UNIVERSITÉ DE NANCY 1

FACULTÉ DES SCIENCES

TRAITEMENT AUTOMATIQUE DE LA GESTION SCOLAIRE D'UNE UNIVERSITE

REALISATION ET EXTENSIONS PREVUES

THÈSE

pour l'obtention du

DOCTORAT de SPECIALITE MATHEMATIQUES (3ème CYCLE)

Soutenue devant le Jury le 9 Décembre 1971

par

Jean-François DUFOURD

Jury : Mr LEGRAS Président

Mr DEPAIX Examinateurs

Mr PAIR

UNIVERSITE DE NANCY I FACULTE DES SCIENCES

TRAITEMENT AUTOMATIQUE DE LA GESTION SCOLAIRE D'UNE UNIVERSITE

REALISATION ET EXTENSIONS PREVUES

par Jean-François DUFOURD

Je remercie vivement Monsieur le Professeur LEGRAS, Directeur de l'Institut Universitaire de Calcul Automatique, de m'avoir dirigé dans la voie de l'informatique de gestion, encore nouvelle pour l'Université, et des orientations fructueuses qu'il a données à mon travail.

Que Monsieur le Professeur PAIR, soit remercié pour les conseils qu'il m'a prodigués pour la partie graphes et langages.

Mes remerciements vont aussi à Monsieur le Professeur DEPAIX, qui a accepté de participer au Jury.

J'exprime toute ma gratitude à Monsieur LE CADRE, programmeur à l'I. U. C. A., pour le travail considérable qu'il a fourni pour la réalisation et l'exploitation du système de gestion des étudiants, pour celui qu'il continue à produire pour sa maintenance et son extension.

Je remercie Madame DESOIZE pour les premiers essais qu'elle a effectués.

Enfin, je remercie Madame KUGLER pour la réalisation effective de cette thèse, toutes les personnes de l'I. U. C. A. - en particulier de l'exploitation - et des divers établissements scolaires qui ont contribué à mener à bien ce travail.

SOMMAIRE

CHAPITRE I

INTRODUCTION

OBJECTIFS DE LA GESTION DES ETUDIANTS

Deux objectifs sont à atteindre : la planification de l'enseignement
et la gestion interne des établissements.

1.1 Place dans la planification de l'enseignement à long terme

Depuis de nombreuses années, à partir de l'après-guerre en particulier, beaucoup de pays s'interrogent sur la planification de l'enseignement. Il paraît maintenant admis très largement que celle-ci est nécessaire pour s'assurer que le système d'enseignement s'acquitte d'une manière efficace de ses lourdes responsabilités vis à vis du reste de la société.

1.1.1 Rôle du planificateur de l'enseignement

Les objectifs du planificateur sont de plusieurs ordres :

- s'assurer que l'on disposera à peu près, à l'avenir, du nombre "convenable" des différents types de qualifications d'enseignement nécessaires à une économie saine.
- tenter que chaque personne reçoive l'éducation qui convient le mieux à ses qualités et désirs personnels.
- Enfin, l'égalité des chances est devenue bien souvent un des buts explicites de la politique d'enseignement.
- Il faut ajouter à tout ceci un souci d'économie par l'adaptation optimale des moyens existants aux objectifs fixés.

"On s'aperçoit que, dans une société moderne complexe avec une participation scolaire augmentant rapidement, on ne peut concilier les buts parfois contradictoires d'égalité des chances, de satisfaction des besoins en main d'œuvre et de liberté individuelle dans le choix de l'enseignement, que moyennant une planification suffisante" (O. C. D. E. [1]).

1.1.2 Diverses approches de la planification de l'enseignement à long terme :

A partir des objectifs définis ci-dessus, on conçoit facilement l'existence de deux méthodes d'approche principales.

1.1.2.1 Méthode par prévision des demandes de places :

Elle affirme que l'accès à toutes les branches d'enseignement doit être possible pour tous ceux qui ont les qualités et connaissances requises et qui désirent y entrer. Pour le planificateur, il s'agira donc souvent de prévoir un taux de scolarisation par branches d'enseignement (par branche d'enseignement, on entend : niveau et type d'enseignement) et de dégager les besoins nécessaires. Cette prévision se faisait traditionnellement par extrapolation des tendances de ces taux de scolarisation en corrélation avec la structure démographique, géographique, socio-économique, etc..., de la population. On basait donc la prévision sur des "photographies" à certains moments du système d'éducation.

En fait, le principal défaut de cette méthode "statique" est la difficulté de prendre en considération les relations mutuelles entre les différentes branches d'enseignement, c'est pourquoi, depuis quelques années, on tâche d'avoir une vue beaucoup plus "dynamique" du système d'éducation : cette méthode, appelée "méthode des flux" considère celui-ci comme un système de production complexe dont les "matières premières" de base sont les individus, en général les enfants ou étudiants, qui subissent des transformations au fur et à mesure de leur progression dans le système et dont les "sorties" sont les différentes catégories de personnes "instruites". En adoptant cette conception, il est relativement aisé de modéliser en partie le système d'enseignement ; ce modèle peut, dans un premier stade, s'appuyer sur la théorie des processus stochastiques et, plus particulièrement, sur les chaînes de Markov.

Rappels sur les processus stochastiques (KARLIN [2])

La théorie des processus stochastiques concerne l'investigation de la structure de familles de variables aléatoires X_t où t est un paramètre parcourant un ensemble T convenablement indexé.

Définitions :

- espace d'état S : espace auquel appartiennent les valeurs possibles de chaque X_t. Si S est un ensemble fini ou dénombrable, le processus est dit à valeurs entières ou processus à états discrets.

- espaces des indices : T ; dans la suite $t \in T$ sera la variable temps ($T = \mathbb{R}$, ensemble des nombres réels)
- processus de Markov :
  c'est un processus possédant la propriété suivante : étant donnée une valeur de X_t, les valeurs de X_s, pour $s > t$, ne dépendent pas des valeurs de X_u, pour $u < t$, c'est-à-dire que la probabilité de tout comportement particulier futur du processus, quand son état présent est connu exactement, n'est pas altéré par son comportement antérieur.

Un processus est donc dit de Markov si :

$$\Pr \{a < X_t \leq b / X_{t_1} = x_1, \ X_{t_2} = x_2, \ \ldots, \ X_{t_n} = x_n \}$$

$$= \Pr \{a < X_t \leq b / X_{t_n} = x_n\}, \text{ si } t_1 < t_2 < \ldots < t_n < t$$

Soit A un intervalle de la droite réelle ; la fonction

$P(x, s ; t, A) = \Pr \{ X_t \in A / X_s = x \}$ pour $t > s$ est appelée fonction de probabilité de transition. La condition ci-dessus s'écrit alors :

$$\Pr \{a < X_t \leq b / X_{t_1} = x_1, \ \ldots, \ X_{t_n} = x_n \} = P(x_n, t_n ; t, A)$$

où $A =]a, b]$

- chaîne de Markov à temps discret : $\{X_n\}$:
  c'est un processus stochastique de Markov dont l'espace d'états S est fini ou dénombrable et pour lequel $T = \mathbb{N}$. Nous appellerons "valeur de X_n" le résultat de la n^e épreuve.

La probabilité de X_{n+1} d'être dans l'état j, étant donné que X_n est dans l'état i (appelée probabilité de transition de pas un) est notée $P_{ij}^{n, n+1}$

$$P_{ij}^{n, n+1} = \Pr \{X_{n+1} = j / X_n = i \}$$

Cette notation souligne que, en général, les probabilités de transition sont fonction, non seulement de l'état initial et final, mais aussi du temps de transition. Quand des probabilités de transition de pas un sont

indépendantes de la variable temps (c'est-à-dire de la valeur de n), nous dirons que le processus de Markov a des probabilités de transition stationnaires. Dans ce cas : $P_{ij}^{n, n+1} = P_{ij}$ est indépendante de n et nous noterons P_{ij} la probabilité pour que la valeur de l'état subisse une transition de i à j dans une épreuve quelconque.

$P = (P_{ij})$ est appelée "matrice de Markov" (ou matrice des probabilités de transition) du processus. La $(i+1)^{ème}$ ligne représente les probabilités des valeurs de X_{n+1} avec la condition $X_n = i$. Si le nombre d'états est fini, P est une matrice carrée dont l'ordre est égal au nombre des états.

Les qualités P_{ij} satisfont aux conditions $P_{ij} \geq 0$ pour $(i, j) \in \mathbb{N}^2$ et

$$\sum_{j=0}^{+\infty} P_{ij} = 1 \quad \text{pour } i \in \mathbb{N}$$

On démontre alors que le processus est parfaitement déterminé lorsque la matrice P et la valeur de X_0 (ou plus généralement sa loi de probabilité) est donnée.

Application à la planification de l'enseignement

Considérons une population d'étudiants (ou d'élèves) d'un établissement scolaire et un ensemble S de m "branches d'enseignement" (caractérisées par niveau et type d'enseignement) et un état de sortie (branche m), caractérisant tout ce qui se trouve à l'extérieur de l'ensemble des m branches d'enseignement .

Si nous appelons x_n^i la proportion d'étudiants (de la population initiale) appartenant à la branche i à l'année n, on peut écrire :

$$\begin{bmatrix} x_{n+1}^0 \\ \vdots \\ x_{n+1}^i \\ \vdots \\ x_{n+1}^m \end{bmatrix} = \begin{bmatrix} P_{0,0}^{n,n+1} & \cdots & P_{0,m}^{n,n+1} \\ \vdots & & \vdots \\ \cdots & P_{i,j}^{n,n+1} & \cdots \\ \vdots & & \vdots \\ P_{m,0}^{n,n+1} & \cdots & P_{m,m}^{n,n+1} \end{bmatrix} \begin{bmatrix} x_n^0 \\ \vdots \\ x_n^j \\ \vdots \\ x_n^m \end{bmatrix}$$

$P_{i,j}^{n,n+1}$: probabilité de passage de la branche i à la branche j entre les années n et n+1. Ce processus est une chaîne de Markov avec :

- variables aléatoires : X_n : "branche d'un étudiant à l'année n"
- espaces des états : les m branches d'enseignement
- espaces des indices : les années 0, 1, ..., n, ...

Donc, connaissant la loi de probabilité de X_0, on en déduit celle de X_{n+1} par le produit des matrices

$$P^{n,n+1} \times P^{n-1,n} \times \cdots \times P^{0,1}$$

On conçoit que, pour la prévision, le modèle n'a d'intérêt que si l'on est capable de connaître à l'avance les valeurs des fonctions de transfert $P_{i,j}^{k,k+1}$.

Dans une première approximation (nous reviendrons sur cette hypothèse on peut le supposer ; dans ce cas, il suffit de connaître la loi du transfert entre 2 années quelconques et la répartition de la population initiale pour obtenir la répartition de celle-ci à une année quelconque.

Bien souvent, ce sont les effectifs globaux pour un établissement qui, nous intéressent ; chaque année, il faut donc ajouter à l'effectif de la population initiale l'effectif des nouveaux arrivants :

Soient e_n^i . (i = 1, ..., m) les effectifs à l'année n

a_n^i (i = 1, ..., m) l'effectif des nouveaux arrivants à l'année n dans la branche i.

Alors :

$$\begin{bmatrix} e_{n+1}^0 \\ \vdots \\ e_{n+1}^m \end{bmatrix} = P^{n,n+1} \begin{bmatrix} e_n^0 \\ \vdots \\ e_n^m \end{bmatrix} + \begin{bmatrix} a_{n+1}^0 \\ \vdots \\ a_{n+1}^m \end{bmatrix}$$

($a_{n+1}^m = 0$, bien évidemment) .

