CALL TOV(L(LFIRST+5),L(LFIRST+6),INT2,RAC2,IR2)
25220000       LL1=IR1+IR2-2
25230000       IF(LL1) 1520,1520,1590
25240000 1520 GO TO (1530,1540,1550,1560,1570),KOD
25250000 1530 INT1=INT1+INT2
25260000       GO TO 1580
25270000 1540 INT1=INT1-INT2
25280000       GO TO 1580
25290000 1550 INT1=INT1*INT2
25300000       GO TO 1580
25310000 1560 INT1=INT1/INT2
25320000       GO TO 1580
25330000 1570 INT1=INT1**INT2
25340000 1580 RAC1=INT1
25350000       GO TO 1670
25360000 1590 GO TO (1610,1620,1630,1640,1650),KOD
25370000 1610 RAC1=RAC1+RAC2
25380000       GO TO 1660
```

```
25390000 1620 RAC1=RAC1-RAC2
25400000      GO TO 1660
25410000 1630 RAC1=RAC1*RAC2
25420000      GO TO 1660
25430000 1640 RAC1=RAC1/RAC2
25440000      GO TO 1660
25450000 1650 RAC1=RAC1**RAC2
25460000 1660 INT1=RAC1
25470000 1670 GO TO (1680,1720),LF1
25480000 1680 CALL INX(L(LFIRST+1),INT1,RAC1)
25490000      GO TO 10
25500000C--------------------------------
25510000C EXECUTING THE INSTRUCTION BRANCH
25520000C--------------------------------
25530000 1700 IF(L(LFIRST+4)-6) 1710,1790,1800
25540000C
25550000C EXECUTING THE CONDITIONAL  BRANCH
25560000C
25570000 1710 LF1=2
25580000      GO TO 1505
25590000 1720 IF(KOD) 1730,1730,1740
25600000 1730 GO TO (1745,1750),IR1
25610000 1740 IF(LL1) 1745,1745,1750
25620000 1745 IF(INT1) 1760,1770,1780
25630000 1750 IF(RAC1) 1754,1770,1752
25640000 1752 IF(RAC1-EPSIL ) 1770,1770,1780
25650000 1754 IF(RAC1+EPSIL) 1760,1770,1770
25660000 1760 IAR=L(LFIRST+1)
25670000      GO TO 20
25680000 1770 IAR=L(LFIRST+7)
25690000      GO TO 20
25700000 1780 IAR=L(LFIRST+8)
25710000      GO TO 20
25720000C
25730000C EXECUTING THE UNCONDITIONAL AND COMPUTED BRANCHS
25740000C
25750000 1790 CALL TOV(L(LFIRST+2),L(LFIRST+3),IAR,RAC,IR1)
25760000      IF(L(LFIRST+2)-1) 1795,20,1795
25770000 1795 CALL BINTRA(LABFIL,INDLAB,IAR,KOD)
25780000      IF(KOD) 1797,1797,1796
25790000 1796 IAR=IA(KOD)
25800000      GO TO 20
25810000 1797 WRITE(KLP,1798)
25820000 1798 FORMAT(' ----   INCORRECT BRANCH'/
25830000     *6X'EXECUTION SUPPRESSED'/)
25840000C**************
25850000      IPROG=1
25860000      RETURN
25870000C**************
25880000C
25890000C EXECUTING THE LOOP
25900000C
25910000 1800 CALL TOV(L(LFIRST+2),L(LFIRST+3),INT1,RAC1,IR1)
25920000      CALL TOV(L(LFIRST+5),L(LFIRST+6),INT2,RAC2,IR2)
```

```
25930000       KOD=L(LFIRST+1)
25940000       KOD=K(KOD)
25950000       KOD=L(KOD+1)
25960000       CALL TXV(KOD,INT,RAC,IR)
25970000       IF(IR1+IR2+IR-3) 1810,1810,1850
25980000 1810  INT=INT+INT2
25990000       IF (INT2) 1820,1815,1815
26000000 1815  IF(INT-INT1) 1830,1830,1930
26010000 1820  IF(INT-INT1) 1930,1830,1830
26020000 1830  IAR=L(LFIRST+1)+1
26030000       CALL INX(KOD,INT,RAC)
26040000       GO TO 20
26050000 1850  RAC=RAC+RAC2
26060000       INT=RAC
26070000       IF(RAC2) 1920,1915,1915
26080000 1915  IF(RAC-RAC1) 1830,1830,1930
26090000 1920  IF(RAC-RAC1) 1930,1830,1830
26100000 1930  CALL INX(KOD,INT,RAC)
26110000       GO TO 10
26120000C------------------------------
26130000C EXECUTING THE INSTRUCTION WAIT
26140000C------------------------------
26150000 2000 GO TO 10
26160000C--------------------------------------
26170000C EXECUTING THE INSTRUCTION SENSITIVITY
26180000C--------------------------------------
26190000C LF , LL = LOOP PARAMETERS OF THE INPUT NODE
26200000C INDEX = OUTPUT NODE
26210000C
26220000 3300 LF=L(LFIRST+1)
26230000       LL=NBTOTC-1
26240000       IF(LF) 3306,3305,3310
26250000 3305 LF=1
26260000       INDEX=NBTOTC
26270000       GO TO 3330
26280000 3306 LF=-LF
26290000       LF=II(LP)
26300000       IF(LF) 3313,3313,3310
26310000 3310 IF(LF-LL) 3320,3320,3313
26320000 3313 WRITE(KLP,3314)
26330000 3314 FORMAT(/' ----   DOMAIN ERROR (SENSITIVITY STATEMENT)'/)
26340000       GO TO 3317
26350000 3315 WRITE(KLP,3316)
26360000 3316 FORMAT(' ----   THE EFFECTIVENESS OF THE OUTPUT NODE IS ZERO'/
26370000      *'       EXECUTION OF SENSIVITIVITY SUPPRESSED'/)
26380000C**************
26390000 3317 IPROG=5
26400000       IENT=2
26410000       RETURN
26420000C**************
26430000 3320 LL=LF
26440000       INDEX=L(LFIRST+2)
26450000       IF(INDEX) 3321,3322,3322
26460000 3321 INDEX=-INDEX
```

```
26470000       INDEX=II(INDEX)
26480000  3322 IF(INDEX-LL) 3313,3313,3325
26490000  3325 IF(INDEX-NBELC) 3313,3313,3326
26500000  3326 IF(INDEX-NBTOTC) 3330,3330,3313
26510000C LOOP FOR COMPUTING THE EFFECTIVENESSES
26520000  3330 IF(E(INDEX)) 3331,3315,3331
26530000  3331 DO 3399 IND=LF,LL
26540000       EFEKT=E(IND)
26550000       DO 3335 IR=1,21
26560000       E(IND)=EXLINE(IR)
26570000       J=IND
26580000  3333 J=ITREE(J)
26590000       CALL NELEFF(J,R)
26600000       IF(J-INDEX) 3333,3335,3336
26610000  3335 EYLINE(IR)=E(INDEX)
26620000C RESTAURATING THE PREVIOUS EFFECTIVENESS VALUES
26630000  3336 E(IND)=EFEKT
26640000       J=IND
26650000  3343 J=ITREE(J)
26660000       CALL NELEFF(J,R)
26670000       IF(J-INDEX) 3343,3345,3313
26680000  3345 EFEKT=E(INDEX)
26690000C COMPUTING THE SENSITIVITY INDICATORS
26700000       ALFA=100.*(EYLINE(21)-EYLINE(1))
26710000       ALFAR=ALFA/EYLINE(21)
26720000       DELPL=100.*(EYLINE(21)-EFEKT)
26730000       DELMIN=100.*(EFEKT-EYLINE(1))
26740000       PI=100.*DELMIN/ALFA
26750000       BETA=(EXLINE(2)-EXLINE(1))*EYLINE(1)
26760000       DO 3350 J=2,20
26770000  3350 BETA=EYLINE(J)*(EXLINE(J+1)-EXLINE(J-1))+BETA
26780000       BETA=0.5*(BETA+EYLINE(21)*(EXLINE(21)-EXLINE(20)))
26790000       BETA=100.*(EYLINE(21)-BETA)/EYLINE(21)
26800000       BETAR=100.*BETA/ALFAR
26810000C INSCRIBING THESE RESULTS IN THE SENSITIVITY FILE
26820000       WRITE(41'IND*8-7,4) ALFA,ALFAR,BETA,BETAR,DELPL,DELMIN,PI,INDEX
26830000C PRINTING THE RESULTS
26840000       IF(L(LFIRST+3)-10) 3370,3352,3355
26850000  3352 IF(IND-LF) 3355,3355,3360
26860000C PRINTING THE RESULTS OF THE SENSITIVITY ANALYSIS
26870000  3355 WRITE(KLP,3356)
26880000  3356 FORMAT(1H1'SENSITIVITY ANALYSIS'/21('-')/)
26890000       GO TO 3362
26900000  3360 WRITE(KLP,3361)
26910000  3361 FORMAT( /' SENSITIVITY ANALYSIS'/21('-')/)
26920000  3362 READ(43'IND*20-19,3,ERR=2)(IB(J),J=1,20)
26930000       WRITE(KLP,3363) IND,(IB(J),J=1,20)
26940000  3363 FORMAT(' INPUT ='I4' = '20A2)
26950000       READ(43'INDEX*20-19,3,ERR=2)(IB(J),J=1,20)
26960000       WRITE(KLP,3365) INDEX,(IB(J),J=1,20)
26970000  3365 FORMAT(' OUTPUT='I4' = '20A2/)
26980000       DO 3367 J=1,21
26990000       XLINE(J)=100.*EXLINE(J)
27000000  3367 YLINE(J)=100.*EYLINE(J)
```

```
27010000       WRITE(KLP,3369) XLINE,YLINE
27020000 3369 FORMAT(9X' IN ='21F5.1/ 9X'OUT ='21F5.1)
27030000       DO 3372 J=2,21
27040000 3372 YLINE(J)=(EYLINE(J)-EYLINE(J-1))/(EXLINE(J)-EXLINE(J-1))+0.005
27050000       YLINE(1)=YLINE(2)
27060000       WRITE(KLP,3374) YLINE
27070000 3374 FORMAT(' D(OUT)/D(IN)='21F5.2)
27080000       WRITE(KLP,3376) ALFA,ALFAR,BETA,BETAR,DELPL,DELMIN,PI
27090000 3376 FORMAT(/2X'ALPHA   ALPHA REL   BETA   BETA REL   DELTA+   DELTA-
27100000      *  PI'/7F9.2/)
27110000C DRAWING THE FUNCTION 'E OUT' OF THE VARIABLE 'E IN'
27120000 3370 IF(L(LFIRST+3)-1) 3399,3375,3377
27130000 3375 WRITE(KLP,3379)
27140000 3379 FORMAT(1H1)
27150000 3377 IF(L(LFIRST+3)-10) 3380,3399,3380
27160000 3380 READ(43'INDEX*20-19,3,ERR=2)(IB(J),J=1,20)
27170000       WRITE(KLP,3381)(IB(J),J=1,20)
27180000 3381 FORMAT( 12X,20A2)
27190000       DO 3385 J=1,21
27200000       XLINE(J)=100.*EXLINE(J)
27210000 3385 YLINE(J)=100.*EYLINE(J)
27220000       CALL DIS100(XLINE,YLINE,IBB,21)
27230000       READ(43'IND*20-19,3,ERR=2)(IB(J),J=1,20)
27240000       WRITE(KLP,3388)(IB(J),J=1,20)
27250000 3388 FORMAT(70X,20A2)
27260000C END OF THE LOOP OF COMPUTING THE EFFECTIVENESSES
27270000 3399 CONTINUE
27280000       GO TO 10
27290000C-------------------------------------
27300000C EXECUTING THE INSTUCTION EFFECTIVENESS
27310000C-------------------------------------
27320000 2700 IPROG=L(LFIRST+1)
27330000       IENT=L(LFIRST+2)
27340000       IF(IPROG) 2800,2710,2710
27350000 2710 CALL BINTRA(NBSUB,NBTOTC,IPROG,LF1)
27360000       IF(LF1) 2720,2720,2740
27370000 2720 WRITE(KLP,2730) IAR
27380000 2730 FORMAT(/' ----   DOMAIN ERROR AT STATEMENT NO.'I5/)
27390000C**************
27400000       IPROG=1
27410000       RETURN
27420000C**************
27430000 2740 CALL BINTRA(NBSUB,NBTOTC,IENT,LL1)
27440000       IF(LL1) 2720,2720,2750
27450000 2750 IF(LF1-NBELC) 2760,2760,2790
27460000 2760 LF=LF1
27470000       IF(LL1-NBELC) 2770,2770,2780
27480000 2770 LL=LL1
27490000       LF1=0
27500000       GO TO 2840
27510000 2780 LL=NBELC
27520000       LF1=NBELC+1
27530000       GO TO 2840
27540000 2790 LF=0
```

```
27550000       GO TO 2840
27560000 2800 GO TO(2810,2820,2830),IENT
27570000 2810 LF=1
27580000       LL=NBELC
27590000       LF1=NBELC+1
27600000       LL1=NBTOTC
27610000       GO TO 2840
27620000 2820 LF=1
27630000       LL=NBELC
27640000       LF1=0
27650000       GO TO 2840
27660000 2830 LF=0
27670000       LF1=NBELC+1
27680000       LL1=NBTOTC
27690000 2840 IF(LF) 2910,2910,2850
27700000 2850 DO 2900 IENT=LF,LL
27710000 2900 CALL ELEFF(IENT,X(IENT),E(IENT))
27720000 2910 IF(LF1) 10,10,2911
27730000 2911 DO 2995 IENT=LF1,LL1
27740000 2995 CALL NELEFF(IENT,R)
27750000C
27760000C PREPARING THE EXIT OUT OF THE SUBROUTINE
27770000   10 IPROG=5
27780000       IENT=2
27790000       RETURN
27800000   20 IPROG=5
27810000       IENT=3
27820000       RETURN
27830000C STANDARDS FOR FILE MANAGEMENT
27840000    1 FORMAT(I10)
27850000    2 CALL EXIT
27860000    3 FORMAT(6X,A2)
27870000    4 FORMAT(7F10.3,I10)
27880000       RETURN
27890000       END
27900000C
27910000 ------SUBROUTINE REAL3(AGGROP) ----------------------------------------
27920000       DIMENSION AGGROP(17),IB1(5),IB2(5)
27930000       DIMENSION BKPWGH(780),X(120),E(210)
27940000       DIMENSION IB(80)
27950000       COMMON VIRTUA(1230),RR(20)
27960000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
27970000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
27980000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
27990000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
28000000       COMMON LFIRST,INDEX,LL,IENT
28010000       COMMON NBOPTI
28020000       COMMON IBB(101)
28030000       EQUIVALENCE (IBB(21),IB(1)),(IB(31),IB1(1)),(IB(36),IB2(1))
28040000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
28050000       EQUIVALENCE (VIRTUA(1),BKPWGH(1)),(VIRTUA(781),X(1))
28060000       EQUIVALENCE (VIRTUA(901),E(1))
28070000       DATA IBL/' '/
28080000       DATA ISTAR/'*'/
```

```
28090000C
28100000C EXECUTING THE INSTRUCTION INPUT
28110000C OP = VECTOR CONTAINING THE AGGREGATION COEFFICIENTS' SYMBOLS
28120000C NOTE : THE COMMON ZONE (FORTRAN) IS THE SAME AS IN THE MAIN PROGRAM
28130000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
28140000C
28150000C -------------------------------
28160000C EXECUTING THE INSTRUCTION INPUT
28170000C -------------------------------
28180000C
28190000C INPUT ELC
28200000C
28210000C READING THE PRINTING AND JUMPING CODES
28220000 2100 READ(KCR,2105) LIST1,INDEX
28230000 2105 FORMAT(A1,3X,A1)
28240000      IF(L(LFIRST+1)) 2110,2110,2300
28250000C EXECUTING THE INSTRUCTION 'INPUT ELC'
28260000 2110 IF(FLOAT(INDEX)-IBL) 2200,2111,2200
28270000 2111 INT1=0
28280000      IPOINT(1)=1
28290000      NBELC=0
28300000 2120 NBELC=NBELC+1
28310000      LF=IPOINT(NBELC)
28320000      LL=LF+4
28330000C TESTING IF OFF LIMITS
28340000      IF(NBELC-IVDIM(3)-1) 2125,2125,2122
28350000 2122 WRITE(KLP,2123)
28360000 2123 FORMAT(/' ----   SUBROUTINE EXCEEDS AVAILABLE STORAGE'
28370000     */6X'EXECUTION DISCONTINUED'/)
28380000C**********
28390000      IPROG=1
28400000      RETURN
28410000C**********
28420000 2125 READ(KCR,2130) NBSUB(NBELC),E(NX),KOD,(IB(INT),INT=1,20),
28430000     *(BKPWGH(IR),ISUB(IR),IR=LF,LL)
28440000 2130 FORMAT(I4,1X,F4.4,A1,20A2,5(F4.0,I2))
28450000      IF(FLOAT(KOD)-IBL) 2170,2135,2170
28460000 2135 M=IFIL13
28470000      WRITE(43'NBELC*20-19,3)(IB(INT),INT=1,20)
28480000      IFIL13=M+20
28490000      IF(NBELC-1) 2149,2149,2142
28500000C TESTING THE BLOCK NUMBER
28510000 2142 IF(NBSUB(NBELC)-NBSUB(NX-1)) 2145,2145,2149
28520000 2145 E(NBELC)=-1.
28530000      IPOINT(NBELC+1)=LF+1
28540000      INT1=1
28550000      GO TO 2120
28560000C TESTING THE MONOTONY OF THE ARGUMENTS
28570000 2149 LF=LF+1
28580000      DO 2155 IR=LF,LL
28590000      IF(BKPWGH(IR)-BKPWGH(IR-1)) 2150,2150,2155
28600000 2150 IF(BKPWGH(IR)) 2145,2160,2145
28610000 2155 CONTINUE
28620000      IR=LL+1
```

```
28630000 2160 IF(IR-LF) 2145,2145,2161
28640000 2161 IPOINT(NBELC+1)=IR
28650000      GO TO 2120
28660000C END OF THE READING LOOP
28670000 2170 NBELC=NBELC-1
28680000      IF(NBELC-1) 2180,2180,2190
28690000 2180 WRITE(KLP,2188)
28700000 2188 FORMAT(/' ----   INCORRECT ELC , EXECUTION SUPPRESSED'/)
28710000C**************
28720000 2189 IPROG=1
28730000      RETURN
28740000C**************
28750000C
28760000 2190 IF(NBELC-IVDIM(3)) 2191,2191,2122
28770000 2191 LL=IPOINT(NBELC+1)-1
28780000      DO 2196 IR=1,LL
28790000      IF(ISUB(IR)-99) 2196,2195,2196
28800000 2195 ISUB(IR)=100
28810000 2196 CONTINUE
28820000C
28830000 2200 IF(NBELC) 2180,2180,2210
28840000 2210 IF(LIST1-IBL) 2220,2260,2220
28850000 2220 WRITE(KLP,2221) (INT2,INT2=1,5)
28860000 2221 FORMAT(1H1,43X'ELEMENTARY CRITERIA'
28870000     */121('-')/'  NO. BLOCK  E    COMPONENT FOR EVALUATION'
28880000     *14X,5('    INPUT'I2,2X),/121('-'))
28890000      DO 2250 INT=1,NBELC
28900000      READ(43'INT*20-19,3,ERR=2)(IB(INT2),INT2=1,20)
28910000      IF(E(INT)) 2225,2235,2235
28920000 2225 WRITE(KLP,2226) INT,NBSUB(INT),(IB(INT2),INT2=1,20)
28930000 2226 FORMAT(2I5,6X,20A2)
28940000      GO TO 2250
28950000 2235 LF=IPOINT(INT)
28960000      LL=IPOINT(INT+1)-1
28970000      RAC=E(INT)*100.
28980000      IF(LF-LL) 2245,2240,2240
28990000 2240 WRITE(KLP,2241) INT,NBSUB(INT),RAC,(IB(INT2),INT2=1,20)
29000000 2241 FORMAT(2I5,F5.1,1X,20A2,5(F9.3,I4))
29010000      GO TO 2250
29020000 2245 WRITE(KLP,2241) INT,NBSUB(INT),RAC,(IB(INT2),INT2=1,20),
29030000     *(BKPWGH(IR),ISUB(IR),IR=LF,LL)
29040000 2250 CONTINUE
29050000      WRITE(KLP,2251)
29060000 2251 FORMAT(121('-'))
29070000 2260 IF(INDEX-IBL) 10,2280,10
29080000 2280 IF(INT1) 2290,10,2290
29090000 2290 IF(LIST1-IBL) 2180,2295,2180
29100000 2295 WRITE(KLP,2296)
29110000 2296 FORMAT(/' ----   LIST OF INVALID STATEMENTS'/)
29120000      INDEX=1
29130000      N=NBELC
29140000      GO TO 2493
29150000C
29160000C INPUT CAS
```

```
29170000C
29180000 2300 IF(NBELC-1) 2180,2180,2310
29190000 2310 IF(INDEX-IBL) 2411,2320,2411
29200000 2320 NBTOTC=NBELC
29210000      INT1=0
29220000C BEGINNING THE READING LOOP
29230000 2323 NBTOTC=NBTOTC+1
29240000C TESTING IF OFF LIMITS BY THE NUMBER OF NON-ELEMENTARY CRITERIA
29250000      IF(NBTOTC-IVDIM(4)-1) 2324,2324,2122
29260000 2324 LF=IPOINT(NBTOTC)
29270000      LL=LF+4
29280000C TESTING IF OFF LIMITS FOR THE VECTORS ISUB AND BKPWGH
29290000      IF(LL-IDSUBK) 2340,2340,2122
29300000 2340 READ(KCR,2341) NBSUB(NBTOTC),RAC,KOD,(IB(INT),INT=1,20),
29310000     *(ISUB(IR),BKPWGH(IR),IR=LF,LL)
29320000 2341 FORMAT(I4,1X,A4,A1,20A2,5(I4,F2.2))
29330000      IF(KOD-IBL) 2410,2342,2410
29340000 2342 IF(NBTOTC-IVDIM(4)) 2345,2345,2122
29350000 2345 M=IFIL13
29360000      WRITE(43'NBTOTC*20-19,3)(IB(INT),INT=1,20)
29370000      IFIL13=M+20
29380000C TESTING THE BLOCK NUMBER
29390000      IF(NBSUB(NBTOTC)-NBSUB(NBTOTC-1)) 2350,2350,2355
29400000C CASE OF ERROR IN THE DATA
29410000 2350 E(NBTOTC)=-1.
29420000 2351 INT1=1
29430000      IPOINT(NBTOTC+1)=LF+1
29440000      GO TO 2323
29450000 2355 E(NBTOTC)=1.
29460000C DETERMINING THE NUMBER OF ENTRIES IN THE CONSIDERED BLOCK
29470000C AND TESTING IF THE SUM OF WEIGHTS IS EQUAL TO ONE OR TWO
29480000      SUM=0.
29490000      DO 2361 INT=LF,LL
29500000      IF(BKPWGH(INT)) 2363,2363,2360
29510000 2360 IF(ISUB(INT)-NBSUB(NBTOTC)) 2361,2350,2350
29520000 2361 SUM=BKPWGH(INT)+SUM
29530000      INT=LL+1
29540000 2363 N=INT-LF
29550000      IF(SUM-1.5) 2369,2369,2364
29560000C CASE OF PARTIAL ABSORPTION OPERATOR
29570000 2364 KOD=1
29580000      N=N-2
29590000      SUM=2.-SUM
29600000      GO TO 2370
29610000 2369 KOD=0
29620000      SUM=1.-SUM
29630000 2370 IF(SUM) 2371,2372,2372
29640000 2371 SUM=-SUM
29650000 2372 IF(SUM-0.0001) 2375,2350,2350
29660000 2375 IF(N-1) 2350,2350,2380
29670000 2380 IPOINT(NBTOTC+1)=INT
29680000C DETECTING THE OPERATOR
29690000      DO 2390 INT=1,17
29700000      IF(RAC-AGGROP(INT)) 2390,2395,2390
```

```
29710000 2390 CONTINUE
29720000      E(NBTOTC)=-2.
29730000      GO TO 2351
29740000C
29750000C INITIALIZING THE VECTOR 'INDEX' ( POINTING TO
29760000C THE AGGREGATION COEFFICIENTS' VECTOR , 'R' )
29770000 2395 INT2=NBTOTC-NBELC
29780000      IF(INT2-IVDIM(4)+IVDIM(3)) 2396,2396,2122
29790000 2396 IPTRR(INT2)=(N-2)*17+INT
29800000      IF(KOD) 2398,2398,2397
29810000 2397 IPTRR(INT2)=-IPTRR(INT2)
29820000C DETERMINING THE INTERNAL ADDRESS
29830000 2398 LL=LF+N-1
29840000      N=NBTOTC-1
29850000      DO 2400 INT=LF,LL
29860000      CALL BINTRA(NBSUB,N,ISUB(INT),INDEX)
29870000      IF(INDEX) 2399,2350,2399
29880000 2399 ITREE(INDEX)=NBTOTC
29890000 2400 ISUB(INT)=INDEX
29900000      GO TO 2323
29910000C TESTING THE PRECISION AT THE END OF THE ICAS-SUBPROGRAM
29920000 2410 NBTOTC=NBTOTC-1
29930000 2411 IF(NBTOTC-NBELC) 2415,2415,2417
29940000 2415 WRITE(KLP,2416)
29950000 2416 FORMAT(/' ----   INCORRECT ICAS , EXECUTION SUPPRESSED'/)
29960000      GO TO 2189
29970000 2417 IF(LIST1-IBL) 2420,2460,2420
29980000 2420 WRITE(KLP,2421)(INT2,INT2=1,5)
29990000 2421 FORMAT(1H1,43X'NON-ELEMENTARY CRITERIA'
30000000     */121('-')/'  NO. BLOCK OP.   COMPONENT FOR EVALUATION'
30010000     *14X,5('    INPUT'I2,2X)/121('-'))
30020000      INDEX=NBELC+1
30030000      DO 2450 INT=INDEX,NBTOTC
30040000      KOD=INT-NBELC
30050000      KOD=IPTRR(KOD)
30060000      IF(KOD) 2422,2422,2423
30070000 2422 IFULL=ISTAR
30080000      KOD=-KOD
30090000      GO TO 2424
30100000 2423 IFULL=IBL
30110000 2424 KOD=KOD-(KOD/17)*17
30120000      READ(43'INT*20-19,3,ERR=2)(IB(INT2),INT2=1,20)
30130000      IF(E(INT)) 2425,2435,2435
30140000C CASE OF ERROR
30150000 2425 IF(E(INT)+1.5)2431,2431,2426
30160000C PARAMETERS OF THE PROGRESSION
30170000 2426 WRITE(KLP,2430) INT,NBSUB(INT),IFULL,AGGROP(KOD),(IB(INT2),INT2=1,
30180000     *20)
30190000 2430 FORMAT(2I5,1X,A1,A4,20A2,22X,'INCORRECT PARAMETERS')
30200000      GO TO 2450
30210000C OPERATOR OF THE PROGRESSION
30220000 2431 WRITE(KLP,2432) INT,NBSUB(INT),(IB(INT2),INT2=1,20)
30230000 2432 FORMAT(2I5,'  *** ',20A2,24X,'INCORRECT OPERATOR')
30240000      GO TO 2450
```

```
30250000 2435 LF=IPOINT(INT)
30260000      LL=IPOINT(INT+1)-1
30270000      IR2=LL-LF
30280000      N=IR2+1
30290000      DO 2441 INT2=1,N
30300000      LINDE=LF+INT2-1
30310000      INT3=ISUB(LINDE)
30320000      IF(INT3) 2437,2436,2437
30330000 2436 IB1(INT2)=0
30340000      GO TO 2441
30350000 2437 IB1(INT2)=NBSUB(INT3)
30360000 2441 IB2(INT2)=100.*BKPWGH(LINDE)+0.5
30370000 2439 FORMAT(2I5,1X,A1,A4,20A2,5(I6,1X'('I3' )'))
30380000      WRITE(KLP,2439) INT,NBSUB(INT),IFULL,AGGROP(KOD),(IB(INT2),INT2=1,
30390000     *20),(IB1(LINDE),IB2(LINDE),LINDE=1,N)
30400000 2450 CONTINUE
30410000      WRITE(KLP,2251)
30420000C FINAL TESTS
30430000 2460 IF(INDEX-IBL) 10,2480,10
30440000 2480 IF(INT1) 2490,10,2490
30450000 2490 IF(LIST1-IBL) 2415,2495,2415
30460000 2495 WRITE(KLP,2296)
30470000      INDEX=NBELC+1
30480000      N=NBTOTC
30490000 2493 DO 2499 INT=INDEX,N
30500000      IF(E(INT)) 2497,2499,2499
30510000 2497 WRITE(KLP,2498) INT,NBSUB(INT)
30520000 2498 FORMAT(I10,3X'( BLOCK'I5' )')
30530000 2499 CONTINUE
30540000      IF(L(LFIRST+1)) 2180,2180,2415
30550000C PREPARING THE EXIT OUT OF THE PROGRAM
30560000   10 IPROG=5
30570000      IENT=2
30580000      RETURN
30590000C STANDARDS FOR FILE MANAGEMENT
30600000    3 FORMAT(A10)
30610000    2 CALL EXIT
30620000      RETURN
30630000      END
30640000C
30650000 ------SUBROUTINE REAL4(R)----------------------------------------------
30660000      DIMENSION R(1),IB4(10),X(120),E(210),BKPWGH(780)
30670000      DIMENSION IB(80)
30680000      DIMENSION XSCAN(102),YSCAN(101)
30690000      DIMENSION C1(160),E1(160),C2(160),E2(160),C10PT(160),C20PT(160),
30700000     *EREZ(160),NF(31)
30710000C THE DIMENSION OF VECTOR NF IS 2*NBINP-1
30720000      COMMON VIRTUA(1230),RR(20)
30730000      COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
30740000     *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
30750000      COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
30760000     *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
30770000      COMMON LFIRST,INDEX,LL,IENT
30780000      COMMON NBOPTI
```

```
30790000       COMMON IBB(101)
30800000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
30810000       EQUIVALENCE (IBB(21),IB(1)),(IB(21),IB4(1))
30820000       EQUIVALENCE (XSCAN(1),X(1)),(XSCAN(102),YSCAN(1))
30830000       EQUIVALENCE (VIRTUA(1),BKPWGH(1)),(VIRTUA(781),X(1))
30840000       EQUIVALENCE (VIRTUA(901),E(1))
30850000       EQUIVALENCE (VIRTUA(111),C1(1)),(VIRTUA(271),E1(1)),(VIRTUA(431),
30860000      *C2(1)),(VIRTUA(591),E2(1)),(VIRTUA(751),C1OPT(1)),(VIRTUA(911),
30870000      *C2OPT(1)),(VIRTUA(1071),EREZ(1))
30880000       EQUIVALENCE (ISUB(780),NF(31))
30890000       DATA ISTAR/'*'/
30900000C
30910000C EXECUTING THE INSTRUCTIONS OUTPUT AND DISPLAY
30920000C R = VECTOR OF THE AGGREGATION EXPONENTS' VALUES
30930000C NOTE : THE COMMON ZONE (FORTRAN) IS THE SAME AS IN THE MAIN PROGRAM
30940000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
30950000C
30960000       GO TO(3000,3500),IENT
30970000C--------------------------------
30980000C EXECUTING THE INSTRUCTION OUTPUT
30990000C--------------------------------
31000000 3000 IR2=L(LFIRST+1)/100
31010000       IR1=L(LFIRST+1)-100*IR2
31020000       IR=IR1/10
31030000       IR1=IR1-10*IR
31040000       NELBUF=0
31050000       WRITE(50'1,1)(ISUB(IND),IND=1,IDSUBK)
31060000       WRITE(51'1,4) (BKPWGH(IND),IND=1,IDSUBK)
31070000       IF(IR2-10) 3180,3010,3010
31080000C
31090000C ELEMENTARY EFFECTIVENESS
31100000C
31110000 3010 WRITE(KLP,3020)
31120000 3020 FORMAT('1ELEMENTARY EFFECTIVENESSES')
31130000       IF(IR) 3030,3030,3060
31140000 3030 WRITE(KLP,3031)
31150000 3031 FORMAT(/'  NO. BLOCK  COMPONENT FOR EVALUATION'
31160000      *20X'X'8X'E'/70('-'))
31170000       DO 3040 J=1,NBELC
31180000       READ(43'J*20-19,3,ERR=2)(IB(IND),IND=1,20)
31190000       EFEKT=100.*E(J)+0.0005
31200000 3040 WRITE(KLP,3050) J,NBSUB(J),(IB(IND),IND=1,20),X(J),EFEKT
31210000 3050 FORMAT(2I5,2X,20A2,2F9.3)
31220000       WRITE(KLP,3056)
31230000 3056 FORMAT(70('-'))
31240000       IF(IR1) 3177,3177,3060
31250000 3060 WRITE(KLP,3061)
31260000 3061 FORMAT(1H+,27X'(SORTED)')
31270000       DO 3070 J=1,NBELC
31280000       INDEX=J+NELBUF
31290000       ISUB(J)=INDEX
31300000 3070 BKPWGH(J)=E(INDEX)
31310000C SORTING THE EFFECTIVENESSES
31320000       CALL SORTRI(BKPWGH,ISUB,NBELC)
```

```
31330000       IF(IR) 3100,3100,3080
31340000 3080 IF(NELBUF) 3081,3081,3083
31350000 3081 WRITE(KLP,3031)
31360000       GO TO 3085
31370000 3083 WRITE(KLP,3084)
31380000 3084 FORMAT(/'  NO. BLOCK  COMPONENT FOR EVALUATION'
31390000      *21X'E'/62('-'))
31400000 3085 DO 3090 J=1,NBELC
31410000       NUM=ISUB(J)
31420000       READ(43'NUM*20-19,3,ERR=2)(IB(IND),IND=1,20)
31430000       EFEKT=100.*E(NUM)
31440000       IF(NELBUF) 3087,3089,3087
31450000 3087 WRITE(KLP,3086) NUM,NBSUB(NUM),(IB(IND),IND=1,20),EFEKT
31460000 3086 FORMAT(2I5,2X,20A2,F10.3)
31470000       GO TO 3090
31480000 3089 WRITE(KLP,3050) NUM,NBSUB(NUM),(IB(IND),IND=1,20),
31490000      *X(NUM),EFEKT
31500000 3090 CONTINUE
31510000       IF(NELBUF) 3091,3091,3092
31520000 3091 WRITE(KLP,3056)
31530000       GO TO 3100
31540000 3092 WRITE(KLP,3093)
31550000 3093 FORMAT(62('-'))
31560000 3100 IF(IR1) 3177,3177,3110
31570000 3110 IF(NELBUF) 3111,3111,3115
31580000 3111 WRITE(KLP,3112)
31590000 3112 FORMAT('1EE HISTOGRAM'/)
31600000       GO TO 3120
31610000 3115 WRITE(KLP,3116)
31620000 3116 FORMAT('1SE HISTOGRAM'/)
31630000 3120 IND=1
31640000       DO 3155 J=10,100,10
31650000       INDEX=J-10
31660000       INT=0
31670000       EFEKT=J/100.
31680000       IF(IND-NBELC) 3125,3125,3133
31690000 3125 DO 3130 INT1=IND,NBELC
31700000       IF(BKPWGH(INT1)-EFEKT) 3130,3130,3131
31710000 3130 INT=INT+1
31720000       IND=NBELC+1
31730000       GO TO 3133
31740000 3131 IND=INT1
31750000 3133 IF(INT) 3140,3140,3143
31760000 3140 WRITE(KLP,3141) INDEX,J,INT
31770000 3141 FORMAT(I3' -'I4,I5' I'100A1)
31780000       GO TO 3150
31790000 3143 IF(INT-100) 3146,3146,3144
31800000 3144 INT1=100
31810000       GO TO 3147
31820000 3146 INT1=INT
31830000 3147 WRITE(KLP,3141) INDEX,J,INT1,(ISTAR,KOD=1,INT1)
31840000 3150 INDEX=J/10
31850000 3155 IB4(INDEX)=INT
31860000       LF1=1
```