Remarques sur les matrices de transition $P^{n, n+1}$:

- ce sont généralement des matrices creuses, les éléments non nuls se trouvant souvent au voisinage supérieur de la diagonale principale. Cette remarque est importante pour les rangements dans des mémoires centrales d'ordinateurs.
- Les probabilités de redoublements de branches sont détectées facilement sur la diagonale principale.
- Intuitivement, on conçoit que la probabilité pour un étudiant d'atteindre le niveau "sortie" croît avec les années, et tendrait vers 1 si n tendait vers $+\infty$.
- Cette chaîne de Markov est non récurrente et apériodique, dans les systèmes d'éducation classiques du moins (recyclage exclu).

Il est nécessaire, comme nous l'avons montré, d'avoir une bonne idée du flux entrant dans le système : celui-ci peut être obtenu à partir de données démographiques précises sur la population pré-scolaire qui permettront d'obtenir une estimation de l'évolution des coefficients a_n^i.

Il est évident que cette vue dynamique n'exclut en rien la nécessité de connaître très exactement et le plus finement possible les effectifs de chaque catégorie d'étudiants à tout moment (statique). Mais on conçoit l'importance de cette étude dynamique pour tenir compte des interactions entre branches d'enseignement, pour détecter les redoublements...

Le problème le plus important est celui de l'estimation de l'évolution probable des fonctions de transferts $P_{ij}^{k, k+1}$. Cette estimation ne peut s'effectuer que si l'on est capable d'analyser les divers comportements et choix scolaires des individus provenant de milieux sociaux et culturels différents.

Il est en effet très probable que de nombreux facteurs appartenant aux caractéristiques sociales principales des étudiants influent sur ces transferts, tels que :

statut professionnel des parents, revenu familial, statut religieux et ethnique, lieu de résidence, etc... Il ne faut toutefois pas exclure le statut personnel des étudiants : âge, sexe, intelligence, ordre de naissance dans la famille, etc...

Il apparaît donc nécessaire, pour expliquer ces transferts, de faire figurer ces variables dans les tableaux de flux précédents pour déceler les facteurs importants.

Le planificateur sera dès lors plus à même, connaissant certaines évolutions probables de la société (niveau d'éducation général croissant, urbanisation croissante, taux de fécondité s'égalisant dans les divers milieux, etc...), d'estimer ces coefficients de transfert pour les années à venir.

Le modèle "demande de places" est cependant critiquable à bien des endroits ([1], pages 34 - 35) : "entrées" confondues avec nombre de places demandées, interdépendance de certaines fonctions de transfert, etc...

1.1.2.2 Méthode par prévision des besoins en main d'œuvre

Au lieu d'avoir pour point de départ les "entrées" dans le système d'éducation, comme la méthode précédente, celle-ci se préoccupe des "sorties" : à partir de besoins estimés de "main d'œuvre", on tente de déterminer quelles seront les proportions de personnes instruites différemment qui seront nécessaires pour pourvoir à ces besoins.

Cette approche se préoccupe donc, en quelque sorte, du "rendement" du système d'éducation, alors que la méthode précédente tenait plus compte du désir de chacun.

De gros obstacles sont rencontrés lors d'un essai d'application de cette méthode :

Il est nécessaire de prévoir les besoins en main d'œuvre, ceci ne pouvant se faire qu'en ayant une juste estimation de la production par branches d'activité ; celle-ci peut alors conduire à estimer les besoins par professions si l'on est capable de dire le nombre de personnes à employer par profession et par secteur d'activité pour atteindre la production prévue.

- Il est nécessaire d'avoir une estimation des divers types d'éducation que nécessite chaque profession, en quelque sorte par l'établissement d'une "matrice enseignement/profession". On constate alors que si l'on est

capable de prévoir le nombre des individus de chaque profession, on pourra obtenir le nombre de diplômés nécessaires pour répondre aux objectifs de production fixés.

Ceci est évidemment un schéma très théorique de ce que pourrait être une méthode par prévision de la main d'œuvre ; il n'est pas à l'heure actuelle réalisé dans aucun pays, mais il montre la quantité d'informations à recueillir et à manipuler pour parvenir à le mettre en œuvre.

Quand on sait avec quel mal beaucoup d'entreprises essaient d'exprimer leurs besoins en main d'oeuvre et surtout les qualifications de cette main d'oeuvre, cela laisse très sceptique quant aux chances de l'application efficace d'une telle méthode.

D'autre part, on court le risque, en fixant ainsi les "sorties" que certaines "entrées" soient impossibles, par manque d'enfants d'âge scolaire ou parce qu'ils ne veulent pas entrer dans les branches d'enseignement ainsi favorisées.

Remarque : il n'y a que dans certains cas particuliers que l'on sent poindre cette méthode ; c'est quand des objectifs de rendement de l'enseignement sont étroitement fixés, pour une profession particulière par exemple.

(exemple : nombre de médecins par habitants ou ... par lits d'hôpitaux)

1.1.3 Cadre théorique d'intégration des diverses conceptions de la planification de l'enseignement :

En fait, les deux méthodes exposées précédemment ne s'excluent pas : tout planificateur réaliste utilise les deux approches.

Un cadre général pour le système d'éducation et ses relations avec la société peut être dessiné : les trois facteurs principaux intervenant dans la planification de l'enseignement sont (comme, l'ont montré les paragraphes précédents) : la démographie, le système d'enseignement, l'économie et la société dans leur ensemble.

Entre ces domaines et à l'intérieur de chacun d'eux, on peut suivre des flux de personnes ainsi que des courants d'idées, d'influences ou de décisions. ([1] pages 61 - 74). Pour aborder le problème du rassemblement des données nécessaires à la planification, on les sépare en trois catégories :

i) , Données sur les élèves du système d'enseignement
ii) Données sur le système d'enseignement lui-même : professeurs, équipements...
iii) Données d'ordre démographique, économique, sociologique, politique ...

Pour chacun des domaines ci-dessus, il sera nécessaire de faire des mesures, d'effectuer un grand nombre d'études statistiques pour prévoir l'évolution du système.

Enfin, le rôle du planificateur est aussi essentiel pour prévoir quelles seront les ressources nécessaires à la réalisation des plans et pour prévoir la distribution des ressources réelles obtenues : à tous les problèmes évoqués précédemment, on doit donc ajouter l'analyse des dépenses passées d'enseignement et la projection des dépenses futures.

1.1.4 Place de la gestion des étudiants dans la planification de l'enseignement

C'est, à l'origine, dans un but d'observation et de planification qu'a été lancée vers 1964, dans certaines académies de France ce que l'on a coutume d'appeler "gestion des étudiants". Elle concerne tout l'enseignement supérieur d'une académie et permet d'obtenir des relevés statistiques nécessaires à la planification pour l'ensemble des établissements d'enseignement supérieur. (par "établissement", on entendra "groupe d'UER" dans les structures actuelles). Ces relevés statistiques ne concernent que les étudiants, ce qui montre bien, d'après les paragraphes précédents, qu'à lui seul, ce système est insuffisant pour résoudre les problèmes de planification de l'enseignement supérieur. Les données recueillies doivent être complétées par une quantité d'autres informations appartenant aux domaines cités précédemment. Cependant, ce système

semble être apte s'il est suffisamment développé dans l'avenir, à fournir un nombre considérable d'observations sur la population étudiante qui seront la base nécessaire à une planification efficace.

Il existe diverses méthodes pour obtenir des données statistiques (qui peuvent d'ailleurs être combinées entre elles) :

- recensement régulier d'élèves dans chaque établissement d'enseignement.
- enquêtes par sondages périodiques sur les élèves dans tous les établissements ou dans un échantillonnage d'établissements d'enseignement.
- enquêtes ad hoc à intervalles irréguliers ...

Les techniques retenues pour l'enseignement supérieur, s'appuient sur des informations recueillies chaque année au moment de l'inscription des étudiants dans les divers établissements. Chaque étudiant remplit un questionnaire où on lui demande des renseignements sur son état civil, sa situation familiale et sociale, ses études antérieures et ses études actuelles. Ceci permet l'étude statique et dynamique du système d'éducation, puisqu'il est possible d'établir des flux entre deux années, et aussi une étude des milieux sociologique et économique de la population étudiante.

Ce système est critiquable par la nécessité de devoir reprendre chaque année les mêmes renseignements sur des individus qui étaient déjà inscrits l'année précédente. Il est, d'autre part, assez compliqué de suivre l'expérience d'un étudiant à travers le système d'éducation entre des années qui ne sont pas consécutives, puisqu'il faudra faire appel à des informations que ne sont pas, à priori, dans le même fichier. Ceci ne doit donc constituer qu'une première étape vers un système de fiches individualisées beaucoup plus complet : chaque fiche doit récapituler la vie scolaire d'un étudiant dès son entrée dans le système ; on doit établir la relation entre chaque nouvelle expérience d'un étudiant avec ses activités précédentes pour constituer un "système de données individualisées". C'est ce que nous étudierons dans le projet à long terme (chapitre 2).

En fait, il faut bien prendre garde de ne pas séparer le rassemblement

et le traitement de données statistiques de l'administration scolaire proprement dite.

1.2 Place dans la gestion administrative d'un établissement

Le système de gestion des étudiants ne paraît être qu'un rouage dans un système de planification au niveau national. Cependant, il est facile d'imaginer que des données individualisées peuvent servir à des fins de gestion interne à chaque établissement très importantes, si importantes même qu'elles peuvent en devenir l'objectif premier ; il ne faut évidemment pas que le rassemblement des données soit trop subordonné à des convenances administratives, mais on peut considérer que les statistiques nécessaires à la planification doivent apparaître comme les sous-produits d'une gestion administrative bien comprise. C'est d'ailleurs ce qui semble se produire dans la plupart des entreprises industrielles et c'est cette considération qui nous a guidé lors de l'étude et de la mise en place de notre système.

Il faut d'ailleurs remarquer que la collecte d'informations au niveau d'une administration est une opération suffisamment lourde pour qu'il soit nécessaire de motiver les membres de cette administration en les soulageant un peu de leurs problèmes routiniers. Il faut aussi éviter l'erreur de séparer totalement les points de vue informatique et organisation administrative car ils sont intimement liés, comme nous tâcherons de le montrer dans les chapitres suivants. C'est toujours avec beaucoup de prudence que nous nous sommes aventurés dans le domaine administratif en prenant bien garde que la liaison reste toujours établie avec le système informatique, ce qui n'est pas le point le moins délicat.

CHAPITRE II

ASPECTS THEORIQUES DE LA GESTION DES ETUDIANTS: PROJET A LONG TERME - OUTILS MATHEMATIQUES, INFORMATIQUES ET PROCEDURES ADMINISTRATIVES NECESSAIRES A CETTE GESTION.

Il apparaît donc que le principal objectif de la gestion des étudiants est l'élaboration d'un fichier historique (système de données individualisées) qui retrace de la manière la plus complète possible la vie des étudiants à travers le système d'enseignement supérieur.

Il n'y a pas, théoriquement, de grosses difficultés à envisager un fichier central unique commun à tous les établissements d'une même académie. Les problèmes seraient surtout d'ordre pratique (informatiques : pour un matériel plus important et un software élaboré, administratives : pour une harmonisation des procédures de gestion interne) et c'est pourquoi nous considérons, dans cette étude, que chaque établissement d'enseignement supérieur fonctionne de manière autonome et possède son propre système de gestion d'un fichier historique qui lui appartient.

2.1 Le fichier historique : structure et contenu

C'est un fichier qui contient de nombreux renseignements dont certains sont constants et d'autres susceptibles d'être modifiés ou complétés :

(A) --- des informations d'identification de l'étudiant et renseignements d'état-civil constants :
- un n° caractéristique de l'étudiant (par exemple : n° Insee)
- Nom et prénom
- Date de naissance

(B) --- Des informations sur les études passées de l'étudiant (constantes, mais susceptibles d'être complétées au cours des années) :
- Baccalauréat (ou équivalence) : type, année, département ;
- Diplômes obtenus dans l'enseignement supérieur : nature, date d'obtention, lieu d'obtention.

(C) --- Par année de présence dans l'établissement, donc répétées autant de fois que d'années :
- L'année considérée ;
- Des informations d'état-civil variables : situation familiale, nombre d'enfants, nationalité ;
- Autre établissement fréquenté, dernier établissement fréquenté ;
- Des informations sur l'environnement socio-économique de l'étudiant :

situation militaire, logement, emploi, emploi du conjoint, profession des parents...

— par niveau d'étude :
  - Diplômes préparés cette année ;
  - Eléments préparés cette année, et résultats.

On peut alors considérer que la trame de la gestion des étudiants est la gestion de ce fichier historique, c'est-à-dire sa mise à jour. Cette gestion présente, dans le régime actuel des études, un cycle annuel et la mise à jour peut être scindée en deux périodes principales concernant :

- l'inscription annuelle des étudiants
- la prise en compte des résultats d'examens et la délivrance des diplômes

Dans la suite, nous étudierons, donc les problèmes posés par
- l'inscription annuelle des étudiants (§ 2.2),
- la gestion des examens (§ 2.3)
- la délivrance des diplômes (§ 2.4).

D'autres travaux annexes seront envisagés (§ 2.5) ainsi que l'élaboration des statistiques nécessaires à la planification (§ 2.6).

2.2 Inscription annuelle des etudiants

2.2.1 Procédure d'inscription

C'est au moment de l'inscription des étudiants dans les secrétariats d'établissement que la première mise à jour annuelle du fichier historique pourra être effectuée.

Deux cas sont à considérer :

a) L'étudiant s'inscrit pour la première fois dans l'établissement : on lui demande de remplir un formulaire complet nécessaire à son insertion dans le fichier historique : les renseignements démandés sont alors (A), (B) et une occurence de (C) concernant l'année en cours.

b) L'étudiant était déjà inscrit lors d'une année précédente dans l'établissement ; il y a alors deux possibilités :

  i) l'étudiant était présent l'année précédente : il ne fournit que les renseignements d'une occurence de (C).

  ii) l'étudiant n'était pas présent l'année précédente : il fournit alors des renseignements sur la période intermédiaire pendant laquelle il a quitté l'établissement (sous la forme des diplômes obtenus, du lieu et de l'année d'obtention) pour compléter l'information (B) sur les études passées, et il fournit les renseignements d'une occurence de (C).

Remarques : il est bien évident que, dans une procédure plus élaborée de saisie de l'information, on pourra ne demander aux étudiants de la catégorie (b) que les renseignements qui auront été modifiés par rapport à l'année précédente. Cela revient à associer à chaque information une date pour laquelle cette information est vraie.

- On ne doit pas éliminer du fichier historique un étudiant qui a quitté l'établissement, car cette absence ne pourrait être que momentanée. Il serait souhaitable de déterminer un "seuil" d'absence au delà duquel cette élimination serait automatique (par exemple 10 ans). Les renseignements concernant les étudiants éliminés seraient placés dans un fichier-archives.

Une partie importante de toute création ou mise à jour de fichier est la validation de l'information. Nous passerons ici sur les problèmes d'insuffisance de l'information fournie, de validité de type, de vraisemblance (qui seront examinés en détail au chapitre 3), pour nous intéresser à une validation caractéristique d'un fichier historique : il s'agit de la comparaison de l'information introduite dans le fichier avec celle qui y est déjà.

2.2.2 Le graphe des diplômes

Une question fondamentale que l'on doit se poser au moment de l'inscription d'un étudiant est :

- Est-ce que cet étudiant a le droit de s'inscrire comme il le fait ?

A ce niveau, on peut faire l'hypothèse que l'avancement de tout individu dans le système d'enseignement peut s'exprimer en termes de "diplômes" et "éléments" :

- l'élément est l'ensemble d'épreuves le plus petit auquel on puisse s'inscrire (certificat, unité de valeur ...)

- un diplôme est caractérisé généralement par :
  - un nombre d'éléments à obtenir
  - un ensemble d'éléments nécessaires
  - un ensemble d'éléments ne pouvant pas participer à la constitution du diplôme
  - un ensemble d'éléments optionnels rattachés à ce diplôme.

exemple : La maîtrise de mathématiques comprend nécessairement les éléments C_1, C_2, C_3 de mathématiques plus un élément C_4 ne pouvant être choisi pa une certaine liste d'éléments (qui seraient redondants avec C_1, C_2, C_3). Le C_4 d'analyse est un élément optionnel de cette maîtrise.

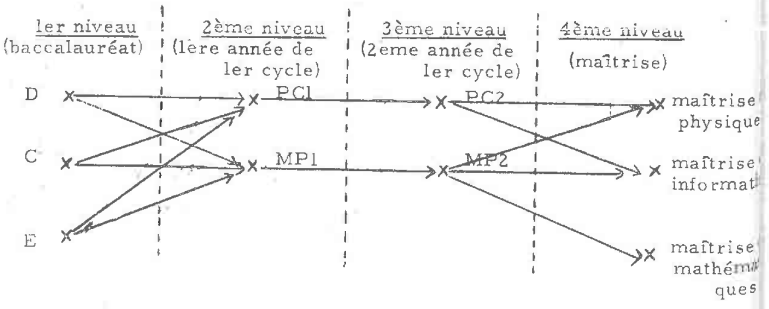
Enfin, l'ensemble des diplômes peut être partitionné en différents "niveaux" correspondants aux états d'avancement possibles dans le système et pouvant être numérotés par ordre d'importance.

Soit $D = (d_i)_{i = 1, \ldots, p}$ l'ensemble des diplômes préparés dans un établissement. Dans D, on peut définir une relation binaire R de la façon suivante :

$x \in D$, $y \in D$: $x R y \Leftrightarrow$ On peut s'inscrire au diplôme y si l'on possède le diplôme x

On obtient donc un graphe (D, R). Si la relation R n'est vraie qu'entre des diplômes de niveaux successifs, le graphe (D, R) est un graphe multiparti.

exemple : si l'on considère les 1er et 2ème cycles d'un groupe d'UER "sciences" une partie du graphe pourrait être :



Remarque : La notion de diplôme considérée ici n'est pas toujours celle que l'on utilise habituellement : il peut être très utile d'introduire des diplômes fictifs, en particulier pour les questions qui concernent les équivalences à certains d'entre eux.

Le problème initial peut alors s'exprimer comme suit, puisque l'on connaît l'ensemble des diplômes possédés et l'ensemble des diplômes postulés pour un étudiant :

pour chaque diplôme postulé, existe-t-il un arc du graphe (D, R) ayant pour origine un diplôme acquis et pour extrémité le diplôme considéré ? La résolution de ce problème est triviale si l'on a réussi à construire le graphe (D, R) pour un établissement.

2.2.3 Le graphe des diplômes et des éléments

Une autre question que l'on peut se poser est : est-ce que l'étudiant a choisi convenablement ses éléments, compte-tenu de ses inscriptions aux diplômes ?

Soit $E = \{c_i\}_{i=1, \ldots, n}$ l'ensemble des éléments pour un établissement donné.

Dans l'ensemble $E \cup D$, on peut définir les relations binaires suivantes : $x \in E$ et $y \in D$:

$x R_1 y \Leftrightarrow$ l'élément x est nécessaire à la composition du diplôme y.

$x R_2 y \Leftrightarrow$ l'élément x appartient au diplôme y sans être nécessaire à sa composition.

$x R_3 y \Leftrightarrow$ l'élément x ne peut pas faire partie des composés du diplôme y.

Les graphes (E, D, R_1), (E, D, R_2), (E, D, R_3) sont des graphes bipartis : on peut parler de "multigraphe biparti".

Remarques : - on n'impose pas qu'un élément appartienne à un diplôme précis ; il peut appartenir à 2 diplômes distincts. (exemple : le C_1 calcul différentiel appartient à la maîtrise de mathématiques et à la M. A. A. F).

- Les relations R_1, R_2 et R_3 sont exclusives deux à deux. Pour représenter le multigraphe précédent, on peut représenter chaque graphe séparément ou non ; si on décide de les représenter ensemble, on

peut constituer la matrice (n x p) $A = (a_{ij})$ suivante :

$$a_{ij} = \begin{cases} 1 \text{ si } c_i \ R_1 \ d_j \\ 2 \text{ si } c_i \ R_2 \ d_j \\ 3 \text{ si } c_i \ R_3 \ d_j \\ 0 \text{ sinon} \end{cases}$$

Donc, un diplôme y doit être constitué des éléments de $R_1^{-1}(y)$ et d'éléments appartenant à $E - R_1^{-1}(y) - R_3^{-1}(y)$. Le nombre total de ces éléments est caractéristique du diplôme y. Le problème initial peut alors s'exprimer ainsi :

pour chaque élément c_i choisi par l'étudiant, existe-t-il un diplôme d_j tel que $c_i \ R_1 \ d_j$ ou $c_i \ R_2 \ d_j$, c'est-à-dire $a_{ij} = 1$ ou 2 ?

Nous verrons plus loin l'utilité de la relation R_3 (§ 2-4).

Remarque : pour diminuer les dimensions de la matrice représentant le multigraphe, il est généralement possible de partitionner D et E en sous-ensembles dont chacun correspond à un niveau déterminé. On ne représentera alors que des sous-graphes.

Remarque : si l'on considère l'ensemble $E \cup D$ et les relations binaires R_1, R_2, R_3 et R, on constate que l'on doit gérer une structure d'information (multigraphe) relativement complexe dont la mise à jour automatique posera quelques problèmes. Dans l'avenir, ce type de problème pourra être résolu grâce aux banques de données qui permettent de gérer des informations en même temps que des relations sur ces informations.

2.3 Gestion des examens

Nous avons vu que le deuxième problème de mise à jour du fichier historique était constitué par la notification dans ce fichier des résultats d'examens. Un autre important problème que rencontrent les grands établissements, à ce niveau, est celui de l'élaboration d'un planning d'examens.

2.3.1 Procédure de remontée dans le fichier-historique des résultats d'examens

Nous insisterons ici sur l'aspect fonctionnel de cette procédure, les détails de réalisation se trouvant dans le chapitre 3.

Après chaque session d'examens, il est nécessaire de notifier les résultats pour permettre la mise à jour. Cette notification peut se faire aisément si l'on a pris soin de fournir au secrétariat de l'établissement et aux professeurs des bordereaux faciles à remplir. Cette "remontée des résultats" s'effectue alors en trois étapes.

2.3.1.1 Etablissement de bordereaux de notification des résultats d'examens

Dans le fichier historique, pour chaque étudiant, nous possédons la liste complète des éléments (examens) auxquels l'étudiant s'est inscrit en début d'année. On peut alors soit considérer que tous les étudiants inscrits peuvent présenter l'élément, soit exiger des étudiants une inscription spéciale aux examens (ceci dans un but de diminution du nombre de salles nécessaires au déroulement des épreuves).

Dans le premier cas, on peut facilement sortir pour chaque examen la liste complète des étudiants inscrits. Dans le deuxième cas, on pourra d'abord sortir la liste des étudiants inscrits et demander à ceux-ci de faire savoir s'ils seront effectivement candidats ou non. Cette réponse permettra alors, après notification sur la liste (par une procédure tout à fait analogue à celle que nous allons décrire) de mettre à jour le fichier-historique. On pourra alors obtenir, de la même façon que dans le premier cas, la liste des étudiants présentant chaque élément.

Ces listes sont généralement nominatives et on prendra soin d'en garder une copie sur support magnétique, ordonnée de la même façon (voir chapitre 3).

2.3.1.2 Notification des résultats d'examens :

Les listes établies précédemment sont communiquées aux professeurs qui peuvent y notifier les résultats des éléments qu'ils contrôlent : en regard de chaque nom, le professeur indique les résultats obtenus par l'étudiant.

Remarque : il arrive fréquemment qu'un étudiant non inscrit à un élémen régulièrement (en début d'année) obtienne une dérogation pour se présenter tout de même à l'examen. Il est donc nécessaire de prévoi des dispositions spéciales pour pouvoir prendre en compte ses résultats d'examens (voir chapitre 3).