```
31870000       DO 3175 J=10,100,10
31880000       INDEX=J-10
31890000       INT=J/10
31900000       INT=IB4(INT)
31910000       IF(INT) 3160,3175,3160
31920000 3160 WRITE(KLP,3161) INDEX,J
31930000 3161 FORMAT(/' EFFECTIVENESS FROM'I3' TO'I4/29('-'))
31940000       LL1=LF1+INT-1
31950000       DO 3170 IND=LF1,LL1
31960000       NUM=ISUB(IND)
31970000       READ(43'NUM*20-19,3,ERR=2)(IB(KOD),KOD=1,20)
31980000 3170 WRITE(KLP,3171) (IB(KOD),KOD=1,20)
31990000 3171 FORMAT(1X,20A2)
32000000       LF1=LL1+1
32010000 3175 CONTINUE
32020000C
32030000C NON-ELEMENTARY EFFECTIVENESSES
32040000C
32050000 3177 IF(IR2-10) 3180,3250,3180
32060000 3180 IF(NELBUF) 3185,3185,3182
32070000C RESTAURATING NBELC
32080000 3182 NBELC=NELBUF
32090000       GO TO 3250
32100000 3185 NELBUF=NBELC
32110000       NBELC=NBTOTC-NBELC
32120000       WRITE(KLP,3220)
32130000 3220 FORMAT('1SUBSYSTEM EFFECTIVENESSES')
32140000       IF(IR) 3230,3230,3060
32150000 3230 LL1=NELBUF+1
32160000       WRITE(KLP,3084)
32170000       DO 3240 J=LL1,NBTOTC
32180000       READ(43'J*20-19,3,ERR=2)(IB(IND),IND=1,20)
32190000       EFEKT=100.*E(J)+0.0005
32200000 3240 WRITE(KLP,3086) J,NBSUB(J),(IB(IND),IND=1,20),EFEKT
32210000       WRITE(KLP,3093)
32220000       IF(IR1) 3177,3177,3060
32230000 3250 READ(50'1,1,ERR=2)(ISUB(IND),IND=1,IDSUBK)
32240000       READ(51'1,4,ERR=2) (BKPWGH(IND),IND=1,IDSUBK)
32250000       GO TO 10
32260000C-----------------------------------
32270000C EXECUTING THE INSTRUCTION DISPLAY
32280000C-----------------------------------
32290000 3500 KOD=L(LFIRST+1)
32300000       GO TO(3505,3670,3675,3733,3733),KOD
32310000C
32320000C DISPLAYING THE ELEMENTARY EFFECTIVENESS FUNCTIONS
32330000C
32340000 3505 IF(IVDIM(3)+IVDIM(4)-202) 3538,3510,3510
32350000 3510 WRITE(51'1,4) X,E
32360000       KOD=L(LFIRST+2)
32370000       IF(KOD) 3515,3511,3525
32380000 3511 LF=1
32390000       LL=NBELC
32400000       GO TO 3545
```

```
32410000 3515 KOD=-KOD
32420000      KOD=II(KOD)
32430000      IF(KOD) 3535,3535,3525
32440000 3525 IF(KOD-NBELC) 3540,3540,3535
32450000 3535 WRITE(KLP,3536)
32460000 3536 FORMAT(/' ----    DOMAIN ERROR (DISPLAY STATEMENT)'/)
32470000C**********
32480000      IPROG=1
32490000      RETURN
32500000C**********
32510000 3538 WRITE(KLP,3539)
32520000 3539 FORMAT(/' ----    INSUFFICIENT STORAGE FOR DISPLAY EXECUTION'/)
32530000      GO TO 10
32540000 3540 LF=KOD
32550000      LL=KOD
32560000 3545 DO 3546 J=1,101
32570000 3546 XSCAN(J)=J-1
32580000      DO 3650 IND=LF,LL
32590000      READ(43'IND*20-19,3,ERR=2)(IB(J),J=1,20)
32600000      WRITE(KLP,3550)(IB(J),J=1,20)
32610000 3550 FORMAT(1H1,'EEF FOR ',20A2/)
32620000      LF1=IPOINT(IND)
32630000      LL1=IPOINT(IND+1)-1
32640000      DO 3555 J=1,6
32650000      YSCAN(J)=ISUB(LF1)
32660000 3555 YSCAN(J+95)=ISUB(LL1)
32670000      IF(BKPWGH(LF1)) 3560,3560,3570
32680000 3560 DELTA=BKPWGH(LL1)/95
32690000      DO 3565 J=2,96
32700000      XP=(J-1)*DELTA
32710000      CALL ELEFF(IND,XP,EFEKT)
32720000 3565 YSCAN(J)=100.*EFEKT
32730000      GO TO 3590
32740000 3570 DELTA=(BKPWGH(LL1)-BKPWGH(LF1))/90
32750000      DO 3575 J=7,96
32760000      XP=BKPWGH(LF1)+(J-6)*DELTA
32770000      CALL ELEFF(IND,XP,EFEKT)
32780000 3575 YSCAN(J)=100.*EFEKT
32790000 3590 CALL DIS100(XSCAN,YSCAN,IBB,101)
32800000      WRITE(KLP,3600)(BKPWGH(J),J=LF1,LL1)
32810000 3600 FORMAT(/10X'X =',5F10.3)
32820000 3650 CONTINUE
32830000C RESTAURATING THE VECTORS X AND E
32840000      READ(51'1,4,ERR=2) X,E
32850000      GO TO 10
32860000C
32870000C DISPLAYING THE SENSITIVITY TABLES
32880000C
32890000 3670 LF=1
32900000      LL=NBELC
32910000      WRITE(KLP,3671)
32920000 3671 FORMAT(1H1'ELEMENTARY CRITERIA SENSITIVITY TABLE')
32930000      GO TO 3680
32940000 3675 LF=NBELC+1
```

```
32950000       LL=NBTOTC-1
32960000       WRITE(KLP,3676)
32970000 3676 FORMAT(1H1'NON-ELEMENTARY CRITERIA SENSITIVITY TABLE')
32980000 3680 WRITE(50'1,1) ISUB
32990000       WRITE(51'1,4) BKPWGH,X
33000000       KOD=L(LFIRST+2)
33010000       IF(KOD) 3690,3682,3700
33020000 3682 INDEX=0
33030000       DO 3683 J=LF,LL
33040000       INDEX=INDEX+1
33050000 3683 ISUB(INDEX)=J
33060000       GO TO 3720
33070000 3690 KOD=-KOD
33080000       KOD=II(KOD)
33090000       IF(KOD) 3535,3535,3695
33100000 3695 IF(KOD-7) 3700,3700,3535
33110000C SORTING
33120000 3700 INDEX=0
33130000       DO 3705 J=LF,LL
33140000       READ(41'J*8-7,5,ERR=2)(X(IND),IND=1,7)
33150000       INDEX=INDEX+1
33160000       ISUB(INDEX)=J
33170000 3705 BKPWGH(INDEX)=X(KOD)
33180000       CALL SORTRI(BKPWGH,ISUB,INDEX)
33190000       WRITE(KLP,3711) KOD
33200000 3711 FORMAT(1H+,28X,'(COLUMN'I2' SORTED)')
33210000 3720 WRITE(KLP,3721)
33220000 3721 FORMAT(/' NO.  COMPONENT FOR EVALUATION'20X,
33230000      *'ALPHA ALPHA REL  BETA  BETA REL  DELTA+  DELTA-    PI  '
33240000      *'OUTPUT'/110('-'))
33250000       DO 3730 J=1,INDEX
33260000       IND=ISUB(J)
33270000       READ(43'IND*20-19,3,ERR=2)(IB(J),J=1,20)
33280000       READ(41'IND*8-7,5,ERR=2)(X(KOD),KOD=1,7),LL1
33290000 3730 WRITE(KLP,3731) IND,(IB(KOD),KOD=1,20),(X(KOD),KOD=1,7),LL1
33300000 3731 FORMAT(I4,2X,20A2,2X,7F8.2,I5)
33310000       WRITE(KLP,3732)
33320000 3732 FORMAT(110('-'))
33330000C RESTAURATING THE VECTORS ISUB , BKPWGH , AND X
33340000       READ(50'1,1,ERR=2) ISUB
33350000       READ(51'1,4,ERR=2) BKPWGH,X
33360000       GO TO 10
33370000C
33380000C DISPLAYING THE OPTIMIZATION TABLES
33390000C
33400000 3733 LL1=KOD-4
33410000       KOD=L(LFIRST+2)
33420000       IF(KOD) 3734,3736,3735
33430000 3734 KOD=-KOD
33440000       KOD=II(KOD)
33450000       IF(KOD) 3535,3535,3735
33460000 3735 IF(KOD-NBTOTC) 3737,3737,3535
33470000 3736 LF=1
33480000       LL=NBTOTC
```

```
33490000       GO TO 3738
33500000  3737 LF=KOD
33510000       LL=KOD
33520000  3738 IF(LL-2*NBINP) 3739,3535,3535
33530000  3739 WRITE(51'1,4) VIRTUA
33540000       DO 3760 IND=LF,LL
33550000       KOD=NF(IND)
33560000       LF1=3*IND
33570000       READ(93'LF1*480-479,4,ERR=2)(C1(J),J=1,KOD)
33580000       READ(93'LF1*480-319,4,ERR=2)(C2(J),J=1,KOD)
33590000       READ(93'LF1*480-159,4,ERR=2)(E1(J),J=1,KOD)
33600000       EMAX=0.
33610000       CMAX=0.
33620000       IF(IND-NBELC) 3742,3742,3740
33630000  3740 DO 3741 J=1,KOD
33640000  3741 C2(J)=C1(J)+C2(J)
33650000  3742 DO 3750 J=1,KOD
33660000       IF(LL1) 3746,3746,3743
33670000  3743 IF(C2(J)) 3744,3744,3745
33680000  3744 E1(J)=0.
33690000       GO TO 3746
33700000  3745 E1(J)=E1(J)/C2(J)
33710000  3746 IF(C2(J)-CMAX) 3748,3748,3747
33720000  3747 CMAX=C2(J)
33730000  3748 IF(E1(J)-EMAX) 3750,3750,3749
33740000  3749 EMAX=E1(J)
33750000  3750 CONTINUE
33760000C
33770000       DO 3751 J=1,KOD
33780000       C2(J)=C2(J)/CMAX*100.
33790000  3751 E1(J)=E1(J)/EMAX*100.
33800000       READ(43'IND*20-19,3,ERR=2)(IB(J),J=1,20)
33810000       IF(LL1) 3752,3752,3754
33820000  3752 WRITE(KLP,3753)(IB(J),J=1,20),EMAX
33830000  3753 FORMAT(1H1,6X'OPTIMAL EFFECTIVENESS FOR '20A2' (MAX='F8.5')'/)
33840000       GO TO 3756
33850000  3754 WRITE(KLP,3755)(IB(J),J=1,20),EMAX
33860000  3755 FORMAT(1H1,6X'OPTIMAL EFFECTIVENESS/COST FOR '
33870000      *20A2' (MAX='F9.5 ')'/)
33880000  3756 CALL DIS100(C2,E1,IBB,KOD)
33890000  3757 FORMAT(/88X'COST (MAX='F11.4')   ')
33900000  3760 WRITE(KLP,3757) CMAX
33910000       READ(51'1,4,ERR=2) VIRTUA
33920000C PREPARING THE EXIT OUT OF THE PROGRAM
33930000    10 IPROG=5
33940000       IENT=2
33950000       RETURN
33960000C STANDARDS FOR FILE MANAGEMENT
33970000     1 FORMAT(I10)
33980000     2 CALL EXIT
33990000     3 FORMAT(6X,A2)
34000000     4 FORMAT(F10.3)
34010000     5 FORMAT(7F10.3,I10)
34020000       RETURN
```

```
34030000       END
34040000C
34050000 ------SUBROUTINE REAL5(R) ------------------------------------------
34060000       DIMENSION R(1)
34070000       DIMENSION BKPWGH(780),IB(80)
34080000       DIMENSION C1(160),E1(160),C2(160),E2(160),C1OPT(160),C2OPT(160),
34090000      *EREZ(160),NF(31)
34100000C THE DIMENSION OF VECTOR NF IS 2*NBINP-1
34110000       DIMENSION RAZ(160)
34120000       COMMON VIRTUA(1230),RR(20)
34130000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
34140000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
34150000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
34160000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
34170000       COMMON LFIRST,INDEX,LL,IENT
34180000       COMMON NBOPTI
34190000       COMMON IBB(101)
34200000       EQUIVALENCE (VIRTUA(111),C1(1)),(VIRTUA(271),E1(1)),(VIRTUA(431),
34210000      *C2(1)),(VIRTUA(591),E2(1)),(VIRTUA(751),C1OPT(1)),(VIRTUA(911),
34220000      *C2OPT(1)),(VIRTUA(1071),EREZ(1))
34230000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
34240000       EQUIVALENCE (ISUB(780),NF(31))
34250000       EQUIVALENCE (ISUB(701),RAZ(160))
34260000       EQUIVALENCE (IB(1),IBB(1))
34270000       EQUIVALENCE(VIRTUA(1),BKPWGH(1))
34280000C
34290000C EXECUTING THE INSTRUCTIONS DATA AND OPTIMIZE
34300000C R = VECTOR OF THE AGGREGATION EXPONENTS' VALUES
34310000C NOTE : THE COMMON ZONE (FORTRAN) IS THE SAME AS IN THE MAIN PROGRAM
34320000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
34330000C
34340000       GO TO(3800,4000),IENT
34350000C ------------------------------
34360000C EXECUTING THE INSTRUCTION DATA
34370000C ------------------------------
34380000C TESTING THE OPPORTUNITY OF TRANSLATING THE INSTRUCTION DATA
34390000 3800 IF(NBELC-NBINP) 3810,3810,3805
34400000 3805 WRITE(KLP,3806)
34410000 3806 FORMAT(/' ----   INVALID TREE STRUCTURE'/)
34420000C**************
34430000 3808 IPROG=1
34440000       RETURN
34450000C**************
34460000 3810 IF(NBELC) 3811,3811,3813
34470000 3811 WRITE(KLP,3812)
34480000 3812 FORMAT(/' ----   ELC AND CAS NONEXISTENT'/)
34490000       GO TO 3808
34500000C THE RESERVED MEMORY FOR BKPWGH AND ISUB IS :
34510000C 5*NBINP+2*(NBINP-1)=7*NBINP-2
34520000 3813 KOD=IPOINT(NBTOTC+1)-7*NBINP+1
34530000C TESTING THE OPPORTUNITY OF INCREASING
34540000C THE MEMORY IN THE CASE OF PARTIAL ABSORPTION OPERATORS
34550000       IF(KOD) 3816,3816,3814
34560000 3814 WRITE(KLP,3807) KOD
```

```
34570000 3807 FORMAT(/' ----   'I4' ADDITIONAL WORDS REQUIRED FOR BKPWGH AND
34580000     *ISUB'/)
34590000       GO TO 3808
34600000 3816 KOD=L(LFIRST+1)
34610000       IF(KOD) 3815,3825,3825
34620000 3815 KOD=-KOD
34630000       KOD=II(KOD)
34640000       IF(KOD) 3820,3820,3825
34650000 3820 WRITE(KLP,3821)
34660000 3821 FORMAT(/' ----   DOMAIN ERROR (DATA)'/)
34670000       GO TO 3808
34680000 3825 IF(KOD-NBELC) 3827,3827,3820
34690000C  MEMORY SAVE ( OF AN EVENTUAL CONTENTS OF VECTORS K , E AND C)
34700000 3827 WRITE(51'1,4) VIRTUA
34710000 3828 READ(KCR,3830) LL
34720000 3830 FORMAT(I5)
34730000       IF(LL) 3835,3835,3832
34740000 3832 IF(LL-NBOPTI) 3840,3840,3835
34750000 3835 WRITE(KLP,3836)
34760000 3836 FORMAT(/' ----   INCORRECT INPUT DATA'/8X' EXECUTION SUPPRESSED'/
34770000C**************
34780000       IPROG=1
34790000       RETURN
34800000C**************
34810000 3840 NF(KOD)=LL
34820000C READING THE PARAMETERS
34830000       READ(KCR,3845) (C1(IND),IND=1,LL)
34840000 3845 FORMAT(8F10.0)
34850000C COMPUTING THE EFFECTIVENESSES
34860000       DO 3850 IND=1,LL
34870000 3850 CALL ELEFF(KOD,C1(IND),EREZ(IND))
34880000C READING THE AMOUNT
34890000       READ(KCR,3845) (C2(IND),IND=1,LL)
34900000       IF(L(LFIRST+2)) 3860,3880,3860
34910000C PRINTING THE READ INPUT VALUES ( OPTION 'PRINT' )
34920000 3860 READ(43'KOD*20-19,3,ERR=2) (IB(IND),IND=1,20)
34930000       WRITE(KLP,3865) (IB(IND),IND=1,20)
34940000 3865 FORMAT(/' AVAILABLE VALUES FOR '20A2/'  NO.'7X'X'11X'C'11X'E'/
34950000      ,40('-'))
34960000       DO 3870 IND=1,LL
34970000 3870 WRITE(KLP,3871) IND,C1(IND),C2(IND),EREZ(IND)
34980000 3871 FORMAT(I4,3F12.4)
34990000C TRANSFERRING THE READ INPUT VALUES
35000000C TO THE SPECIALIZED DISK FILE AND RESTAURATING THE PREVIOUS MEMORY
35010000C PARAMETERS
35020000 3880 WRITE(93'KOD*480-479,4) (C1(IND),IND=1,LL)
35030000C PRICES
35040000       WRITE(93'KOD*480-319,4) (C2(IND),IND=1,LL)
35050000C EFFECTIVENESSES
35060000       WRITE(93'KOD*480-159,4) (EREZ(IND),IND=1,LL)
35070000       READ(51'1,4,ERR=2) VIRTUA
35080000       GO TO 10
35090000C-----------------------------------
35100000C EXECUTING THE INSTRUCTION OPTIMIZE
```

```
35110000C-----------------------------------
35120000C NOTE : PREVIOUSLY MUST HAVE BEEN EXECUTED THE INSTRUCTIONS 'DATA'
35130000C DETERMINING THE BLOCK NUMBER
35140000 4000 KOD=L(LFIRST+1)
35150000      IF(KOD) 4005,4010,4020
35160000 4005 KOD=-KOD
35170000      KOD=II(KOD)
35180000      IF(KOD) 4010,4010,4020
35190000 4010 WRITE(KLP,4011)
35200000 4011 FORMAT(/' ----   DOMAIN ERROR (OPTIMIZE)'/)
35210000      GO TO 3808
35220000 4020 IF(NBELC) 3811,3811,4021
35230000 4021 IF(KOD-NBELC) 4010,4010,4025
35240000 4025 IF(KOD-NBTOTC) 4030,4030,4010
35250000C DETERMINING THE LIMITS IN THE VECTORS BKPWGH AND ISUB
35260000 4030 LF=IPOINT(KOD)
35270000      LF1=IPOINT(KOD+1)-1
35280000C TEST OF WHETHER THE STRUCTURE IS A BINARY TREE
35290000C WITH TOTAL OR PARTIAL ABSORPTION
35300000      INDEX=LF1-LF+1
35310000      IF(INDEX-2) 3805,4040,4031
35320000 4031 IF(INDEX-4) 3805,4032,3805
35330000 4032 INDEX=KOD-NBELC
35340000      INDEX=IPTRR(INDEX)
35350000      IF(INDEX) 4040,3805,3805
35360000C DETERMINING THE ADDRESS OF THE TWO FIRST BLOCKS
35370000 4040 LF1=ISUB(LF+1)
35380000      LF=ISUB(LF)
35390000C DETERMINING THE VECTOR'S LENGHT ( SIZE OF THE CEM )
35400000      LL=NF(LF)
35410000      LL1=NF(LF1)
35420000C TESTING OF WHETHER THE INPUT NODES HAVE BEEN PREVIOUSLY DEFINED
35430000      IF(LL) 4010,4010,4041
35440000 4041 IF(LL-NBOPTI) 4042,4042,4010
35450000 4042 IF(LL1) 4010,4010,4043
35460000 4043 IF(LL1-NBOPTI) 4044,4044,4010
35470000C WRITING THE VECTOR ON DISK
35480000 4044 WRITE(51'1,4) VIRTUA
35490000      J=LF
35500000      READ(93'LF*480-319,4,ERR=2) (C1(IND),IND=1,LL)
35510000      IF(J-NBELC) 4060,4060,4045
35520000 4045 READ(93'LF*480-479,4,ERR=2) (E1(IND),IND=1,LL)
35530000      DO 4050 IND=1,LL
35540000 4050 C1(IND)=E1(IND)+C1(IND)
35550000 4060 READ(93'LF*480-159,4,ERR=2) (E1(IND),IND=1,LL)
35560000      J=LF1
35570000      READ(93'LF1*480-319,4,ERR=2) (C2(IND),IND=1,LL1)
35580000      IF(J-NBELC) 4080,4080,4065
35590000 4065 READ(93'LF1*480-479,4,ERR=2)(E2(IND),IND=1,LL1)
35600000      DO 4070 IND=1,LL1
35610000 4070 C2(IND)=E2(IND)+C2(IND)
35620000 4080 READ(93'LF1*480-159,4,ERR=2)  (E2(IND),IND=1,LL1)
35630000C DETERMINING THE LIMIT
35640000      LIMIT=L(LFIRST+3)
```

```
35650000       IF(LIMIT) 4081,4082,4082
35660000 4081 LIMIT=-LIMIT
35670000       LIMIT=II(LIMIT)
35680000 4082 IF(LIMIT-4) 4010,4010,4083
35690000 4083 IF(LIMIT-NBOPTI) 4085,4085,4010
35700000C DETERMINING THE INITIAL VALUES FOR AVERAGING
35710000 4085 JFIRST=IPOINT(KOD)
35720000       INDPTR=KOD-NBELC
35730000       INDPTR=IPTRR(INDPTR)
35740000       J=INDPTR
35750000       IF(J) 4087,4087,4088
35760000 4087 J=-J
35770000       W1=BKPWGH(JFIRST+2)
35780000       W2=BKPWGH(JFIRST+3)
35790000       W3=BKPWGH(JFIRST)
35800000       W4=BKPWGH(JFIRST+1)
35810000       GO TO 4089
35820000 4088 W1=BKPWGH(JFIRST)
35830000       W2=BKPWGH(JFIRST+1)
35840000 4089 RP=R(J)
35850000C ***********************************
35860000C OPTIMIZATION ALGORITHM
35870000C ***********************************
35880000C DETERMINING THE INITIAL VALUES FOR OPTIMIZATION
35890000       UPPER=1.1*(C1(LL)+C2(LL1))
35900000       INDEX=LIMIT+1
35910000       IRED=0
35920000C BEGINNING OF THE LOOP FOR CHANGING THE LIMIT
35930000 4090 EMAX=-1.
35940000       IND1=LL+1
35950000       IND2=1
35960000C LIMITS PROCESSING ( FROM TOP TO BOTTOM )
35970000 4099 IND1=IND1-1
35980000       IF(C1(IND1)+C2(IND2)-UPPER) 4130,4130,4110
35990000 4110 IF(IND1-1) 4234,4234,4099
36000000C MOVING TO THE RIGHT
36010000 4120 IND2=IND2+1
36020000 4130 J=IND2+1
36030000       IF(J-LL1) 4135,4135,4144
36040000 4135 IF(C1(IND1)+C2(J)-UPPER) 4120,4120,4144
36050000C COMPUTING THE EFFECTIVENESS
36060000 4144 EFEK1=E1(IND1)
36070000       EFEK2=E2(IND2)
36080000C BEGINNING FOR THE CASE OF TOTAL ABSORPTION
36090000       IF(INDPTR) 4158,4158,4151
36100000 4151 IF(EFEK1) 4152,4152,4155
36110000 4152 IF(RP) 4160,4160,4153
36120000 4153 IF(EFEK2) 4160,4160,4154
36130000 4154 EFEKT=EFEK2*EXP(ALOG(W2)/RP)
36140000       GO TO 4200
36150000 4155 IF(EFEK2) 4156,4156,4161
36160000 4156 IF(RP) 4160,4160,4157
36170000 4157 EFEKT=EFEK1*EXP(ALOG(W1)/RP)
36180000       GO TO 4200
```

```
36190000C CASE OF PARTIAL ABSORPTION
36200000 4158 EFEK2=W3*EFEK1+W4*EFEK2
36210000      IF(EFEK1) 4159,4159,4161
36220000 4159 IF(RP) 4160,4160,4154
36230000C SEQUEL FOR THE CASE OF TOTAL ABSORPTION
36240000 4160 EFEKT=0.
36250000      GO TO 4200
36260000 4161 EFEKT=(EFEK1-EFEK2)*RP
36270000      IF(EFEKT) 4163,4162,4162
36280000 4162 EFEKT=EFEK2*(W1*(EFEK1/EFEK2)**RP+W2)**(1./RP)
36290000      GO TO 4200
36300000 4163 EFEKT=EFEK1*(W2*(EFEK2/EFEK1)**RP+W1)**(1./RP)
36310000C DETERMINING THE MAXIMAL EFFECTIVENESS
36320000 4200 IF(EFEKT-EMAX) 4220,4220,4210
36330000 4210 EMAX=EFEKT
36340000      I10PT=IND1
36350000      I20PT=IND2
36360000 4220 IF(IND1-1) 4234,4234,4221
36370000 4221 IF(IND2-LL1) 4222,4234,4234
36380000 4222 IND2=IND2+1
36390000      GO TO 4099
36400000C SAVING INTO MEMORY THE OBTAINED MAXIMA IN THE VECTORS EREZ,C1OPT
36410000C AND C2OPT
36420000 4234 IF(EMAX-0.0001) 4235,4235,4237
36430000 4235 I10PT=1
36440000      I20PT=1
36450000 4237 INDEX=INDEX-1
36460000      IF(INDEX) 4240,4240,4250
36470000C VECTOR'S REDUCTION
36480000 4240 IEND=LIMIT-1
36490000C COMPUTING THE INITIAL DIFFERENCES
36500000      EFEKT=EREZ(2)-EMAX
36510000      RAZ(1)=EFEKT
36520000      IF(IRED) 4244,4242,4244
36530000 4242 DO 4243 J=2,IEND
36540000 4243 RAZ(J)=EREZ(J+1)-EREZ(J-1)
36550000C TRACING THE MINIMAL DIFFERENCE
36560000 4244 IND=1
36570000      IRED=IRED+1
36580000      DO 4246 J=2,IEND
36590000      IF(RAZ(J)-EFEKT) 4245,4246,4246
36600000 4245 EFEKT= RAZ(J)
36610000      IND=J
36620000 4246 CONTINUE
36630000C ELIMINATING THE TRACED ELEMENT (IND)
36640000      IF(IND-1) 4247,4249,4247
36650000 4247 JEND=IND
36660000      DO 4248 J=2,IND
36670000      EREZ(JEND)=EREZ(JEND-1)
36680000      C1OPT(JEND)=C1OPT(JEND-1)
36690000      C2OPT(JEND)=C2OPT(JEND-1)
36700000      RAZ(JEND)=RAZ(JEND-1)
36710000 4248 JEND=JEND-1
36720000 4249 EREZ(1)=EMAX
```

```
36730000       C10PT(1)=C1(I10PT)
36740000       C20PT(1)=C2(I20PT)
36750000       INDEX=1
36760000C DIFFERENCE CORRECTION
36770000       IF(IND-1) 4255,4256,4255
36780000 4255 RAZ(IND)=EREZ(IND+1)-EREZ(IND-1)
36790000       IF(IND-IEND) 4256,4251,4256
36800000 4256 RAZ(IND+1)=EREZ(IND+2)-EREZ(IND)
36810000       GO TO 4251
36820000C CASE IF NO REDUCTION
36830000 4250 EREZ(INDEX)=EMAX
36840000       C10PT(INDEX)=C1(I10PT)
36850000       C20PT(INDEX)=C2(I20PT)
36860000C TESTING THE END OF OPTIMIZATION
36870000 4251 IF(I10PT+I20PT-2) 4333,4333,4252
36880000C DETERMINING THE NEW LIMIT
36890000 4252 UPPER=0.999*(C1(I10PT)+C2(I20PT))
36900000C ENDING THE LOOP FOR CHANGING THE LIMIT
36910000       GO TO 4090
36920000C DETERMINING THE DIMENSION OF THE RESULTING VECTOR
36930000 4333 NF(KOD)=LIMIT-INDEX+1
36940000C **********************************************
36950000C END OF THE OPTIMIZATION ALGORITHM
36960000C **********************************************
36970000C OPTION OF PRINTING THE RESULTS
36980000       IF(L(LFIRST+2)) 4340,4360,4340
36990000 4340 READ(43'KOD*20-19,3,ERR=2)(IB(IND),IND=1,20)
37000000       WRITE(KLP,4341)(IB(IND),IND=1,20)
37010000 4341 FORMAT(1H1'OPTIMIZATION FOR '20A2/' NO.'7X
37020000      *'C10PT' 7X'C20PT' 9X'C'10X'EOPT' 7X'QREL'/64('-'))
37030000       EMAX=0.
37040000       DO 4348 IND=INDEX,LIMIT
37050000       C1(IND)=C10PT(IND)+C20PT(IND)
37060000       IF(C1(IND)) 4343,4343,4344
37070000 4343 E1(IND)=0.
37080000       GO TO 4348
37090000 4344 E1(IND)=EREZ(IND)/C1(IND)
37100000       IF(E1(IND)-EMAX) 4348,4348,4347
37110000 4347 EMAX=E1(IND)
37120000 4348 CONTINUE
37130000       J=0
37140000       DO 4350 IND=INDEX,LIMIT
37150000       J=J+1
37160000       EFEKT=E1(IND)/EMAX
37170000       EFEK1=EREZ(IND)
37180000 4350 WRITE(KLP,4351) J,C10PT(IND),C20PT(IND),C1(IND),EFEK1,EFEKT
37190000 4351 FORMAT(I4,5F12.3)
37200000       IF(IRED) 4353,4360,4353
37210000 4353 WRITE(KLP,4355) IRED
37220000 4355 FORMAT(/' ----  'I3'  POINTS WERE OMITTED DUE TO THE LIMITATION')
37230000C SAVING THE RESULTS ON DISK
37240000 4360 WRITE(93'KOD*480-479,4)(C10PT(IND),IND=INDEX,LIMIT)
37250000       WRITE(93'KOD*480-319,4)(C20PT(IND),IND=INDEX,LIMIT)
37260000       WRITE(93'KOD*480-159,4)(EREZ(IND),IND=INDEX,LIMIT)
```

```
37270000       READ(51'1,4,ERR=2) VIRTUA
37280000C PREPARING THE EXIT OUT OF THE PROGRAM
37290000    10 IPROG=5
37300000       IENT=2
37310000       RETURN
37320000C STANDARDS FOR FILE MANAGEMENT
37330000     1 FORMAT(I10)
37340000     2 CALL EXIT
37350000     3 FORMAT(6X,A2)
37360000     4 FORMAT(F10.3)
37370000       RETURN
37380000       END
37390000C
37400000 ------ SUBROUTINE REAL6 ------------------------------------------------
37410000       DIMENSION CEN(160)
37420000       DIMENSION ACTUAL(31)
37430000       DIMENSION BKPWGH(780),IB(80)
37440000       DIMENSION C1(160),E1(160),C2(160),E2(160),C10PT(160),C20PT(160),
37450000      *EREZ(160),NF(31)
37460000C THE DIMENSION OF VECTOR NF IS 2*NBINP-1
37470000       DIMENSION RAZ(160)
37480000       COMMON VIRTUA(1230),RR(20)
37490000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
37500000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
37510000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
37520000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
37530000       COMMON LFIRST,INDEX,LL,IENT
37540000       COMMON NBOPTI
37550000       COMMON IBB(101)
37560000       EQUIVALENCE (VIRTUA(111),C1(1)),(VIRTUA(271),E1(1)),(VIRTUA(431),
37570000      *C2(1)),(VIRTUA(591),E2(1)),(VIRTUA(751),C10PT(1)),(VIRTUA(911),
37580000      *C20PT(1)),(VIRTUA(1071),EREZ(1))
37590000       EQUIVALENCE (NBELC,II(99)),(NBTOTC,II(100))
37600000       EQUIVALENCE (ISUB(780),NF(31))
37610000       EQUIVALENCE (ISUB(701),RAZ(160))
37620000       EQUIVALENCE (IB(1),IBB(1))
37630000       EQUIVALENCE (VIRTUA(1),BKPWGH(1))
37640000C
37650000C EXECUTING THE INSTRUCTIONS PRESENT VALUE AND ALLOCATE
37660000C  DUJMOVIC / VAN BASTELAER / ARNOTTE    JANUARY 1977
37670000C NOTE : THE COMMON ZONE (FORTRAN) IS THE SAME AS IN THE MAIN PROGRAM
37680000C
37690000       GO TO (5000,6000)  ,IENT
37700000C-------------------------------------
37710000C EXECUTING THE INSTRUCTION PRESENT VALUE
37720000C-------------------------------------
37730000 5000 WRITE(KLP,5001)
37740000 5001 FORMAT(1H1)
37750000 5002 READ(KCR,5005) CASHFL,ACTRAT,MONTHS,LFPAY,LLPAY,(IB(J),J=1,50)
37760000 5005 FORMAT(F10.0,F5.0,3I5,50A1)
37770000       IF(CASHFL) 5020,5010,5020
37780000 5010 WRITE(KLP,5011)(IB(J),J=1,50)
37790000 5011 FORMAT(1X,50A1)
37800000       GO TO 5002
```

```
5020 WRITE(KLP,5021)
5021 FORMAT(/68('-')/
     ,          12X'RATE OF INTEREST'/6X'CASH'3X'PER ONE PERIOD'3X
     ,'PERIODS'9X'TOTAL'4X'PRESENT VALUE'/6X'FLOW'4X'(PERCENTAGE)'3X
     ,'FROM  TO'8X'AMOUNT'5X'OF CASH FLOW'/68('-'))
      DO 5015 J=2,31
5015 ACTUAL(J)=0.
      ACNTOT=0.
      ACTTOT=0.
5030 RATPER=ACTRAT*MONTHS/12.
      Q=1./(1.+0.01*RATPER)
      ACTNPA=CASHFL*(1+LLPAY-LFPAY)
      ACNTOT=ACTNPA+ACNTOT
      ACTPAR=CASHFL*(Q**(LLPAY+1)-Q**LFPAY)/(Q-1.)
      ACTTOT=ACTPAR+ACTTOT
      WRITE(KLP,5040) CASHFL,RATPER,LFPAY,LLPAY,ACTNPA,ACTPAR,(IB(J),J=1
     *,50)
5040 FORMAT(1X,F12.3,F11.5,I7,I6,2F15.3,50A1)
      DO 5045 J=2,31
      Q=1./(1.+0.01*(J-1)*MONTHS/12.)
5045 ACTUAL(J)=CASHFL*(Q**(LLPAY+1)-Q**LFPAY)/(Q-1.)+ACTUAL(J)
      READ(KCR,5005) CASHFL,ACTRAT,MONTHS,LFPAY,LLPAY,(IB(J),J=1,50)
      IF(CASHFL) 5030,5050,5030
5050 J=ACTTOT
      CALL INX(L(LFIRST+1),J,ACTTOT)
      WRITE(KLP,5060) ACNTOT,ACTTOT
5060 FORMAT(68('=')/26X'T O T A L  '2F15.3/68('-')/)
      ACTUAL(1)=ACNTOT
      WRITE(KLP,5082)
      WRITE(KLP,5070)
5070 FORMAT(' RATE OF INTEREST PER YEAR'/8X'(PERCENTAGE)'11X,
     *'TOTAL PRESENT VALUE'/50('-'))
      DO 5080 J=1,31
      LLPAY=J-1
5080 WRITE(KLP,5081) LLPAY,ACTUAL(J)
      WRITE(KLP,5082)
5081 FORMAT(I15,F35.3)
5082 FORMAT(50('-'))
      GO TO 10
C-----------------------------------
C EXECUTING THE INSTRUCTION ALLOCATE
C-----------------------------------
6000 WRITE(51'1,4) VIRTUA
      NBELC1=NBELC+1
C DETERMINING THE BLOCK
      IBLOK=L(LFIRST+3)
      IF(IBLOK) 6020,6006,6040
C ERROR CASE ( NEXT INSTRUCTION )
6006 WRITE(KLP,6010)
6010 FORMAT(' ----   DOMAIN ERROR(ALLOCATE)'/)
6011 READ(51'1,4,ERR=2) VIRTUA
C**************
      IPROG=5
      IENT=2
```

```
38350000       RETURN
38360000C**************
38370000C DETERMINING THE BLOCK ON WHICH THE ALLOCATION IS MADE
38380000 6020 IBLOK=-IBLOK
38390000       IF(IBLOK-IVDIM(1)) 6030,6030,6006
38400000 6030 IBLOK=II(IBLOK)
38410000 6040 IF(IBLOK-NBELC) 6006,6006,6050
38420000 6050 IF(IBLOK-NBTOTC) 6060,6060,6006
38430000C READING THE DATA FOR THE CONSIDERED BLOCK
38440000 6060 KOLIC=NF(IBLOK)
38450000       READ(93'IBLOK*480-479,4,ERR=2) (C1OPT(I),I=1,KOLIC)
38460000       READ(93'IBLOK*480-319,4,ERR=2) (C2OPT(I),I=1,KOLIC)
38470000       READ(93'IBLOK*480-159,4,ERR=2) (EREZ(I),I=1,KOLIC)
38480000       DO 6070 I=1,KOLIC
38490000 6070 RAZ(I)=C1OPT(I)+C2OPT(I)
38500000C PUNCH OPTION
38510000       IPAR=L(LFIRST+4)
38520000       IBUSI=0
38530000       IF(IPAR) 6072,6006,6077
38540000 6072 IBUSI=1
38550000       IPAR=-IPAR
38560000C DETERMINING THE NUMBER OF AMOUNTS
38570000 6077 IF(L(LFIRST+1)) 6100,6080,6100
38580000 6080 IF(L(LFIRST+2)) 6100,6090,6100
38590000 6090 NCENA=KOLIC
38600000       IF(NCENA-1) 6006,6006,6110
38610000 6100 NCENA=1
38620000       CALL TOV(L(LFIRST+1),L(LFIRST+2),I,CENA,INDEX)
38630000C AMOUNT MODIFICATION LOOP
38640000 6110 DO 6444 ICENA=1,NCENA
38650000       DO 6120 I=1,NBTOTC
38660000 6120 VIRTUA(I)=-1.
38670000       INDEX=ICENA
38680000       IF(NCENA-1) 6140,6130,6140
38690000 6130 CALL BITROX(RAZ,KOLIC,CENA,INDEX)
38700000 6140 CENAR=RAZ(INDEX)
38710000       EFEKT=EREZ(INDEX)
38720000C SEARCHING THE SUBORDINATE BLOCKS
38730000       IND1=IPOINT(IBLOK)
38740000       IND2=IND1+1
38750000       IND1=ISUB(IND1)
38760000       IND2=ISUB(IND2)
38770000C FILLING THE VECTOR 'VIRTUA'
38780000       VIRTUA(IND1)=C1OPT(INDEX)
38790000       IF(IND1-NBELC) 6150,6150,6170
38800000 6150 GO TO (6170,6160),IPAR
38810000 6160 LIM=NF(IND1)
38820000       READ(93'IND1*480-479,4,ERR=2) (E2(I),I=1,LIM)
38830000       READ(93'IND1*480-319,4,ERR=2) (CEN(I),I=1,KOLIC)
38840000       CALL BITROX(CEN,NF(IND1),C1OPT(INDEX),INDE)
38850000       VIRTUA(IND1)=E2(INDE)
38860000 6170 VIRTUA(IND2)=C2OPT(INDEX)
38870000       IF(IND2-NBELC) 6180,6180,6200
38880000 6180 GO TO (6200,6190),IPAR
```

```
38890000 6190 READ(93'IND2*480-479,4,ERR=2) E2,CEN
38900000      CALL BITROX(CEN,NF(IND2),C2OPT(INDEX),INDE)
38910000      VIRTUA(IND2)=E2(INDE)
38920000C BEGINNING THE ALLOCATION ALONG THE TREE
38930000C SCANNING
38940000 6200 DO 6210 INDEX=NBELC1,NBTOTC
38950000      IF(VIRTUA(INDEX)) 6210,6220,6220
38960000 6210 CONTINUE
38970000C END OF THE ALLOCATION ALONG THE TREE
38980000      GO TO 6300
38990000C ALLOCATION FROM THE DETECTED BLOCK 'INDEX'
39000000 6220 CENA=VIRTUA(INDEX)
39010000      VIRTUA(INDEX)=-1.
39020000C READING THE VECTORS OF PRICES
39030000      IND1=NF(INDEX)
39040000      READ(93'INDEX*480-479,4,ERR=2) (C1(I),I=1,IND1)
39050000      READ(93'INDEX*480-319,4,ERR=2) (C2(I),I=1,IND1)
39060000      DO 6230 I=1,IND1
39070000 6230 E1(I)=C1(I)+C2(I)
39080000C BINARY TRACING
39090000      CALL BITROX (E1,IND1,CENA,INDEX)
39100000C SEARCHING THE SUBORDINATE BLOCKS
39110000      IND1=IPOINT(INDEX)
39120000      IND2=IND1+1
39130000      IND1=ISUB(IND1)
39140000      IND2=ISUB(IND2)
39150000C FILLING THE VECTOR 'VIRTUA'
39160000      VIRTUA(IND1)=C1(INDEX)
39170000      IF(IND1-NBELC) 6240,6240,6260
39180000 6240 GO TO (6260,6250),IPAR
39190000 6250 LIM1=NF(IND1)
39200000      READ(93'IND1*480-479,4,ERR=2) (E2(I),I=1,LIM1)
39210000      READ(93'IND1*480-319,4,ERR=2) (CEN(I),I=1,LIM1)
39220000      CALL BITROX(CEN,NF(IND1),C1(INDEX),INDE)
39230000      VIRTUA(IND1)=E2(INDE)
39240000 6260 VIRTUA(IND2)=C2(INDEX)
39250000      IF(IND2-NBELC) 6270,6270,6200
39260000 6270 GO TO (6200,6280),IPAR
39270000 6280 LIM2=NF(IND2)
39280000      READ(93'IND2*480-479,4,ERR=2) (E2(I),I=1,LIM2)
39290000      READ(93'IND2*480-319,4,ERR=2) (CEN(I),I=1,LIM2)
39300000      CALL BITROX(CEN,NF(IND2),C2(INDEX),INDE)
39310000      VIRTUA(IND2)=E2(INDE)
39320000C END OF THE ALLOCATION ALONG THE TREE
39330000      GO TO 6200
39340000C TESTING THE PRECISION OF THE RESULTS
39350000 6300 DO 6330 INDEX=1,NBELC
39360000      IF(VIRTUA(INDEX)) 6310,6330,6330
39370000 6310 WRITE(KLP,6320)
39380000 6320 FORMAT(' ----   INVALID RESULTS(ALLOCATE)'/)
39390000      GO TO 6011
39400000 6330 CONTINUE
39410000C PRINTING AND PUNCHING THE RESULTS
39420000      IF(NBELC-8) 6350,6350,6370
```

```
9430000  6350 WRITE(KLP,6360) ICENA,CENAR,EFEKT,(VIRTUA(I),I=1,NBELC)
9440000  6360 FORMAT(I4,F15.6,F7.4,8F11.4)
9450000       GO TO 6390
9460000  6370 WRITE(KLP,6360) ICENA,CENAR,EFEKT,(VIRTUA(I),I=1,8)
9470000       WRITE(KLP,6380)(VIRTUA(I),I=9,NBELC)
9480000  6380 FORMAT(32X,8F11.4)
9490000  6390 IF(IBUSI) 6400,6444,6400
9500000  6400 WRITE(KLP,6410)
9510000  6410 FORMAT(' ----    RESTRICTED PUNCHING OPTION (NO PROCESSING)' )
9520000  6444 CONTINUE
9530000C
9540000       READ(51'1,4,ERR=2) VIRTUA
9550000C PREPARING THE EXIT OUT OF THE SUBROUTINE
9560000    10 IPROG=5
9570000       IENT=2
9580000       RETURN
9590000C STANDARDS FOR FILE MANAGEMENT
9600000     1 FORMAT(I10)
9610000     2 CALL EXIT
9620000     4 FORMAT(F10.3)
9630000       RETURN
9640000       END
9650000C
9660000 ------PROGRAM SELMP ----------------------------------------------------
9670000C
9680000C---------------------------------------------------------------------
9690000C MAIN PROGRAM (INITIALIZATIONS AND BRANCHS TO THE MAIN SUBROUTINES)
9700000C---------------------------------------------------------------------
9710000       DIMENSION INJCL(33),ICHAR(11),IDIG(10),IAZ(2),R(68),AGGROP(17),
9720000      *INSEL(51),R2(17),R3(17),R4(17),R5(17),IUSVEC(5)
9730000       COMMON VIRTUA(1230),RR(20)
9740000       COMMON II(100),K(100),L(400),IPOINT(211),ISUB(780),NBSUB(211),
9750000      *IPTRR(90),ITREE(210),IA(30),LABFIL(30),IVDIM(5)
9760000       COMMON KDIM,LDIM,IDSUBK,INDLAB,NBLAB,KCR,KLP,KCP,LIST1,NIN,IAR,
9770000      *NBFILE,NBCDS,NISEL,IFIL12,NBINP,IPROG
9780000       COMMON LFIRST,INDEX,LL,IENT
9790000       COMMON NBOPTI
9800000       COMMON IBB(101)
9810000       EQUIVALENCE (R2(1),R(1)),(R3(1),R(18)),(R4(1),R(35)),(R5(1),R(52))
9820000       EQUIVALENCE (IVDIM(1),IDIM),(IVDIM(2),IRDIM),(IVDIM(3),IXDIM),
9830000      *(IVDIM(4),IEDIM),(IVDIM(5),ICDIM)
9840000       DATA INJCL/'J','O','B','S','E','L','E','X','E','R','E','A',' ',' '
9850000      *,' ','S','T','O','D','E','L','P','R','I','N','E','W','E','X','I',
9860000      *'L','I','S'/
9870000       DATA ICHAR/' ','*',',','.','''','(',')','=','+','-','/'/
9880000       DATA IDIG/'0','1','2','3','4','5','6','7','8','9'/
9890000       DATA IUSVEC/'K','A','X','E','C'/
9900000       DATA IAZ/'A','Z'/
9910000       DATA INSEL/'S','T','O','T','I','T','R','E','P','R','E','A',
9920000      *'P','R','I','A','R','I','B','R','A','W','A','I','I','N','P',
9930000      *'E','F','F','O','U','T','S','E','N','D','I','S','D','A','T',
9940000      *'O','P','T','P','R','E','A','L','L'/
9950000       DATA AGGROP/'C','C++','C+','C+-','CA','C-+','C-','C--','A','D--',
9960000      *'D-','D-+','DA','D+-','D+','D++','D'/
```

```
39970000       DATA R2/-10000.,-9.06004318,-3.51004579,-1.65482313,-0.72025393,-0
39980000      *.14780327,0.26117236,0.61944543,1.,1.44897276,2.01847121,2.7916911
39990000      *9,3.92929444,5.80233017,9.52073910,20.63030641,10000./
40000000       DATA R3/-10000.,-7.63927928,-3.11311435,-1.54928097,-0.73137681,-0
40010000      *.20652040,0.19527910,0.57297598,1.,1.51896230,2.18730967,3.1010826
40020000      *0,4.45005266,6.67475054,11.09478925,24.31768349,10000./
40030000       DATA R4/-10000.,-6.71004102,-2.82322180,-1.45190267,-0.71453444,-0
40040000      *.22476119,0.16778557,0.54727369,1.,1.56388979,2.30194765,3.3180711
40050000      *1,4.82501816,7.31711354,12.27659848,27.13457935,10000./
40060000       DATA R5/-10000.,-6.2723,-2.6404,-1.3627,-0.6697,-0.2025,0.1787,0.5
40070000      *423,1.,1.5838,2.3624,3.4427,5.0542,7.7294,13.0662,29.0810,10000./
40080000       DEFINE FILE 31(1600,22,L,IFIL1),32(1600,22,L,IFIL2),
40090000      *33(1600,22,L,IFIL3),
40100000      *34(1600,22,L,IFIL4),35(1600,22,L,IFIL5),36(1600,22,L,IFIL6),
40110000      *37(1600,22,L,IFIL7),
40120000      *38(1600,22,L,IFIL8),39(1600,22,L,IFIL9), 40(1600,22,L,IFIL10)
40130000       DEFINE FILE 41(792,22,L,IFIL11)
40140000       DEFINE FILE 42(1037,22,L,IFIL12)
40150000       DEFINE FILE 43(7960,22,L,IFIL13)
40160000       DEFINE FILE 50(780,22,L,IFIL50),51(1230,22,L,IFIL51)
40170000       DEFINE FILE 93(14880,22,L,IFIL93)
40180000C
40190000C OPTIMIZATION AND COMPARISON OF COMPLEX SYSTEMS BY EVALUATION
40200000C  DUJMOVIC / VAN BASTELAER / ARNOTTE       JANUARY 1977
40210000C
40220000       JOB=0
40230000       NBLAB=30
40240000       KCR=5
40250000       KLP=6
40260000       KCP=7
40270000       MAXF=10
40280000       NISEL=17
40290000       NBCDS=10
40300000       NBINP=16
40310000       NBOPTI=160
40320000       NBFILE=10
40330000       KDIM=100
40340000       LDIM=400
40350000       IDIM=100
40360000       IRDIM=20
40370000       IXDIM=120
40380000       IEDIM=210
40390000       ICDIM=120
40400000       IDSUBK=780
40410000       IFIL1=1
40420000       IFIL2=1
40430000       IFIL3=1
40440000       IFIL4=1
40450000       IFIL5=1
40460000       IFIL6=1
40470000       IFIL7=1
40480000       IFIL8=1
40490000       IFIL9=1
40500000       IFIL10=1
```

```
40510000       IFIL11=1
40520000       IFIL12=1
40530000       IFIL13=1
40540000       IFIL20=1
40550000       IFIL93=1
40560000       IPROG=1
40570000       WRITE (KLP,10)
40580000    10 FORMAT(1H1/5X,32('-')/5X,'I  SYSTEMS EVALUATION LANGUAGE I'/5X'I'
40590000      *30X,'I',/5X,'I** VERSION SEL77 (ENGLISH) ***I'/5X,'I',30X,'I',/5X
40600000      *'I DUJMOVIC/V.BASTELAER/ARNOTTE I',/5X,32('-')/)
40610000    20 GO TO(100,200,300,400,500,600,700,800,900,1000,1100),IPROG
40620000   100 CALL MONIT(INJCL,INSEL,ICHAR,IDIG,JOB)
40630000       GO TO 20
40640000   200 CALL TRAN1(INSEL,ICHAR,IDIG,IAZ,IUSVEC)
40650000       GO TO 20
40660000   300 CALL TRAN2(ICHAR,IDIG,IAZ,IUSVEC)
40670000       GO TO 20
40680000   400 CALL TRAN3(ICHAR,IDIG,IUSVEC)
40690000       GO TO 20
40700000   500 CALL REAL1(IDIG,IUSVEC)
40710000       GO TO 20
40720000   600 CALL REAL2(R)
40730000       GO TO 20
40740000   700 CALL REAL3(AGGROP)
40750000       GO TO 20
40760000   800 CALL REAL4(R)
40770000       GO TO 20
40780000   900 CALL REAL5(R)
40790000       GO TO 20
40800000  1000 CALL REAL6
40810000       GO TO 20
40820000  1100 CALL EXIT
40830000       STOP
40840000       END
```

C H A P I T R E 5

LISTINGS DES EXEMPLES PIRC ET CPIRC

Le présent chapitre comporte 4 listings : la source et les résultats des exemples PIRC et CPIRC .

Ces deux exemples sont exposés au chapitre 5 du 1° tome (pages 55 à 73) .

Chaque source constitue un fichier de données pour le système SEL . Chaque ensemble de résultats constitue un fichier de sorties de ce même système SEL , à ceci près que nous le présentons sous une forme compactée .

Les résultats obtenus appellent deux commentaires :

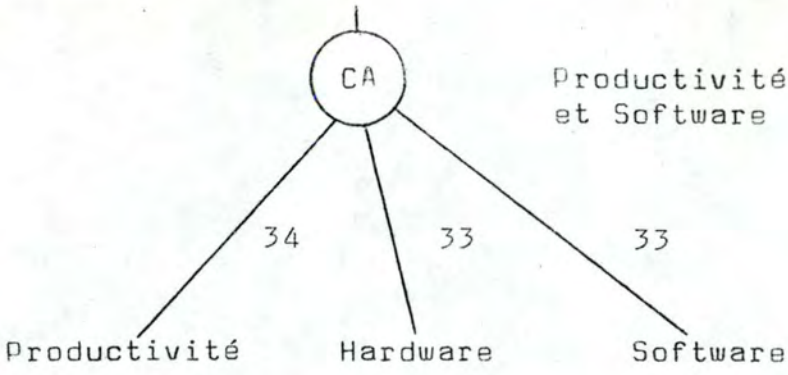
- pour PIRC , les résultats sont ceux d'une des premières versions du problème ; elle diffère de la version finale (1° tome , page 67) , car entre-temps de nouvelles propositions des vendeurs ont rehaussé l'ensemble des efficacités globales (1) ; la liste rangée reste inchangée .

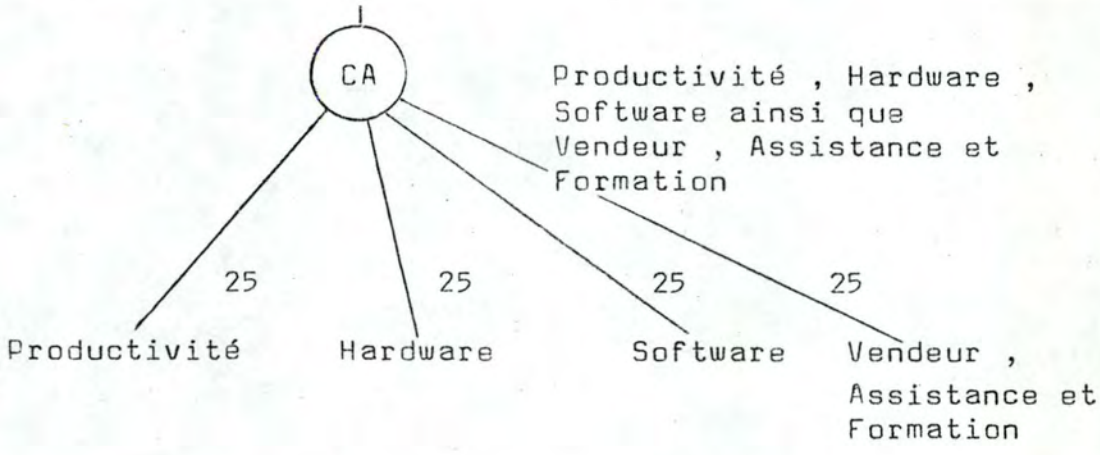
- pour CPIRC , un petit changement de la CEM du CE 6 a modifié légèrement les CEM des CNE 9 et 11 ; le reste de l'exemple est inchangé .

Signalons que les CNE des colonnes 3 et 4 de la page 67 du 1° tome ne figurent pas dans l'arbre de PIRC . Leurs efficacités sont calculées séparément , de la façon suivante (2) :

(1) sauf pour le système numéro 4 , dont l'efficacité globale est toujours nulle .

(2) par analogie au système global , on utilise l'opérateur d'aggrégation CA , et un même poids pour chaque entrée .





La source de l'exemple PIRC commence à la page , sous le nom VBA.ARN.PIRC , et ses résultats , à la page , sous le nom VBA.ARN.LISTING (qui est la forme compactée du véritable fichier des résultats , VBA.ARN.LIST) .

La source de l'exemple CPIRC commence à la page , sous le nom VBA.ARN.CPIRC , et ses résultats , à la page , sous le nom VBA.ARN.S1 (qui est la forme compactée du véritable fichier des résultats , VBA.ARN.S) .

Les ordinogrammes du système SEL ont été remis à M. Ph. Van Bastelaer dans une farte spéciale .

```
C ------------------------------------
C EXEMPLE PIRC ( CENTRE D'INFORMATIQUE
C DE PIROT , YOUGOSLAVIE )
C LE PRESENT EXEMPLE EST UN CAS REEL
C DE CHOIX D'UN ORDINATEUR ( 1976 )
C ------------------------------------
*JOB
*SEL,LIS
  INPUT ELC
  INPUT CAS
1 TITLE 2,1
  READ X(1,K99)
  EFFECTIVENESS
  OUTPUT (E) SORT,HIST
  REPEAT 1,4
  STOP
  END
*EXECUTE
```

USER I.D.- FUN96A
ACCT. NO.- MEMO
FILENAME - VBA.ARN.PIRC
MAILING ADDR.- FACULTE N.D.
DATE - 09/09/77

83 critères élémentaires

Données de l'instruction INPUT ELC

```
LIST
  10      TRAINING FOR PROGRAMMERS                    0 0 10099
  20      TRAINING FOR SYSTEM ANALYSTS                0 0 10099
  30      TRAINING FOR SYSTEM PROGRAMMERS             0 0 10099
  40      TRAINING FOR OPERATORS                      0 0 10099
  50      NUMB. & CONTENTS OF AVAILABLE TRAININGS     0 0 10099
  60      QUALITY & QUANTITY OF TRAIN. LITERATURE     0 0 10099
  70      PERIOD OF GUARANT. SYST. ING. ASSIST.       0 0 10099
  80      POSSIBILITY OF CHANGING THE SYST. ENG.      0 0   199
  90      POSSIB. OF CHOOSING THE SYST. ENG. ASS.     0 0 10099
 100      ARRIVAL SPEED ON CALL                     299   550  24 0
 110      ASSISTANCE IN NIGHT SHIFT , WEEK-END ...    0 0   199
 120      DISTANCE OF THE NEXT SELLER'S OFFICE      099 40070 900 0
 130      ASSISTANCE FOR EXTERNAL HARDWARE            0 0   199
 140      PERIOD OF FIXED ASSISTANCE COSTS            0 0   599
 150      PERIOD OF CONTRACTUAL ASSISTANCE            3 0   550  1099
 160      DISTANCE OF THE NEXT SELLER'S OFFICE      099 600 0
 170      NUMBER OF SYST. ING. AT THIS OFFICE         0 0   599
 180      NUMBER OF ASSISTANCE TECHNICIANS            0 0   599
 190      REPRESENTATION ORGANIZ. IN YUGOSLAVIA       0 0 10099
 193      BENCHMARKS EXECUTION QUALITY                0 0 10099
 195      VALIDATION TEST WITH BENCHMARKS             0 0 10099
 200      NUMBER OF SAME OR GREATER CONFIG. IN Y U     0 0   399
 210      NUMBER OF AVAILABLE BACK-UP CONFIGURAT .     0 0   280   399
 220      DISTANCE OF THE NEXT BACK-UP CONFIGURAT.  099 500 0
 230      COMPATIBILITY PC-BUC (MACHINE LANG.)        0 0   199
 240      PACKAGE OF WORKER'S MANAGEMENT              0 0   199
 250      PACKAGE OF EMPLOYEE'S MANAGEMENT            0 0 10099
 260      PACKAGE OF PRODUCTS INVENTORY MANAGEMENT    0 0 10099
 270      PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT     0 0 10099
 280      PACKAGE OF GROSS MATERIAL INVENT. MANAG.    0 0 10099
 290      PACKAGE OF FINANCIAL ACCOUNTING             0 0 10099
 300      PACKAGE OF RESOURCES PLANNING               0 0 10099
 310      PACKAGE OF TRANSACTIONS' ACCOUNTING         0 0 10099
 320      PACKAGE OF CIVIL ENGINEERING                0 0 10099
 330      DISTANCE OF THE PROVIS. CONFIGURATION     099 400 0
```

```
340       NUMBER OF BUC FOR PROVIS. WORKING          0 0    499
350       OTHER PACKAGES FOR THE COMPUTER USER       0 0  10099
360       AVAILABLE TOOLS FOR PROV. WORKING          0 0    499
400       PRODUCTIVITY FOR THE MONOPROGR. TEST      199   3 0
410       PRODUCTIVITY FOR THE MULTIPROGR. TEST     199   3 0
420       SPEED OF CENTRAL PROCESSOR                199   3 0
430       MEMORY CONSUMPTION IN THE MONOPR. TEST     199   3 0
450       NUMBER OF INDEX REGISTERS                  0 0    880  1699
460       NUMBER OF ACCUMULATORS                     0 0    880  1699
470       REPERTOIR OF MACHINE INSTRUCTIONS          0 0  10099
480       ADDRESS NUMBER PER INSTRUCTION             0 0    160   299
490       INSTRUCTION FORMAT                         0 0    150   280   399
500       CAPACITY OF USERS' MEMORY                 50 0  10099
510       POSSIBILITY OF MEMORY EXTENSION          100 0  20099
515       NUMBER OF ACCESSES TO DISK MEMORY          2 0    499
520       TOTAL CAPACITY OF DISK MEMORIES           20 0   6099
530       POSSIBILITY OF DISK MEMORY EXTENSION      80 0  16099
550       TRANSFER SPEED OF THE 9-CHANN. TAPES      10 0   4080  6099
555       CONTROL FLAGS ON THE 9-CHANN. TAPES        0 0    299
560       AUTOMATIC MONTAGE OF THE 9-CH. TAPES       0 0    199
580       DECIDING THE DATA PREPARATION AND INPUT    0 0  10099
590       DECIDING THE PROGRAM PREP. AND INPUT       0 0  10099
600       GLOBAL SPEED OF THE LINE PRINTER         300 0  80099
610       BUILDING PRINCIPLE OF THE LINE PRINTER     0 0    150   280   399
620       OUTPUT ON THE DIRECTING CONSOLE            0 0    140   280   399
700       WAY OF GIVING MEMORY TO THE PROGRAM        0 0    799
705       USER PROGRAM ADAPTATION TO THE MEMORY      0 0    140   260   380   599
710       VIRTUAL INPUT ORGANS (SPOOL-IN)            0 0    199
715       VIRTUAL OUTPUT ORGANS  ( SPOOL-OUT )       0 0    199
720       PROGRAM INVARIANCE DUE TO CAPACITY         0 0    199
730       JCL (USER'S INSTRUCTIONS)                  0 0  10099
740       CONTROL LANGUAGE FOR THE OPERATOR          0 0  10099
750       ACCOUNTING , STATISTICAL & COMPUT. MESS.   0 0    280   399
760       ACCOUNTING OF FAILURES AND ERRORS          0 0    499
770       MEASURING PROGRAM (SOFTWARE MONITOR)       0 0    160   299
790       O.S. GLOBAL EFFICIENCY                     0 0  10099
800       COBOL COMPILER                             0 0  10099
810       SPEED OF COBOL COMPILER                    0 0  10099
820       FORTRAN COMPILER                           0 0  10099
830       SPEED OF FORTRAN COMPILER                199  10 0
840       PL/1 COMPILER                              0 0  10099
850       RPG OR SIMILAR LANGUAGE                    0 0    199
860       SYMBOLIC MACHINE LANGUAGE                  0 0    180   299
870       POSSIBILITY OF SORTING PROGRAMS            0 0  10099
880       EFFECTIVENESS OF THE SORTING PROGRAM      199   6 0
890       DATA STRUCTURES                            0 0  10099
900       SYSTEMS FOR DATA BASE MANAGING             0 0  10099
910       EQUIPMENT FOR THE PROGRAMMATION SYSTEM     0 0  10099
      END
```

Fin des données de l'instruction INPUT ELC

```
IST
010 C-    TRAINING PROPOSALS                      1030  2030  3020  4020
020 C-    TOTAL TRAINING POSSIBILITIES            5060  6040
030 CA    TRAINING                          10105010 2050    30    70
040 C-    SYSTEM ENGINEERS ASSISTANCE             7080  8010  9010
```

Données de l'instruction INPUT CAS

```
1050 C-+  TRAINING AND ASSISTANCE BY SYST. ENG.   103075104025
1060 C-+  PROPOSED MAINTENANCE                     10060 11010 12030
1070 C-   CONDITIONAL MAINTENANCE                  13020 14040 15040
1080 C-+  GLOBAL ASSISTANCE                       106080107020
1090 C--  POTENTIALITIES OF THE LOCAL BR. OFF.     16020 17020 18020 19040
1095 C-+  LOCAL BRANCH OFFICE                     109050 19345 195 5
1100 C-+  BUC FOR RUNNING                         20010 21030 22030 23030
1110 C-+  * COMPUTER CONSTRUCTOR & REPRESENTATION 105035108030109515110020
1120 C-   RUNNING PART DURING PROVIS. WORKING     24010 25010 26020 27040 28020
1130 C-   PROPOSED APPPLICATION FOR PROVIS. WORK. 112035 29015 30010 310 5 32035
1140 C-   CONFIGURATION FOR PROV. WORKING         33040 34010 35010 36040
1150 C-+  * PROVIS. WORK FOR PIRC DEVELOPMENT     113080114020
1160 C-+  MEASURED PERFORMANCE OF THE COMPUTER     40075 41025
1180 A    PERFORMANCE , SPEED AND MEMORY USAGE    116060 42020 43020
1190 CA   * COMPUTER PRODUCTIVITY                 116050118050
1200 C-   PROCESSOR ORGANIZATION                  45015 46015 47030 48020 49020
1210 C-+  USER'S MEMORY                           50080 51020
1220 C-+  DISK MEMORY                             51545 52045 53010
1230 C--  MEMORY OF 9-CHANNEL TAPES               55070 55520 56010
1240 C-+  INPUT-OUTPUT ORGANS                     58015 59020 60050 61005 62010
1250 C-+  * HARDWARE                              120015121030122025123010124020
1255 C-   GROUND CHARACTERISTICS OF THE O.S.      70040 70520 71015 71515 72010
1260 C-   JCL , MEASURING & ACCOUNTING PROGRAMS   73020 74020 75020 76020 77020
1280 C-+  O.S.                                    125520126020 79060
1290 C-+  COBOL AND FORTRAN                       80045 81030 82015 83010
1300 A    RPG , ASSEMBLER AND PL/1                84020 85060 86020
1310 CA   PROGRAMMING LANGUAGES                   129060130040    50    50
1320 C-+  SORTING SYSTEMS AND WORK ON FILES       87020 88080
1322 C-+  APPLICATION SYSTEMS                     132050 89015 90015 91020
1330 C-+  * SOFTWARE                              128040131040132220
1340 CA   *** GLOBAL COMPUTER                     111020115020119020125020133020
          END
SYSTEM NUMBER 1
 100   93   19   74   70   40   50    1    1   20    1  700    1    5   10  310
   3    2   30   40    1    1    2  310  0.5   90   10  100  100   10   90   10
  50  100  310    1   50    0    1    1    1 2.32    3    5   89  1.5    2   64
 192    4  210  560 43.2    1    0  100  100  900    1    2    7    5    1    1
   1  100  100    3    4    2 84.8   90    1  100    1   60   90    2  100 1.65
  70   90   50
SYSTEM NUMBER 2
  69   60   29   74   41   30   60    1    1    2    1  310    1    5   10  310
   5    3   50   50    1  1.5    3   70    1   90   90   90   90   90   90   90
  10  100   70    2   50    0 1.43 1.07 2.94 2.36    8    8   93  1.5    1   99
 480    3  140  400   30    1    1  100  100  800    3    3    6    5    1    1
   1  100  100    2    4    0 48.5   90  1.9  100 6.06   30   90    0  100 3.18
  80  100   50
SYSTEM NUMBER 3
 100   37   60   70   60  100   90  0.8    1    2    1  310    1    3   10  190
   5    5  100  100  0.5    3    2   70    1  100  100  100  100  100  100  100
  10   30   70    3   90    5 3.63 3.17  3.0 2.14    2    0   83    2    4   95
 223    6   65   90   40    2    0   90  100  500    2    1    1    5    1    1
   1  100  100    3    3    0 46.8   85    3   90    7    0   90    2  100    1
  70   70   80
SYSTEM NUMBER 4
```

Données de l'instruction READ

Données de l'instruction TITLE

Fin des données de l'instruction INPUT CAS

à cinq reprises

```
  100   100   100   100   100   100   100  0.8     1     4     1   310     1     3    10   190
    5     5   100    90   0.5     2     3   200   0.5   100   100   100   100   100   100   100
   10   100    70     3    80     5   1.5  1.85  2.21  1.06     8     8   100     2     3    70
  300     4   210   560    40     2     0    90   100   900     2     2     5     5     1     1
    1   100   100     3     4     1  43.6    95  1.29   100  3.59   100   100     2   100  1.69
  100   100   100
SYSTEM NUMBER 5
  100    47    64   100    50    50    50     1     1     5     1   310     1     5    10   310
    5     3    60    20     1     1     1   310     1    10    10    10    10    10    10    10
   10    10   310     1   100     0  1.57  1.18  2.72     1     8    12    86     2     3   100
  480     4    90   480    40     1     1   100   100   450     2     3     3     4     1     1
    1   100   100     2     4     0  42.7    90  1.19    60  1.18     0    90     2   100   3.8
   90    90    50
*EXIT
```

```
    I  SYSTEMS EVALUATION LANGUAGE I
    I                              I
    I** VERSION SEL77 (ENGLISH) ***I
    I                              I
    I DUJMOVIC/V.BASTELAER/ARNOTTE I
    --------------------------------
C -----------------------------------
C EXEMPLE PIRC ( CENTRE D'INFORMATIQUE
C DE PIROT , YOUGOSLAVIE )
C LE PRESENT EXEMPLE EST UN CAS REEL
C DE CHOIX D'UN ORDINATEUR ( 1976 )
C -----------------------------------
*JOB
*SEL,LIS
    INPUT ELC
    INPUT CAS
  1 TITLE 2,1
    READ X(1,K99)
    EFFECTIVENESS
    OUTPUT (E) SORT,HIST
    REPEAT 1,4
    STOP
    END
NO ERROR IN ABOVE PROGRAM
END OF TRANSLATION
*EXECUTE
```

USER I.D.
ACCT. NO.
FILENAME
MAILING A
DATE -

Numéro interne du CE

Numéro externe du CE

ELEMENTARY CRIT

NO.	BLOCK	E	COMPONENT FOR EVALUATION
1	10	0.0	TRAINING FOR PROGRAMMERS
2	20	0.0	TRAINING FOR SYSTEM ANALYSTS
3	30	0.0	TRAINING FOR SYSTEM PROGRAMMERS
4	40	0.0	TRAINING FOR OPERATORS
5	50	0.0	NUMB. & CONTENTS OF AVAILABLE TRAININGS
6	60	0.0	QUALITY & QUANTITY OF TRAIN. LITERATURE
7	70	0.0	PERIOD OF GUARANT. SYST. ING. ASSIST.
8	80	0.0	POSSIBILITY OF CHANGING THE SYST. ENG.
9	90	0.0	POSSIB. OF CHOOSING THE SYST. ENG. ASS.
10	100	0.0	ARRIVAL SPEED ON CALL
11	110	0.0	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
12	120	0.0	DISTANCE OF THE NEXT SELLER'S OFFICE
13	130	0.0	ASSISTANCE FOR EXTERNAL HARDWARE
14	140	0.0	PERIOD OF FIXED ASSISTANCE COSTS
15	150	0.0	PERIOD OF CONTRACTUAL ASSISTANCE
16	160	0.0	DISTANCE OF THE NEXT SELLER'S OFFICE
17	170	0.0	NUMBER OF SYST. ING. AT THIS OFFICE
18	180	0.0	NUMBER OF ASSISTANCE TECHNICIANS
19	190	0.0	REPRESENTATION ORGANIZ. IN YUGOSLAVIA.
20	193	0.0	BENCHMARKS EXECUTION QUALITY
21	195	0.0	VALIDATION TEST WITH BENCHMARKS
22	200	0.0	NUMBER OF SAME OR GREATER CONFIG. IN Y U
23	210	0.0	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .

VOIR PAGE 151 bis

- FUN96A
- MEMO
- VBA.ARN.LISTING

DDR.- FACULTE N.D.
09/09/77

1
2
3
4
5
6
7
8
9 — Efficacité immédiate (inutilisée ici)

Libellé du CE

ERIA

INPUT 1		INPUT 2		INPUT 3		INPUT 4		INPUT 5
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	1.000	100					
0.000	0	100.000	100					
2.000	100	5.000	50	24.000	0			
0.000	0	1.000	100					
0.000	100	400.000	70	900.000	0			
0.000	0	1.000	100					
0.000	0	5.000	100					
3.000	0	5.000	50	10.000	100			
0.000	100	600.000	0					
0.000	0	5.000	100					
0.000	0	5.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	100.000	100					
0.000	0	3.000	100					
0.000	0	2.000	80	3.000	100			

Abscisse
Ordonnée
] du 2° point critique du CE numéro 1

7
8
9

délai d'arrivée sur appel maintenance en jours ouvrables

→ dist.

24	220	0.0	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
25	230	0.0	COMPATIBILITY PC-BUC (MACHINE LANG.)
26	240	0.0	PACKAGE OF WORKER'S MANAGEMENT
27	250	0.0	PACKAGE OF EMPLOYEE'S MANAGEMENT
28	260	0.0	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
29	270	0.0	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
30	280	0.0	PACKAGE OF GROSS MATERIAL INVENT. MANAG.
31	290	0.0	PACKAGE OF FINANCIAL ACCOUNTING
32	300	0.0	PACKAGE OF RESOURCES PLANNING
33	310	0.0	PACKAGE OF TRANSACTIONS' ACCOUNTING
34	320	0.0	PACKAGE OF CIVIL ENGINEERING
35	330	0.0	DISTANCE OF THE PROVIS. CONFIGURATION
36	340	0.0	NUMBER OF BUC FOR PROVIS. WORKING
37	350	0.0	OTHER PACKAGES FOR THE COMPUTER USER
38	360	0.0	AVAILABLE TOOLS FOR PROV. WORKING
39	400	0.0	PRODUCTIVITY FOR THE MONOPROGR. TEST
40	410	0.0	PRODUCTIVITY FOR THE MULTIPROGR. TEST
41	420	0.0	SPEED OF CENTRAL PROCESSOR
42	430	0.0	MEMORY CONSUMPTION IN THE MONOPR. TEST
43	450	0.0	NUMBER OF INDEX REGISTERS
44	460	0.0	NUMBER OF ACCUMULATORS
45	470	0.0	REPERTOIR OF MACHINE INSTRUCTIONS
46	480	0.0	ADDRESS NUMBER PER INSTRUCTION
47	490	0.0	INSTRUCTION FORMAT
48	500	0.0	CAPACITY OF USERS' MEMORY
49	510	0.0	POSSIBILITY OF MEMORY EXTENSION
50	515	0.0	NUMBER OF ACCESSES TO DISK MEMORY
51	520	0.0	TOTAL CAPACITY OF DISK MEMORIES
52	530	0.0	POSSIBILITY OF DISK MEMORY EXTENSION
53	550	0.0	TRANSFER SPEED OF THE 9-CHANN. TAPES
54	555	0.0	CONTROL FLAGS ON THE 9-CHANN. TAPES
55	560	0.0	AUTOMATIC MONTAGE OF THE 9-CH. TAPES
56	580	0.0	DECIDING THE DATA PREPARATION AND INPUT
57	590	0.0	DECIDING THE PROGRAM PREP. AND INPUT
58	600	0.0	GLOBAL SPEED OF THE LINE PRINTER
59	610	0.0	BUILDING PRINCIPLE OF THE LINE PRINTER
60	620	0.0	OUTPUT ON THE DIRECTING CONSOLE
61	700	0.0	WAY OF GIVING MEMORY TO THE PROGRAM
62	705	0.0	USER PROGRAM ADAPTATION TO THE MEMORY
63	710	0.0	VIRTUAL INPUT ORGANS (SPOOL-IN)
64	715	0.0	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
65	720	0.0	PROGRAM INVARIANCE DUE TO CAPACITY
66	730	0.0	JCL (USER'S INSTRUCTIONS)
67	740	0.0	CONTROL LANGUAGE FOR THE OPERATOR
68	750	0.0	ACCOUNTING , STATISTICAL & COMPUT. MESS.
69	760	0.0	ACCOUNTING OF FAILURES AND ERRORS
70	770	0.0	MEASURING PROGRAM (SOFTWARE MONITOR)
71	790	0.0	O.S. GLOBAL EFFICIENCY
72	800	0.0	COBOL COMPILER
73	810	0.0	SPEED OF COBOL COMPILER
74	820	0.0	FORTRAN COMPILER
75	830	0.0	SPEED OF FORTRAN COMPILER
76	840	0.0	PL/1 COMPILER
77	850	0.0	RPG OR SIMILAR LANGUAGE

Voir page 152 bis