Le problème de la notification des résultats n'est cependant pas toujours aussi simple : dans certains établissements, on désire préserver l'anonymat des composants à un examen en regard du correcteur. Les listes nominatives pour la notification sont alors mal adaptées ; une procédure peut alors être la suivante :

pour chaque élément, on affecte un numéro de manière aléatoire à chaque étudiant et on ordonne les listes sur support magnétique dans l'ord de ces numéros. On édite une liste comportant pour chaque étudiant le numéro qu'on lui a affecté et ces numéros sont reportés sur chaque copie. L nom de l'étudiant est détaché de celle-ci, le professeur ne connaîtra donc les étudiants que par des numéros. C'est sur une liste ne comportant que des numéros que le professeur notifiera les résultats qui pourront être enregistrés de la même façon que précédemment.

La procédure de numérotation des r candidats d'un élément pourrait, par exemple, être la suivante :

on dispose en mémoire centrale de l'ordinateur d'une table indiquant quels numéros ont déjà été affectés entre 1 et r (table booléenne).

pour chaque étudiant :

- génération d'un nombre au hasard i entre 1 et r par une procédure classique.
- recherche du nombre le plus proche de i qui n'est pas encor affecté à un étudiant. (le sens de la recherche peut dépend de i).
- affectation de ce nombre à l'étudiant.

2.3.1.3 Mise à jour du fichier-historique :

Comme nous le verrons au chap. 3, il est facile, grâce aux bordereaux codifiés dans les secrétariats, de mettre à jour les listes d'étudiants par éléments qui ont été gardées sur support magnétique. On regroupe alors les résultats d'examens pour chaque étudiant et l'on effectue la mise à jour du fichier historique.

A la session suivante, on recommencera la même opération avec seulement les étudiants qui n'auront pas été reçus aux sessions précédentes.

Remarque : il est évident que pour tirer les listes par éléments, il est inutile de manipuler le volume considérable d'information contenue dans le fichier historique ; un fichier plus réduit caractérisant l'étudiant pour l'année en cours peut être construit directement à partir de l'historique. Dans la plupart des opérations élémentaires de gestion interne, d'ailleurs, il suffit de manipuler ce "fichier-annuel" (chap. 3).

2.3.2 Problème de l'établissement d'un planning d'examens :

2.3.2.1 Position du problème :

2.3.2.1.1 Le graphe G des conflits :

Soit $E = \{c_i\}_{i=1,\ldots,n}$ l'ensemble des éléments (examens) préparés dans un établissement. A chaque session d'examens, il est nécessaire de préparer un planning compte-tenu du fait que :

(1) : la session d'examens a une durée maximale déterminée ; on doit faire passer les examens sur un ensemble de périodes de nombre fixé m_0, généralement très inférieur à n ; donc, certains examens doivent avoir lieu durant la même période.

(2) : certains examens ne peuvent pas se tenir en même temps parce que un étudiant peut préparer plusieurs éléments à la fois.

Pour la contrainte (2), il est possible de connaître, par les textes régissant les inscriptions aux examens, les examens qui ne peuvent avoir lieu en même temps qu'un examen donné :

dans E, on peut alors définir la relation binaire Γ' :

$(x, y) \in E \times E \quad x \Gamma' y \Leftrightarrow x \neq y$ et, par les textes, x et y ne peuvent pas se passer en même temps.

On risque, par cette méthode, d'avoir un grand nombre de contraintes. Or, le fichier des étudiants permet de savoir à quels éléments s'inscrit chaque étudiant : le nombre de contraintes sera donc moindre si, au lieu de Γ', on tient compte de la relation binaire Γ :

$(x, y) \in E \times E$, $x \Gamma y \Leftrightarrow x \neq y$ et il existe un étudiant (dans le fichier) qui prépare x et y.

Il est évident (si les inscriptions sont bien faites) que $x \Gamma y \Rightarrow x \Gamma' y$, donc le nombre de contraintes est moindre pour la relation Γ.

Cette relation est non réflexive, symétrique et non transitive ; elle définit un graphe $G = (E, \Gamma)$, symétrique et sans boucle, sur E.

Soit $R = (r_{ij})$ la matrice booléenne associée au graphe G :

$$r_{ij} = \begin{cases} 1 \text{ si } c_i \Gamma c_j \\ 0 \text{ sinon} \end{cases}$$

Dans la suite, nous appellerons R matrice des conflits (ou matrice des incompatibilités).

La matrice R peut être aisément obtenue à partir du fichier des étudiants à condition que le nombre des éléments ne soit pas trop grand. Toutefois, elle est symétrique, donc on peut réduire son encombrement-mémoire en la considérant comme triangulaire (voir chap. 3).

La construction de ce graphe a déjà rendu, dans les grands établissements surtout, des services estimables (présentation sous forme de tableau avec codes des éléments en abscisse et en ordonnée). Elle ne résout pas pour autant le problème du planning d'examens.

2.3.2.1.2 Problème du planning = coloration du graphe G

Il s'agit de trouver une partition de E en $E_1, \ldots, E_m$ telle que :

$(\forall i)\ (x \in E_i \text{ et } x \Gamma y \Rightarrow y \notin E_i)$

ou : $(\forall i)\ (E_i \cap \Gamma(E_i) = \emptyset)$

avec $m \leq m_0$, nombre des périodes permises, chaque sous-ensemble E_i correspondant à une période (ou à plusieurs).

Définition : sous-ensemble intérieurement stable (e. s. i.) :

Soit un graphe $G = (E, \Gamma)$, un sous-ensemble $S \subset E$ est dit intérieurement stable si $S \cap \Gamma(S) = \emptyset$.

c'est-à-dire : aucun sommet de S n'est adjacent à un autre sommet de S. Il s'agit de trouver une partition de E en e. s. i. : $E_1, \ldots, E_m$.

I_G, ensemble des e. s. i. d'un graphe G, est un demi-treillis inférieur pour la relation d'ordre inclusion : toute intersection d'e. s. i. est un e. s. i., toute partie d'e. s. i. est un e. s. i. (l'ensemble vide est un e. s. i.).

Pour connaître tous les e. s. i. d'un graphe, il suffit de connaître les éléments maximaux de I_G pour la relation d'ordre inclusion :

définition : sous-ensemble intérieurement stable maximal (e. s. i. m) :

un e. s. i. est maximal s'il n'existe aucun e. s. i. qui le contient strictement.

Il est très facile de montrer que :
pour un graphe symétrique et sans boucle $G = (E, \Gamma)$, S est un e. s. i. m. équivaut à

$$\text{et} \quad \begin{cases} S \cap \Gamma(S) = \emptyset \\ S \cup \Gamma(S) = E \end{cases}$$

Remarque : Dans ce cas, S est un noyau du graphe G.

- Les e. s. i. m. d'un graphe G sont les cliques maximales (au sens de Berge [3]) du graphe complémentaire $\overline{G}$ de G.

Ce problème de recherche d'une partition de E en e. s. i. de $G = (E, \Gamma)$ est un problème de coloration de graphe (une couleur correspondant à une période de la session d'examens).

Définitions :

- Un graphe est p-chromatique s'il est possible de colorier ses sommets avec p couleurs de sorte que deux sommets adjacents ne soient pas de la même couleur.
- Le plus petit nombre p tel que le graphe soit p-chromatique est appelé nombre chromatique et noté $\alpha(G)$.

Il est donc nécessaire que $\alpha(G) \leq m_0$, sinon le problème n'admet pas de solution.

Nous allons examiner plusieurs méthodes pour parvenir à une coloration du graphe $G = (E, \Gamma)$:

- la première est une méthode heuristique directe permettant d'obtenir une coloration telle que le nombre de couleurs est, sinon $\alpha(G)$, du moins majoré par un entier obtenu facilement à partir de G .
- les suivantes procèdent en deux étapes :
  - on énumère d'abord tous les e. s. i. m. du graphe.
  - on recherche ensuite différents recouvrements (ou "couvertures") de E par les e. s. i. m :

définition :

Etant donné le graphe biparti $H = (Y, X, \Delta)$, tout sous-ensemble F de Y vérifiant $\bigcup_{y \in F} \Delta(y) = X$ est appelé couverture de X.

Dans notre problème, Y est l'ensemble des e. s. i. m. du graphe G, X est l'ensemble E et la relation Δ est la relation :

$y \in Y, \quad x \in X \quad y \Delta x \Leftrightarrow x \in y$ (x est un élément de l'e. s. i. m. y).

De la même façon que nous avons défini les e. s. i. m à partir des e. s. i., il est possible de définir des couvertures minimales.

Définition :

Une couverture F d'un ensemble X est minimale si elle ne contient strictement aucune couverture de X.

La connaissance de toutes les couvertures minimales entraîne la connaissance de toutes les couvertures (Il suffit de leur adjoindre des éléments).

Une condition nécessaire et suffisante pour que la couverture F de X soit minimale pour le graphe biparti $H = (Y, X, \Delta)$ est :

$$(\forall y \in F)\ (\Delta(F) \not\subset \Delta(F - \{y\})).$$

Cette proposition sera utile pour la résolution du problème.

Nous développerons d'ailleurs un algorithme capable de construire un (ou plusieurs) recouvrement (s) de E avec un nombre minimum d'e. s. i. m. Il est clair que, dans ce cas, on atteint effectivement le nombre chromatique $\alpha(G)$.

Malheureusement, dans la pratique, le nombre des e. s. i. m. est souvent trop grand pour qu'une telle méthode soit applicable. De plus, ce n'est pas tant d'atteindre le nombre chromatique qui nous intéresse ici que d'obtenir une "bonne" partition de E.

Aussi, on peut transformer le problème légèrement en adjoignant des contraintes supplémentaires (naturelles) sous la forme de fonctions de poids :

1. Le poids Pl d'une partie de E peut être défini comme la somme des poids de ses éléments :

$$S \subset E \qquad Pl(S) = \sum_{x \in S} Pl(x).$$

On pourra alors se contenter de déterminer, dans les méthodes du 2è groupe, les e. s. i. m. ayant un poids supérieur ou égal à un certain seuil P_1^0, d'où un gain d'efficacité.

Dans notre problème, le poids d'un élément x peut s'envisager d'au moins deux façons :

- Pl (x) = 1 quel que soit x ;
  on recherchera alors les e. s. i. m. dont le nombre d'éléments (examens) est supérieur ou égal à P_1^0, à l'exclusion des autres, en vue d'obtenir des nombres d'examens par couleurs les plus grands possibles (quitte à créer ensuite des couleurs (périodes) supplémentaires).

- P1 (x) = nombre des étudiants passant l'examen x, en vue d'obteni des nombres d'étudiants par couleurs les plus grands possibles (pour parvenir par exemple à une meilleure utilisation des salles d'examens).

2. On peut définir aussi le poids d'une couverture de E de plusieurs faç

Soit F une partie de l'ensemble des e. s. i. m :

$$P_2(F) = \sum_{y \in F} P2(y).$$

On pourra se contenter alors de rechercher les couvertures dont le poids est inférieur ou égal à un certain seuil P_2^0.

- $P_2(y) = 1$ quel que soit y :

Cette fonction de poids conduit alors à déterminer des colorations telles que le nombre des couleurs est "le plus près possible" du nombre chromatique. En fait, comme nous le faisions remarquer précédemment, le nombre chromatique sera effectivement atte à condition de partir de tous les e. s. i. m du graphe G.

- $P_2(y) = \text{card}(y)$ (ou $\text{card}\,\Delta(y)$) :

Cette fonction poids conduit alors à minimiser le nombre des élé ments dans les "chevauchements" d'e. s. i. m. (Les éléments d'une couverture ne forment pas nécessairement une partition de E).

Cette minimisation s'effectuera sans doute au détriment du nombre des couleurs, mais simplifiera le partage des éléments en chevauchement dans les différents e. s. i. de la partition cherchée. Cette fonction-poids rejoint le but fixé par la 1ère fonction de poid P1 : $P_1(x) = 1$ quelque soit x.