```
  0.000 100   500.000   0
  0.000   0     1.000 100
  0.000   0     1.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000 100   400.000   0
  0.000   0     4.000 100
  0.000   0   100.000 100
  0.000   0     4.000 100
  1.000 100     3.000   0
  1.000 100     3.000   0
  1.000 100     3.000   0
  1.000 100     3.000   0
  0.000   0     8.000  80   16.000 100
  0.000   0     8.000  80   16.000 100
  0.000   0   100.000 100
  0.000   0     1.000  60    2.000 100
  0.000   0     1.000  50    2.000  80    3.000 100
 50.000   0   100.000 100
100.000   0   200.000 100
  2.000   0     4.000 100
 20.000   0    60.000 100
 80.000   0   160.000 100
 10.000   0    40.000  80   60.000 100
  0.000   0     2.000 100
  0.000   0     1.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
300.000   0   800.000 100
  0.000   0     1.000  50    2.000  80    3.000 100
  0.000   0     1.000  40    2.000  80    3.000 100
  0.000   0     7.000 100
  0.000   0     1.000  40    2.000  60    3.000  80    5.000 100
  0.000   0     1.000 100
  0.000   0     1.000 100
  0.000   0     1.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0     2.000  80    3.000 100
  0.000   0     4.000 100
  0.000   0     1.000  60    2.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  0.000   0   100.000 100
  1.000 100    10.000   0
  0.000   0   100.000 100
  0.000   0     1.000 100
```

nb de registre d'index
nb d'accumulateurs
op instr mach
adresse de la instruction
format des instructions

compil. ~~Fortran~~ Cobol
vitesse compil ~~Fortran~~ cobol
compil ~~cobol~~ Fortran
~~compil~~ vitesse compil Fortran

78	860	0.0	SYMBOLIC MACHINE LANGUAGE
79	870	0.0	POSSIBILITY OF SORTING PROGRAMS
80	880	0.0	EFFECTIVENESS OF THE SORTING PROGRAM
81	890	0.0	DATA STRUCTURES
82	900	0.0	SYSTEMS FOR DATA BASE MANAGING
83	910	0.0	EQUIPMENT FOR THE PROGRAMMATION SYSTEM

NON-ELEMENTA

NO.	BLOCK	OP.	COMPONENT FOR EVALUATION
84	1010	C-	TRAINING PROPOSALS
85	1020	C-	TOTAL TRAINING POSSIBILITIES
86	1030	*CA	TRAINING
87	1040	C-	SYSTEM ENGINEERS ASSISTANCE
88	1050	C-+	TRAINING AND ASSISTANCE BY SYST. ENG.
89	1060	C-+	PROPOSED MAINTENANCE
90	1070	C-	CONDITIONAL MAINTENANCE
91	1080	C-+	GLOBAL ASSISTANCE
92	1090	C--	POTENTIALITIES OF THE LOCAL BR. OFF.
93	1095	C-+	LOCAL BRANCH OFFICE
94	1100	C-+	BUC FOR RUNNING
95	1110	C-+	* COMPUTER CONSTRUCTOR & REPRESENTATION
96	1120	C-	RUNNING PART DURING PROVIS. WORKING
97	1130	C-	PROPOSED APPPLICATION FOR PROVIS. WORK.
98	1140	C-	CONFIGURATION FOR PROV. WORKING
99	1150	C-+	* PROVIS. WORK FOR PIRC DEVELOPMENT
100	1160	C-+	MEASURED PERFORMANCE OF THE COMPUTER
101	1180	A	PERFORMANCE , SPEED AND MEMORY USAGE
102	1190	CA	* COMPUTER PRODUCTIVITY
103	1200	C-	PROCESSOR ORGANIZATION
104	1210	C-+	USER'S MEMORY
105	1220	C-+	DISK MEMORY
106	1230	C--	MEMORY OF 9-CHANNEL TAPES
107	1240	C-+	INPUT-OUTPUT ORGANS
108	1250	C-+	* HARDWARE
109	1255	C-	GROUND CHARACTERISTICS OF THE O.S.
110	1260	C-	JCL , MEASURING & ACCOUNTING PROGRAMS
111	1280	C-+	O.S.
112	1290	C-+	COBOL AND FORTRAN
113	1300	A	RPG , ASSEMBLER AND PL/1
114	1310	*CA	PROGRAMMING LANGUAGES
115	1320	C-+	SORTING SYSTEMS AND WORK ON FILES
116	1322	C-+	APPLICATION SYSTEMS
117	1330	C-+	* SOFTWARE
118	1340	CA	*** GLOBAL COMPUTER

Voir page 153 bis

SYSTEM NUMBER 1

* ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION
38	360	AVAILABLE TOOLS FOR PROV. WORKING
55	560	AUTOMATIC MONTAGE OF THE 9-CH. TAPES

0.000	0	1.000	80	2.000 100
0.000	0	100.000	100	
1.000	100	6.000	0	
0.000	0	100.000	100	
0.000	0	100.000	100	
0.000	0	100.000	100	

Numéro interne du CNE — Numéro externe du CNE — Libellé du CNE — Opérateur d'aggrégation

RY CRITERIA

INPUT 1	INPUT 2	INPUT 3	INPUT 4	INPUT 5
10 (30)	20 (30)	30 (20)	40 (20)	
50 (60)	60 (40)			
1010 (50)	1020 (50)	0 (30)	0 (70)	
70 (80)	80 (10)	90 (10)		
1030 (75)	1040 (25)			
100 (60)	110 (10)	120 (30)		
130 (20)	140 (40)	150 (40)		
1060 (80)	1070 (20)			
160 (20)	170 (20)	180 (20)	190 (40)	
1090 (50)	193 (45)	195 (5)		
200 (10)	210 (30)	220 (30)	230 (30)	
1050 (35)	1080 (30)	1095 (15)	1100 (20)	
240 (10)	250 (10)	260 (20)	270 (40)	280 (20)
1120 (35)	290 (15)	300 (10)	310 (5)	320 (35)
330 (40)	340 (10)	350 (10)	360 (40)	
1130 (80)	1140 (20)			
400 (75)	410 (25)			
1160 (60)	420 (20)	430 (20)		
1160 (50)	1180 (50)			
450 (15)	460 (15)	470 (30)	480 (20)	490 (20)
500 (80)	510 (20)			
515 (45)	520 (45)	530 (10)		
550 (70)	555 (20)	560 (10)		
580 (15)	590 (20)	600 (50)	610 (5)	620 (10)
1200 (15)	1210 (30)	1220 (25)	1230 (10)	1240 (20)
700 (40)	705 (20)	710 (15)	715 (15)	720 (10)
730 (20)	740 (20)	750 (20)	760 (20)	770 (20)
1255 (20)	1260 (20)	790 (60)		
800 (45)	810 (30)	820 (15)	830 (10)	
840 (20)	850 (60)	860 (20)		
1290 (60)	1300 (40)	0 (50)	0 (50)	
870 (20)	880 (80)			
1320 (50)	890 (15)	900 (15)	910 (20)	
1280 (40)	1310 (40)	1322 (20)		
1110 (20)	1150 (20)	1190 (20)	1250 (20)	1330 (20)

Numéro externe du 3° CIS — Poids du 3° CIS — du CNE 1040

X	E
0.000	0.000
0.000	0.000

9	90	POSSIB. OF CHOOSING THE SYST. ENG. ASS.	1.000	1.000
21	195	VALIDATION TEST WITH BENCHMARKS	1.000	1.000
X73	810	SPEED OF COBOL COMPILER	1.000	1.000
27	250	PACKAGE OF EMPLOYEE'S MANAGEMENT	10.000	10.000
30	280	PACKAGE OF GROSS MATERIAL INVENT. MANAG.	10.000	10.000
32	300	PACKAGE OF RESOURCES PLANNING	10.000	10.000
10	100	ARRIVAL SPEED ON CALL	20.000	10.526
3	30	TRAINING FOR SYSTEM PROGRAMMERS	19.000	19.000
35	330	DISTANCE OF THE PROVIS. CONFIGURATION	310.000	22.500
36	340	NUMBER OF BUC FOR PROVIS. WORKING	1.000	25.000
48	500	CAPACITY OF USERS' MEMORY	64.000	28.000
12	120	DISTANCE OF THE NEXT SELLER'S OFFICE	700.000	28.000
19	190	REPRESENTATION ORGANIZ. IN YUGOSLAVIA	30.000	30.000
X 43	450	NUMBER OF INDEX REGISTERS	3.000	30.000 ←
22	200	NUMBER OF SAME OR GREATER CONFIG. IN Y U	1.000	33.333
42	430	MEMORY CONSUMPTION IN THE MONOPR. TEST	2.320	34.000
24	220	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.	310.000	38.000
6	60	QUALITY & QUANTITY OF TRAIN. LITERATURE	40.000	40.000
18	180	NUMBER OF ASSISTANCE TECHNICIANS	2.000	40.000
20	193	BENCHMARKS EXECUTION QUALITY	40.000	40.000
16	160	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	48.333
44	460	NUMBER OF ACCUMULATORS	5.000	50.000 ←
7	70	PERIOD OF GUARANT. SYST. ING. ASSIST.	50.000	50.000
33	310	PACKAGE OF TRANSACTIONS' ACCOUNTING	50.000	50.000
37	350	OTHER PACKAGES FOR THE COMPUTER USER	50.000	50.000
54	555	CONTROL FLAGS ON THE 9-CHANN. TAPES	1.000	50.000
59	610	BUILDING PRINCIPLE OF THE LINE PRINTER	1.000	50.000
83	910	EQUIPMENT FOR THE PROGRAMMATION SYSTEM	50.000	50.000
25	230	COMPATIBILITY PC-BUC (MACHINE LANG.)	0.500	50.000
17	170	NUMBER OF SYST. ING. AT THIS OFFICE	3.000	60.000
76	840	PL/1 COMPILER	60.000	60.000
5	50	NUMB. & CONTENTS OF AVAILABLE TRAININGS	70.000	70.000
81	890	DATA STRUCTURES	70.000	70.000
4	40	TRAINING FOR OPERATORS	74.000	74.000
23	210	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .	2.000	80.000
46	480	ADDRESS NUMBER PER INSTRUCTION	1.500	80.000 ←
47	490	INSTRUCTION FORMAT	2.000	80.000 ←
60	620	OUTPUT ON THE DIRECTING CONSOLE	2.000	80.000
53	550	TRANSFER SPEED OF THE 9-CHANN. TAPES	43.200	83.200
71	790	O.S. GLOBAL EFFICIENCY	84.800	84.800
80	880	EFFECTIVENESS OF THE SORTING PROGRAM	1.650	87.000
45	470	REPERTOIR OF MACHINE INSTRUCTIONS	89.000	89.000 ←
31	290	PACKAGE OF FINANCIAL ACCOUNTING	90.000	90.000
X72	800	COBOL COMPILER	90.000	90.000
82	900	SYSTEMS FOR DATA BASE MANAGING	90.000	90.000
49	510	POSSIBILITY OF MEMORY EXTENSION	192.000	92.000
2	20	TRAINING FOR SYSTEM ANALYSTS	93.000	93.000
1	10	TRAINING FOR PROGRAMMERS	100.000	100.000
8	80	POSSIBILITY OF CHANGING THE SYST. ENG.	1.000	100.000
11	110	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...	1.000	100.000
13	130	ASSISTANCE FOR EXTERNAL HARDWARE	1.000	100.000
14	140	PERIOD OF FIXED ASSISTANCE COSTS	5.000	100.000
15	150	PERIOD OF CONTRACTUAL ASSISTANCE	10.000	100.000
26	240	PACKAGE OF WORKER'S MANAGEMENT	90.000	100.000

28	260	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT	100.000	100.000
29	270	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT	100.000	100.000
34	320	PACKAGE OF CIVIL ENGINEERING	100.000	100.000
39	400	PRODUCTIVITY FOR THE MONOPROGR. TEST	1.000	100.000
40	410	PRODUCTIVITY FOR THE MULTIPROGR. TEST	1.000	100.000
41	420	SPEED OF CENTRAL PROCESSOR	1.000	100.000
50	515	NUMBER OF ACCESSES TO DISK MEMORY	4.000	100.000
51	520	TOTAL CAPACITY OF DISK MEMORIES	210.000	100.000
52	530	POSSIBILITY OF DISK MEMORY EXTENSION	560.000	100.000
56	580	DECIDING THE DATA PREPARATION AND INPUT	100.000	100.000
57	590	DECIDING THE PROGRAM PREP. AND INPUT	100.000	100.000
58	600	GLOBAL SPEED OF THE LINE PRINTER	900.000	100.000
61	700	WAY OF GIVING MEMORY TO THE PROGRAM	7.000	100.000
62	705	USER PROGRAM ADAPTATION TO THE MEMORY	5.000	100.000
63	710	VIRTUAL INPUT ORGANS (SPOOL-IN)	1.000	100.000
64	715	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)	1.000	100.000
65	720	PROGRAM INVARIANCE DUE TO CAPACITY	1.000	100.000
66	730	JCL (USER'S INSTRUCTIONS)	100.000	100.000
67	740	CONTROL LANGUAGE FOR THE OPERATOR	100.000	100.000
68	750	ACCOUNTING , STATISTICAL & COMPUT. MESS.	3.000	100.000
69	760	ACCOUNTING OF FAILURES AND ERRORS	4.000	100.000
70	770	MEASURING PROGRAM (SOFTWARE MONITOR)	2.000	100.000
×74	820	FORTRAN COMPILER	100.000	100.000
×75	830	SPEED OF FORTRAN COMPILER	1.000	100.000
77	850	RPG OR SIMILAR LANGUAGE	90.000	100.000
78	860	SYMBOLIC MACHINE LANGUAGE	2.000	100.000
79	870	POSSIBILITY OF SORTING PROGRAMS	100.000	100.000

EE HISTOGRAM

```
 0 -  10     8 I********
10 -  20     2 I**
20 -  30     6 I******
30 -  40     6 I******
40 -  50     8 I********
50 -  60     3 I***
60 -  70     2 I**
70 -  80     5 I*****
80 -  90     7 I*******
90 - 100    36 I************************************
```

EFFECTIVENESS FROM 0 TO 10

AVAILABLE TOOLS FOR PROV. WORKING
AUTOMATIC MONTAGE OF THE 9-CH. TAPES
POSSIB. OF CHOOSING THE SYST. ENG. ASS.
VALIDATION TEST WITH BENCHMARKS
SPEED OF COBOL COMPILER
PACKAGE OF EMPLOYEE'S MANAGEMENT
PACKAGE OF GROSS MATERIAL INVENT. MANAG.
PACKAGE OF RESOURCES PLANNING
EFFECTIVENESS FROM 10 TO 20

ARRIVAL SPEED ON CALL
TRAINING FOR SYSTEM PROGRAMMERS
EFFECTIVENESS FROM 20 TO 30

DISTANCE OF THE PROVIS. CONFIGURATION
NUMBER OF BUC FOR PROVIS. WORKING
CAPACITY OF USERS' MEMORY
DISTANCE OF THE NEXT SELLER'S OFFICE
REPRESENTATION ORGANIZ. IN YUGOSLAVIA
NUMBER OF INDEX REGISTERS
EFFECTIVENESS FROM 30 TO 40

NUMBER OF SAME OR GREATER CONFIG. IN Y U
MEMORY CONSUMPTION IN THE MONOPR. TEST
DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
QUALITY & QUANTITY OF TRAIN. LITERATURE
NUMBER OF ASSISTANCE TECHNICIANS
BENCHMARKS EXECUTION QUALITY
EFFECTIVENESS FROM 40 TO 50

DISTANCE OF THE NEXT SELLER'S OFFICE
NUMBER OF ACCUMULATORS
PERIOD OF GUARANT. SYST. ING. ASSIST.
PACKAGE OF TRANSACTIONS' ACCOUNTING
OTHER PACKAGES FOR THE COMPUTER USER
CONTROL FLAGS ON THE 9-CHANN. TAPES
BUILDING PRINCIPLE OF THE LINE PRINTER
EQUIPMENT FOR THE PROGRAMMATION SYSTEM
EFFECTIVENESS FROM 50 TO 60

COMPATIBILITY PC-BUC (MACHINE LANG.)
NUMBER OF SYST. ING. AT THIS OFFICE
PL/1 COMPILER
EFFECTIVENESS FROM 60 TO 70

NUMB. & CONTENTS OF AVAILABLE TRAININGS
DATA STRUCTURES
EFFECTIVENESS FROM 70 TO 80

TRAINING FOR OPERATORS
NUMBER OF AVAILABLE BACK-UP CONFIGURAT .
ADDRESS NUMBER PER INSTRUCTION
INSTRUCTION FORMAT
OUTPUT ON THE DIRECTING CONSOLE
EFFECTIVENESS FROM 80 TO 90

TRANSFER SPEED OF THE 9-CHANN. TAPES
O.S. GLOBAL EFFICIENCY
EFFECTIVENESS OF THE SORTING PROGRAM
REPERTOIR OF MACHINE INSTRUCTIONS
PACKAGE OF FINANCIAL ACCOUNTING
COBOL COMPILER
SYSTEMS FOR DATA BASE MANAGING
EFFECTIVENESS FROM 90 TO 100

POSSIBILITY OF MEMORY EXTENSION
TRAINING FOR SYSTEM ANALYSTS

TRAINING FOR PROGRAMMERS
POSSIBILITY OF CHANGING THE SYST. ENG.
ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
ASSISTANCE FOR EXTERNAL HARDWARE
PERIOD OF FIXED ASSISTANCE COSTS
PERIOD OF CONTRACTUAL ASSISTANCE
PACKAGE OF WORKER'S MANAGEMENT
PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
PACKAGE OF CIVIL ENGINEERING
PRODUCTIVITY FOR THE MONOPROGR. TEST
PRODUCTIVITY FOR THE MULTIPROGR. TEST
SPEED OF CENTRAL PROCESSOR
NUMBER OF ACCESSES TO DISK MEMORY
TOTAL CAPACITY OF DISK MEMORIES
POSSIBILITY OF DISK MEMORY EXTENSION
DECIDING THE DATA PREPARATION AND INPUT
DECIDING THE PROGRAM PREP. AND INPUT
GLOBAL SPEED OF THE LINE PRINTER
WAY OF GIVING MEMORY TO THE PROGRAM
USER PROGRAM ADAPTATION TO THE MEMORY
VIRTUAL INPUT ORGANS (SPOOL-IN)
VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
PROGRAM INVARIANCE DUE TO CAPACITY
JCL (USER'S INSTRUCTIONS)
CONTROL LANGUAGE FOR THE OPERATOR
ACCOUNTING , STATISTICAL & COMPUT. MESS.
ACCOUNTING OF FAILURES AND ERRORS
MEASURING PROGRAM (SOFTWARE MONITOR)
FORTRAN COMPILER
SPEED OF FORTRAN COMPILER
RPG OR SIMILAR LANGUAGE
SYMBOLIC MACHINE LANGUAGE
POSSIBILITY OF SORTING PROGRAMS
* SUBSYSTEM EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	E
98	1140	CONFIGURATION FOR PROV. WORKING	1.255
112	1290	COBOL AND FORTRAN	14.012
114	1310	PROGRAMMING LANGUAGES	14.012
89	1060	PROPOSED MAINTENANCE	16.803
91	1080	GLOBAL ASSISTANCE	23.165
99	1150	☆ PROVIS. WORK FOR PIRC DEVELOPMENT	23.663
93	1095	LOCAL BRANCH OFFICE	30.917
104	1210	USER'S MEMORY	34.952
95	1110	☆ COMPUTER CONSTRUCTOR & REPRESENTATION	37.825
117	1330	☆ SOFTWARE	38.177
87	1040	SYSTEM ENGINEERS ASSISTANCE	40.802
92	1090	POTENTIALITIES OF THE LOCAL BR. OFF.	40.973
118	1340	☆☆☆ GLOBAL COMPUTER	42.610
94	1100	BUC FOR RUNNING	50.323
96	1120	RUNNING PART DURING PROVIS. WORKING	55.043
88	1050	TRAINING AND ASSISTANCE BY SYST. ENG.	55.951

EFFECTIVENESS FROM 50 TO 60

BUC FOR RUNNING
RUNNING PART DURING PROVIS. WORKING
TRAINING AND ASSISTANCE BY SYST. ENG.
TOTAL TRAINING POSSIBILITIES
EFFECTIVENESS FROM 60 TO 70

TRAINING
MEMORY OF 9-CHANNEL TAPES
PROPOSED APPPLICATION FOR PROVIS. WORK.
* HARDWARE
PROCESSOR ORGANIZATION
TRAINING PROPOSALS
EFFECTIVENESS FROM 70 TO 80

APPLICATION SYSTEMS
EFFECTIVENESS FROM 80 TO 90

PERFORMANCE , SPEED AND MEMORY USAGE
SORTING SYSTEMS AND WORK ON FILES
EFFECTIVENESS FROM 90 TO 100

O.S.
RPG , ASSEMBLER AND PL/1
* COMPUTER PRODUCTIVITY
INPUT-OUTPUT ORGANS
CONDITIONAL MAINTENANCE
GROUND CHARACTERISTICS OF THE O.S.
JCL , MEASURING & ACCOUNTING PROGRAMS
MEASURED PERFORMANCE OF THE COMPUTER
DISK MEMORY

SYSTEM NUMBER 2

* ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	X	E
38	360	AVAILABLE TOOLS FOR PROV. WORKING	0.000	0.000
70	770	MEASURING PROGRAM (SOFTWARE MONITOR)	0.000	0.000
78	860	SYMBOLIC MACHINE LANGUAGE	0.000	0.000
9	90	POSSIB. OF CHOOSING THE SYST. ENG. ASS.	1.000	1.000
21	195	VALIDATION TEST WITH BENCHMARKS	1.000	1.000
73	810	SPEED OF COBOL COMPILER	1.900	1.900
41	420	SPEED OF CENTRAL PROCESSOR	2.940	3.000
33	310	PACKAGE OF TRANSACTIONS' ACCOUNTING	10.000	10.000
3	30	TRAINING FOR SYSTEM PROGRAMMERS	29.000	29.000
6	60	QUALITY & QUANTITY OF TRAIN. LITERATURE	30.000	30.000
76	840	PL/1 COMPILER	30.000	30.000
42	430	MEMORY CONSUMPTION IN THE MONOPR. TEST	2.360	32.000
5	50	NUMB. & CONTENTS OF AVAILABLE TRAININGS	41.000	41.000
X 75	830	SPEED OF FORTRAN COMPILER	6.060	43.778
16	160	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	48.333
71	790	O.S. GLOBAL EFFICIENCY	48.500	48.500
19	190	REPRESENTATION ORGANIZ. IN YUGOSLAVIA	50.000	50.000

85	1020	TOTAL TRAINING POSSIBILITIES	56.507
86	1030	TRAINING	62.370
106	1230	MEMORY OF 9-CHANNEL TAPES	62.574
97	1130	PROPOSED APPPLICATION FOR PROVIS. WORK.	63.567
108	1250	✡ HARDWARE	63.569
103	1200	PROCESSOR ORGANIZATION	67.296
84	1010	TRAINING PROPOSALS	68.232
116	1322	APPLICATION SYSTEMS	76.347
101	1180	PERFORMANCE , SPEED AND MEMORY USAGE	86.800
115	1320	SORTING SYSTEMS AND WORK ON FILES	89.437
111	1280	O.S.	90.520
113	1300	RPG , ASSEMBLER AND PL/1	92.000
102	1190	✡ COMPUTER PRODUCTIVITY	92.999
107	1240	INPUT-OUTPUT ORGANS	94.207
90	1070	CONDITIONAL MAINTENANCE	100.000
109	1255	GROUND CHARACTERISTICS OF THE O.S.	100.000
110	1260	JCL , MEASURING & ACCOUNTING PROGRAMS	100.000
100	1160	MEASURED PERFORMANCE OF THE COMPUTER	100.000
105	1220	DISK MEMORY	100.000

SE HISTOGRAM

0 - 10	1	I✡
10 - 20	3	I✡✡✡
20 - 30	2	I✡✡
30 - 40	4	I✡✡✡✡
40 - 50	3	I✡✡✡
50 - 60	4	I✡✡✡✡
60 - 70	6	I✡✡✡✡✡✡
70 - 80	1	I✡
80 - 90	2	I✡✡
90 - 100	9	I✡✡✡✡✡✡✡✡✡

EFFECTIVENESS FROM 0 TO 10

CONFIGURATION FOR PROV. WORKING

EFFECTIVENESS FROM 10 TO 20

COBOL AND FORTRAN

PROGRAMMING LANGUAGES

PROPOSED MAINTENANCE

EFFECTIVENESS FROM 20 TO 30

GLOBAL ASSISTANCE

✡ PROVIS. WORK FOR PIRC DEVELOPMENT

EFFECTIVENESS FROM 30 TO 40

LOCAL BRANCH OFFICE

USER'S MEMORY

✡ COMPUTER CONSTRUCTOR & REPRESENTATION

✡ SOFTWARE

EFFECTIVENESS FROM 40 TO 50

SYSTEM ENGINEERS ASSISTANCE

POTENTIALITIES OF THE LOCAL BR. OFF.

✡✡✡ GLOBAL COMPUTER

20	193	BENCHMARKS EXECUTION QUALITY	50.000	50.000
22	200	NUMBER OF SAME OR GREATER CONFIG. IN Y U	1.500	50.000
36	340	NUMBER OF BUC FOR PROVIS. WORKING	2.000	50.000
37	350	OTHER PACKAGES FOR THE COMPUTER USER	50.000	50.000
47	490	INSTRUCTION FORMAT	1.000	50.000
50	515	NUMBER OF ACCESSES TO DISK MEMORY	3.000	50.000
54	555	CONTROL FLAGS ON THE 9-CHANN. TAPES	1.000	50.000
83	910	EQUIPMENT FOR THE PROGRAMMATION SYSTEM	50.000	50.000
53	550	TRANSFER SPEED OF THE 9-CHANN. TAPES	30.000	53.333
80	880	EFFECTIVENESS OF THE SORTING PROGRAM	3.180	56.400
2	20	TRAINING FOR SYSTEM ANALYSTS	60.000	60.000
7	70	PERIOD OF GUARANT. SYST. ING. ASSIST.	60.000	60.000
18	180	NUMBER OF ASSISTANCE TECHNICIANS	3.000	60.000
1	10	TRAINING FOR PROGRAMMERS	69.000	69.000
4	40	TRAINING FOR OPERATORS	74.000	74.000
12	120	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	76.750
39	400	PRODUCTIVITY FOR THE MONOPROGR. TEST	1.430	78.500
43	450	NUMBER OF INDEX REGISTERS	8.000	80.000
44	460	NUMBER OF ACCUMULATORS	8.000	80.000
46	480	ADDRESS NUMBER PER INSTRUCTION	1.500	80.000
68	750	ACCOUNTING , STATISTICAL & COMPUT. MESS.	2.000	80.000
81	890	DATA STRUCTURES	80.000	80.000
35	330	DISTANCE OF THE PROVIS. CONFIGURATION	70.000	82.500
61	700	WAY OF GIVING MEMORY TO THE PROGRAM	6.000	85.714
24	220	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.	70.000	86.000
27	250	PACKAGE OF EMPLOYEE'S MANAGEMENT	90.000	90.000
28	260	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT	90.000	90.000
29	270	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT	90.000	90.000
30	280	PACKAGE OF GROSS MATERIAL INVENT. MANAG.	90.000	90.000
31	290	PACKAGE OF FINANCIAL ACCOUNTING	90.000	90.000
32	300	PACKAGE OF RESOURCES PLANNING	90.000	90.000
72	800	COBOL COMPILER	90.000	90.000
45	470	REPERTOIR OF MACHINE INSTRUCTIONS	93.000	93.000
40	410	PRODUCTIVITY FOR THE MULTIPROGR. TEST	1.070	96.500
42	500	CAPACITY OF USERS' MEMORY	99.000	98.000
8	80	POSSIBILITY OF CHANGING THE SYST. ENG.	1.000	100.000
10	100	ARRIVAL SPEED ON CALL	2.000	100.000
11	110	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...	1.000	100.000
13	130	ASSISTANCE FOR EXTERNAL HARDWARE	1.000	100.000
14	140	PERIOD OF FIXED ASSISTANCE COSTS	5.000	100.000
15	150	PERIOD OF CONTRACTUAL ASSISTANCE	10.000	100.000
17	170	NUMBER OF SYST. ING. AT THIS OFFICE	5.000	100.000
23	210	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .	3.000	100.000
25	230	COMPATIBILITY PC-BUC (MACHINE LANG.)	1.000	100.000
26	240	PACKAGE OF WORKER'S MANAGEMENT	90.000	100.000
34	320	PACKAGE OF CIVIL ENGINEERING	100.000	100.000
49	510	POSSIBILITY OF MEMORY EXTENSION	480.000	100.000
51	520	TOTAL CAPACITY OF DISK MEMORIES	140.000	100.000
52	530	POSSIBILITY OF DISK MEMORY EXTENSION	400.000	100.000
55	560	AUTOMATIC MONTAGE OF THE 9-CH. TAPES	1.000	100.000
56	580	DECIDING THE DATA PREPARATION AND INPUT	100.000	100.000
57	590	DECIDING THE PROGRAM PREP. AND INPUT	100.000	100.000
58	600	GLOBAL SPEED OF THE LINE PRINTER	800.000	100.000
59	610	BUILDING PRINCIPLE OF THE LINE PRINTER	3.000	100.000

160

60	620	OUTPUT ON THE DIRECTING CONSOLE	3.000	100.000
62	705	USER PROGRAM ADAPTATION TO THE MEMORY	5.000	100.000
63	710	VIRTUAL INPUT ORGANS (SPOOL-IN)	1.000	100.000
64	715	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)	1.000	100.000
65	720	PROGRAM INVARIANCE DUE TO CAPACITY	1.000	100.000
66	730	JCL (USER'S INSTRUCTIONS)	100.000	100.000
67	740	CONTROL LANGUAGE FOR THE OPERATOR	100.000	100.000
69	760	ACCOUNTING OF FAILURES AND ERRORS	4.000	100.000
74	820	FORTRAN COMPILER	100.000	100.000
77	850	RPG OR SIMILAR LANGUAGE	90.000	100.000
79	870	POSSIBILITY OF SORTING PROGRAMS	100.000	100.000
82	900	SYSTEMS FOR DATA BASE MANAGING	100.000	100.000

EE HISTOGRAM

```
 0 -  10     8 I********
10 -  20     0 I
20 -  30     3 I***
30 -  40     1 I*
40 -  50    13 I*************
50 -  60     5 I*****
60 -  70     1 I*
70 -  80     8 I********
80 -  90    10 I**********
90 - 100    34 I**********************************
```

EFFECTIVENESS FROM 0 TO 10

AVAILABLE TOOLS FOR PROV. WORKING
MEASURING PROGRAM (SOFTWARE MONITOR)
SYMBOLIC MACHINE LANGUAGE
POSSIB. OF CHOOSING THE SYST. ENG. ASS.
VALIDATION TEST WITH BENCHMARKS
SPEED OF COBOL COMPILER
SPEED OF CENTRAL PROCESSOR
PACKAGE OF TRANSACTIONS' ACCOUNTING
EFFECTIVENESS FROM 20 TO 30

TRAINING FOR SYSTEM PROGRAMMERS
QUALITY & QUANTITY OF TRAIN. LITERATURE
PL/1 COMPILER
EFFECTIVENESS FROM 30 TO 40

MEMORY CONSUMPTION IN THE MONOPR. TEST
EFFECTIVENESS FROM 40 TO 50

NUMB. & CONTENTS OF AVAILABLE TRAININGS
SPEED OF FORTRAN COMPILER
DISTANCE OF THE NEXT SELLER'S OFFICE
O.S. GLOBAL EFFICIENCY
REPRESENTATION ORGANIZ. IN YUGOSLAVIA
BENCHMARKS EXECUTION QUALITY
NUMBER OF SAME OR GREATER CONFIG. IN Y U
NUMBER OF BUC FOR PROVIS. WORKING
OTHER PACKAGES FOR THE COMPUTER USER
INSTRUCTION FORMAT

NUMBER OF ACCESSES TO DISK MEMORY
CONTROL FLAGS ON THE 9-CHANN. TAPES
EQUIPMENT FOR THE PROGRAMMATION SYSTEM
EFFECTIVENESS FROM 50 TO 60

TRANSFER SPEED OF THE 9-CHANN. TAPES
EFFECTIVENESS OF THE SORTING PROGRAM
TRAINING FOR SYSTEM ANALYSTS
PERIOD OF GUARANT. SYST. ING. ASSIST.
NUMBER OF ASSISTANCE TECHNICIANS
EFFECTIVENESS FROM 60 TO 70

TRAINING FOR PROGRAMMERS
EFFECTIVENESS FROM 70 TO 80

TRAINING FOR OPERATORS
DISTANCE OF THE NEXT SELLER'S OFFICE
PRODUCTIVITY FOR THE MONOPROGR. TEST
NUMBER OF INDEX REGISTERS
NUMBER OF ACCUMULATORS
ADDRESS NUMBER PER INSTRUCTION
ACCOUNTING , STATISTICAL & COMPUT. MESS.
DATA STRUCTURES
EFFECTIVENESS FROM 80 TO 90

DISTANCE OF THE PROVIS. CONFIGURATION
WAY OF GIVING MEMORY TO THE PROGRAM
DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
PACKAGE OF EMPLOYEE'S MANAGEMENT
PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
PACKAGE OF GROSS MATERIAL INVENT. MANAG.
PACKAGE OF FINANCIAL ACCOUNTING
PACKAGE OF RESOURCES PLANNING
COBOL COMPILER
EFFECTIVENESS FROM 90 TO 100

REPERTOIR OF MACHINE INSTRUCTIONS
PRODUCTIVITY FOR THE MULTIPROGR. TEST
CAPACITY OF USERS' MEMORY
POSSIBILITY OF CHANGING THE SYST. ENG.
ARRIVAL SPEED ON CALL
ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
ASSISTANCE FOR EXTERNAL HARDWARE
PERIOD OF FIXED ASSISTANCE COSTS
PERIOD OF CONTRACTUAL ASSISTANCE
NUMBER OF SYST. ING. AT THIS OFFICE
NUMBER OF AVAILABLE BACK-UP CONFIGURAT .
COMPATIBILITY PC-B C (MACHINE LANG.)
PACKAGE OF WORKER'S MANAGEMENT
PACKAGE OF CIVIL ENGINEERING
POSSIBILITY OF MEMORY EXTENSION
TOTAL CAPACITY OF DISK MEMORIES
POSSIBILITY OF DISK MEMORY EXTENSION

AUTOMATIC MONTAGE OF THE 9-CH. TAPES
DECIDING THE DATA PREPARATION AND INPUT
DECIDING THE PROGRAM PREP. AND INPUT
GLOBAL SPEED OF THE LINE PRINTER
BUILDING PRINCIPLE OF THE LINE PRINTER
OUTPUT ON THE DIRECTING CONSOLE
USER PROGRAM ADAPTATION TO THE MEMORY
VIRTUAL INPUT ORGANS (SPOOL-IN)
VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
PROGRAM INVARIANCE DUE TO CAPACITY
JCL (USER'S INSTRUCTIONS)
CONTROL LANGUAGE FOR THE OPERATOR
ACCOUNTING OF FAILURES AND ERRORS
FORTRAN COMPILER
RPG OR SIMILAR LANGUAGE
POSSIBILITY OF SORTING PROGRAMS
SYSTEMS FOR DATA BASE MANAGING
* SUBSYSTEM EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	E
98	1140	CONFIGURATION FOR PROV. WORKING	3.340
112	1290	COBOL AND FORTRAN	13.438
114	1310	PROGRAMMING LANGUAGES	18.438
110	1260	JCL , MEASURING & ACCOUNTING PROGRAMS	27.154
117	1330	☆ SOFTWARE	34.051
85	1020	TOTAL TRAINING POSSIBILITIES	36.294
99	1150	☆ PROVIS. WORK FOR PIRC DEVELOPMENT	39.042
93	1095	LOCAL BRANCH OFFICE	40.793
86	1030	TRAINING	46.627
88	1050	TRAINING AND ASSISTANCE BY SYST. ENG.	46.846
87	1040	SYSTEM ENGINEERS ASSISTANCE	47.510
111	1280	O.S.	48.528
118	1340	☆☆☆ GLOBAL COMPUTER	52.198
101	1180	PERFORMANCE , SPEED AND MEMORY USAGE	56.566
106	1230	MEMORY OF 9-CHANNEL TAPES	56.738
84	1010	TRAINING PROPOSALS	56.960
92	1090	POTENTIALITIES OF THE LOCAL BR. OFF.	60.462
115	1320	SORTING SYSTEMS AND WORK ON FILES	63.004
95	1110	☆ COMPUTER CONSTRUCTOR & REPRESENTATION	63.268
113	1300	RPG , ASSEMBLER AND PL/1	66.000
116	1322	APPLICATION SYSTEMS	66.471
102	1190	☆ COMPUTER PRODUCTIVITY	67.484
105	1220	DISK MEMORY	72.307
103	1200	PROCESSOR ORGANIZATION	76.514
100	1160	MEASURED PERFORMANCE OF THE COMPUTER	82.609
108	1250	☆ HARDWARE	82.995
97	1130	PROPOSED APPPLICATION FOR PROVIS. WORK.	85.599
94	1100	BUC FOR RUNNING	88.744
96	1120	RUNNING PART DURING PROVIS. WORKING	90.961
89	1060	PROPOSED MAINTENANCE	92.227
91	1080	GLOBAL ASSISTANCE	93.725
109	1255	GROUND CHARACTERISTICS OF THE O.S.	94.068
104	1210	USER'S MEMORY	98.396