- $P_2(y) = \sum_{x \in y} P_1(x)$

dans le cas où $P_1(x)$ est le nombre des étudiants passant l'examen Ici, on cherche à minimiser le nombre des étudiants dans les "chevauchements". Ceci rejoint le point de vue adopté pour la 2èm fonction de poids $P_1(x)$.

Remarques :

- Si l'on désire se réserver le plus de chances possible d'obtenir une "bonne" couverture de E, il est nécessaire de ne pas prendre un seuil P_1^0 trop élevé, risquant de limiter les choix parmi les e. s. i. m.
- Il reste tout de même à résoudre le problème des "chevauchements" d'e. s. i. m. On peut le règler facilement en rattachant arbitrairement telle ou telle intersection d'e. s. i. m. à l'un d'entre eux, celui de plus faible poids par exemple. Cependant, il serait sans doute souhaitable d'aller plus loin encore dans l'optimisation : chercher à égaliser les poids des classes de la partition $E_1, \ldots, E_m$ par des affectations convenables des éléments en chevauchement.
- La recherche des e. s. i. m ne donne pas la solution du problème de coloration, mais elle peut en être une partie essentielle, la résolution (trouver une partition) pouvant alors se terminer manuellement, comme nous le montrerons.

2.3.2.2 Méthode de coloration de Welsh et Powell [4] :

Cette méthode heuristique consiste à affecter les éléments de l'ensemble E à des périodes dans l'ordre des degrés décroissants (c'est en fait un préordre) : on obtiendra donc ici une seule solution :

Définition :

Etant donné un graphe $G = (E, \Gamma)$ non orienté, on appelle degré de x, $x \in E$, le nombre d'arêtes ayant x pour l'une (et l'une seulement) de ses extrémités.

Ce sont évidemment les sommets de plus haut degré qui entraînent le maximum de contraintes. L'idée consiste à s'en débarasser le plus vite possible (en les affectant), en espérant que l'affectation des autres pourra s'effectuer plus facilement.

La procédure peut se dérouler comme suit :

(1) numéroter les sommets par degrés décroissants : $c_1, \ldots, c_n$.
Poser i = 1.

(2) Soit C_{i_1} le sommet de plus petit indice (de plus haut degré).

On sélectionne parmi les sommets non adjacents à c_{i_1} le sommet de plus petit indice c_{i_2}, puis, parmi les sommets non adjacents à c_{i_1} c_{i_2} celui qui a le plus petit indice c_{i_3} et ainsi de suite aussi longtemps qu'il existe des sommets non adjacents à aucun de ceux préalablement sélectionnés. Quand il n'existe plus, on colore le sommet c_{i_1} et les sommets sélectionnés d'une même couleur.

(3) On supprime du graphe tous les sommets colorés et leurs arêtes adjacentes et l'on retourne à (2) après avoir posé $i = i + 1$.

Cette méthode conduit à l'algorithme suivant dans lequel le graphe $G = (E, \Gamma)$ est représenté par sa matrice carrée booléenne R. N est l'ordre de cette matrice (card (E)) et ELEM est le tableau des codes (supposés numériques) des éléments de E. La numérotation des sommets suppose un tri préalable par degrés décroissants des éléments de ELEM et une permutation des lignes et des colonnes de R pour les ranger dans cet ordre : procédure TRI (R, M, ELEM). La suppression des sommets colorés à chaque étape se traduit par la réduction de la matrice R (on supprime les lignes et les colonnes correspondant aux sommets colorés) et par la réduction du tableau ELEM. A chaque étape, on construit un tableau booléen TAB qui représente les sommets colorés :

$$TAB(I) = \begin{cases} \underline{VRAI} & \text{si ELEM (I) est un sommet coloré} \\ \underline{FAUX} & \text{sinon.} \end{cases}$$

Les codes des sommets colorés sont édités pendant la réduction de la matrice R.

Algorithme de Welsh et Powell : procédure POWELSH :

```
PROCEDURE POWELSH (R, N, ELEM, TRI) ; VALEUR N ; BOOLEEN
                                          TABLEAU R ;
    ENTIER N ; ENTIER TABLEAU ELEM ; PROCEDURE TRI ;
    DEBUT ENTIER M, I, J, K, K1, P ; BOOLEEN TABLEAU TAB [1:N]
    M := N ;
    TRI (R, M, ELEM) ;
ITER : SI M = 0 ALORS ALLERA EXIT ;
    SI M = 1 ALORS DEBUT SORTIR(ELEM[1]) ; ALLERA EXIT FIN ;
```



```
    P:= 1 ; TAB [1] := VRAI ;
    POUR I:= 2 PAS 1 JUSQUA M FAIRE
    SI ¬ R [I,1] ALORS DEBUT P:=P+1 ; TAB [I] := VRAI ;
        POUR J:=I+1 PAS 1 JUSQUA M FAIRE
        R [J,1] := R [J,1] ∨ R [I,J] FIN
            SINON TAB [I] := FAUX ;
REDUC : K := 0 ;
    POUR I:=1 PAS 1 JUSQUA M FAIRE
    SI TAB [I] ALORS DEBUT K:=K+1 ; SORTIR (ELEM [I]) FIN
            SINON DEBUT K1 := 0 ;
    POUR J:= 1 PAS 1 JUSQUA N FAIRE
    SI TAB [J] ALORS K1 := K1 + 1
            SINON R[I-K, J-K1] := R [I, J] FIN ;
    K := 0 ;
    POUR I:=1 PAS 1 JUSQUA M FAIRE
    SI TAB [I] ALORS K:=K+1 SINON ELEM [I-K] := ELEM [I] ;
    M := M-P ; ALLERA ITER ;
EXIT : FIN POWELSH ;
```

On peut améliorer cette méthode en recalculant à chaque étape les degrés des sommets du graphe réduit, en réordonnant dans ELEM les codes par degrés décroissants et en réordonnant lignes et colonnes de R : dans TRI (R, M, ELEM).

Cet algorithme permet de déterminer un majorant du nombre chromatique $\alpha(G)$.

Théorème (Welsh et Powell) :

Les n sommets du graphe G étant numérotés i = 1, ..., n selon l'ordre décroissant de leurs degrés d_i, si k est le numéro du dernier sommet tel que $k \leq d_k + 1$, alors $\alpha(G) \leq k$.

démonstration :

si l'on suppose que, après la phase k, le sommet c_{k+1} de degré d_{k+1} n'a pas été coloré, nécessairement $d_{k+1} \geq k$, ce qui contredit l'hypothèse "k dernier sommet tel que $d_k + 1 \geq k$".

Supposons que, pour $1 \leq p \leq n$, x_{k+p} est coloré après la phase k ; si x_{k+p+1} n'est pas coloré après la phase k ; $d_{k+p+1} \geq k$.

Or $d_{k+1} \geq d_{k+p+1}$ d'après l'ordre sur les degrés,

avec $d_{k+1} + 1 < k + 1$ d'après l'hypothèse,

donc $d_{k+1} < k$ et $k > d_{k+p+1}$ ce qui est impossible, puisque $d_{k+p+1} \geq k$.

On est donc certain que l'algorithme de Welsh et Powell conduit à un maximum de couleurs égal à k.

Le graphe donné dans l'exemple suivant servira à illustrer tous les algorithmes développés ultérieurement.

exemple : Soit E = {A, B, C, D, E, F, G, H} et la relation Γ définie par la matrice booléenne R. Après la première permutation sur lignes et colonnes, nous obtenons la matrice suivante :

	A	B	C	D	E	F	G	H
A		1	1		1	1	1	
B	1				1	1	1	1
C	1				1			1
D								1
E	1	1	1			1		1
F	1	1			1			
G	1	1						
H		1	1	1	1			

⟶

	A	B	E	H	C	F	G	D
A		1	1		1	1	1	
B	1		1	1		1	1	
E	1	1		1	1	1		
H		1	1		1			1
C	1		1	1				
F	1	1	1					
G	1	1						
D				1				

L'application de l'algorithme donne la partition en e. s. i. suivante :

{A, H} , {B, C, D} , {E, G} , {F}

Le plus grand nombre k tel que $k \leq d_k + 1$ est 4, ce qui vérifie le théorème ci-dessus.

Cette méthode est intéressante car elle demande peu de place en mémoire (essentiellement un tableau booléen) et est très facile à mettre en œuvre.

2.3.2.3 Algorithme de recherche des e. s. i. m. (ou des e. s. i.) d'un graphe G = (E, Γ)

Plusieurs méthodes peuvent être utilisées ici.

2.3.2.3.1 Méthode de Maghout : (K. MAGHOUT [5] ou A. KAUFMANN [6])

La recherche d'une partie S de E est la recherche d'un vecteur booléen $(x_1, \ldots, x_n)$ qui caractérise cette partie :

$$x_i = \begin{cases} 1 \text{ si } c_i \in S \\ 0 \text{ sinon} \end{cases}$$

Soit $R = (r_{ij})$ la matrice booléenne du graphe G

$$r_{ij} = \begin{cases} 1 \text{ si } c_i \, \Gamma \, c_j \\ 0 \text{ sinon} \end{cases}$$

Dire que S est intérieurement stable s'exprime par une fonction booléenne à n variables :

$$S \cap \Gamma(S) = \emptyset \Leftrightarrow (\forall i)\ (\forall j) \neg (c_i \, \Gamma \, c_j \wedge c_i \in S \wedge c_j \in S)$$

$$\Leftrightarrow \prod_{i,j} (\overline{r_{ij} \cdot x_i \cdot x_j}) = 1$$

$$\Leftrightarrow \prod_{i,j} (\overline{r_{ij}} + \overline{x_i} + \overline{x_j}) = 1$$

Si le graphe est symétrique :

$$\overline{r_{ij}} + \overline{x_i} + \overline{x_j} = \overline{r_{ji}} + \overline{x_j} + \overline{x_i}$$

Si le graphe est sans boucle : $r_{ii} = 0$

et $\overline{r_{ii}} + \overline{x_i} + \overline{x_i} = 1$

Il suffit donc de faire le produit pour $i < j$:

(1) $$\prod_i \prod_{j>i} (\overline{r_{ij}} + \overline{x_i} + \overline{x_j}) = 1$$

Cette fonction s'exprime par une somme de monômes maximaux obtenue en développant et en réduisant grâce à la propriété $u + uv = u$. Les ensembles intérieurement stables sont obtenus en rendant un de ces monômes égal à 1, c'est-à-dire en prenant :

- $x_i = 1$ si x_i figure dans le monôme
- $x_i = 0$ si $\overline{x_i}$ figure dans le monôme.
- x_i quelconque si ni x_i ni $\overline{x_i}$ ne figurent.

On obtient les ensembles intérieurement stables maximaux en prenant $x_i = 1$ dans ce dernier cas, pour tout x_i ne figurant pas dans le monôme.

En fait, les monômes maximaux ne contiennent que des variables en négation : il suffit donc, pour obtenir les e. s. i. m., de rendre égales à 1 toutes les variables booléennes x_i ne figurant pas dans les monômes.