```
  90 1070  CONDITIONAL MAINTENANCE                          100.000
 107 1240  INPUT-OUTPUT ORGANS                              100.000
-------------------------------------------------------------------
SE HISTOGRAM
 0 -  10     1 I*
10 -  20     2 I**
20 -  30     1 I*
30 -  40     3 I***
40 -  50     5 I*****
50 -  60     4 I****
60 -  70     6 I******
70 -  80     2 I**
80 -  90     4 I****
90 - 100     7 I*******
EFFECTIVENESS FROM  0 TO  10
----------------------------
CONFIGURATION FOR PROV. WORKING
EFFECTIVENESS FROM 10 TO  20
----------------------------
COBOL AND FORTRAN
PROGRAMMING LANGUAGES
EFFECTIVENESS FROM 20 TO  30
----------------------------
JCL , MEASURING & ACCOUNTING PROGRAMS
EFFECTIVENESS FROM 30 TO  40
----------------------------
* SOFTWARE
TOTAL TRAINING POSSIBILITIES
* PROVIS. WORK FOR PIRC DEVELOPMENT
EFFECTIVENESS FROM 40 TO  50
----------------------------
LOCAL BRANCH OFFICE
TRAINING
TRAINING AND ASSISTANCE BY SYST. ENG.
SYSTEM ENGINEERS ASSISTANCE
O.S.
EFFECTIVENESS FROM 50 TO  60
----------------------------
*** GLOBAL COMPUTER
PERFORMANCE , SPEED AND MEMORY USAGE
MEMORY OF 9-CHANNEL TAPES
TRAINING PROPOSALS
EFFECTIVENESS FROM 60 TO  70
----------------------------
POTENTIALITIES OF THE LOCAL BR. OFF.
SORTING SYSTEMS AND WORK ON FILES
* COMPUTER CONSTRUCTOR & REPRESENTATION
RPG , ASSEMBLER AND PL/1
APPLICATION SYSTEMS
* COMPUTER PRODUCTIVITY
EFFECTIVENESS FROM 70 TO  80
----------------------------
DISK MEMORY
PROCESSOR ORGANIZATION
```

EFFECTIVENESS FROM 80 TO 90

MEASURED PERFORMANCE OF THE COMPUTER
☆ HARDWARE
PROPOSED APPPLICATION FOR PROVIS. WORK.
BUC FOR RUNNING
EFFECTIVENESS FROM 90 TO 100

RUNNING PART DURING PROVIS. WORKING
PROPOSED MAINTENANCE
GLOBAL ASSISTANCE
GROUND CHARACTERISTICS OF THE O.S.
USER'S MEMORY
CONDITIONAL MAINTENANCE
INPUT-OUTPUT ORGANS

SYSTEM NUMBER 3

ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	X	E
39	400	PRODUCTIVITY FOR THE MONOPROGR. TEST	3.630	0.000
40	410	PRODUCTIVITY FOR THE MULTIPROGR. TEST	3.170	0.000
41	420	SPEED OF CENTRAL PROCESSOR	3.000	0.000
44	460	NUMBER OF ACCUMULATORS	0.000	0.000
55	560	AUTOMATIC MONTAGE OF THE 9-CH. TAPES	0.000	0.000
70	770	MEASURING PROGRAM (SOFTWARE MONITOR)	0.000	0.000
76	840	PL/1 COMPILER	0.000	0.000
21	195	VALIDATION TEST WITH BENCHMARKS	0.500	0.500
9	90	POSSIB. OF CHOOSING THE SYST. ENG. ASS.	1.000	1.000
73	810	SPEED OF COBOL COMPILER	3.000	3.000
33	310	PACKAGE OF TRANSACTIONS' ACCOUNTING	10.000	10.000
52	530	POSSIBILITY OF DISK MEMORY EXTENSION	90.000	12.500
61	700	WAY OF GIVING MEMORY TO THE PROGRAM	1.000	14.286
43	450	NUMBER OF INDEX REGISTERS	2.000	20.000
34	320	PACKAGE OF CIVIL ENGINEERING	30.000	30.000
75	830	SPEED OF FORTRAN COMPILER	7.000	33.333
2	20	TRAINING FOR SYSTEM ANALYSTS	37.000	37.000
58	600	GLOBAL SPEED OF THE LINE PRINTER	500.000	40.000
60	620	OUTPUT ON THE DIRECTING CONSOLE	1.000	40.000
42	430	MEMORY CONSUMPTION IN THE MONOPR. TEST	2.140	43.000
71	790	O.S. GLOBAL EFFICIENCY	46.800	46.800
3	30	TRAINING FOR SYSTEM PROGRAMMERS	60.000	60.000
5	50	NUMB. & CONTENTS OF AVAILABLE TRAININGS	60.000	60.000
14	140	PERIOD OF FIXED ASSISTANCE COSTS	3.000	60.000
16	160	DISTANCE OF THE NEXT SELLER'S OFFICE	190.000	68.333
4	40	TRAINING FOR OPERATORS	70.000	70.000
81	890	DATA STRUCTURES	70.000	70.000
82	900	SYSTEMS FOR DATA BASE MANAGING	70.000	70.000
36	340	NUMBER OF BUC FOR PROVIS. WORKING	3.000	75.000
69	760	ACCOUNTING OF FAILURES AND ERRORS	3.000	75.000
12	120	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	76.750
23	210	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .	2.000	80.000
53	550	TRANSFER SPEED OF THE 9-CHANN. TAPES	40.000	80.000
59	610	BUILDING PRINCIPLE OF THE LINE PRINTER	2.000	80.000

83	910	EQUIPMENT FOR THE PROGRAMMATION SYSTEM	80.000	80.000
8	80	POSSIBILITY OF CHANGING THE SYST. ENG.	0.800	80.000
35	330	DISTANCE OF THE PROVIS. CONFIGURATION	70.000	82.500
45	470	REPERTOIR OF MACHINE INSTRUCTIONS	83.000	83.000
72	800	COBOL COMPILER	85.000	85.000
24	220	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.	70.000	86.000
7	70	PERIOD OF GUARANT. SYST. ING. ASSIST.	90.000	90.000
37	350	OTHER PACKAGES FOR THE COMPUTER USER	90.000	90.000
48	500	CAPACITY OF USERS' MEMORY	95.000	90.000
56	580	DECIDING THE DATA PREPARATION AND INPUT	90.000	90.000
74	820	FORTRAN COMPILER	90.000	90.000
1	10	TRAINING FOR PROGRAMMERS	100.000	100.000
6	60	QUALITY & QUANTITY OF TRAIN. LITERATURE	100.000	100.000
10	100	ARRIVAL SPEED ON CALL	2.000	100.000
11	110	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...	1.000	100.000
13	130	ASSISTANCE FOR EXTERNAL HARDWARE	1.000	100.000
15	150	PERIOD OF CONTRACTUAL ASSISTANCE	10.000	100.000
17	170	NUMBER OF SYST. ING. AT THIS OFFICE	5.000	100.000
18	180	NUMBER OF ASSISTANCE TECHNICIANS	5.000	100.000
19	190	REPRESENTATION ORGANIZ. IN YUGOSLAVIA	100.000	100.000
20	193	BENCHMARKS EXECUTION QUALITY	100.000	100.000
22	200	NUMBER OF SAME OR GREATER CONFIG. IN Y U	3.000	100.000
25	230	COMPATIBILITY PC-BUC (MACHINE LANG.)	1.000	100.000
26	240	PACKAGE OF WORKER'S MANAGEMENT	100.000	100.000
27	250	PACKAGE OF EMPLOYEE'S MANAGEMENT	100.000	100.000
28	260	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT	100.000	100.000
29	270	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT	100.000	100.000
30	280	PACKAGE OF GROSS MATERIAL INVENT. MANAG.	100.000	100.000
31	290	PACKAGE OF FINANCIAL ACCOUNTING	100.000	100.000
32	300	PACKAGE OF RESOURCES PLANNING	100.000	100.000
38	360	AVAILABLE TOOLS FOR PROV. WORKING	5.000	100.000
46	480	ADDRESS NUMBER PER INSTRUCTION	2.000	100.000
47	490	INSTRUCTION FORMAT	4.000	100.000
49	510	POSSIBILITY OF MEMORY EXTENSION	223.000	100.000
50	515	NUMBER OF ACCESSES TO DISK MEMORY	6.000	100.000
51	520	TOTAL CAPACITY OF DISK MEMORIES	65.000	100.000
54	555	CONTROL FLAGS ON THE 9-CHANN. TAPES	2.000	100.000
57	590	DECIDING THE PROGRAM PREP. AND INPUT	100.000	100.000
62	705	USER PROGRAM ADAPTATION TO THE MEMORY	5.000	100.000
63	710	VIRTUAL INPUT ORGANS (SPOOL-IN)	1.000	100.000
64	715	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)	1.000	100.000
65	720	PROGRAM INVARIANCE DUE TO CAPACITY	1.000	100.000
66	730	JCL (USER'S INSTRUCTIONS)	100.000	100.000
67	740	CONTROL LANGUAGE FOR THE OPERATOR	100.000	100.000
68	750	ACCOUNTING , STATISTICAL & COMPUT. MESS.	3.000	100.000
77	850	RPG OR SIMILAR LANGUAGE	90.000	100.000
78	860	SYMBOLIC MACHINE LANGUAGE	2.000	100.000
79	870	POSSIBILITY OF SORTING PROGRAMS	100.000	100.000
80	880	EFFECTIVENESS OF THE SORTING PROGRAM	1.000	100.000

EE HISTOGRAM

```
 0 -  10    11 I**********
10 -  20     3 I***
20 -  30     1 I*
```

```
30 -  40     4 I****
40 -  50     2 I**
50 -  60     3 I***
60 -  70     4 I****
70 -  80     7 I*******
80 -  90    10 I**********
90 - 100    38 I**************************************
EFFECTIVENESS FROM  0 TO  10
----------------------------
PRODUCTIVITY FOR THE MONOPROGR. TEST
PRODUCTIVITY FOR THE MULTIPROGR. TEST
SPEED OF CENTRAL PROCESSOR
NUMBER OF ACCUMULATORS
AUTOMATIC MONTAGE OF THE 9-CH. TAPES
MEASURING PROGRAM (SOFTWARE MONITOR)
PL/1 COMPILER
VALIDATION TEST WITH BENCHMARKS
POSSIB. OF CHOOSING THE SYST. ENG. ASS.
SPEED OF COBOL COMPILER
PACKAGE OF TRANSACTIONS' ACCOUNTING
EFFECTIVENESS FROM 10 TO  20
----------------------------
POSSIBILITY OF DISK MEMORY EXTENSION
WAY OF GIVING MEMORY TO THE PROGRAM
NUMBER OF INDEX REGISTERS
EFFECTIVENESS FROM 20 TO  30
----------------------------
PACKAGE OF CIVIL ENGINEERING
EFFECTIVENESS FROM 30 TO  40
----------------------------
SPEED OF FORTRAN COMPILER
TRAINING FOR SYSTEM ANALYSTS
GLOBAL SPEED OF THE LINE PRINTER
OUTPUT ON THE DIRECTING CONSOLE
EFFECTIVENESS FROM 40 TO  50
----------------------------
MEMORY CONSUMPTION IN THE MONOPR. TEST
O.S. GLOBAL EFFICIENCY
EFFECTIVENESS FROM 50 TO  60
----------------------------
TRAINING FOR SYSTEM PROGRAMMERS
NUMB. & CONTENTS OF AVAILABLE TRAININGS
PERIOD OF FIXED ASSISTANCE COSTS
EFFECTIVENESS FROM 60 TO  70
----------------------------
DISTANCE OF THE NEXT SELLER'S OFFICE
TRAINING FOR OPERATORS
DATA STRUCTURES
SYSTEMS FOR DATA BASE MANAGING
EFFECTIVENESS FROM 70 TO  80
----------------------------
NUMBER OF BUC FOR PROVIS. WORKING
ACCOUNTING OF FAILURES AND ERRORS
DISTANCE OF THE NEXT SELLER'S OFFICE
```

NUMBER OF AVAILABLE BACK-UP CONFIGURAT .
TRANSFER SPEED OF THE 9-CHANN. TAPES
BUILDING PRINCIPLE OF THE LINE PRINTER
EQUIPMENT FOR THE PROGRAMMATION SYSTEM
EFFECTIVENESS FROM 80 TO 90

POSSIBILITY OF CHANGING THE SYST. ENC.
DISTANCE OF THE PROVIS. CONFIGURATION
REPERTOIR OF MACHINE INSTRUCTIONS
COBOL COMPILER
DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
PERIOD OF GUARANT. SYST. ING. ASSIST.
OTHER PACKAGES FOR THE COMPUTER USER
CAPACITY OF USERS' MEMORY
DECIDING THE DATA PREPARATION AND INPUT
FORTRAN COMPILER
EFFECTIVENESS FROM 90 TO 100

TRAINING FOR PROGRAMMERS
QUALITY & QUANTITY OF TRAIN. LITERATURE
ARRIVAL SPEED ON CALL
ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
ASSISTANCE FOR EXTERNAL HARDWARE
PERIOD OF CONTRACTUAL ASSISTANCE
NUMBER OF SYST. ING. AT THIS OFFICE
NUMBER OF ASSISTANCE TECHNICIANS
REPRESENTATION ORGANIZ. IN YUGOSLAVIA
BENCHMARKS EXECUTION QUALITY
NUMBER OF SAME OR GREATER CONFIG. IN Y U
COMPATIBILITY PC-BUC (MACHINE LANG.)
PACKAGE OF WORKER'S MANAGEMENT
PACKAGE OF EMPLOYEE'S MANAGEMENT
PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
PACKAGE OF GROSS MATERIAL INVENT. MANAG.
PACKAGE OF FINANCIAL ACCOUNTING
PACKAGE OF RESOURCES PLANNING
AVAILABLE TOOLS FOR PROV. WORKING
ADDRESS NUMBER PER INSTRUCTION
INSTRUCTION FORMAT
POSSIBILITY OF MEMORY EXTENSION
NUMBER OF ACCESSES TO DISK MEMORY
TOTAL CAPACITY OF DISK MEMORIES
CONTROL FLAGS ON THE 9-CHANN. TAPES
DECIDING THE PROGRAM PREP. AND INPUT
USER PROGRAM ADAPTATION TO THE MEMORY
VIRTUAL INPUT ORGANS (SPOOL-IN)
VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
PROGRAM INVARIANCE DUE TO CAPACITY
JCL (USER'S INSTRUCTIONS)
CONTROL LANGUAGE FOR THE OPERATOR
ACCOUNTING , STATISTICAL & COMPUT. MESS.
RPG OR SIMILAR LANGUAGE
SYMBOLIC MACHINE LANGUAGE

POSSIBILITY OF SORTING PROGRAMS
EFFECTIVENESS OF THE SORTING PROGRAM
* SUBSYSTEM EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	E
100	1160	MEASURED PERFORMANCE OF THE COMPUTER	0.000
102	1190	☆ COMPUTER PRODUCTIVITY	0.000
118	1340	☆☆☆ GLOBAL COMPUTER	0.000
101	1180	PERFORMANCE , SPEED AND MEMORY USAGE	8.600
112	1290	COBOL AND FORTRAN	21.810
114	1310	PROGRAMMING LANGUAGES	21.810
110	1260	JCL , MEASURING & ACCOUNTING PROGRAMS	26.734
103	1200	PROCESSOR ORGANIZATION	29.230
117	1330	☆ SOFTWARE	36.336
111	1280	O.S.	42.107
109	1255	GROUND CHARACTERISTICS OF THE O.S.	49.689
107	1240	INPUT-OUTPUT ORGANS	55.198
97	1130	PROPOSED APPPLICATION FOR PROVIS. WORK.	60.971
93	1095	LOCAL BRANCH OFFICE	61.226
84	1010	TRAINING PROPOSALS	63.184
86	1030	TRAINING	63.184
88	1050	TRAINING AND ASSISTANCE BY SYST. ENG.	63.704
108	1250	☆ HARDWARE	64.236
87	1040	SYSTEM ENGINEERS ASSISTANCE	65.294
99	1150	☆ PROVIS. WORK FOR FIRC DEVELOPMENT	65.667
106	1230	MEMORY OF 9-CHANNEL TAPES	70.123
85	1020	TOTAL TRAINING POSSIBILITIES	74.212
95	1110	☆ COMPUTER CONSTRUCTOR & REPRESENTATION	74.903
105	1220	DISK MEMORY	77.646
113	1300	RPG , ASSEMBLER AND PL/1	80.000
90	1070	CONDITIONAL MAINTENANCE	82.016
116	1322	APPLICATION SYSTEMS	85.688
98	1140	CONFIGURATION FOR PROV. WORKING	89.105
94	1100	BUC FOR RUNNING	89.295
91	1080	GLOBAL ASSISTANCE	90.073
104	1210	USER'S MEMORY	91.905
89	1060	PROPOSED MAINTENANCE	92.227
92	1090	POTENTIALITIES OF THE LOCAL BR. OFF.	93.233
96	1120	RUNNING PART DURING PROVIS. WORKING	100.000
115	1320	SORTING SYSTEMS AND WORK ON FILES	100.000

SE HISTOGRAM

0 - 10	4	I☆☆☆☆
10 - 20	0	I
20 - 30	4	I☆☆☆☆
30 - 40	1	I☆
40 - 50	2	I☆☆
50 - 60	1	I☆
60 - 70	8	I☆☆☆☆☆☆☆☆
70 - 80	4	I☆☆☆☆
80 - 90	5	I☆☆☆☆☆
90 - 100	6	I☆☆☆☆☆☆

EFFECTIVENESS FROM 0 TO 10

MEASURED PERFORMANCE OF THE COMPUTER
✡ COMPUTER PRODUCTIVITY
✡✡✡ GLOBAL COMPUTER
PERFORMANCE , SPEED AND MEMORY USAGE
EFFECTIVENESS FROM 20 TO 30

COBOL AND FORTRAN
PROGRAMMING LANGUAGES
JCL , MEASURING & ACCOUNTING PROGRAMS
PROCESSOR ORGANIZATION
EFFECTIVENESS FROM 30 TO 40

✡ SOFTWARE
EFFECTIVENESS FROM 40 TO 50

O.S.
GROUND CHARACTERISTICS OF THE O.S.
EFFECTIVENESS FROM 50 TO 60

INPUT-OUTPUT ORGANS
EFFECTIVENESS FROM 60 TO 70

PROPOSED APPPLICATION FOR PROVIS. WORK.
LOCAL BRANCH OFFICE
TRAINING PROPOSALS
TRAINING
TRAINING AND ASSISTANCE BY SYST. ENG.
✡ HARDWARE
SYSTEM ENGINEERS ASSISTANCE
✡ PROVIS. WORK FOR PIRC DEVELOPMENT
EFFECTIVENESS FROM 70 TO 80

MEMORY OF 9-CHANNEL TAPES
TOTAL TRAINING POSSIBILITIES
✡ COMPUTER CONSTRUCTOR & REPRESENTATION
DISK MEMORY
EFFECTIVENESS FROM 80 TO 90

RPG , ASSEMBLER AND PL/1
CONDITIONAL MAINTENANCE
APPLICATION SYSTEMS
CONFIGURATION FOR PROV. WORKING
BUC FOR RUNNING
EFFECTIVENESS FROM 90 TO 100

GLOBAL ASSISTANCE
USER'S MEMORY
PROPOSED MAINTENANCE
POTENTIALITIES OF THE LOCAL BR. OFF.
RUNNING PART DURING PROVIS. WORKING
SORTING SYSTEMS AND WORK ON FILES

SYSTEM NUMBER 4

✡ ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	X	E
55	560	AUTOMATIC MONTAGE OF THE 9-CH. TAPES	0.000	0.000
21	195	VALIDATION TEST WITH BENCHMARKS	0.500	0.500
9	90	POSSIB. OF CHOOSING THE SYST. ENG. ASS.	1.000	1.000
73	810	SPEED OF COBOL COMPILER	1.290	1.290
33	310	PACKAGE OF TRANSACTIONS' ACCOUNTING	10.000	10.000
41	420	SPEED OF CENTRAL PROCESSOR	2.210	39.500
48	500	CAPACITY OF USERS' MEMORY	70.000	40.000
71	790	O.S. GLOBAL EFFICIENCY	43.600	43.600
25	230	COMPATIBILITY PC-BUC (MACHINE LANG.)	0.500	50.000
40	410	PRODUCTIVITY FOR THE MULTIPROGR. TEST	1.850	57.500
14	140	PERIOD OF FIXED ASSISTANCE COSTS	3.000	60.000
70	770	MEASURING PROGRAM (SOFTWARE MONITOR)	1.000	60.000
24	220	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.	200.000	60.000
22	200	NUMBER OF SAME OR GREATER CONFIG. IN Y U	2.000	66.667
10	100	ARRIVAL SPEED ON CALL	4.000	66.667
16	160	DISTANCE OF THE NEXT SELLER'S OFFICE	190.000	68.333
75	830	SPEED OF FORTRAN COMPILER	3.590	71.222
61	700	WAY OF GIVING MEMORY TO THE PROGRAM	5.000	71.429
36	340	NUMBER OF BUC FOR PROVIS. WORKING	3.000	75.000
39	400	PRODUCTIVITY FOR THE MONOPROGR. TEST	1.500	75.000
12	120	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	76.750
37	350	OTHER PACKAGES FOR THE COMPUTER USER	80.000	80.000
43	450	NUMBER OF INDEX REGISTERS	8.000	80.000
44	460	NUMBER OF ACCUMULATORS	8.000	80.000
53	550	TRANSFER SPEED OF THE 9-CHANN. TAPES	40.000	80.000
59	610	BUILDING PRINCIPLE OF THE LINE PRINTER	2.000	80.000
60	620	OUTPUT ON THE DIRECTING CONSOLE	2.000	80.000
8	80	POSSIBILITY OF CHANGING THE SYST. ENG.	0.800	80.000
35	330	DISTANCE OF THE PROVIS. CONFIGURATION	70.000	82.500
80	880	EFFECTIVENESS OF THE SORTING PROGRAM	1.690	86.200
20	193	BENCHMARKS EXECUTION QUALITY	90.000	90.000
56	580	DECIDING THE DATA PREPARATION AND INPUT	90.000	90.000
72	800	COBOL COMPILER	95.000	95.000
42	430	MEMORY CONSUMPTION IN THE MONOPR. TEST	1.060	97.000
1	10	TRAINING FOR PROGRAMMERS	100.000	100.000
2	20	TRAINING FOR SYSTEM ANALYSTS	100.000	100.000
3	30	TRAINING FOR SYSTEM PROGRAMMERS	100.000	100.000
4	40	TRAINING FOR OPERATORS	100.000	100.000
5	50	NUMB. & CONTENTS OF AVAILABLE TRAININGS	100.000	100.000
6	60	QUALITY & QUANTITY OF TRAIN. LITERATURE	100.000	100.000
7	70	PERIOD OF GUARANT. SYST. ING. ASSIST.	100.000	100.000
11	110	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...	1.000	100.000
13	130	ASSISTANCE FOR EXTERNAL HARDWARE	1.000	100.000
15	150	PERIOD OF CONTRACTUAL ASSISTANCE	10.000	100.000
17	170	NUMBER OF SYST. ING. AT THIS OFFICE	5.000	100.000
18	180	NUMBER OF ASSISTANCE TECHNICIANS	5.000	100.000
19	190	REPRESENTATION ORGANIZ. IN YUGOSLAVIA	100.000	100.000
23	210	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .	3.000	100.000
26	240	PACKAGE OF WORKER'S MANAGEMENT	100.000	100.000
27	250	PACKAGE OF EMPLOYEE'S MANAGEMENT	100.000	100.000
28	260	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT	100.000	100.000

29	270	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT	100.000	100.000
30	280	PACKAGE OF GROSS MATERIAL INVENT. MANAG.	100.000	100.000
31	290	PACKAGE OF FINANCIAL ACCOUNTING	100.000	100.000
32	300	PACKAGE OF RESOURCES PLANNING	100.000	100.000
34	320	PACKAGE OF CIVIL ENGINEERING	100.000	100.000
38	360	AVAILABLE TOOLS FOR PROV. WORKING	5.000	100.000
45	470	REPERTOIR OF MACHINE INSTRUCTIONS	100.000	100.000
46	480	ADDRESS NUMBER PER INSTRUCTION	2.000	100.000
47	490	INSTRUCTION FORMAT	3.000	100.000
49	510	POSSIBILITY OF MEMORY EXTENSION	300.000	100.000
50	515	NUMBER OF ACCESSES TO DISK MEMORY	4.000	100.000
51	520	TOTAL CAPACITY OF DISK MEMORIES	210.000	100.000
52	530	POSSIBILITY OF DISK MEMORY EXTENSION	560.000	100.000
54	555	CONTROL FLAGS ON THE 9-CHANN. TAPES	2.000	100.000
57	590	DECIDING THE PROGRAM PREP. AND INPUT	100.000	100.000
58	600	GLOBAL SPEED OF THE LINE PRINTER	900.000	100.000
62	705	USER PROGRAM ADAPTATION TO THE MEMORY	5.000	100.000
63	710	VIRTUAL INPUT ORGANS (SPOOL-IN)	1.000	100.000
64	715	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)	1.000	100.000
65	720	PROGRAM INVARIANCE DUE TO CAPACITY	1.000	100.000
66	730	JCL (USER'S INSTRUCTIONS)	100.000	100.000
67	740	CONTROL LANGUAGE FOR THE OPERATOR	100.000	100.000
68	750	ACCOUNTING , STATISTICAL & COMPUT. MESS.	3.000	100.000
69	760	ACCOUNTING OF FAILURES AND ERRORS	4.000	100.000
74	820	FORTRAN COMPILER	100.000	100.000
76	840	PL/1 COMPILER	100.000	100.000
77	850	RPG OR SIMILAR LANGUAGE	100.000	100.000
78	860	SYMBOLIC MACHINE LANGUAGE	2.000	100.000
79	870	POSSIBILITY OF SORTING PROGRAMS	100.000	100.000
81	890	DATA STRUCTURES	100.000	100.000
82	900	SYSTEMS FOR DATA BASE MANAGING	100.000	100.000
83	910	EQUIPMENT FOR THE PROGRAMMATION SYSTEM	100.000	100.000

EE HISTOGRAM

```
 0 -  10     5 I*****
10 -  20     0 I
20 -  30     0 I
30 -  40     2 I**
40 -  50     1 I*
50 -  60     4 I****
60 -  70     4 I****
70 -  80    11 I***********
80 -  90     5 I*****
90 - 100    51 I**************************************************
```

EFFECTIVENESS FROM 0 TO 10

AUTOMATIC MONTAGE OF THE 9-CH. TAPES
VALIDATION TEST WITH BENCHMARKS
POSSIB. OF CHOOSING THE SYST. ENG. ASS.
SPEED OF COBOL COMPILER
PACKAGE OF TRANSACTIONS' ACCOUNTING

EFFECTIVENESS FROM 30 TO 40

SPEED OF CENTRAL PROCESSOR

CAPACITY OF USERS' MEMORY
EFFECTIVENESS FROM 40 TO 50

O.S. GLOBAL EFFICIENCY
EFFECTIVENESS FROM 50 TO 60

COMPATIBILITY PC-BUC (MACHINE LANG.)
PRODUCTIVITY FOR THE MULTIPROGR. TEST
PERIOD OF FIXED ASSISTANCE COSTS
MEASURING PROGRAM (SOFTWARE MONITOR)
EFFECTIVENESS FROM 60 TO 70

DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
NUMBER OF SAME OR GREATER CONFIG. IN Y U
ARRIVAL SPEED ON CALL
DISTANCE OF THE NEXT SELLER'S OFFICE
EFFECTIVENESS FROM 70 TO 80

SPEED OF FORTRAN COMPILER
WAY OF GIVING MEMORY TO THE PROGRAM
NUMBER OF BUC FOR PROVIS. WORKING
PRODUCTIVITY FOR THE MONOPROGR. TEST
DISTANCE OF THE NEXT SELLER'S OFFICE
OTHER PACKAGES FOR THE COMPUTER USER
NUMBER OF INDEX REGISTERS
NUMBER OF ACCUMULATORS
TRANSFER SPEED OF THE 9-CHANN. TAPES
BUILDING PRINCIPLE OF THE LINE PRINTER
OUTPUT ON THE DIRECTING CONSOLE
EFFECTIVENESS FROM 80 TO 90

POSSIBILITY OF CHANGING THE SYST. ENG.
DISTANCE OF THE PROVIS. CONFIGURATION
EFFECTIVENESS OF THE SORTING PROGRAM
BENCHMARKS EXECUTION QUALITY
DECIDING THE DATA PREPARATION AND INPUT
EFFECTIVENESS FROM 90 TO 100

COBOL COMPILER
MEMORY CONSUMPTION IN THE MONOPR. TEST
TRAINING FOR PROGRAMMERS
TRAINING FOR SYSTEM ANALYSTS
TRAINING FOR SYSTEM PROGRAMMERS
TRAINING FOR OPERATORS
NUMB. & CONTENTS OF AVAILABLE TRAININGS
QUALITY & QUANTITY OF TRAIN. LITERATURE
PERIOD OF GUARANT. SYST. ING. ASSIST.
ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
ASSISTANCE FOR EXTERNAL HARDWARE
PERIOD OF CONTRACTUAL ASSISTANCE
NUMBER OF SYST. ING. AT THIS OFFICE
NUMBER OF ASSISTANCE TECHNICIANS
REPRESENTATION ORGANIZ. IN YUGOSLAVIA
NUMBER OF AVAILABLE BACK-UP CONFIGURAT .

PACKAGE OF WORKER'S MANAGEMENT
PACKAGE OF EMPLOYEE'S MANAGEMENT
PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
PACKAGE OF GROSS MATERIAL INVENT. MANAG.
PACKAGE OF FINANCIAL ACCOUNTING
PACKAGE OF RESOURCES PLANNING
PACKAGE OF CIVIL ENGINEERING
AVAILABLE TOOLS FOR PROV. WORKING
REPERTOIR OF MACHINE INSTRUCTIONS
ADDRESS NUMBER PER INSTRUCTION
INSTRUCTION FORMAT
POSSIBILITY OF MEMORY EXTENSION
NUMBER OF ACCESSES TO DISK MEMORY
TOTAL CAPACITY OF DISK MEMORIES
POSSIBILITY OF DISK MEMORY EXTENSION
CONTROL FLAGS ON THE 9-CHANN. TAPES
DECIDING THE PROGRAM PREP. AND INPUT
GLOBAL SPEED OF THE LINE PRINTER
USER PROGRAM ADAPTATION TO THE MEMORY
VIRTUAL INPUT ORGANS (SPOOL-IN)
VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
PROGRAM INVARIANCE DUE TO CAPACITY
JCL (USER'S INSTRUCTIONS)
CONTROL LANGUAGE FOR THE OPERATOR
ACCOUNTING , STATISTICAL & COMPUT. MESS.
ACCOUNTING OF FAILURES AND ERRORS
FORTRAN COMPILER
PL/1 COMPILER
RPG OR SIMILAR LANGUAGE
SYMBOLIC MACHINE LANGUAGE
POSSIBILITY OF SORTING PROGRAMS
DATA STRUCTURES
SYSTEMS FOR DATA BASE MANAGING
EQUIPMENT FOR THE PROGRAMMATION SYSTEM
* SUBSYSTEM EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	E
112	1290	COBOL AND FORTRAN	15.949
114	1310	PROGRAMMING LANGUAGES	15.949
117	1330	* SOFTWARE	35.898
104	1210	USER'S MEMORY	47.583
111	1280	O.S.	57.307
93	1095	LOCAL BRANCH OFFICE	58.642
118	1340	*** GLOBAL COMPUTER	63.843
94	1100	BUC FOR RUNNING	66.342
101	1180	PERFORMANCE , SPEED AND MEMORY USAGE	69.367
102	1190	* COMPUTER PRODUCTIVITY	69.737
100	1160	MEASURED PERFORMANCE OF THE COMPUTER	70.111
106	1230	MEMORY OF 9-CHANNEL TAPES	70.123
87	1040	SYSTEM ENGINEERS ASSISTANCE	71.435
89	1060	PROPOSED MAINTENANCE	72.307
91	1080	GLOBAL ASSISTANCE	74.138

108	1250	☆ HARDWARE	74.928
95	1110	☆ COMPUTER CONSTRUCTOR & REPRESENTATION	75.213
90	1070	CONDITIONAL MAINTENANCE	82.016
109	1255	GROUND CHARACTERISTICS OF THE O.S.	87.619
98	1140	CONFIGURATION FOR PROV. WORKING	88.069
115	1320	SORTING SYSTEMS AND WORK ON FILES	88.776
99	1150	☆ PROVIS. WORK FOR PIRC DEVELOPMENT	90.342
110	1260	JCL , MEASURING & ACCOUNTING PROGRAMS	90.619
97	1130	PROPOSED APPPLICATION FOR PROVIS. WORK.	90.920
88	1050	TRAINING AND ASSISTANCE BY SYST. ENG.	91.789
92	1090	POTENTIALITIES OF THE LOCAL BR. OFF.	93.233
103	1200	PROCESSOR ORGANIZATION	93.612
116	1322	APPLICATION SYSTEMS	94.183
107	1240	INPUT-OUTPUT ORGANS	95.127
84	1010	TRAINING PROPOSALS	100.000
85	1020	TOTAL TRAINING POSSIBILITIES	100.000
86	1030	TRAINING	100.000
96	1120	RUNNING PART DURING PROVIS. WORKING	100.000
105	1220	DISK MEMORY	100.000
113	1300	RPG , ASSEMBLER AND PL/1	100.000

SE HISTOGRAM