A tout monôme, on associe donc un e. s. i. m. formé des point c_i tels que $\overline{x_i}$ ne figure pas dans le monôme ; il n'y a pas d'autres e. s. i. m.

exemple : en reprenant le graphe du paragraphe précédent, si a, b, c, d, e, f, g, h sont les variables booléennes associées à A, B, C, D, E, F, G, H alors :

$\overline{r_{12}} + \bar{a} + \bar{b} = \bar{a} + \bar{b}$ $\qquad$ $\overline{r_{16}} + \bar{a} + \bar{f} = \bar{a} + \bar{f}$

$\overline{r_{13}} + \bar{a} + \bar{c} = \bar{a} + \bar{c}$ $\qquad$ $\overline{r_{17}} + \bar{a} + \bar{g} = \bar{a} + \bar{g}$

$\overline{r_{14}} + \bar{a} + \bar{d} = 1$ $\qquad$ $\overline{r_{18}} + \bar{a} + \bar{h} = 1$

$\overline{r_{15}} + \bar{a} + \bar{e} = \bar{a} + \bar{e}$

et $$\prod_{j>1} (\overline{r_{1j}} + \bar{a} + \overline{x_j}) = \bar{a} + \bar{b}\,\bar{c}\,\bar{e}\,\bar{f}\,\bar{g}$$

en calculant de la même façon les facteurs du 1er membre de (1) ; cette expression s'écrit :

$(\bar{a} + \bar{b}\,\bar{c}\,\bar{e}\,\bar{f}\,\bar{g})\,(\bar{b} + \bar{e}\bar{f}\bar{g}\bar{h}) + (\bar{c} + \bar{e}\bar{h})\,(\bar{d} + \bar{h})\,(\bar{e} + \bar{f}\bar{h}) = 1$

en développant et en simplifiant grâce à $u + uv = u$:

$(\bar{a}\bar{b} + \bar{a}\bar{e}\bar{f}\bar{g}\bar{h} + \bar{b}\bar{c}\bar{e}\bar{f}\bar{g})\,(\bar{c}\bar{d} + \bar{c}\bar{h} + \bar{e}\bar{h})\,(\bar{e} + \bar{f}\bar{h}) = 1$

$\bar{a}\bar{b}\bar{c}\bar{d}\bar{e} + \bar{a}\bar{b}\bar{c}\bar{f}\bar{h} + \bar{a}\bar{b}\bar{e}\bar{h} + \bar{a}\bar{e}\bar{f}\bar{g}\bar{h} + \bar{b}\bar{c}\bar{d}\bar{e}\bar{f}\bar{g} + \bar{b}\bar{c}\bar{e}\bar{f}\bar{g}\bar{h} = 1$

tous les e. s. i. m. sont alors déterminés par les monômes ci-dessus, comme nous l'avons dit ; ce sont :

{F, G, H}, {D, E, G}, {C, D, F, G}, {B, C, D}, {A, H}, {A, D}.

2. 3. 2. 3. 2) Algorithme pour la construction des e. s. i. m.) d'un graphe (proposé par J. HALL [7] et B. ROY [8])

Soit G = (E, Γ) un graphe non orienté et sans boucle avec $E = \{c_1, \ldots, c_n\}$ et $G_1, \ldots, G_i, \ldots, G_n$ les sous-graphes engendrés par les sous-ensembles $F_1 = \{c_1\}, \ldots, F_i = \{c_1, \ldots c_i\}, \ldots F_n = E$ et les relations Γ_i, restrictions de Γ à F_i.

Tout e. s. i. m. S de G_{i-1} engendre pour G_i les e. s. i. suivants :

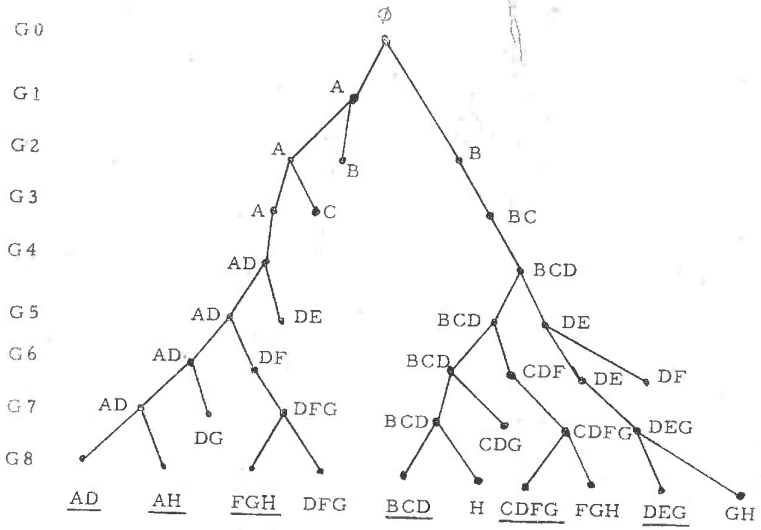
- si $S \cap \Gamma_i (c_i) \neq \emptyset$: S et $S \cup \{c_i\} - \Gamma_i (c_i)$
- si $S \cap \Gamma_i (c_i) = \emptyset$: $S \cup \{c_i\}$.

On peut démontrer que tous les e. s. i. m. sont engendrés, mais on peut fort bien avoir répétition de certains e. s. i. m., car rien n'empêche que l'on ait, avec $S \neq S'$:

$$S \cup \{c_i\} - \Gamma_i (c_i) = S' \cup \{c_i\} - \Gamma_i (c_i)$$

D'autre part, on engendre des e. s. i. non maximaux comme le montre l'exemple suivant :

exemple : avec le graphe précédent (2. 3. 2. 2), on obtient :



On constate, comme prévu, la nécessité de développer, à chaque, étape, une phase d'élimination des e. s. i. non maximaux et des e. s. i. m. redondants.

En fait, dans beaucoup d'algorithmes proposés, on ne peut éviter cette lourde phase d'élimination. De plus, la méthode présentée ici a le gros inconvénient de ne pas donner de résultat avant la fin : aucun e. s. i. du graphe final n'est connu comme résultat partiel. Nous montrerons un algorithme qui évite ces inconvénients.

En fait, nous verrons deux algorithmes, l'un pour la détermination des e. s. i. m., l'autre pour la recherche des couvertures minimales, qui procèdent de la même manière arborescente. Avant de les développer, il est nécessaire de voir les idées de bases sur lesquelles ils reposent et comment ils fonctionnent. C'est l'objet du paragraphe suivant que nous illustrerons par le premier de ces algorithmes.

2.3.2.3.3) Procédures par séparation et évaluation
(voir P. HERVE [9])

a) Introduction

Soit à résoudre le problème suivant :

"Etant donné l'ensemble U des solutions admissibles d'un problème, trouver dans U des solutions u_0 qui satisfont un critère donné".

Les procédures arborescentes telles qu'elles sont développées plus loin consistent à examiner des sous-ensembles de U de plus en plus petits, jusqu'à pouvoir mettre en évidence des solutions satisfaisant à ce critère.

exemple : étant donné le graphe $G = (E, \Gamma)$ non orienté et sans boucle, proposons-nous de rechercher les e. s. i. m. u_0 de G tels que leur poids $P(u_0)$ soit supérieur ou égal à P_0.

On pose alors

U = ensemble des e. s. i. de G.

Il s'agit de trouver dans U des e. s. i. satisfaisant aux deux conditions suivantes :

- être maximaux
- avoir un poids supérieur ou égal à P_0.

b) Procédure de séparation :

En général, la procédure de séparation des éléments de U s'exprime par une relation binaire Φ définissant un graphe dans $\rho(U)$ satisfaisant aux conditions suivantes :

axiome de finitude :

(1) $\hat{\Phi}(U)$ est fini ($\hat{\Phi} = \bigvee_{i=0}^{+\infty} \Phi^{(i)}$, où $\Phi^{(0)}$ est l'égalité, est la fermeture transitive de Φ).

axiome de conservation :

$$(2)\ \forall X \in \hat{\Phi}(U) : \Phi(X) \neq \emptyset \Rightarrow \begin{cases} \forall\ Y \in \Phi(X) : \emptyset \subset Y \subset X \\ \bigcup Y = X \\ Y \in \Phi(X) \end{cases}$$

autrement dit :

(1) : l'ensemble des parties de U obtenues par séparation est fini.

(2) : Si une partie X peut être séparée :

- les parties obtenues par séparation ne sont pas vides et sont strictement incluses dans X.
- La réunion de ces parties est l'ensemble X.

Si on pose $\mathcal{U} = \hat{\Phi}(U)$, on obtient ainsi le graphe $\mathcal{G} = (\mathcal{U}, \Phi)$

exemple : en reprenant notre problème, on peut définir très simplement la relation binaire Φ :

Si les sommets de E sont numérotées $c_1, \ldots, c_n$ et ordonnés da l'ordre de la numérotation :

le premier élément de E, c_1, permet de séparer U en deux parti

- les e. s. i. qui contiennent c_1 (et donc qui ne contiennent aucun élément de $\Gamma(c_1)$)
- les e. s. i. qui ne contiennent pas c_1.

On sépare ensuite, suivant le même principe, chacune des parties constituées grâce à un élément (parmi les suivants de c_1) défini comme suit :

- pour la première partie, on prend le premier sommet (dans l'ordr de numérotation) de $E - \{c_1\} - \Gamma(c_1)$
- pour la deuxième partie, on prend c_2.

On obtient ainsi le graphe $\mathcal{G} = (\mathcal{U}, \Phi)$:

- chaque sommet de ce graphe représente une partie de U définie comme l'ensemble des e. s. i. contenant tous les éléments d'un certain ensemble A et aucun de ceux d'un ensemble $\overline{A}$ ($A \subset E$ et $\overline{A} \subset E$)
- tout sommet du graphe tel que $E-A-\overline{A}$ est non vide a deux suivants définis à partir du premier élément x de cet ensemble par :

$$\begin{cases} A \cup \{x\}, \ \overline{A} \cup \Gamma(x) & \text{pour le premier} \\ A, \ \overline{A} \cup \{x\} & \text{pour le second.} \end{cases}$$

D'après ce procédé de formation de A et $\overline{A}$, on a toujours :

$$A \cap \overline{A} = \emptyset$$

et $\overline{A} \supset \Gamma(A)$ donc A est un e. s. i.

Montrons que Φ satisfait aux axiomes (1) et (2) :

(1) $\hat{\Phi}(U)$ est fini puisque $\hat{\Phi}(U) \subset \wp(U)$ qui est fini.

(2) Quel que soit X sommet du graphe caractérisé par A et $\overline{A}$ et non feuille ($E - A - \overline{A} \neq \emptyset$) :

$X = X_1 \cup X_2$ où

- X_1 est l'ensemble des e. s. i. qui contiennent A et x (donc pas $\overline{A} \cup \Gamma(x)$).
- X_2 est l'ensemble des e. s. i. qui contiennent A et pas x (donc pas $\overline{A} \cup \{x\}$).

De plus $X_1 \cap X_2 = \emptyset$

et X_1 et X_2 ne sont pas vides puisqu'ils contiennent respectivement $A \cup \{x\}$ et A qui sont des e. s. i., ce qui démontre que ni X_1 ni X_2 ne sont égaux à X.

On a démontré aussi que la relation Φ définit une partition (X_1, X_2) dans X.

De toute évidence, le graphe $\mathcal{G} = (\mathcal{U}, \Phi)$ est une arborescence de racine U et dont les feuilles sont telles que $E - A - \overline{A} = \emptyset$. On peut facilement démontrer (B. ROY [8]) qu'il y a bijection entre les feuilles de cette arborescence et les e. s. i. du graphe $G = (E, \Gamma)$.

Nous disposons donc d'une procédure de séparation de tous les e. s. i. d'un graphe non orienté et sans boucle.

c) examen d'un sommet de $\mathcal{G}$:

A chaque étape de la procédure, on va examiner un sommet s de $\mathcal{G}$: cet examen consiste en certains tests et calculs fournissant une information sur s : cette information peut être, par exemple :

- connaissance d'une solution u_0 de s.
- connaissance d'une évaluation (selon une certaine fonction économique) de la valeur des solutions u_0 de s.
- connaissance du fait que s est vide.
- connaissance du fait que s ne contient pas de solution u_0 du problème posé, etc...

exemple : dans notre problème, il est possible de savoir si un sommet s du graphe ne contient pas de solution u_0 par un test reposant sur deux conditions nécessaires d'existence d'une solution. Si l'une de ces conditions n'est pas vérifiée, le sommet s ne possède pas de solution u_0.

proposition :

Etant donnés un graphe G = (E, Γ) sans boucle, deux sous-ensembles de E, A et $\overline{A}$, tels que $A \cap \overline{A} = \emptyset$,

S'il existe un e. s. i. m. S tel que $A \subset S$ et $\overline{A} \cap S = \emptyset$, alors la propriété ($\mathcal{C}$) est vérifiée ;

($\mathcal{C}$) : Tout sommet $x \in B = \overline{A} - \Gamma(A)$ admet au moins un successeur qui n'appartient ni à B ni à Γ (A).

démonstration :

Soit S un e. s. i. m. tel que $A \subset S$ et $\overline{A} \cap S = \emptyset$. Si la propriété ($\mathcal{C}$) n'est pas satisfaite, il existe $x_0 \in B$ tel que

$\Gamma(x_0) \subset \Gamma(A) \cup B = \overline{A} \cup \Gamma(A)$ donc

$\Gamma(x_0) \cap S \subset (\overline{A} \cup \Gamma(A)) \cap S$ qui est vide puisque $S \cap \overline{A} = \emptyset$

et $S \cap \Gamma(A) = \emptyset$ (S est un e. s. i. m. qui contient A).

Donc $(S \cup \{x_0\}) \cap \Gamma(S \cup \{x_0\}) = \emptyset$ puisque le graphe est sans boucle et S n'est pas maximal puisque $S \cup \{x_0\}$ est un e. s. i.

Cette propriété donne une condition nécessaire d'existence d'un e. s. i. m. dans un sommet du graphe caractérisé par A et $\overline{A}$. (La condition ($\mathcal{C}$) est supposée vraie quand $B = \emptyset$).

Une autre condition nécessaire à vérifier est que le poids de $E - \Gamma(A)$ soit supérieur ou égal à P_0, puisque tout présumé e. s. i. m. est inclus dans cet ensemble.

Si, pour un sommet s, l'une de ces deux propriétés n'est pas vérifiée il ne contient pas de solution ; il est alors inutile d'examiner ses descendants dans l'arborescence qui sont dans le même cas. Si ces deux conditions sont vraies, on vérifiera si s est une feuille de l'arborescence, c'est-à-dire si

$E - A - \overline{A}$ (ou $E - A - \Gamma(A) - B) = \emptyset$. Dans ce cas, on sait trouver une solution u_0 du problème posé : A est un e. s. i. m. de poids supérieur ou égal à P_0.

d) procédure de résolution :

A chaque fois qu'on examine un sommet de $\mathcal{C}$, on le marque : ainsi, à chaque étape, un sommet est marqué ou non marqué.

A une étape de la procédure, un sommet peut devenir suffisamment connu, ce qui signifie qu'on estime n'avoir plus besoin de le connaître plus en détail dans la suite du calcul.

Si un sommet est suffisamment connu, il en est de même de tous ses descendants, et réciproquement. On dira qu'à une étape, un sommet est fermé si tous ses suivants non marqués sont suffisamment connus. Il est ouvert dans le cas contraire.

Si s ($\mathcal{R}$) exprime que le sommet s possède la propriété $\mathcal{R}$; c'est-à-dire :

- S = suffisamment connu
- M = marqué
- F = fermé

On a les relations :

(1) $s(F) \Leftrightarrow \forall t \in \Phi(s) : t\ (M \vee S)$

(2) $s(S) \Leftrightarrow \forall t \in \hat{\Phi}(s) - \{s\} : t\ (S)$

exemple :

Dans notre problème, nous pouvons considérer comme marqués les sommets que l'on examine, comme suffisamment connus les sommets tels que l'examen a été négatif (une des deux conditions nécessaires non vérifiée) ou tel qu'il ait montré que le sommet est une feuille ; dans le premier cas, les descendants des sommets suffisamment connus (s'ils en ont) sont eux aussi suffisamment connus.

C'est la remarque que nous avions faite précédemment.

e) Description de la procédure :

1) On marque le sommet racine U.

2) à chaque étape :

a) s'il existe des sommets marqués et ouverts, on en choisit un, s, sinon on va à 3).

b) Dans Φ (s), on choisit un sommet t non marqué et non suffisamment connu. (Il en existe sinon s serait fermé).

c) On marque t.

d) On examine t ; on déduit si certains sommets de $\mathcal{G}$ sont devenus suffisamment connus ou fermés.

e) On retourne à 2).

3) Fin de la procédure.

Si l'on fait l'hypothèse fondamentale suivante :

L'examen d'un sommet terminal le rend suffisamment connu.

Cette procédure se justifie aisément ; en effet :

- Elle se termine toujours puisque le nombre maximum de sommets que l'on peut marquer est fini (en vertu de l'axiome de finitude).
- Quand la procédure est achevée, tous les sommets de $\mathcal{G}$ sont suffisamment connus :

d'apres la relation (2), il suffit de le montrer pour U.

Si la procédure est achevée : $\forall s \in \mathcal{G} : s(M) \Rightarrow s(F)$.

Si l'on avait $U(M \wedge \bar{S})$, il en résulterait :

- $U(M) \Rightarrow U(F)$ entraînerait $\forall s \in \Phi(U) : s(M \vee S)$
- $U(\bar{S})$ entraînerait : $\exists s \in \Phi(U) : s(\bar{S})$

donc, il existerait $s \in \Phi(U)$ tel que $s(M \wedge \bar{S})$.

Par récurrence, il existerait un chemin issu de U et aboutissant à un dont tous les sommets seraient marqués et insuffisamment connus : ceci est impossible puisque le sommet terminal est suffisamment connu.

Cette méthode s'applique à des problèmes très divers : combinatoire, programmes à variables entières ou mixtes, programmes linéaires où le critère est souvent de rechercher des solutions ε- minimales de U : étant donnés une fonction $f : U \to \mathbb{R}$, et un nombre $\varepsilon > 0$, une solution u_0 de U est dite ε - minimale si

$$(\forall u \in U)\ (f(u_0) < f(u) + \varepsilon)$$

De l'examen d'une feuille s de $\mathcal{G}$, on doit alors déduire une évaluation par défaut des solutions minimales de s.

f) Choix d'un sommet ouvert et marqué

Deux règles de choix se dégagent, conduisant à deux grands ensembles de méthodes :

1. Règle L. I. F. O. (last in first out) : on choisit systématiquement à chaque étape le sommet ouvert marqué le plus récemment : ceci revient à parcourir les chemins de $\mathcal{G}$ en marquant les sommets qu'on rencontre jusqu'à ce qu'on ne puisse plus continuer, c'est-à-dire jusqu'à ce que le dernier sommet marqué soit fermé ; dans ce cas, on remonte le chemin parcouru en sens inverse jusqu'à rencontrer un sommet ouvert à partir duquel on repart en avant de nouveau. (C'est la méthode que nous utiliserons pour nos problèmes). Ces méthodes portent le nom de P. S. E. S. (procédures par séparation et évaluation séquentielle).

2. Règle de l'évaluation minimale : si, à chaque sommet marqué, on peut associer une évaluation par défaut de ses solutions minimales, la regle consiste à choisir à chaque étape un sommet ouvert marqué d'évaluation minimale. Ces méthodes portent le nom de P. S. E. P. (procédures par séparation et évaluation progressive), dont les procédures "Branch and Bound" sont une sous-classe.

Application : les P. S. E. S. à graphe arborescent :

c'est très exactement le cas de l'exemple que nous avons étudié ; nous avons vu que le graphe $\mathcal{G}$ est une arborescence et que chaque sommet est origine de 0 ou 2 arcs. Il s'agit donc de cheminer dans une arborescence : on choisira donc systématiquement le dernier sommet marqué et ouvert s (règle L. I. F. O.) et l'on se fixera un ordre pour le choix du sommet t de Φ (s) non marqué que l'on examine. (Ordre transverse).

Dans notre exemple, nous prendrons toujours comme premier sommet celui qui, à partir de $(A, \bar{A})$ est défini par $(A \cup \{x\},\ \bar{A} \cup \Gamma^{+}(x))$ où x est le

premier élément de E-A-Γ (A) - B.

Ceci revient à examiner les sommets dans un certain ordre (dit ordre de Tarry sur l'arborescence [8]). Dans notre exemple, cet ordre correspond très exactement à l'ordre lexicographique sur les éléments de E.

D'autre part, il faut pouvoir connaître les sommets marqués et ouverts, donc en tenir une liste (d'une façon ou d'une autre) ; et, comme on choisit systématiquement le dernier sommet marqué qui est ouvert, il est naturel de les mettre dans une pile.

Celle-ci à deux buts :

- permettre un cheminement dans l'arborescence (en indiquant où l'on en est).
- Tenir une liste des sommets marqués et ouverts dans l'ordre de la marque. (Quand un sommet est fermé, il quitte la pile).

En fait, dans une procédure à graphe arborescent, il est inutile de faire figurer dans la pile le dernier sommet t non marqué et insuffisamment connu que l'on choisit dans Φ (s) : en effet, l'examen peut le rendre suffisamment connu, donc ce sommet devrait être retiré de la pile, ainsi que son prédécesseur qui deviendrait fermé : on dépilerait donc deux fois au moins.

De plus, il est inutile de faire figurer U dans la pile au départ : on sait que U est marqué et ouvert jusqu'à la fin de la procédure : quand la hauteur de la pile est nulle, la procédure est terminée (puisque tous les sommets sont suffisamment connus, et que U l'est aussi).

Ces remarques conduisent alors à l'algorithme suivant (issu de la procédure générale) pour les P. S. E. S. à graphe arborescent et dichotomique :

- pile (h) est une pile dans laquelle on range les sommets
- s est le sommet marqué et ouvert le plus récemment
- t_s est le plus petit successeur de s dans Φ (s) non marqué et insuffisamment connu.

On particularise le cas de U puisqu'il ne figure jamais dans la pile.

1) - s = U ; h = 0 ;
   - empiler t_s : h = h + 1 ; pile (h) = t_s ;
   - examen : si t_s est suffisamment connu : dépiler : h = h - 1,
     sinon s = t_s ;
2) - si t_s n'est pas le dernier élément de Φ (s), empiler t_s : h = h + 1
     pile (h) = t_s ;
   - examen : si t_s est suffisamment connu : si h = 0 fin de la procédure
     sinon dépiler : h = h - 1 ;
   - s = pile (h) ; aller à 2).

C'est très exactement la procédure que l'on retrouvera pour nos deux algorithmes, aux améliorations de programmation près.

2.3.2.3.4 P. S. E. S. pour la recherche des e. s. i. m. de poids supérieur ou égal à P_0 dans le graphe G = (E, Γ).

- Presque tout a été dit dans le paragraphe précédent concernant cette procédure. Il faut cependant ajouter certaines modalités pratiques. Un sommet de l'arborescence $\mathcal{C}$ est caractérisé par deux parties de E : A et $\bar{A}$, ou, si l'on veut, par A et B = $\overline{A}$ - Γ (A). Pour ranger un sommet sur la pile, il suffirait alors de ranger A et B. En fait, nous rangerons les parties A_0 et B_0, correspondant au prédécesseur du sommet considéré et l'élément x, premier élément dans l'ordre lexicographique de E - A_0 - Γ (A_0) - B_0, qui sert à définir A = $A_0 \cup \{x\}$ et B = $B_0 - \Gamma (A_0 \cup \{x\})$, ce qui revient au même, mais permet de récupérer x au moment ou l'on dépile pour entrer dans la 2è branche, caractérisée par A_0 et $B_0 \cup \{x\}$.

- Chaque partie X de E peut être représentée sous la forme d'un vecteur booléen à n composantes VX [i] :

$$VX[i] = \begin{cases} 1 \text{ si } c_i \in X \\ 0 \text{ sinon.} \end{cases}$$

- Les opérations sur les ensembles s'expriment alors par des opérations booléennes sur les vecteurs :

$A \cap B$ correspond à $VA \wedge VB$

$A \cup B$ correspond à $VA \vee VB$ ...

- Le graphe peut être représenté par une matrice booléenne R, éventuellement triangulaire, et, dans ce cas, rangée sous la forme d'un tableau unidimensionnel.

- La pile peut être représentée par un tableau booléen à 3 dimensions PILE [1:3, 1:N, 1:N] ; en effet, sa hauteur est majorée par n, nombre d'éléments de E ; chaque élément de la pile correspond au rangement des vecteurs booléens correspondant à A_0, B_0 et $\{x\}$.