```
 0 -  10     0 I
10 -  20     2 I☆☆
20 -  30     0 I
30 -  40     1 I☆
40 -  50     1 I☆
50 -  60     2 I☆☆
60 -  70     4 I☆☆☆☆
70 -  80     7 I☆☆☆☆☆☆☆
80 -  90     4 I☆☆☆☆
90 - 100    14 I☆☆☆☆☆☆☆☆☆☆☆☆☆☆
```

EFFECTIVENESS FROM 10 TO 20

COBOL AND FORTRAN
PROGRAMMING LANGUAGES
EFFECTIVENESS FROM 30 TO 40

☆ SOFTWARE
EFFECTIVENESS FROM 40 TO 50

USER'S MEMORY
EFFECTIVENESS FROM 50 TO 60

O.S.
LOCAL BRANCH OFFICE
EFFECTIVENESS FROM 60 TO 70

☆☆☆ GLOBAL COMPUTER
BUC FOR RUNNING
PERFORMANCE , SPEED AND MEMORY USAGE
☆ COMPUTER PRODUCTIVITY
EFFECTIVENESS FROM 70 TO 80

MEASURED PERFORMANCE OF THE COMPUTER
MEMORY OF 9-CHANNEL TAPES
SYSTEM ENGINEERS ASSISTANCE
PROPOSED MAINTENANCE
GLOBAL ASSISTANCE
☆ HARDWARE
☆ COMPUTER CONSTRUCTOR & REPRESENTATION
EFFECTIVENESS FROM 80 TO 90

CONDITIONAL MAINTENANCE
GROUND CHARACTERISTICS OF THE O.S.
CONFIGURATION FOR PROV. WORKING
SORTING SYSTEMS AND WORK ON FILES
EFFECTIVENESS FROM 90 TO 100

☆ PROVIS. WORK FOR PIRC DEVELOPMENT
JCL , MEASURING & ACCOUNTING PROGRAMS
PROPOSED APPPLICATION FOR PROVIS. WORK.
TRAINING AND ASSISTANCE BY SYST. ENG.
POTENTIALITIES OF THE LOCAL BR. OFF.
PROCESSOR ORGANIZATION
APPLICATION SYSTEMS
INPUT-OUTPUT ORGANS
TRAINING PROPOSALS
TOTAL TRAINING POSSIBILITIES
TRAINING
RUNNING PART DURING PROVIS. WORKING
DISK MEMORY
RPG , ASSEMBLER AND PL/1

SYSTEM NUMBER 5

ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	X	E
38	360	AVAILABLE TOOLS FOR PROV. WORKING	0.000	0.000
70	770	MEASURING PROGRAM (SOFTWARE MONITOR)	0.000	0.000
76	840	PL/1 COMPILER	0.000	0.000
9	90	POSSIB. OF CHOOSING THE SYST. ENG. ASS.	1.000	1.000
21	195	VALIDATION TEST WITH BENCHMARKS	1.000	1.000
73	810	SPEED OF COBOL COMPILER	1.190	1.190
27	250	PACKAGE OF EMPLOYEE'S MANAGEMENT	10.000	10.000
28	260	PACKAGE OF PRODUCTS INVENTORY MANAGEMENT	10.000	10.000
29	270	PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT	10.000	10.000
30	280	PACKAGE OF GROSS MATERIAL INVENT. MANAG.	10.000	10.000
31	290	PACKAGE OF FINANCIAL ACCOUNTING	10.000	10.000
32	300	PACKAGE OF RESOURCES PLANNING	10.000	10.000
33	310	PACKAGE OF TRANSACTIONS' ACCOUNTING	10.000	10.000
34	320	PACKAGE OF CIVIL ENGINEERING	10.000	10.000
41	420	SPEED OF CENTRAL PROCESSOR	2.720	14.000
20	193	BENCHMARKS EXECUTION QUALITY	20.000	20.000
35	330	DISTANCE OF THE PROVIS. CONFIGURATION	310.000	22.500
36	340	NUMBER OF BUC FOR PROVIS. WORKING	1.000	25.000
58	600	GLOBAL SPEED OF THE LINE PRINTER	450.000	30.000
22	200	NUMBER OF SAME OR GREATER CONFIG. IN Y U	1.000	33.333

24	220	DISTANCE OF THE NEXT BACK-UP CONFIGURAT.	310.000	38.000
23	210	NUMBER OF AVAILABLE BACK-UP CONFIGURAT .	1.000	40.000
71	790	O.S. GLOBAL EFFICIENCY	42.700	42.700
61	700	WAY OF GIVING MEMORY TO THE PROGRAM	3.000	42.857
80	880	EFFECTIVENESS OF THE SORTING PROGRAM	3.800	44.000
2	20	TRAINING FOR SYSTEM ANALYSTS	47.000	47.000
16	160	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	48.333
5	50	NUMB. & CONTENTS OF AVAILABLE TRAININGS	50.000	50.000
6	60	QUALITY & QUANTITY OF TRAIN. LITERATURE	50.000	50.000
7	70	PERIOD OF GUARANT. SYST. ING. ASSIST.	50.000	50.000
10	100	ARRIVAL SPEED ON CALL	5.000	50.000
54	555	CONTROL FLAGS ON THE 9-CHANN. TAPES	1.000	50.000
83	910	EQUIPMENT FOR THE PROGRAMMATION SYSTEM	50.000	50.000
18	180	NUMBER OF ASSISTANCE TECHNICIANS	3.000	60.000
19	190	REPRESENTATION ORGANIZ. IN YUGOSLAVIA	60.000	60.000
74	820	FORTRAN COMPILER	60.000	60.000
3	30	TRAINING FOR SYSTEM PROGRAMMERS	64.000	64.000
39	400	PRODUCTIVITY FOR THE MONOPROGR. TEST	1.570	71.500
12	120	DISTANCE OF THE NEXT SELLER'S OFFICE	310.000	76.750
43	450	NUMBER OF INDEX REGISTERS	8.000	80.000
53	550	TRANSFER SPEED OF THE 9-CHANN. TAPES	40.000	80.000
59	610	BUILDING PRINCIPLE OF THE LINE PRINTER	2.000	80.000
68	750	ACCOUNTING , STATISTICAL & COMPUT. MESS.	2.000	80.000
45	470	REPERTOIR OF MACHINE INSTRUCTIONS	86.000	86.000
44	460	NUMBER OF ACCUMULATORS	12.000	90.000
62	705	USER PROGRAM ADAPTATION TO THE MEMORY	4.000	90.000
72	800	COBOL COMPILER	90.000	90.000
81	890	DATA STRUCTURES	90.000	90.000
82	900	SYSTEMS FOR DATA BASE MANAGING	90.000	90.000
40	410	PRODUCTIVITY FOR THE MULTIPROGR. TEST	1.180	91.000
75	830	SPEED OF FORTRAN COMPILER	1.180	98.000
1	10	TRAINING FOR PROGRAMMERS	100.000	100.000
4	40	TRAINING FOR OPERATORS	100.000	100.000
8	80	POSSIBILITY OF CHANGING THE SYST. ENG.	1.000	100.000
11	110	ASSISTANCE IN NIGHT SHIFT , WEEK-END ...	1.000	100.000
13	130	ASSISTANCE FOR EXTERNAL HARDWARE	1.000	100.000
14	140	PERIOD OF FIXED ASSISTANCE COSTS	5.000	100.000
15	150	PERIOD OF CONTRACTUAL ASSISTANCE	10.000	100.000
17	170	NUMBER OF SYST. ING. AT THIS OFFICE	5.000	100.000
25	230	COMPATIBILITY PC-BUC (MACHINE LANG.)	1.000	100.000
26	240	PACKAGE OF WORKER'S MANAGEMENT	10.000	100.000
37	350	OTHER PACKAGES FOR THE COMPUTER USER	100.000	100.000
42	430	MEMORY CONSUMPTION IN THE MONOPR. TEST	1.000	100.000
46	480	ADDRESS NUMBER PER INSTRUCTION	2.000	100.000
47	490	INSTRUCTION FORMAT	3.000	100.000
48	500	CAPACITY OF USERS' MEMORY	100.000	100.000
49	510	POSSIBILITY OF MEMORY EXTENSION	480.000	100.000
50	515	NUMBER OF ACCESSES TO DISK MEMORY	4.000	100.000
51	520	TOTAL CAPACITY OF DISK MEMORIES	90.000	100.000
52	530	POSSIBILITY OF DISK MEMORY EXTENSION	480.000	100.000
55	560	AUTOMATIC MONTAGE OF THE 9-CH. TAPES	1.000	100.000
56	580	DECIDING THE DATA PREPARATION AND INPUT	100.000	100.000
57	590	DECIDING THE PROGRAM PREP. AND INPUT	100.000	100.000
60	620	OUTPUT ON THE DIRECTING CONSOLE	3.000	100.000

63	710	VIRTUAL INPUT ORGANS (SPOOL-IN)	1.000	100.000
64	715	VIRTUAL OUTPUT ORGANS (SPOOL-OUT)	1.000	100.000
65	720	PROGRAM INVARIANCE DUE TO CAPACITY	1.000	100.000
66	730	JCL (USER'S INSTRUCTIONS)	100.000	100.000
67	740	CONTROL LANGUAGE FOR THE OPERATOR	100.000	100.000
69	760	ACCOUNTING OF FAILURES AND ERRORS	4.000	100.000
77	850	RPG OR SIMILAR LANGUAGE	90.000	100.000
78	860	SYMBOLIC MACHINE LANGUAGE	2.000	100.000
79	870	POSSIBILITY OF SORTING PROGRAMS	100.000	100.000

EE HISTOGRAM

```
 0 -  10   14 I**************
10 -  20    2 I**
20 -  30    3 I***
30 -  40    3 I***
40 -  50   11 I***********
50 -  60    3 I***
60 -  70    1 I*
70 -  80    6 I******
80 -  90    6 I******
90 - 100   34 I**********************************
```

EFFECTIVENESS FROM 0 TO 10

AVAILABLE TOOLS FOR PROV. WORKING
MEASURING PROGRAM (SOFTWARE MONITOR)
PL/1 COMPILER
POSSIB. OF CHOOSING THE SYST. ENG. ASS.
VALIDATION TEST WITH BENCHMARKS
SPEED OF COBOL COMPILER
PACKAGE OF EMPLOYEE'S MANAGEMENT
PACKAGE OF PRODUCTS INVENTORY MANAGEMENT
PACKAGE OF CUSTOMER'S CREDIT MANAGEMENT
PACKAGE OF GROSS MATERIAL INVENT. MANAG.
PACKAGE OF FINANCIAL ACCOUNTING
PACKAGE OF RESOURCES PLANNING
PACKAGE OF TRANSACTIONS' ACCOUNTING
PACKAGE OF CIVIL ENGINEERING
EFFECTIVENESS FROM 10 TO 20

SPEED OF CENTRAL PROCESSOR
BENCHMARKS EXECUTION QUALITY
EFFECTIVENESS FROM 20 TO 30

DISTANCE OF THE PROVIS. CONFIGURATION
NUMBER OF BUC FOR PROVIS. WORKING
GLOBAL SPEED OF THE LINE PRINTER
EFFECTIVENESS FROM 30 TO 40

NUMBER OF SAME OR GREATER CONFIG. IN Y U
DISTANCE OF THE NEXT BACK-UP CONFIGURAT.
NUMBER OF AVAILABLE BACK-UP CONFIGURAT .
EFFECTIVENESS FROM 40 TO 50

O.S. GLOBAL EFFICIENCY

WAY OF GIVING MEMORY TO THE PROGRAM
EFFECTIVENESS OF THE SORTING PROGRAM
TRAINING FOR SYSTEM ANALYSTS
DISTANCE OF THE NEXT SELLER'S OFFICE
NUMB. & CONTENTS OF AVAILABLE TRAININGS
QUALITY & QUANTITY OF TRAIN. LITERATURE
PERIOD OF GUARANT. SYST. ING. ASSIST.
ARRIVAL SPEED ON CALL
CONTROL FLAGS ON THE 9-CHANN. TAPES
EQUIPMENT FOR THE PROGRAMMATION SYSTEM
EFFECTIVENESS FROM 50 TO 60

NUMBER OF ASSISTANCE TECHNICIANS
REPRESENTATION ORGANIZ. IN YUGOSLAVIA
FORTRAN COMPILER
EFFECTIVENESS FROM 60 TO 70

TRAINING FOR SYSTEM PROGRAMMERS
EFFECTIVENESS FROM 70 TO 80

PRODUCTIVITY FOR THE MONOPROGR. TEST
DISTANCE OF THE NEXT SELLER'S OFFICE
NUMBER OF INDEX REGISTERS
TRANSFER SPEED OF THE 9-CHANN. TAPES
BUILDING PRINCIPLE OF THE LINE PRINTER
ACCOUNTING , STATISTICAL & COMPUT. MESS.
EFFECTIVENESS FROM 80 TO 90

REPERTOIR OF MACHINE INSTRUCTIONS
NUMBER OF ACCUMULATORS
USER PROGRAM ADAPTATION TO THE MEMORY
COBOL COMPILER
DATA STRUCTURES
SYSTEMS FOR DATA BASE MANAGING
EFFECTIVENESS FROM 90 TO 100

PRODUCTIVITY FOR THE MULTIPROGR. TEST
SPEED OF FORTRAN COMPILER
TRAINING FOR PROGRAMMERS
TRAINING FOR OPERATORS
POSSIBILITY OF CHANGING THE SYST. ENG.
ASSISTANCE IN NIGHT SHIFT , WEEK-END ...
ASSISTANCE FOR EXTERNAL HARDWARE
PERIOD OF FIXED ASSISTANCE COSTS
PERIOD OF CONTRACTUAL ASSISTANCE
NUMBER OF SYST. ING. AT THIS OFFICE
COMPATIBILITY PC-BUC (MACHINE LANG.)
PACKAGE OF WORKER'S MANAGEMENT
OTHER PACKAGES FOR THE COMPUTER USER
MEMORY CONSUMPTION IN THE MONOPR. TEST
ADDRESS NUMBER PER INSTRUCTION
INSTRUCTION FORMAT
CAPACITY OF USERS' MEMORY
POSSIBILITY OF MEMORY EXTENSION

NUMBER OF ACCESSES TO DISK MEMORY
TOTAL CAPACITY OF DISK MEMORIES
POSSIBILITY OF DISK MEMORY EXTENSION
AUTOMATIC MONTAGE OF THE 9-CH. TAPES
DECIDING THE DATA PREPARATION AND INPUT
DECIDING THE PROGRAM PREP. AND INPUT
OUTPUT ON THE DIRECTING CONSOLE
VIRTUAL INPUT ORGANS (SPOOL-IN)
VIRTUAL OUTPUT ORGANS (SPOOL-OUT)
PROGRAM INVARIANCE DUE TO CAPACITY
JCL (USER'S INSTRUCTIONS)
CONTROL LANGUAGE FOR THE OPERATOR
ACCOUNTING OF FAILURES AND ERRORS
RPG OR SIMILAR LANGUAGE
SYMBOLIC MACHINE LANGUAGE
POSSIBILITY OF SORTING PROGRAMS
* SUBSYSTEM EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	E
98	1140	CONFIGURATION FOR PROV. WORKING	1.436
99	1150	☆ PROVIS. WORK FOR PIRC DEVELOPMENT	6.968
97	1130	PROPOSED APPPLICATION FOR PROVIS. WORK.	11.039
96	1120	RUNNING PART DURING PROVIS. WORKING	13.203
112	1290	COBOL AND FORTRAN	14.574
114	1310	PROGRAMMING LANGUAGES	14.574
118	1340	☆☆☆ GLOBAL COMPUTER	26.395
110	1260	JCL , MEASURING & ACCOUNTING PROGRAMS	27.154
93	1095	LOCAL BRANCH OFFICE	27.669
117	1330	☆ SOFTWARE	28.670
87	1040	SYSTEM ENGINEERS ASSISTANCE	40.802
111	1280	O.S.	42.744
94	1100	BUC FOR RUNNING	49.827
85	1020	TOTAL TRAINING POSSIBILITIES	50.000
95	1110	☆ COMPUTER CONSTRUCTOR & REPRESENTATION	51.260
115	1320	SORTING SYSTEMS AND WORK ON FILES	51.450
107	1240	INPUT-OUTPUT ORGANS	52.277
88	1050	TRAINING AND ASSISTANCE BY SYST. ENG.	55.574
116	1322	APPLICATION SYSTEMS	60.050
89	1060	PROPOSED MAINTENANCE	60.544
86	1030	TRAINING	61.801
92	1090	POTENTIALITIES OF THE LOCAL BR. OFF.	64.706
91	1080	GLOBAL ASSISTANCE	66.740
101	1180	PERFORMANCE , SPEED AND MEMORY USAGE	68.330
109	1255	GROUND CHARACTERISTICS OF THE O.S.	70.762
102	1190	☆ COMPUTER PRODUCTIVITY	71.936
84	1010	TRAINING PROPOSALS	73.602
106	1230	MEMORY OF 9-CHANNEL TAPES	75.371
100	1160	MEASURED PERFORMANCE OF THE COMPUTER	75.883
113	1300	RPG , ASSEMBLER AND PL/1	80.000
108	1250	☆ HARDWARE	83.638
103	1200	PROCESSOR ORGANIZATION	91.038
90	1070	CONDITIONAL MAINTENANCE	100.000
104	1210	USER'S MEMORY	100.000

105 1220 DISK MEMORY 100.000

SE HISTOGRAM

```
 0 -  10    2 I**
10 -  20    4 I****
20 -  30    4 I****
30 -  40    0 I
40 -  50    4 I****
50 -  60    4 I****
60 -  70    6 I******
70 -  80    5 I*****
80 -  90    2 I**
90 - 100    4 I****
```

EFFECTIVENESS FROM 0 TO 10

CONFIGURATION FOR PROV. WORKING
* PROVIS. WORK FOR PIRC DEVELOPMENT
EFFECTIVENESS FROM 10 TO 20

PROPOSED APPPLICATION FOR PROVIS. WORK.
RUNNING PART DURING PROVIS. WORKING
COBOL AND FORTRAN
PROGRAMMING LANGUAGES
EFFECTIVENESS FROM 20 TO 30

*** GLOBAL COMPUTER
JCL , MEASURING & ACCOUNTING PROGRAMS
LOCAL BRANCH OFFICE
* SOFTWARE
EFFECTIVENESS FROM 40 TO 50

SYSTEM ENGINEERS ASSISTANCE
O.S.
BUC FOR RUNNING
TOTAL TRAINING POSSIBILITIES
EFFECTIVENESS FROM 50 TO 60

---

* COMPUTER CONSTRUCTOR & REPRESENTATION
SORTING SYSTEMS AND WORK ON FILES
INPUT-OUTPUT ORGANS
TRAINING AND ASSISTANCE BY SYST. ENG.
EFFECTIVENESS FROM 60 TO 70

APPLICATION SYSTEMS
PROPOSED MAINTENANCE
TRAINING
POTENTIALITIES OF THE LOCAL BR. OFF.
GLOBAL ASSISTANCE
PERFORMANCE , SPEED AND MEMORY USAGE
EFFECTIVENESS FROM 70 TO 80

GROUND CHARACTERISTICS OF THE O.S.
* COMPUTER PRODUCTIVITY
TRAINING PROPOSALS

MEMORY OF 9-CHANNEL TAPES
MEASURED PERFORMANCE OF THE COMPUTER
EFFECTIVENESS FROM 80 TO 90

RPG , ASSEMBLER AND PL/1
✡ HARDWARE
EFFECTIVENESS FROM 90 TO 100

PROCESSOR ORGANIZATION
CONDITIONAL MAINTENANCE
USER'S MEMORY
DISK MEMORY
END OF SEL PROGRAM EXECUTION
✡EXIT

```
C ----------------------------------------
C EXEMPLE CPIRC ( COMPLEMENT A PIRC )
C  LE PRESENT PROGRAMME EN LANGAGE SEL
C  EST UN EXEMPLE ARTIFICIEL SIMPLEMENT
C  DESTINE A ILLUSTRER LES INSTRUCTIONS NON
C  UTILISEES PAR L'EXEMPLE PRINCIPAL PIRC
C ----------------------------------------
```

```
USER I.D.- FUN96A
ACCT. NO.- MEMO
FILENAME - VBA.ARN.CPIRC
MAILING ADDR.- FACULTE  N.D.
DATE  -  09/09/77
```

```
☆JOB
☆SEL,LIS
  INPUT ELC
  INPUT CAS
  DISPLAY ( ELC ) 5
  READ X(1,K99)
  EFFECTIVENESS
  TITLE 3 , ' RESULTATS DE L'EVALUATION '
  OUTPUT ( E ) SORT , HISTOGRAM
  TITLE 3 , ' ANALYSE DE SENSIBILITE POUR LE CRITERE GLOBAL '
  SENSITIVITY ANALYSIS ( ALL ) DISPLAY , PRINT
1 K1=1
  DATA FOR CE (K1) , PRINT
  BRANCH TO /1/ K99,1
2 K1=K99+1
  OPTIMIZE (K1) , PRINT
  BRANCH TO /2/ K100,1
  PRESENT VALUE = A1
  TITLE 2 , ' VALEUR TOTALE ACTUALISEE @ 6 % : '
  PRINT A1
  STOP
  END
☆NEW PAGE
☆PRINT INSEL PROGRAM
☆NEW PAGE
☆EXECUTE INSEL PROGRAM
LIST
  10       CE NUMERO 10
  20       CE NUMERO 20
  30       CE NUMERO 30
  40       CE NUMERO 40
  50       CE NUMERO 50
  60       CE NUMERO 60
         END
LIST
  70 A     CNE 70 ( CE 10 ET 20 )
  80 A     CNE 80 ( CE 30 ET 40 )
  90 C-+   CNE 90 ( CE 50 ET 60 )
 100 A     CNE 100 ( CE 70 ET 80 )
 110 CA    CRITERE GLOBAL ( CE 90 ET 100 )
         END
```

Données de l'instruction INPUT CAS

```
  0 0 10099
  0 0 10099
  0 0  8099
  0 0  8099
  0 0  3080  9099
  0 0  3080  9099
```

Données de l'instruction INPUT ELC

```
1050  2050
3080  4020
5060  6040    50    50
7030  8070
9050 10050
```

```
  80   70   60   90   85   40
   2
       55       100
      100      1000
   2
       55       100
```

Données de l'instruction READ

Données de l'instruction DATA

```
      100          1000
  3
           30           80          100
           50          100          500
  3
           30           80          100
           50          100          500
  3
           10           60          100
         1000         1500         2000
  3
           10           60          100
         1000         1500         2000
CONSTRUCTION FINANCIERE POUR UN ORDINATEUR
AU TAUX D'ACTUALISATION DE 6 % ANNUELS ( CAS REEL
  93214.2     6      6      0      0
 131836.2     6      6      1     13
 113626.0     6      6      4      4
 147150.8     6      6      5     13
   6430.0     6      1      1     84

*EXIT
```

Fin des données de l'instruction DATA

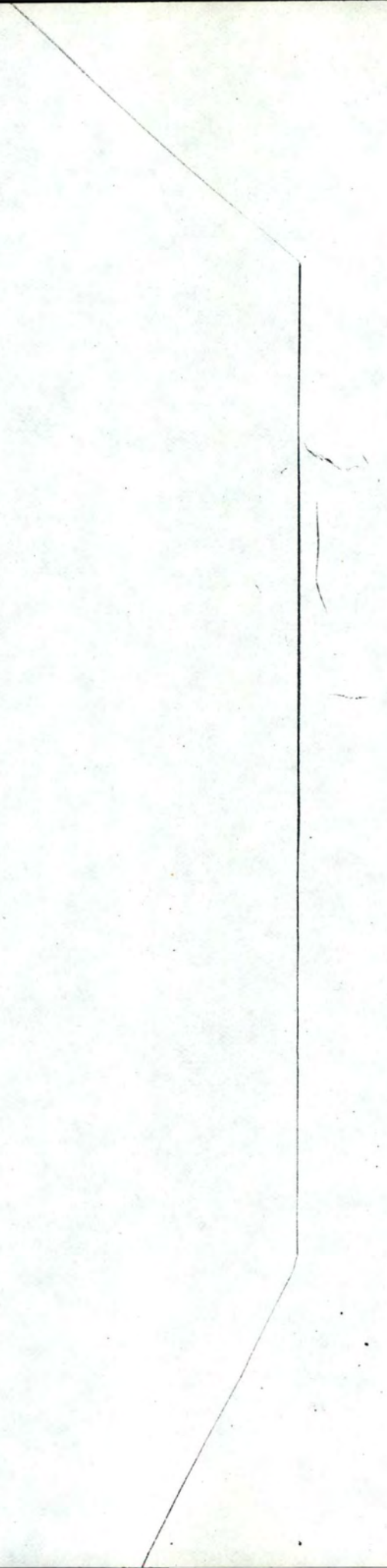
Données de l'instruction PRESENT VALUE

```
    ---------------------------------
    I  SYSTEMS EVALUATION LANGUAGE I
    I                               I
    I** VERSION SEL77 (ENGLISH) ***I
    I                               I
    I DUJMOVIC/V.BASTELAER/ARNOTTE I
    ---------------------------------
C ----------------------------------------
C EXEMPLE CPIRC ( COMPLEMENT A PIRC )
C  LE PRESENT PROGRAMME EN LANGAGE SEL
C  EST UN EXEMPLE ARTIFICIEL SIMPLEMENT
C  DESTINE A ILLUSTRER LES INSTRUCTIONS NON
C  UTILISEES PAR L'EXEMPLE PRINCIPAL PIRC
C ----------------------------------------
*JOB
*SEL,LIS
    INPUT ELC
    INPUT CAS
    DISPLAY ( ELC ) 5
    READ X(1,K99)
    EFFECTIVENESS
    TITLE 3 , ' RESULTATS DE L'EVALUATION '
    OUTPUT ( E ) SORT , HISTOGRAM
    TITLE 3 , ' ANALYSE DE SENSIBILITE POUR LE CRITERE GLOBAL '
    SENSITIVITY ANALYSIS ( ALL ) DISPLAY , PRINT
  1 K1=1
    DATA FOR CE (K1) , PRINT
    BRANCH TO /1/ K99,1
  2 K1=K99+1
    OPTIMIZE (K1) , PRINT
    BRANCH TO /2/ K100,1
    PRESENT VALUE = A1
    TITLE 2 , ' VALEUR TOTALE ACTUALISEE @ 6 % : '
    PRINT A1
    STOP
    END
NO ERROR IN ABOVE PROGRAM
END OF TRANSLATION
*NEW PAGE
*PRINT INSEL PROGRAM
INSEL PROGRAM LISTING
```

NO.	NAME	OP	D1	D2	D3	D4	D5	D6	D7	D8
1	INP	9	0							
2	INP	9	1							
3	DIS	13	1	5						
4	REA	4	300	1	-99	.				
5	EFF	10	-1	1						
6	TIT	2	3	1	27					
7	OUT	11	1111							
8	TIT	2	3	28	74					
9	SEN	12	0	0	11					
10	ARI	6	1001	1	1	0	0	0		
11	DAT	14	-1	1						

USER I.D.- FUN96A
ACCT. NO.- MEMO
FILENAME - VBA.ARN.S1
MAILING ADDR.- FACULTE
DATE - 09/09/77

Voir page 185 bis



```
 12   BRA          7     10       3    1099       7      1      1      0
 13   ARI          6   1001       3    1099       1      1      1
 14   OPT         15     -1       1     160
 15   BRA          7     13       3    1100       7      1      1      0
 16   PRE         16   2001
 17   TIT          2      2      75     108
 18   PRI          5    200       1       1
 19   STO          1
```

```
*NEW PAGE
*EXECUTE INSEL PROGRAM
```

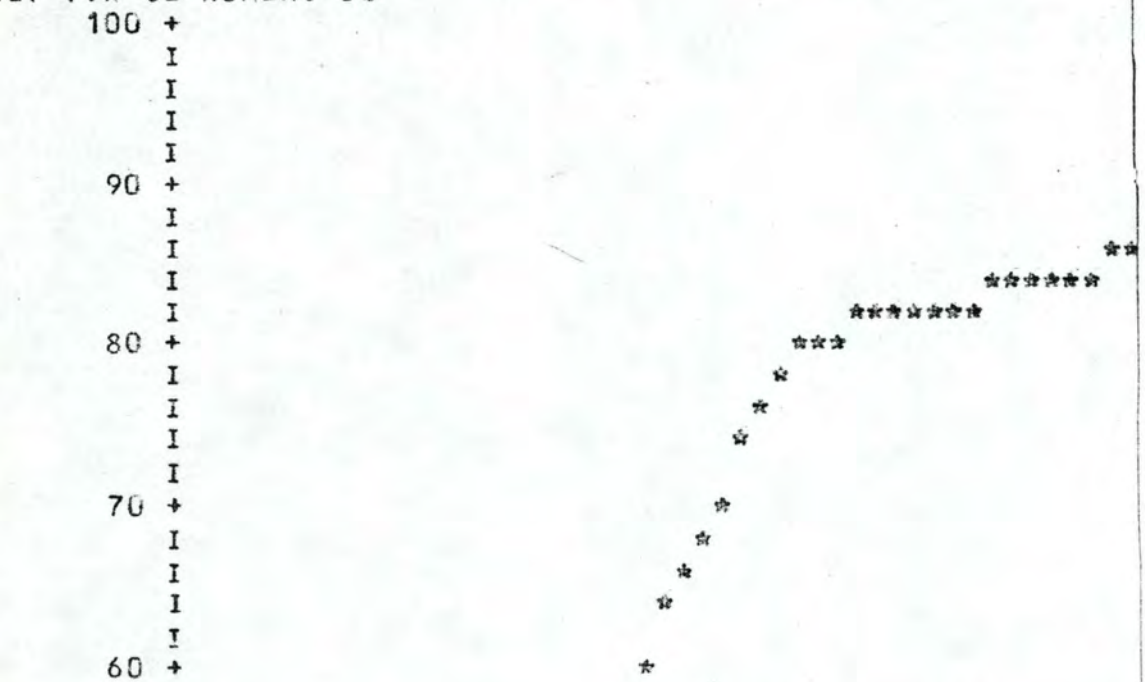
ELEMENTARY CRIT

NO.	BLOCK	E	COMPONENT FOR EVALUATION
1	10	0.0	CE NUMERO 10
2	20	0.0	CE NUMERO 20
3	30	0.0	CE NUMERO 30
4	40	0.0	CE NUMERO 40
5	50	0.0	CE NUMERO 50
6	60	0.0	CE NUMERO 60

NON-ELEMENTARY

NO.	BLOCK	OP.	COMPONENT FOR EVALUATION
7	70	A	CNE 70 (CE 10 ET 20)
8	80	A	CNE 80 (CE 30 ET 40)
9	90	*C-+	CNE 90 (CE 50 ET 60)
10	100	A	CNE 100 (CE 70 ET 80)
11	110	CA	CRITERE GLOBAL (CE 90 ET 100)

EEF FOR CE NUMERO 50



0

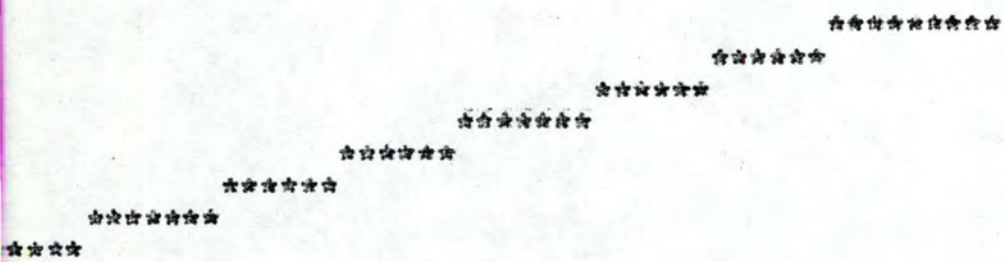
0

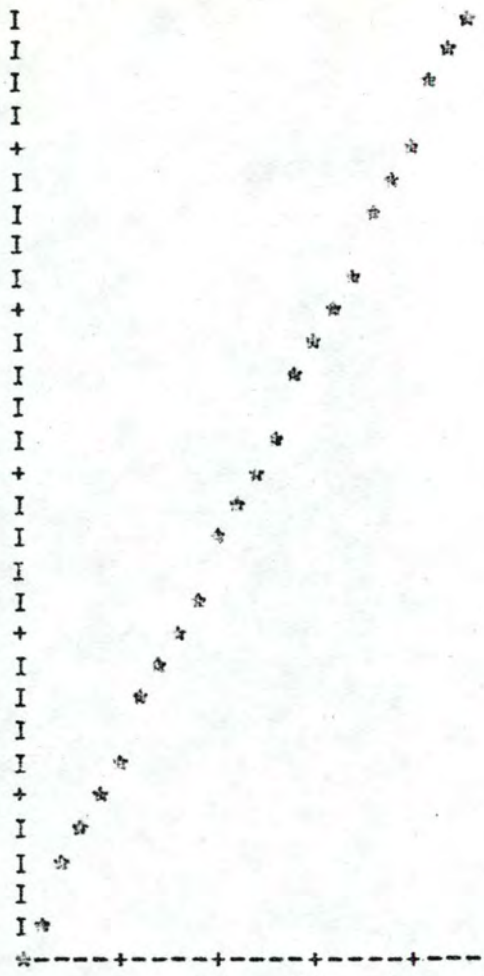
RIA

INPUT 1	INPUT 2	INPUT 3	INPUT 4	INPUT 5
0.000 0	100.000 100			
0.000 0	100.000 100			
0.000 0	80.000 100			
0.000 0	80.000 100			
0.000 0	30.000 80	90.000 100		
0.000 0	30.000 80	90.000 100		

RITERIA

INPUT 1	INPUT 2	INPUT 3	INPUT 4	INPUT 5
10 (50)	20 (50)			
30 (80)	40 (20)			
50 (60)	60 (40)	0 (50)	0 (50)	
70 (30)	80 (70)			
90 (50)	100 (50)			





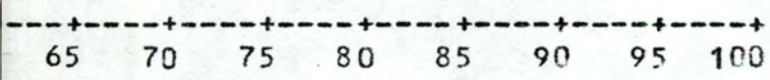
RESULTATS DE L'EVALUATION
ELEMENTARY EFFECTIVENESSES

(SORTED)

NO.	BLOCK	COMPONENT FOR EVALUATION	X	E
2	20	CE NUMERO 20	70.000	70.000
3	30	CE NUMERO 30	60.000	75.000
1	10	CE NUMERO 10	80.000	80.000
6	60	CE NUMERO 60	40.000	83.333
5	50	CE NUMERO 50	85.000	98.333
4	40	CE NUMERO 40	90.000	100.000

EE HISTOGRAM

0 - 10	0	I
10 - 20	0	I
20 - 30	0	I
30 - 40	0	I
40 - 50	0	I
50 - 60	0	I
60 - 70	1	I*
70 - 80	2	I**
80 - 90	1	I*


65
70
75
80
85
90
95
100

```
90 - 100     2 I**
EFFECTIVENESS FROM 60 TO  70
----------------------------
CE NUMERO 20
EFFECTIVENESS FROM 70 TO  80
----------------------------
CE NUMERO 30
CE NUMERO 10
EFFECTIVENESS FROM 80 TO  90
----------------------------
CE NUMERO 60
EFFECTIVENESS FROM 90 TO 100
----------------------------
CE NUMERO 50
CE NUMERO 40
SUBSYSTEM EFFECTIVENESSES
                              (SORTED)
 NO. BLOCK  COMPONENT FOR EVALUATION                                  E
-------------------------------------------------------------------------
   7   70  CNE 70 ( CE 10 ET 20 )                                  75.000
  10  100  CNE 100 ( CE 70 ET 80 )                                 78.500
   8   80  CNE 80 ( CE 30 ET 40 )                                  80.000
  11  110  CRITERE GLOBAL ( CE 90 ET 100 )                         84.935
   9   90  CNE 90 ( CE 50 ET 60 )                                  92.333
-------------------------------------------------------------------------
SE HISTOGRAM
 0 -  10     0 I
10 -  20     0 I
20 -  30     0 I
30 -  40     0 I
40 -  50     0 I
50 -  60     0 I
60 -  70     0 I
70 -  80     3 I***
80 -  90     1 I*
90 - 100     1 I*
EFFECTIVENESS FROM 70 TO  80
----------------------------
CNE 70 ( CE 10 ET 20 )
CNE 100 ( CE 70 ET 80 )
CNE 80 ( CE 30 ET 40 )
EFFECTIVENESS FROM 80 TO  90
----------------------------
CRITERE GLOBAL ( CE 90 ET 100 )
EFFECTIVENESS FROM 90 TO 100
----------------------------
CNE 90 ( CE 50 ET 60 )
 ANALYSE DE SENSIBILITE POUR LE CRITERE GLOBAL
SENSITIVITY ANALYSIS
--------------------
INPUT =    1 = CE NUMERO 10
OUTPUT=   11 = CRITERE GLOBAL ( CE 90 ET 100 )
          IN =  0.0  1.0  2.0  3.0  4.0  5.0  7.0  9.0 12.0 15.0
         OUT = 77.6 77.7 77.8 77.9 78.0 78.1 78.3 78.5 78.8 79.1
```

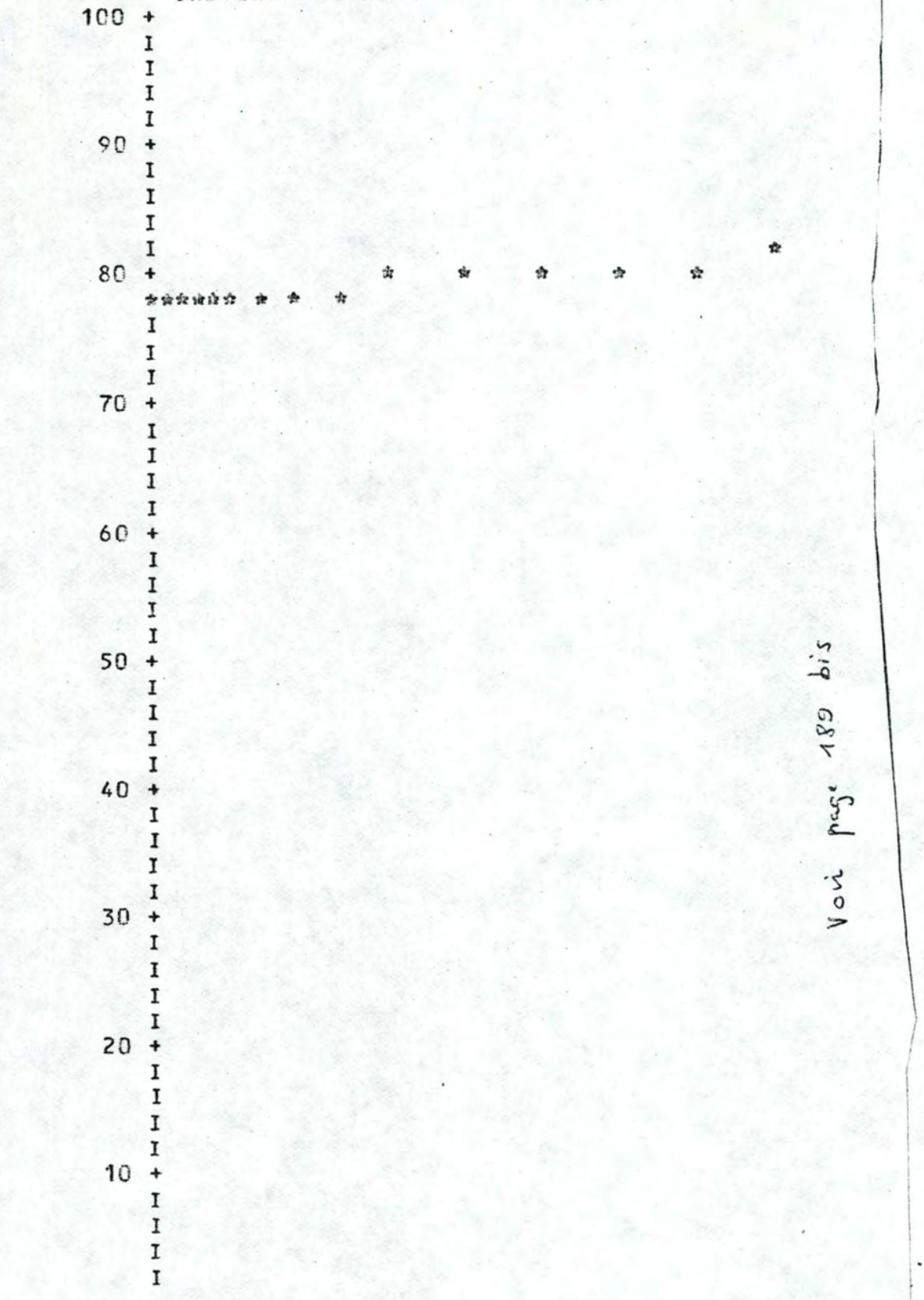
Voir page 188 bis

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
79.5	80.0	80.5	80.9	81.4	82.3	83.2	84.1	84.9	85.8	86.6

D(OUT)/D(IN)= 0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.10

ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-
9.02	10.41	5.07	48.67	1.69	7.33

CRITERE GLOBAL (CE 90 ET 100)



Voir page 189 bis

0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.09 0.09 0.09 0.09 0.09

PI

81.25

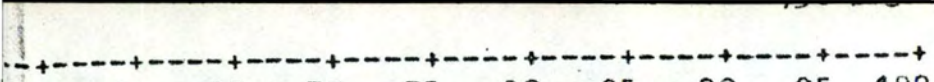
```
  0 +----+----+----+----+----+----+----+----+----+----+---
          5   10   15   20   25   30   35   40   45   50
```

SENSITIVITY ANALYSIS

```
INPUT =    2 = CE NUMERO 20
OUTPUT=   11 = CRITERE GLOBAL ( CE 90 ET 100 )
         IN =  0.0  1.0  2.0  3.0  4.0  5.0  7.0  9.0 12.0 15.0
        OUT = 78.6 78.7 78.8 78.9 79.0 79.1 79.2 79.4 79.7 80.0
D(OUT)/D(IN)= 0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.10 0.10
 ALPHA    ALPHA REL    BETA    BETA REL    DELTA+    DELTA-      PI
   8.88       10.15    4.94       48.69      2.52      6.36   71.63
```

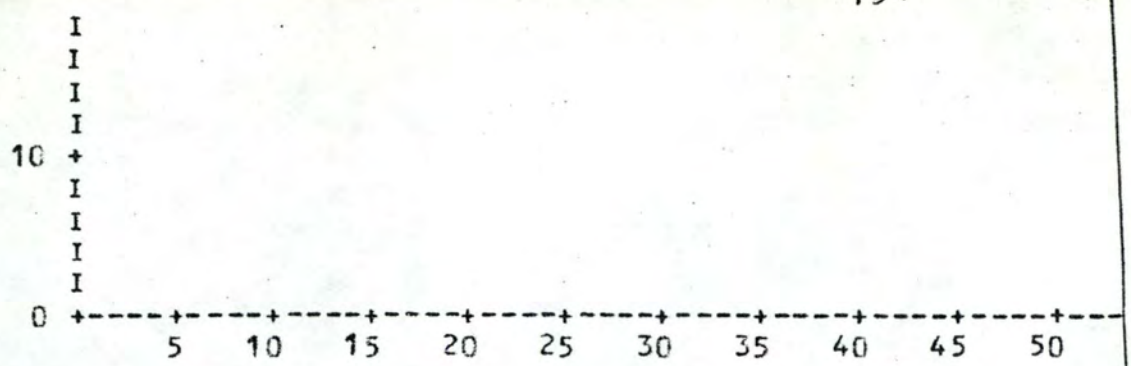
CRITERE GLOBAL (CE 90 ET 100)

```
 100 +
     I
     I
     I
     I
     I
  90 +
     I
     I
     I                                                        *
     I                                    *     *     *
  80 +     * * * * * *     *     *
     *****
     I
     I
     I
  70 +
     I
     I
     I
     I
  60 +
     I
     I
     I
     I
  50 +
     I
     I
     I
     I
  40 +
     I
     I
     I
     I
  30 +
     I
     I
     I
     I
  20 +
```



CE NUMERO 10

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
80.5	80.9	81.4	81.8	82.3	83.2	84.1	84.9	85.8	86.6	87.5
0.10	0.10	0.10	0.10	0.10	0.09	0.09	0.09	0.09	0.09	0.09

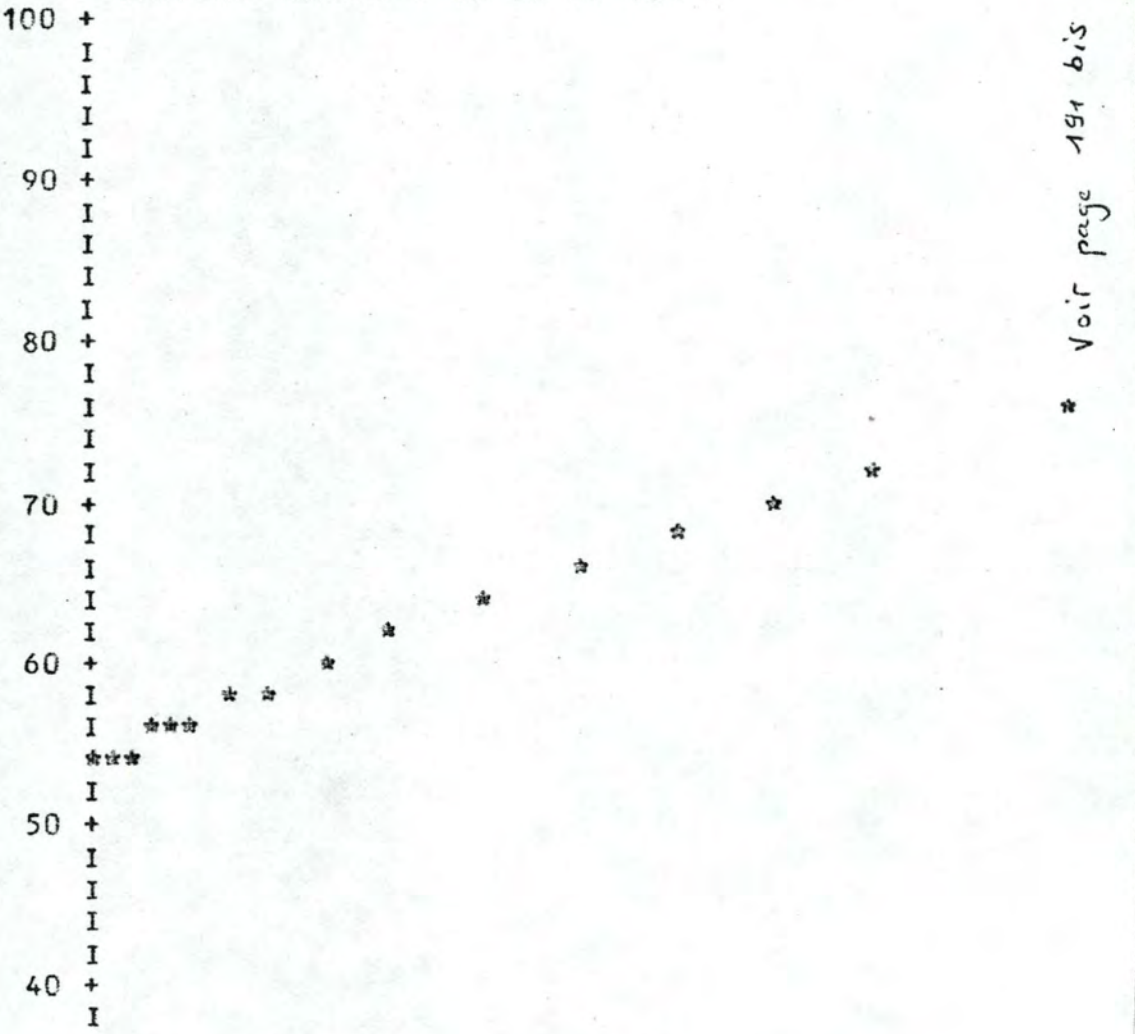


SENSITIVITY ANALYSIS

INPUT = 3 = CE NUMERO 30
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	53.8	54.3	54.9	55.4	55.9	56.5	57.5	58.5	60.0	61.5
D(OUT)/D(IN)=	0.55	0.55	0.54	0.54	0.53	0.53	0.52	0.51	0.50	0.49

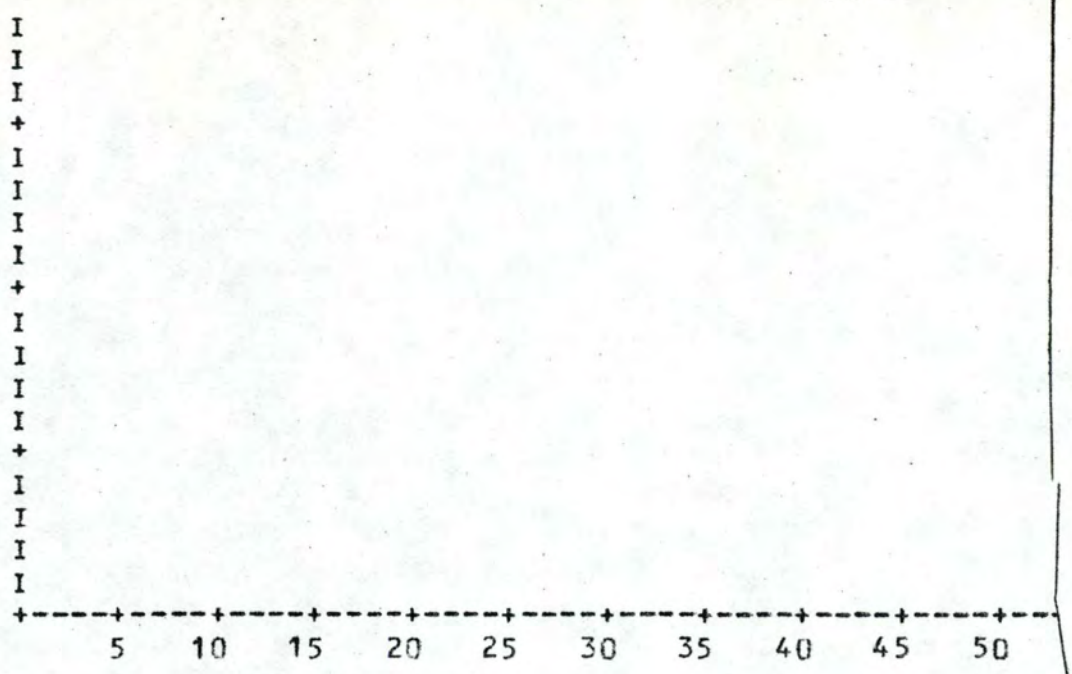
ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
38.62	41.79	18.59	44.49	7.48	31.14	80.63



* Voir page 191 bis

```
-+----+----+----+----+----+----+----+----+----+
55   60   65   70   75   80   85   90   95  100
      CE NUMERO 20
```

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
63.8	66.1	68.3	70.4	72.4	76.3	79.9	83.3	86.5	89.5	92.4
0.48	0.46	0.44	0.43	0.41	0.39	0.37	0.35	0.33	0.31	0.29

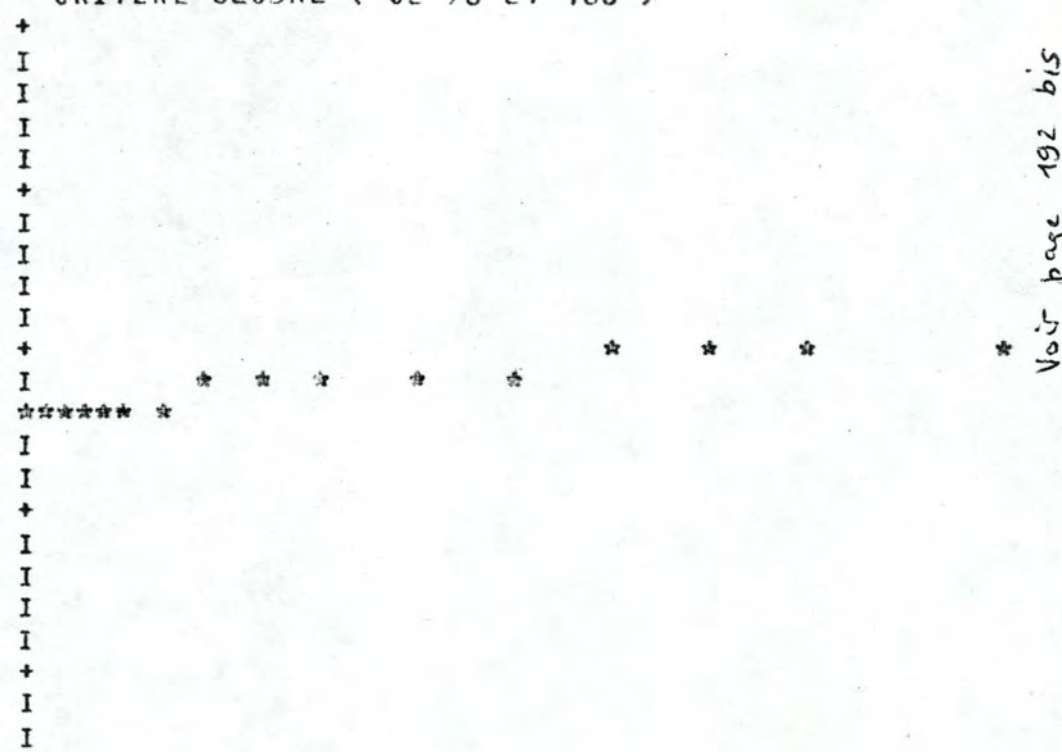


SENSITIVITY ANALYSIS

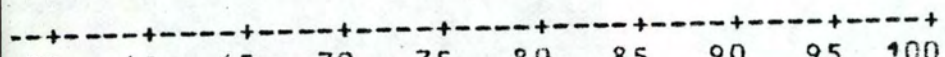
INPUT = 4 = CE NUMERO 40
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	76.3	76.4	76.5	76.6	76.7	76.8	76.9	77.1	77.4	77.7
D(OUT)/D(IN)=	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10

ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
8.65	10.18	4.96	48.73	0.00	8.65	100.00

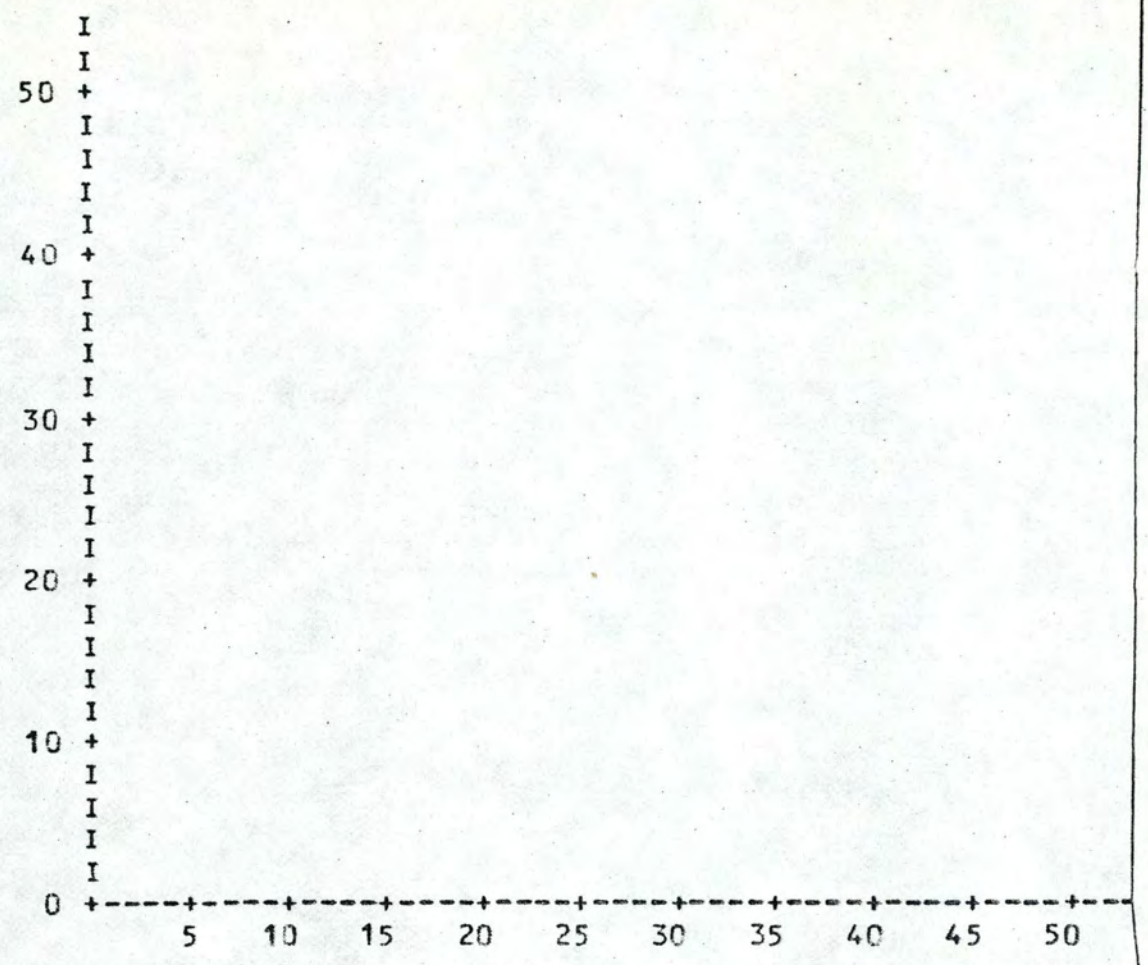


Voir page 192 bis



CE NUMERO 30

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
78.1	78.6	79.0	79.5	79.9	80.8	81.6	82.5	83.3	84.1	84.9
0.10	0.10	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09	0.09



SENSITIVITY ANALYSIS

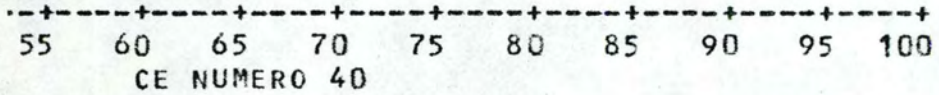
INPUT = 5 = CE NUMERO 50
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	0.0	2.5	4.8	6.9	9.0	10.9	14.6	18.1	22.8	27.2
D(OUT)/D(IN)=	2.47	2.47	2.30	2.17	2.06	1.97	1.85	1.72	1.59	1.45

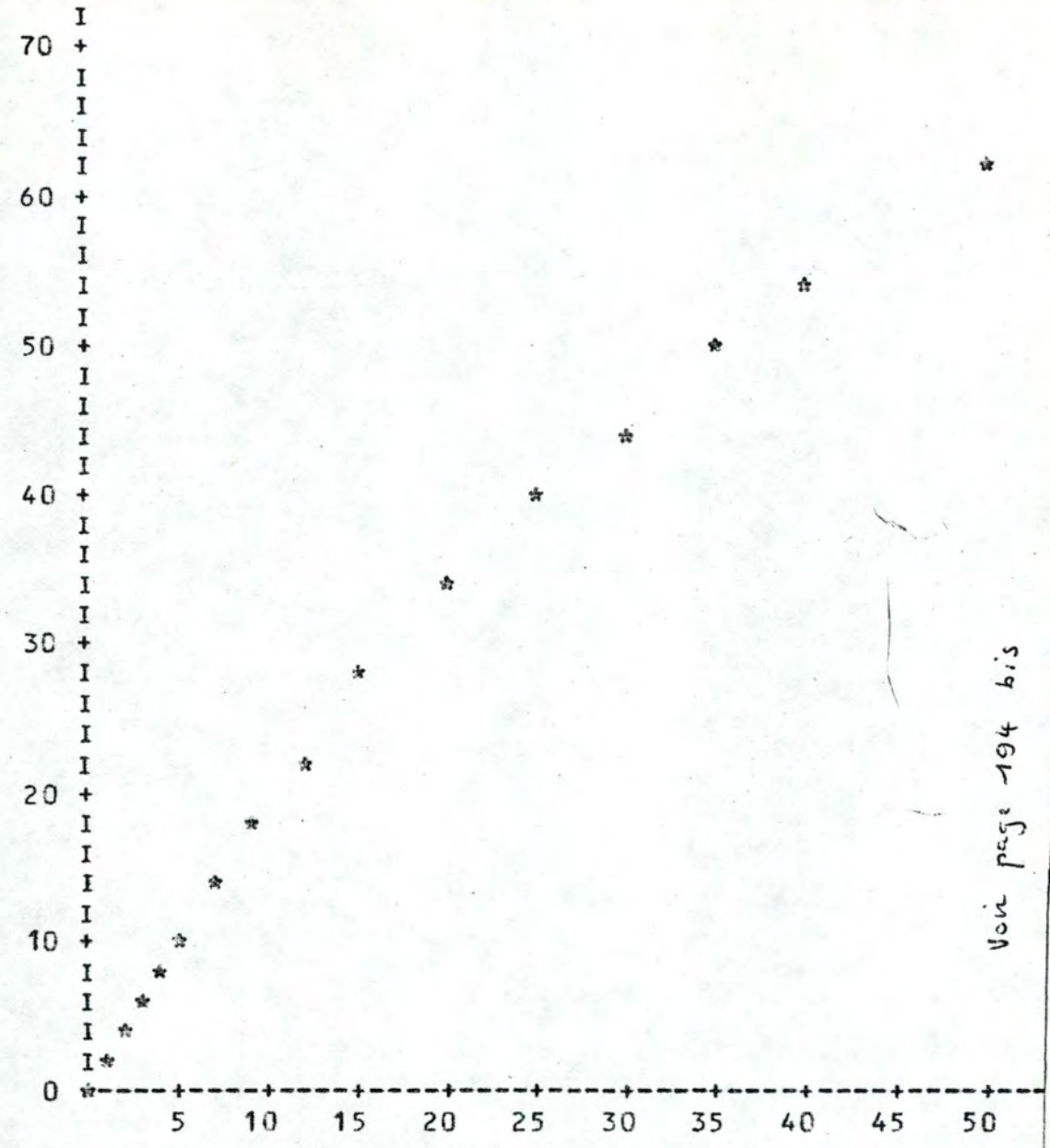
ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
85.37	100.00	34.22	34.22	0.43	84.93	99.50

CRITERE GLOBAL (CE 90 ET 100)

Voir page 193 bis



20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
33.7	39.5	44.7	49.5	53.8	61.5	68.2	74.0	79.2	82.7	85.4
1.31	1.17	1.05	0.95	0.87	0.78	0.67	0.59	0.53	0.35	0.27



Voir page 194 bis

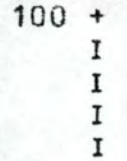
SENSITIVITY ANALYSIS

INPUT = 6 = CE NUMERO 60
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

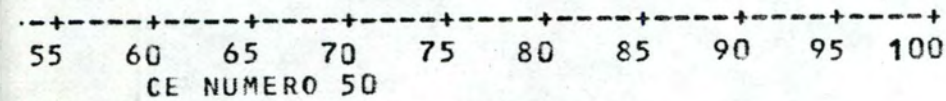
IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	67.6	67.8	68.1	68.3	68.6	68.8	69.3	69.8	70.5	71.2
D(OUT)/D(IN)=	0.26	0.26	0.26	0.25	0.25	0.25	0.25	0.25	0.24	0.24

ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
19.90	22.75	10.40	45.69	2.52	17.38	87.32

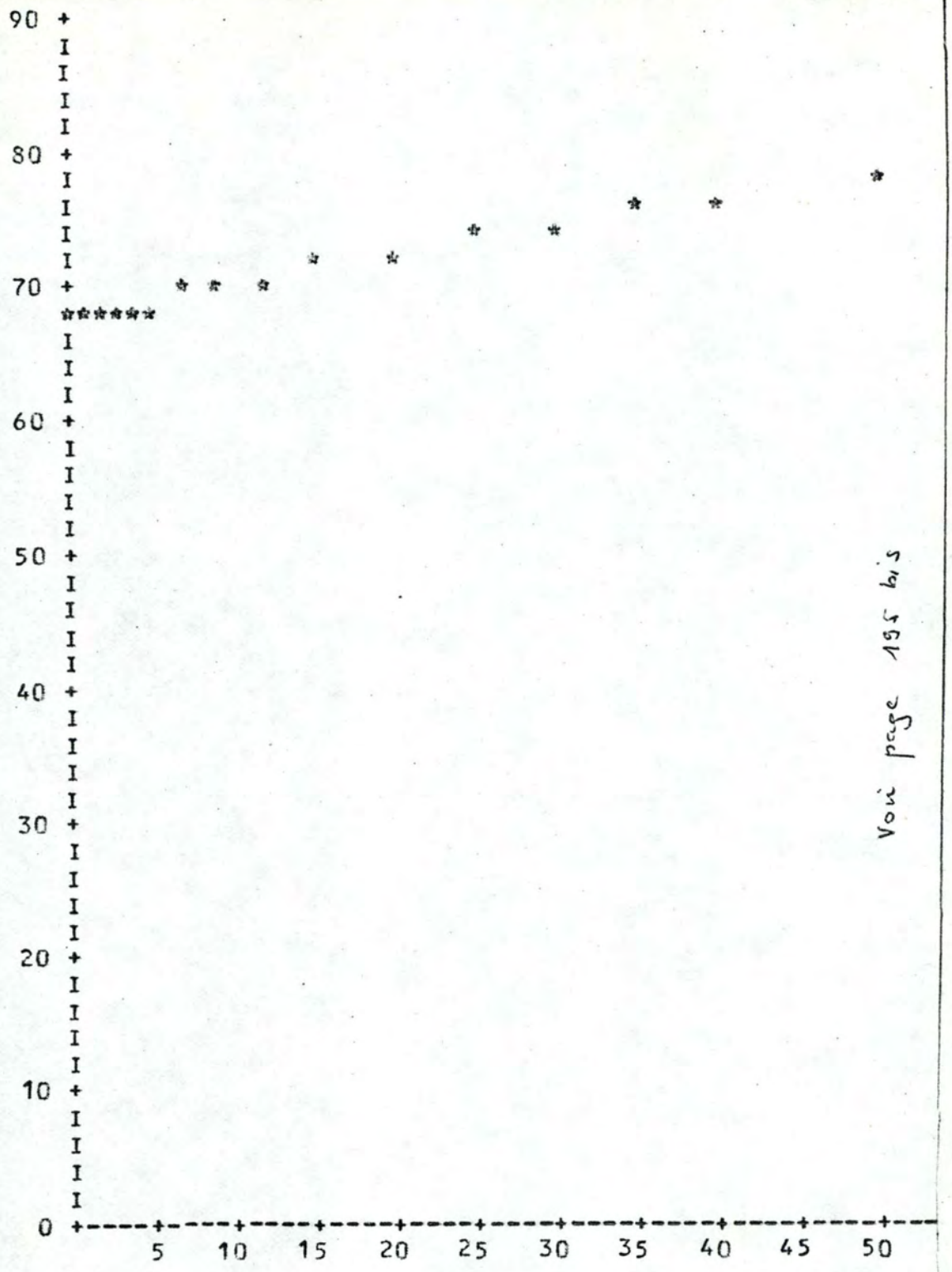
CRITERE GLOBAL (CE 90 ET 100)



*



20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
72.4	73.5	74.6	75.7	76.7	78.7	80.7	82.6	84.4	86.1	87.5
0.24	0.23	0.23	0.22	0.22	0.21	0.20	0.19	0.18	0.18	0.14



Voir page 195 bis

SENSITIVITY ANALYSIS

INPUT = 7 = CNE 70 (CE 10 ET 20)
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	70.3	70.5	70.8	71.0	71.2	71.4	71.8	72.3	72.9	73.6

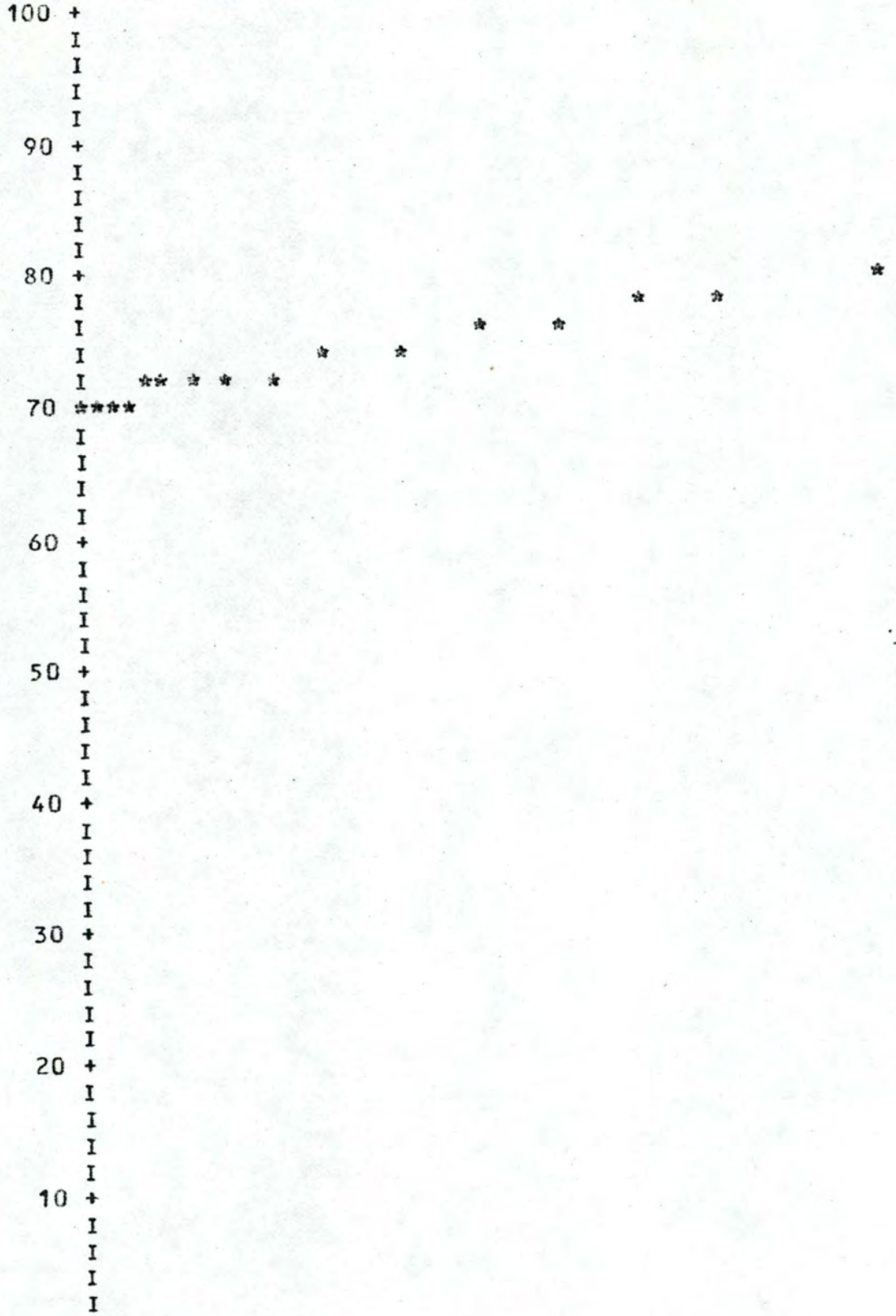
```
-+----+----+----+----+----+----+----+----+----+----+
55   60   65   70   75   80   85   90   95  100
       CE NUMERO 60
```

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
74.6	75.6	76.6	77.6	78.6	80.5	82.3	84.1	85.8	87.5	89.1

D(OUT)/D(IN)= 0.23 0.23 0.23 0.22 0.22 0.22 0.22 0.22 0.22 0.22

ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
18.75	21.06	9.95	47.26	4.14	14.62	77.95

CRITERE GLOBAL (CE 90 ET 100)



Voir page 196 bis

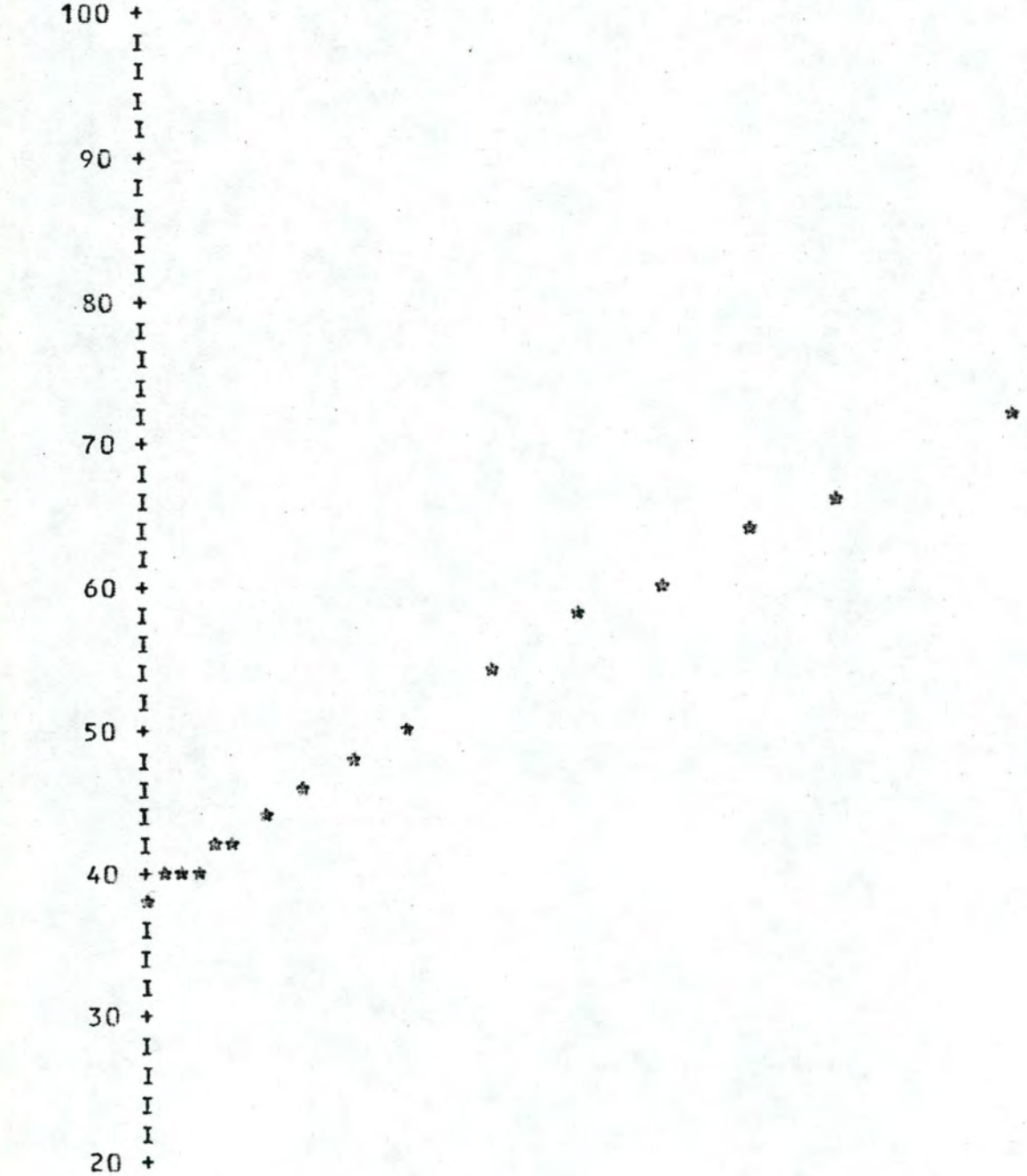
0.21 0.21 0.21 0.20 0.20 0.19 0.19 0.18 0.18 0.17 0.17

```
  0 +----+----+----+----+----+----+----+----+----+----+---
          5   10   15   20   25   30   35   40   45   50
```

SENSITIVITY ANALYSIS

```
INPUT =    8 = CNE 80 ( CE 30 ET 40 )
OUTPUT=   11 = CRITERE GLOBAL ( CE 90 ET 100 )
         IN =  0.0  1.0  2.0  3.0  4.0  5.0  7.0  9.0 12.0 15.0
        OUT = 38.4 39.2 40.1 40.9 41.8 42.6 44.2 45.8 48.1 50.3
D(OUT)/D(IN)= 0.88 0.88 0.86 0.85 0.84 0.83 0.81 0.79 0.77 0.74
 ALPHA    ALPHA REL    BETA    BETA REL    DELTA+    DELTA-      PI
  54.05      58.48     24.85     42.48      7.48     46.57     86.16
```

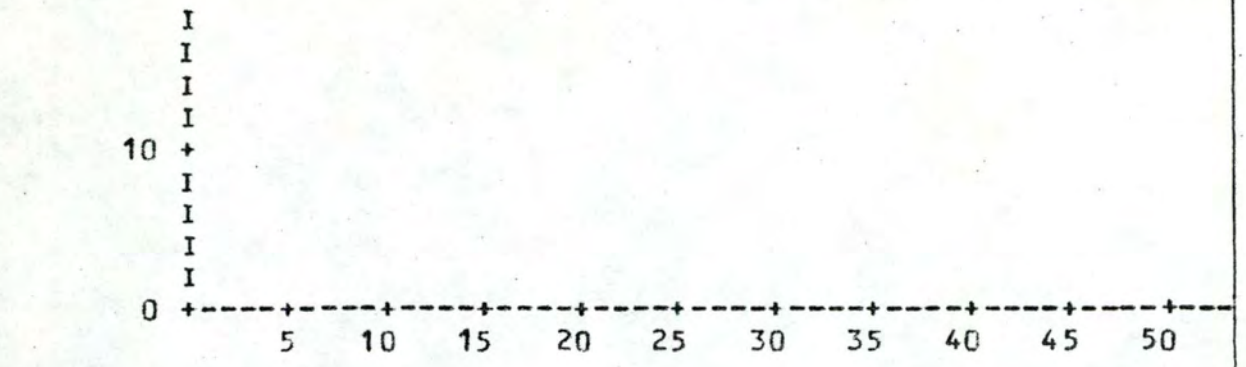
CRITERE GLOBAL (CE 90 ET 100)



Voir page 197 bis

```
-+----+----+----+----+----+----+----+----+----+----+
55    60    65    70    75    80    85    90    95   100
        CNE 70 ( CE 10 ET 20 )
```

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
53.8	57.1	60.3	63.3	66.1	71.4	76.3	80.8	84.9	88.8	92.4
0.71	0.67	0.63	0.60	0.57	0.54	0.49	0.45	0.42	0.39	0.37

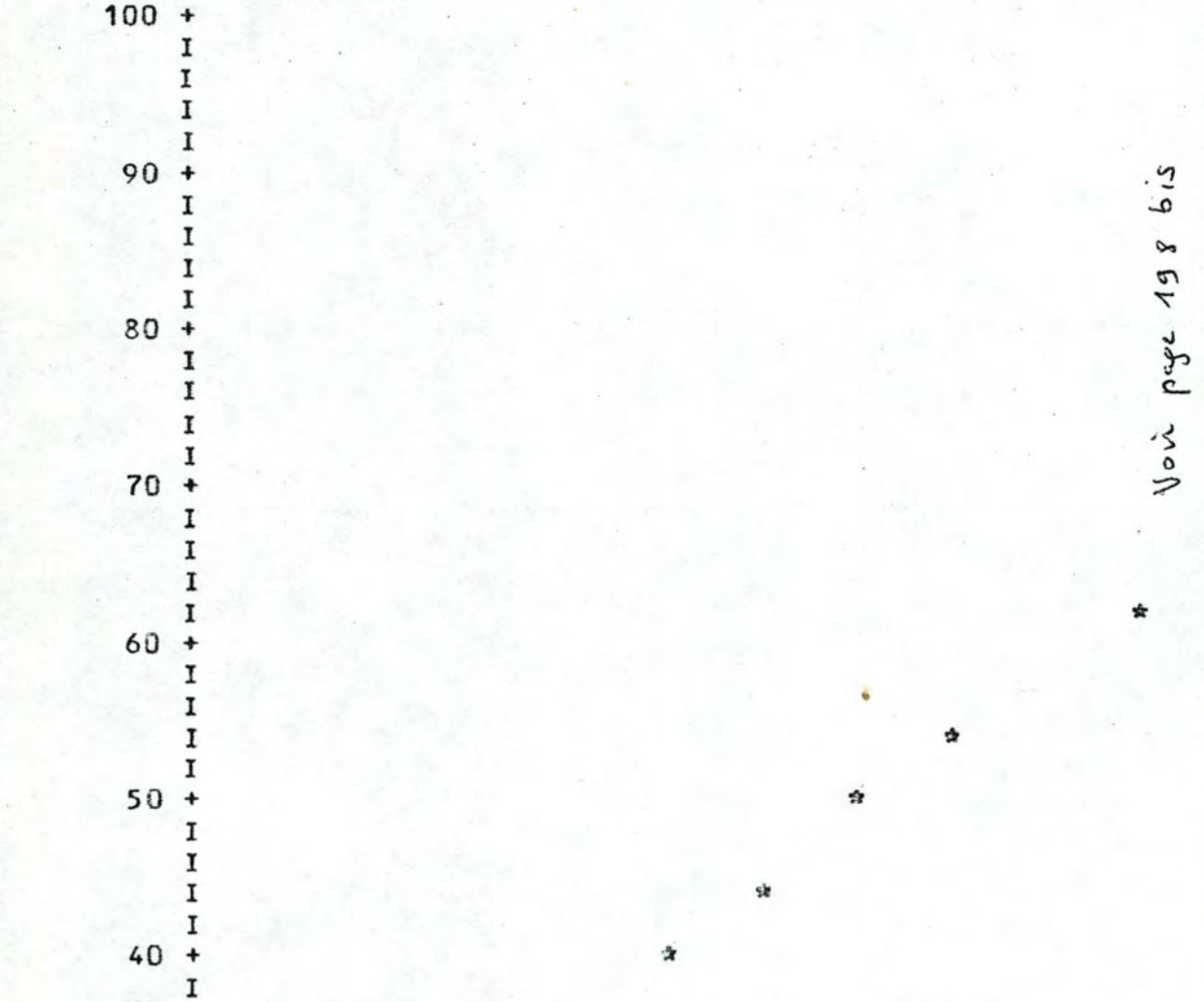


SENSITIVITY ANALYSIS

INPUT = 9 = CNE 90 (CE 50 ET 60)
OUTPUT= 11 = CRITERE GLOBAL (CE 90 ET 100)

IN =	0.0	1.0	2.0	3.0	4.0	5.0	7.0	9.0	12.0	15.0
OUT =	0.0	2.5	4.8	6.9	9.0	10.9	14.6	18.1	22.8	27.2
D(OUT)/D(IN)=	2.47	2.47	2.30	2.17	2.06	1.97	1.85	1.72	1.59	1.45

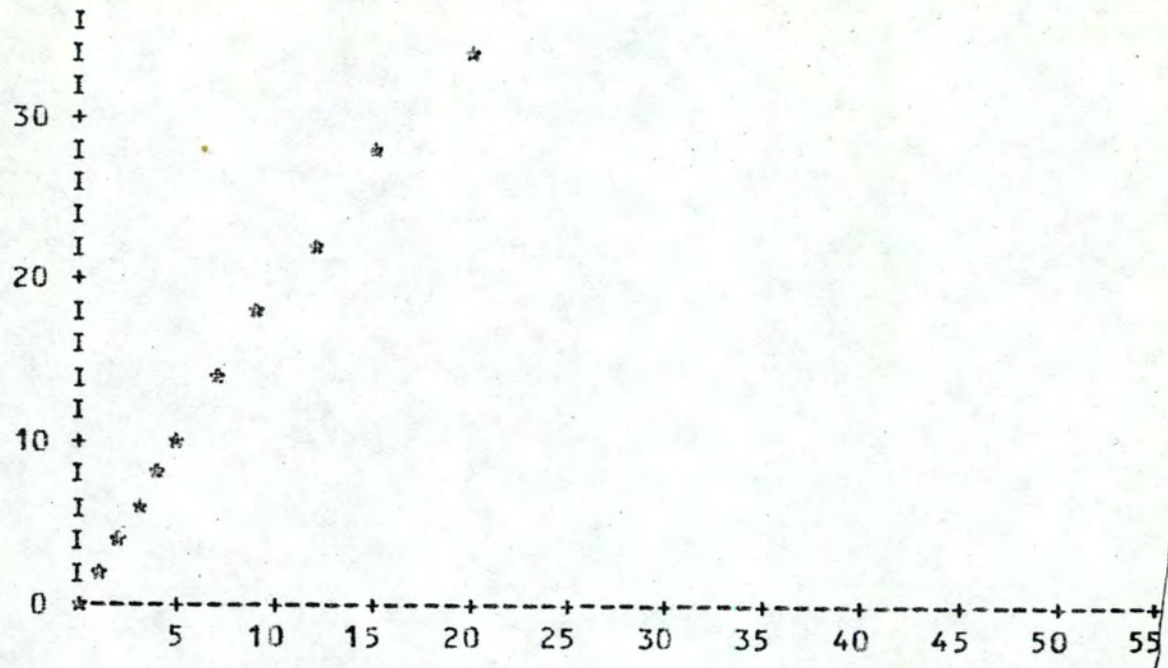
ALPHA	ALPHA REL	BETA	BETA REL	DELTA+	DELTA-	PI
88.13	100.00	35.99	35.99	3.20	84.93	96.37



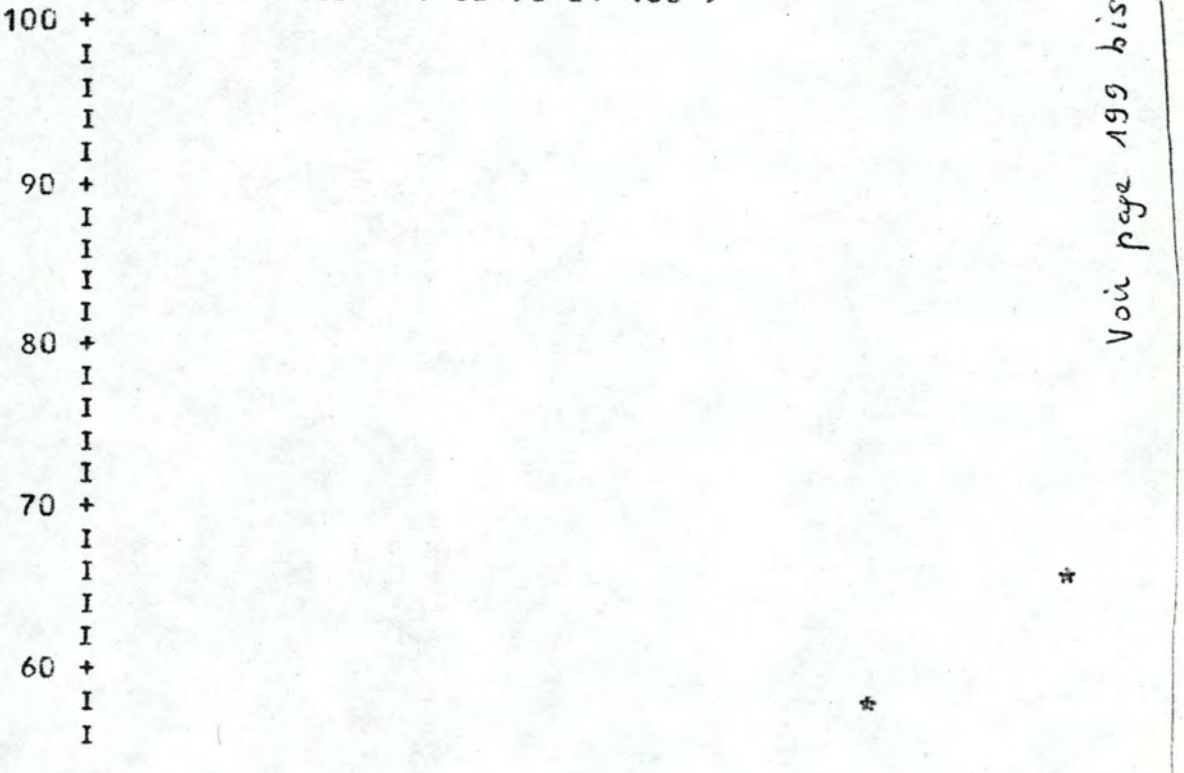
Voir page 198 bis

```
-+----+----+----+----+----+----+----+----+----+
55    60    65    70    75    80    85    90    95   100
        CNE 80 ( CE 30 ET 40 )
```

20.0	25.0	30.0	35.0	40.0	50.0	60.0	70.0	80.0	90.0	100.0
33.7	39.5	44.7	49.5	53.8	61.5	68.2	74.0	79.2	83.9	88.1
1.31	1.17	1.05	0.95	0.87	0.78	0.67	0.59	0.53	0.47	0.43



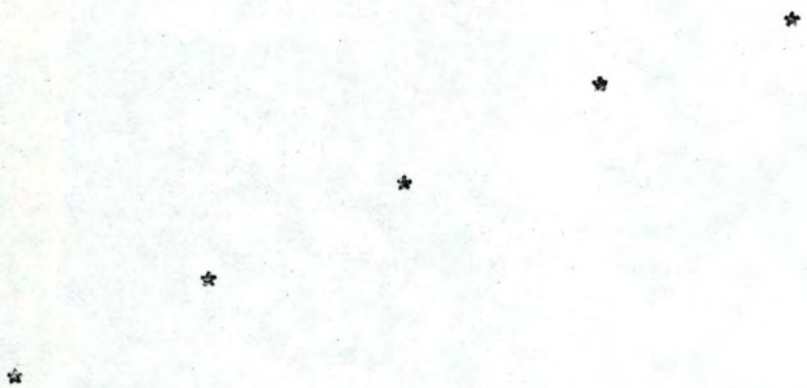
```
SENSITIVITY ANALYSIS
--------------------
INPUT =   10 = CNE 100 ( CE 70 ET 80 )
OUTPUT=   11 = CRITERE GLOBAL ( CE 90 ET 100 )
          IN =  0.0  1.0  2.0  3.0  4.0  5.0  7.0  9.0 12.0 15.0
         OUT =  0.0  2.5  4.8  7.0  9.1 11.2 15.0 18.6 23.6 28.2
D(OUT)/D(IN)= 2.49 2.49 2.33 2.21 2.11 2.03 1.92 1.80 1.67 1.54
 ALPHA    ALPHA REL    BETA    BETA REL    DELTA+    DELTA-      PI
  96.04     100.00     37.05     37.05     11.10     84.93     88.44
```

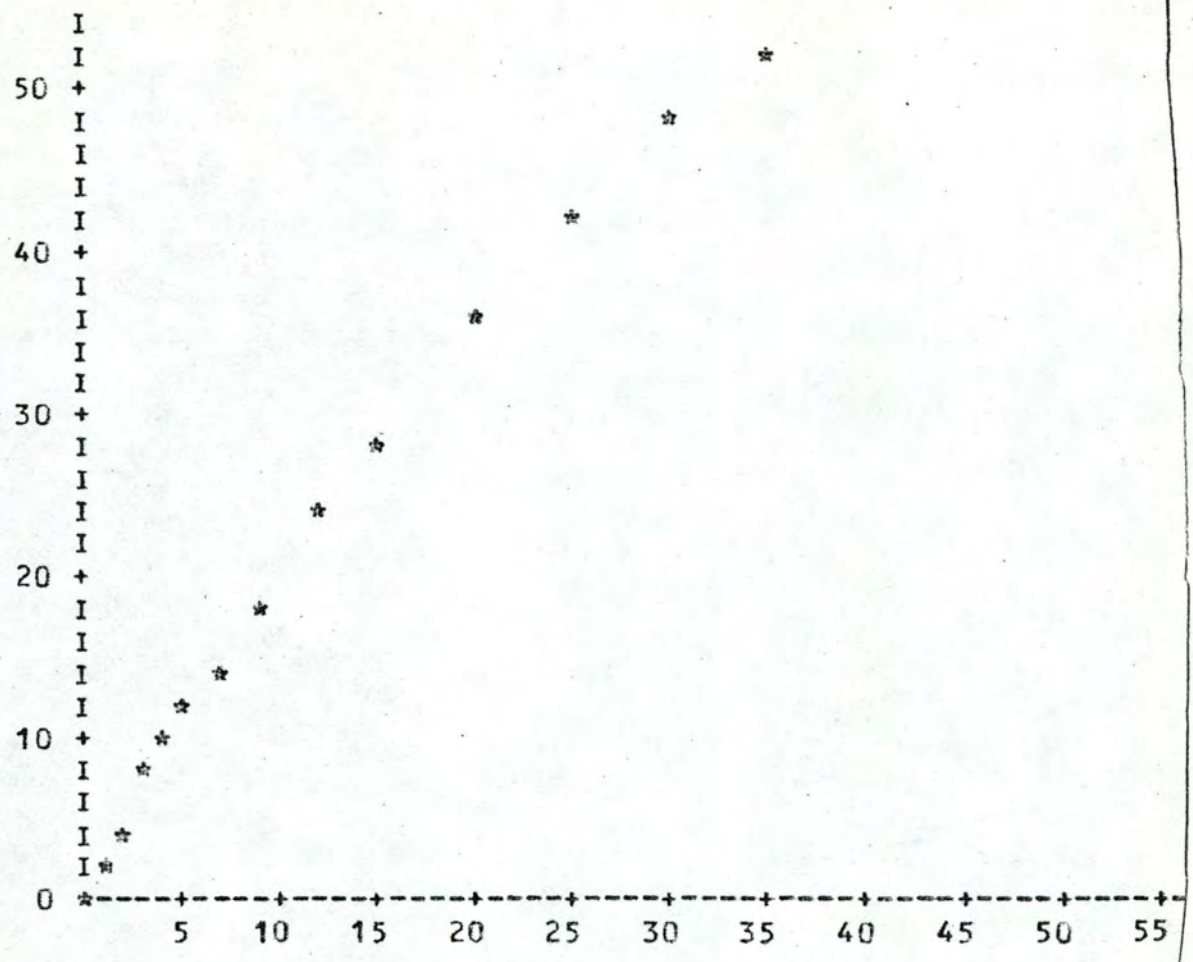


Voir page 199 bis

```
----+----+----+----+----+----+----+----+----+
    60   65   70   75   80   85   90   95  100
     CNE 90 ( CE 50 ET 60 )
```

```
20.0 25.0 30.0 35.0 40.0 50.0 60.0 70.0 80.0 90.0100.0
55.2 41.4 47.1 52.3 57.1 65.7 73.2 79.8 85.8 91.2 96.0
1.40 1.26 1.14 1.05 0.96 0.86 0.76 0.67 0.60 0.54 0.49
```





AVAILABLE VALUES FOR CE NUMERO 10

NO.	X	C	E
1	55.0000	100.0000	0.5500
2	100.0000	1000.0000	1.0000

AVAILABLE VALUES FOR CE NUMERO 20

NO.	X	C	E
1	55.0000	100.0000	0.5500
2	100.0000	1000.0000	1.0000

AVAILABLE VALUES FOR CE NUMERO 30

NO.	X	C	E
1	30.0000	50.0000	0.3750
2	80.0000	100.0000	1.0000
3	100.0000	500.0000	1.0000

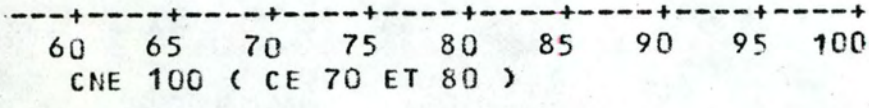
AVAILABLE VALUES FOR CE NUMERO 40

NO.	X	C	E
1	30.0000	50.0000	0.3750
2	80.0000	100.0000	1.0000
3	100.0000	500.0000	1.0000

AVAILABLE VALUES FOR CE NUMERO 50

NO.	X	C	E

Voir pge 200 bis


60 65 70 75 80 85 90 95 100
CNE 100 (CE 70 ET 80)

1	10.0000	1000.0000	0.2667
2	60.0000	1500.0000	0.9000
3	100.0000	2000.0000	1.0000

AVAILABLE VALUES FOR CE NUMERO 60

NO.	X	C	E
1	10.0000	1000.0000	0.2667
2	60.0000	1500.0000	0.9000
3	100.0000	2000.0000	1.0000

OPTIMIZATION FOR CNE 70 (CE 10 ET 20)

NO.	C1OPT	C2OPT	C	EOPT	QREL
1	100.000	100.000	200.000	0.550	1.000
2	1000.000	100.000	1100.000	0.775	0.256
3	1000.000	1000.000	2000.000	1.000	0.182

OPTIMIZATION FOR CNE 80 (CE 30 ET 40)

NO.	C1OPT	C2OPT	C	EOPT	QREL
1	50.000	50.000	100.000	0.375	0.643
2	100.000	50.000	150.000	0.875	1.000
3	100.000	100.000	200.000	1.000	0.857
4	500.000	100.000	600.000	1.000	0.286
5	500.000	500.000	1000.000	1.000	0.171

OPTIMIZATION FOR CNE 90 (CE 50 ET 60)

NO.	C1OPT	C2OPT	C	EOPT	QREL
1	1000.000	1000.000	2000.000	0.267	0.438
2	1500.000	1000.000	2500.000	0.761	1.000
3	1500.000	1500.000	3000.000	0.900	0.985
4	2000.000	1500.000	3500.000	0.980	0.919
5	2000.000	2000.000	4000.000	1.000	0.821

OPTIMIZATION FOR CNE 100 (CE 70 ET 80)

NO.	C1OPT	C2OPT	C	EOPT	QREL
1	200.000	100.000	300.000	0.427	0.641
2	200.000	150.000	350.000	0.777	1.000
3	200.000	200.000	400.000	0.865	0.973
4	200.000	600.000	800.000	0.865	0.487
5	200.000	1000.000	1200.000	0.865	0.324
6	1100.000	200.000	1300.000	0.932	0.323
7	1100.000	600.000	1700.000	0.932	0.247
8	1100.000	1000.000	2100.000	0.932	0.200
9	2000.000	200.000	2200.000	1.000	0.205
10	2000.000	600.000	2600.000	1.000	0.173
11	2000.000	1000.000	3000.000	1.000	0.150

OPTIMIZATION FOR CRITERE GLOBAL (CE 90 ET 100)

NO.	C1OPT	C2OPT	C	EOPT	QREL
1	2000.000	300.000	2300.000	0.331	0.515
2	2000.000	350.000	2350.000	0.412	0.628
3	2000.000	400.000	2400.000	0.426	0.635
4	2500.000	300.000	2800.000	0.553	0.707
5	2500.000	350.000	2850.000	0.769	0.966

6	2500.000	400.000	2900.000	0.810	1.000
7	2500.000	800.000	3300.000	0.810	0.879
8	3000.000	350.000	3350.000	0.835	0.892
9	3000.000	400.000	3400.000	0.882	0.929
10	3000.000	800.000	3800.000	0.882	0.831
11	3500.000	400.000	3900.000	0.919	0.844
12	3500.000	800.000	4300.000	0.919	0.765
13	4000.000	400.000	4400.000	0.928	0.755
14	3500.000	1300.000	4800.000	0.955	0.713
15	3500.000	1700.000	5200.000	0.955	0.658
16	4000.000	1300.000	5300.000	0.965	0.652
17	3500.000	2200.000	5700.000	0.990	0.622
18	3500.000	2600.000	6100.000	0.990	0.581
19	4000.000	2200.000	6200.000	1.000	0.577
20	4000.000	2600.000	6600.000	1.000	0.542
21	4000.000	3000.000	7000.000	1.000	0.511

CONSTRUCTION FINANCIERE POUR UN ORDINATEUR
AU TAUX D'ACTUALISATION DE 6 % ANNUELS (CAS REEL

CASH FLOW	RATE OF INTEREST PER ONE PERIOD (PERCENTAGE)	PERIODS FROM	TO	TOTAL AMOUNT	PRESENT VALUE OF CASH FLOW
93214.187	3.00000	0	0	93214.187	93214.125
131836.180	3.00000	1	13	1713870.000	1402075.000
113626.000	3.00000	4	4	113626.000	100955.430
147150.810	3.00000	5	13	1324357.000	1017971.300
6430.000	0.50000	1	84	540120.000	440168.560
		T O T A L		3785187.000	3054383.000

RATE OF INTEREST PER YEAR (PERCENTAGE)	TOTAL PRESENT VALUE
0	3785187.000
1	3647781.000
2	3517052.000
3	3392699.000
4	3274354.000
5	3161687.000
6	3054383.000
7	2952114.000
8	2854618.000
9	2761646.000
10	2672923.000
11	2588248.000
12	2507371.000
13	2430109.000
14	2356265.000
15	2285663.000
16	2218134.000
17	2153531.000
18	2091673.000

19	2032435.000
20	1975682.000
21	1921291.000
22	1869151.000
23	1819136.000
24	1771146.000
25	1725082.000
26	1680859.000
27	1638381.000
28	1597556.000
29	1558313.000
30	1520577.000

```
 VALEUR TOTALE ACTUALISEE @ 6 % :
 0.30544E 07
END OF SEL PROGRAM EXECUTION
*EXIT
```

C H A P I T R E 6

LISTINGS DES PROCEDURES USUELLES

Nous présentons ici 6 procédures dont l'emploi mène à un appréciable gain de temps pour le praticien .

Pour une simple application SEL , nous proposons la procédure VBA.ARN.C.SEL (1) , qui gère toute l'exécution de cette application .

Pour une manipulation du système SEL , nous proposons 4 procédures .

Signalons tout d'abord que par ' manipulation ' , nous entendons un changement quelconque de la source VBA.ARN.SEL . Pour mettre à jour les modules objets et chargeable de SEL , la solution simple que nous proposons est de rassembler l'ensemble des sous-routines modifiées dans un fichier .

Ce fichier contient ainsi les données des procédures VBA.ARN.CBL ou VBA.ARN.CBL.IDA , qui compilent les sous-routines ' données , les mettent dans la bibliothèque d'objets SEL , VBA.ARN.O.SEL, et les lient en un module chargeable , VBA.ARN.L.SEL . La procédure VBA.ARN.CBL.IDA a pour effet supplémentaire de permettre l'utilisation de IDA sur ces mêmes sous-routines , lors de l'exécution de VBA.ARN.L.SEL .

(1) Les noms de nos fichiers sont formés de
- l'en-tête VBA.ARN. , d'après les noms du responsable et de l'exécutant de notre travail , ainsi que
- du nom propre du fichier éventuellement précédé de C pour CALL , ER pour ERASE , O pour OBJECT et L pour LOAD .

Pour cette utilisation de IDA, nous proposons les deux procédures VBA.ARN.C.FILES et VBA.ARN.ER.FILES , qui exécutent une partie de VBA.ARN.C.SEL , mais laissent au praticien le soin de lancer la commande /LOAD et de donner les paramètres nécessaires pour IDA .

Enfin , la petite procédure VBA.ARN.INIT a pour double but d'éviter tout listing non demandé explicitement , et d'éliminer l'en-tête de l'Editeur de Fichiers .

On notera que plusieurs paramètres non cités ci-après sont utilisés avec leurs valeurs preset .

```
%C E223 LOGON ACCEPTED FROM LINE #026 AT 1519 ON 09/09/77, TSN 4200 ASSIGNED.
%CONTINUE?  (Y,N)  n
/priority 4200,6
/do vba.arn.init
/exec $edt
 %C P500
   1:      àrea'vba.arn.init's ----------------------------------------------

   5.      àp

   1.0000 /PROCEDURE N
   2.0000 /SETSW ON=4
   3.0000 /SYSFILE SYSLST=*DUMMY
   4.0000 /ENDP

   5.      àd

   1:      àrea'vba.arn.cbl's -----------------------------------------------

  19:      àp

   1.0000 /PROCEDURE N,(&FILE)
   2.0000 /REMARK   COMPILATION , MISE EN BIBLIOTHEQUE ET LINKAGE
   3.0000 /SYSFILE SYSDTA=&FILE
   4.0000 /SYSFILE SYSLST=*DUMMY
   5.0000 /PARAM ERRFIL=YES
   6.0000 /EXEC $TSOS.LOCK.BGFOR
   7.0000 /SYSFILE SYSDTA=(SYSCMD)
   8.0000 /EXEC LMR
   9.0000  CONTROL OUTFILE=VBA.ARN.O.SEL
  10.0000  COPY ALL SOURCE=*
  11.0000  END
  12.0000 /EXEC TSOSLNK
  13.0000  PROG SEL,FILENAM=VBA.ARN.L.SEL
  14.0000  INCLUDE SELMP,VBA.ARN.O.SEL
  15.0000  RESOLVE ,VBA.ARN.O.SEL
  16.0000  RESOLVE ,$FUN51.FOREL
  17.0000  END
  18.0000 /ENDP

  19:      àd

   1:
```

Fichier contenant une ou plusieurs sous-routines du système SEL

Voir la commande /REMARK

Bibliotheque d'objets de SEL

Module chargeable SEL

Bibliotheque Fortran

```
 1:      àrea'vba.arn.cbl.ida's
20:      àp

 1.0000 /PROCEDURE N,(&FILE)
 2.0000 /REMARK    COMPILATION , MISE EN BIBLIOTHEQUE ET LINKAGE
 3.0000 /REMARK    ( AVEC IDA )
 4.0000 /SYSFILE SYSDTA=&FILE
 5.0000 /SYSFILE SYSLST=*DUMMY
 6.0000 /PARAM ERRFIL=YES,SYMDIC=YES
 7.0000 /EXEC $TSOS.LOCK.BGFOR
 8.0000 /SYSFILE SYSDTA=(SYSCMD)
 9.0000 /EXEC LMR
10.0000  CONTROL OUTFILE=VBA.ARN.O.SEL
11.0000  COPY ALL SOURCE=*
12.0000  END
13.0000 /EXEC TSOSLNK
14.0000  PROG SEL,FILENAM=VBA.ARN.L.SEL,IDA=Y
15.0000  INCLUDE SELMP,VBA.ARN.O.SEL
16.0000  RESOLVE ,VBA.ARN.O.SEL
17.0000  RESOLVE ,$FUN51.FOREL
18.0000  END
19.0000 /ENDP

20:      àd

 1:
```

Création d'un dictionnaire de symboles pour la (les) sous-routine(s) du fichier d'entrée

Pour inclure le dictionnaire de symboles dans le module chargeable

```
 1:      àrea'vba.arn.c.sel's
42.      àp
 1.0000 /PROCEDURE N,(&SYSDTA,&SYSLST)
 2.0000 /REMARK AFFECTATION DES FICHIERS SYSTEME A DES FICHIERS PRIVES
 3.0000 /SYSFILE SYSDTA=&SYSDTA
 4.0000 /SYSFILE SYSLST=&SYSLST
 5.0000 /REMARK AFFECTATION DES FICHIERS FORTRAN AUX FICHIERS SEL
 6.0000 /FILE VBA.ARN.31,LINK=DSET31
 7.0000 /FILE VBA.ARN.32,LINK=DSET32
 8.0000 /FILE VBA.ARN.33,LINK=DSET33
 9.0000 /FILE VBA.ARN.34,LINK=DSET34
10.0000 /FILE VBA.ARN.35,LINK=DSET35
11.0000 /FILE VBA.ARN.36,LINK=DSET36
12.0000 /FILE VBA.ARN.37,LINK=DSET37
13.0000 /FILE VBA.ARN.38,LINK=DSET38
14.0000 /FILE VBA.ARN.39,LINK=DSET39
15.0000 /FILE VBA.ARN.40,LINK=DSET40
16.0000 /FILE VBA.ARN.41,LINK=DSET41
17.0000 /FILE VBA.ARN.42,LINK=DSET42
18.0000 /FILE VBA.ARN.43,LINK=DSET43
19.0000 /FILE VBA.ARN.50,LINK=DSET50
20.0000 /FILE VBA.ARN.51,LINK=DSET51
21.0000 /FILE VBA.ARN.93,LINK=DSET93
22.0000 /EXEC  VBA.ARN.L.SEL,TIME=300          Module chargeable SEL
23.0000 /SYSFILE SYSDTA=(SYSCMD)
24.0000 /SYSFILE SYSLST=(PRIMARY)
25.0000 /ER  VBA.ARN.31
26.0000 /ER  VBA.ARN.32
27.0000 /ER  VBA.ARN.33
28.0000 /ER  VBA.ARN.34
29.0000 /ER  VBA.ARN.35
30.0000 /ER  VBA.ARN.36
31.0000 /ER  VBA.ARN.37
32.0000 /ER  VBA.ARN.38
33.0000 /ER  VBA.ARN.39
34.0000 /ER  VBA.ARN.40
35.0000 /ER  VBA.ARN.41
36.0000 /ER  VBA.ARN.42
37.0000 /ER  VBA.ARN.43
38.0000 /ER  VBA.ARN.50
39.0000 /ER  VBA.ARN.51
40.0000 /ER  VBA.ARN.93
41.0000 /ENDP
42:      àd
 1.
```

```
 1.      àrea'vba.arn.c.files's
20.      àp
 1.0000 /PROCEDURE N
 2.0000 /REMARK AFFECTATION DES FICHIERS FORTRAN AUX FICHIERS SEL
 3.0000 /FILE VBA.ARN.31,LINK=DSET31
 4.0000 /FILE VBA.ARN.32,LINK=DSET32
 5.0000 /FILE VBA.ARN.33,LINK=DSET33
 6.0000 /FILE VBA.ARN.34,LINK=DSET34
 7.0000 /FILE VBA.ARN.35,LINK=DSET35
 8.0000 /FILE VBA.ARN.36,LINK=DSET36
 9.0000 /FILE VBA.ARN.37,LINK=DSET37
10.0000 /FILE VBA.ARN.38,LINK=DSET38
11.0000 /FILE VBA.ARN.39,LINK=DSET39
12.0000 /FILE VBA.ARN.40,LINK=DSET40
13.0000 /FILE VBA.ARN.41,LINK=DSET41
14.0000 /FILE VBA.ARN.42,LINK=DSET42
15.0000 /FILE VBA.ARN.43,LINK=DSET43
16.0000 /FILE VBA.ARN.50,LINK=DSET50
17.0000 /FILE VBA.ARN.51,LINK=DSET51
18.0000 /FILE VBA.ARN.93,LINK=DSET93
19.0000 /ENDP
20.      àd
```

```
 1.      àrea'vba.arn.er.files's
20.      àp
 1.0000 /PROCEDURE N
 2.0000 /REMARK ECRASEMENT DES FICHIERS SEL
 3.0000 /ER VBA.ARN.31
 4.0000 /ER VBA.ARN.32
 5.0000 /ER VBA.ARN.33
 6.0000 /ER VBA.ARN.34
 7.0000 /ER VBA.ARN.35
 8.0000 /ER VBA.ARN.36
 9.0000 /ER VBA.ARN.37
10.0000 /ER VBA.ARN.38
11.0000 /ER VBA.ARN.39
12.0000 /ER VBA.ARN.40
13.0000 /ER VBA.ARN.41
14.0000 /ER VBA.ARN.42
15.0000 /ER VBA.ARN.43
16.0000 /ER VBA.ARN.50
17.0000 /ER VBA.ARN.51
18.0000 /ER VBA.ARN.93
19.0000 /ENDP
20.      àd
 1.
```

C H A P I T R E 7

QUELQUES SUGGESTIONS AU PRATICIEN

Ce dernier chapitre est consacré à présenter nos quelques suggestions à l'utilisateur de la méthode CAT et du système SEL .

Nous pensons devoir attirer son attention sur les questions à se poser face à un problème d' évaluation et/ou comparaison de système(s) complexe(s):

1 / Sur le choix d'une approche

Cat et SEL forment une approche évoluée , dont l'utilisation n'est ni facile ni peu coûteuse .

Il convient donc de se demander si la complexité du problème justifie l'utilisation de CAT et SEL , ou si une approche plus simple ne serait pas préférable .

En matière de choix d'un ordinateur , CAT et SEL nous semblent conseillables vu le grand nombre de critères en jeu .

Dans ce cadre , il existe deux étapes où le praticien se doit d'être vigilant :

2 / Sur la rédaction du programme SEL

Le programme SEL doit être simple , sans artifice inutile . Il peut faire l'objet des vérifications suivantes:

- il existe au moins une commande *JOB ; la première d'entre elles ne peut être précédée que de commentaires ; s'il y en a plusieurs , elles doivent être séparées par au moins une commande *SEL ou *READ

- entre une commande *JOB et la plus proche commande *SEL ou *READ ne peuvent se trouver que les commandes *DELETE ou *NEW PAGE

- au moins une commande différente de *JOB et *EXIT doit se trouver entre la première commande *JOB et la commande *EXIT

- après une commande *SEL , la carte qui précède immédiatement la commande suivante doit être l'instruction END

- au moins une instruction doit se trouver entre la commande *SEL et la première instruction END qui la suit

- une commande *EXECUTE doit être précédée d'au moins une commande *SEL ou *READ

- toute commande ou instruction doit être précédée de l'entrée ou de la fabrication de ses données : *STORE avant *READ , INPUT et READ avant EFFECTIVENESS , DATA avant OPTIMIZE , OPTIMIZE avant ALLOCATE , etc

- Chaque groupe d'instructions doit contenir au moins une instruction STOP ; le programme formé de ces instructions doit passer au moins une fois par cette instruction STOP , lors de son exécution

- la dernière instruction physique d'un groupe d'instructions doit être END .

3 / Sur la création de l'arbre

La création de l'arbre peut être totalement heuristique ; on peut aussi utiliser la technique TOP-DOWN pour les niveaux supérieurs , et la technique BOTTOM-UP pour les niveaux inférieurs .

La création de l'arbre doit se fonder sur les besoins de l'utilisateur ; usuellement , on estime plus efficace de grouper la plupart des exigences sine qua non dans le cahier de charges , et de réserver l'arbre aux éléments plutôt souhaitables qu'indispensables .

Une fois créé , l'arbre est susceptible des vérifications suivantes , dont la sévérité provient de ce que l'arbre forme le critère par lequel la sélection sera assurée (1):

- chaque critère est-il CIS d'au moins un CNE (le critère global mis à part) ?
- une offre obtenant l'Efficacité 100% pour chaque CE est-elle vraiment parfaite ?
- vérifier que les proportions entre les poids des CIS d'un même CNE sont correctes , surtout si le rapport poids maximum / poids minimum est supérieur à 3
- vérifier qu'il n'y a aucun opérateur d'aggrégation de type A , D ou C supérieure à C-+ dans les niveaux supérieurs , et qu'en général aucun opérateur n'a comme ascendant un opérateur moins conjonctif que lui , càd que plus on monte dans l'arbre , plus les opérateurs sont conjonctifs
- réexaminer le bien-fondé des opérateurs D , A , C supérieure à C-+ , ainsi que tous les cas d' Absorption Partielle
- enfin , il importe que les limitations du système SEL soient respectées (Voir le 1° tome , pages 53 à 54) .

Nous suggérons au praticien de se référer à ce chapitre pour ses premières armes dans la méthode CAT et dans le système SEL .

(1) Abstraction faite du Coût , si l'on utilise le rapport Efficacité/Coût .

FIN DU SECOND TOME