- Le vecteur représentant Γ (A) est obtenu simplement par la multiplication booléenne de la matrice R par le vecteur VA.

- De même, il n'y a aucune difficulté à calculer le vecteur représentant le premier élément, dans l'ordre lexicographique, d'un ensemble.

Aussi, pour la clarté de l'exposé de cet algorithme, même si la syntaxe en est empruntée à Algol 60, nous conserverons les principales opérations ensemblistes, en sachant bien qu'elles se traduisent pas des opérations sur des vecteurs booléens.

Nous nous contenterons d'introduire :

- 2 procédures : EMPILER (U, V, W) : et DEPILER (U, V, W) (voir progr
- 2 "pseudo fonctions-procédures" :
  - PREMIER (A), dont la valeur est un vecteur représentant le premier élément de l'ensemble correspondant au vecteu
  - Γ (A) dont la valeur est un vecteur représentant l'ensemble des successeurs de l'ensemble correspondant à A pour le graphe $G = (E, \Gamma)$, en sachant bien que ces "fonctions-procédures" pourraient aisément s'écrire sous forme de procédures Algol.
- E qui est le vecteur unité.
- ϕ qui est le vecteur nul.
- Une instruction d'écriture SORTIR (A) qui a pour effet d'écrire les éléments du vecteur booléen A.

- Une fonction-procédure P (A) dont l'argument est un vecteur A représentant une partie de E et la valeur un nombre réel représentant le poids de cette partie. Ces valeurs se trouveront éventuellement dans un tableau.

Les opérations - affectation des valeurs d'un vecteur à un autre.
- différence de deux vecteurs (correspondant à la différence d'ensembles) sont admises.

Le prédicat $(\forall x \in B)\ (\Gamma(x) \cap (E - \Gamma(A) - B) \neq \phi)$ sera conservé tel quel, mais pourrait facilement se traduire par une fonction-procédure booléenne, à condition de disposer de procédures pour calculer Γ (A) et pour isoler un vecteur composante d'un vecteur (projection).

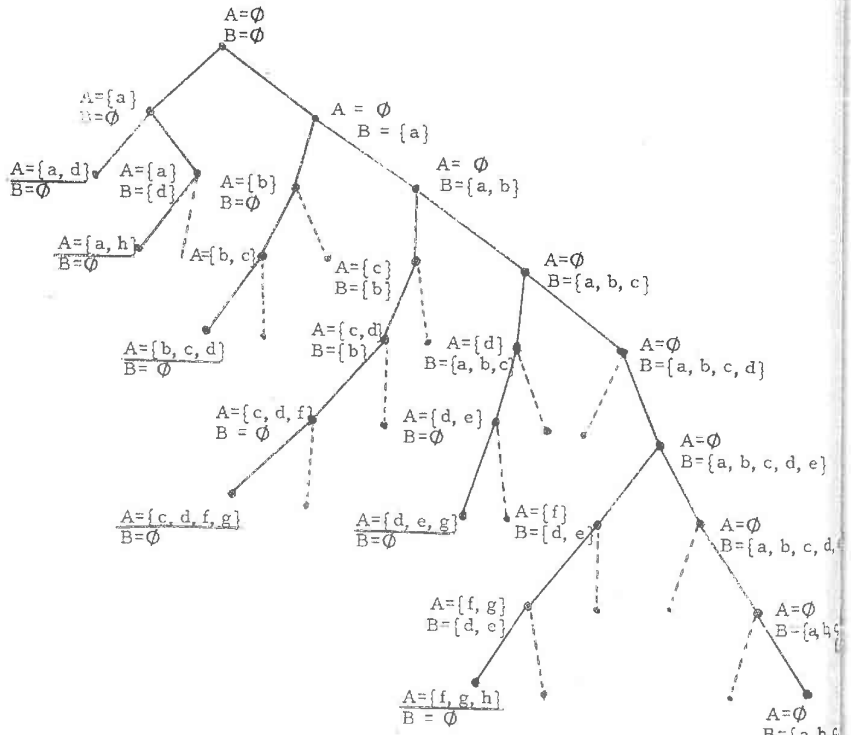
```
PROCEDURE ESIM (R, N, P, P0) ; BOOLEEN TABLEAU R ; ENTIER N ;
      REEL PROCEDURE P ; REEL P0 ; VALEUR N ;
DEBUT COMMENTAIRE Cette procédure énumère les e. s. i. m. d'un gra
                  (dont la matrice booléenne est R) dont le poids est supérieu
                  ou égal à P0 ;
BOOLEEN TABLEAU PILE [1:3, 1:N, 1:N], E, A, B, Ø, X, GA [1:N] ;
ENTIER H, I, J ;
PROCEDURE EMPILER (U, V, W) ; BOOLEEN TABLEAU U, V, W ;
DEBUT COMMENTAIRE Cette procédure range au sommet de la pile PI
                  les trois vecteurs U, V, W après avoir augmenté sa hauteur
                  de 1.
ENTIER K ; H := H + 1 ;
POUR K := 1 PAS 1 JUSQUA N FAIRE
DEBUT PILE [1, H, K] := U [K] ; PILE [2, H, K] := V [K] ; PILE
                                    [3, H, K] := W [K] FIN
FIN procédure empiler ;
PROCEDURE DEPILER (U, V, W) ; BOOLEEN TABLEAU U, V, W ;
DEBUT COMMENTAIRE Cette procédure affecte les valeurs qui sont au
                  sommet de la pile PILE aux vecteurs booléens U, V, W, puis
                  elle diminue la hauteur H de la pile de 1 ;
ENTIER K ;
POUR K:= 1 PAS 1 JUSQUA N FAIRE
DEBUT U [K] := PILE [1, H, K] ; V [K] := PILE [2, H, K] ; W [K] :=
                                    PILE [3, H, K] ;
FIN ; H := H-1 ; FIN procédure dépiler ;
```

autres déclarations :

$\Gamma(A)$, PREMIER (A), $(\forall X \in B)(\Gamma(X) \wedge Y \neq \emptyset)$


```
INIT :  POUR I:=1 PAS 1 JUSQUA N FAIRE
        DEBUT E[I] := VRAI ; Ø [I] := FAUX FIN ;
        H := 0 ; X:= PREMIER (E) ; GA := Γ (X) ;
        EMPILER (Ø, Ø, X) ;
        SI P (E - GA) ≥ P0 ALORS
        DEBUT SI E - X - GA = Ø ALORS ALLERA SØRT
              SINON DEBUT A := X ; B:= Ø ; ALLERA EXAM FIN
        SINON ALLERA DEPIL ;

MARQ :  SI E - A - GA - B = Ø ALORS ALLERA SØRT ;
        X := PREMIER (E - A - GA - B) ;
        EMPILER (A, B, X) ;
        A := A ∨ X ; GA := Γ (A) ; B := B ∨ GA ;

EXAM :  SI P (E - GA - B) ≥ P0 ∧ (∀ X ∈ B) (Γ (X) ∧ (E - GA - B) ≠ Ø)
        ALORS ALLERA MARQ ;

DEPIL : SI H = 0 ALORS ALLERA EXIT
        SINON DEBUT DEPILER (A, B, X) ; B := B∨X ; GA := Γ (A) ;
        ALLERA EXAM FIN ;

SØRT :  SORTIR (A) ; ALLERA DEPIL ;

EXIT :  FIN procédure ESIM ;
```

exemple : Si nous reprenons le graphe qui a servi à illuster la méthode de Welsh et Powell (§ 2.3.2.2), si nous notons les éléments en lettres minuscules, si $P_0 = 0$, $P(x) = 1$ pour tout $x \in B$, $P(\phi)$, nous obtenons l'arborescence $\mathcal{E}$ que nous limiterons : les arcs pendants de l'arborescence dessinés en pointillés aboutissent à une feuille telle que l'une des deux conditions au moins de EXAM, est fausse, d'où nécessité de faire un retour en arrière en dépilant et aller à DEPIL.



Les e. s. i. m. de poids supérieur à $P_0 = 0$ sont soulignés. Nous retrouvons bien les résultats de la méthode de Maghout.

Remarque :

- Cet algorithme permet aussi de déterminer soit tous les e. s. i. m. (en posant $P_0 = 0$), soit l'e. s. i. m. ou les e. s. i. m. de poids maximum : au départ, il suffit de poser $P_0 = 0$ et, à chaque fois que l'on détermine un nouvel e. s. i. m. de poids supérieur ou égal à P_0, de poser P_0 = poids de cet e. s. i. m.

2.3.2.4 Algorithmes de recherche d'une couverture de E par des e. s. i. du graphe G = (E, Γ)

2.3.2.4.1 méthode directe

Cette méthode triviale consiste à considérer les e. s. i. m. obtenus par l'une des méthodes précédentes et numérotés d'une manière quelconque (par exemple dans l'ordre des poids décroissants) S_1, S_2, ... et à recouvrir E par les sous-ensembles stables intérieurement, S_1, $S_2 - S_1$, $S_3 - S_2 - S_1$, ... etc. On y parvient toujours, puisque chaque élément de E appartient à un e. s. i. m. au moins, mais on risque d'obtenir une couverture dont le poids est éloigné du poids minimal. L'algorithme suivant propose de rechercher toutes les couvertures de E par des e. s. i. m. telles que leurs poids soient inférieurs ou égaux à un nombre P_0 donné.

2.3.2.4.2 P. S. E. S. pour la recherche d'une couverture minimale de X dans le graphe biparti H = (Y, X, Δ) dont le poids est inférieur ou égal à P_0.

Nous avons vu précédemment comment notre problème de coloration pouvait se ramener à cette recherche d'une couverture minimale de poids inférieur ou égal à un seuil P_0. L'algorithme que nous allons développer s'appuie sur quelques remarques :

- si un élément $a \in X$ n'a qu'un prédécesseur y (il existe y unique tel que $a \in \Delta(y)$), il est certain que toute couverture contient y. On peut donc retirer du graphe le sommet y et $\Delta(y)$ et travailler sur le graphe ainsi réduit.

- si un sommet $y \in Y$ est isolé (c'est-à-dire qu'il n'existe pas d'ar partant de y), alors ce sommet peut être retiré puisqu'il n'appar à aucune couverture minimale.

Notre algorithme va chercher à fabriquer de tels sommets pour travail sur des graphes de plus en plus réduits.

En nous appuyant à nouveau sur la théorie générale, nous allons dé nir le graphe $\mathcal{C} = (\mathcal{U}, \Phi)$ dans lequel nous cheminerons à la recherche des couvertures minimales. Nous poserons H (V, W), le graphe $H = (Y, X, \Delta)$ réduit aux ensembles $V \subset Y$ et $W \subset X$.

Nous supposerons les éléments de Y numérotés $y_1, \ldots, y_m$ et ordonnés dans l'ordre de cette numérotation.

Pour la racine U de $\mathcal{C}$, on pose : U est l'ensemble des parties de I qui contiennent A et aucun élément de B, A et B étant définis comme suit :

- A est l'ensemble des éléments de Y prédécesseurs d'un élément de $H = (Y, X, \Delta)$ dont ils sont les seuls prédécesseurs.
- B est l'ensemble des éléments de Y - A isolés dans le graphe réduit H (Y - A, X - Δ (A)).

Il est évident que U contient toutes les couvertures minimales de X pour le graphe H.

L'ensemble $\mathcal{U}$ et la relation Φ sont alors définis par récurrence à partir de U :

Chaque sommet de $\mathcal{C}$ est l'ensemble Z des parties de Y qui contiennent tous les éléments d'une partie A de Y et aucun élément d'une partie B de Y. Φ (Z) est alors constitué de (si Y - A - B $\neq$ 0 et y premier élément de Y - A - B dans l'ordre lexicographique) :

- Z1 : ensemble des parties de Y caractérisé par :

$$\begin{cases} A1 = A \cup \{y\} \\ B1 = B \cup \{\text{sommets isolés dans le sous-graphe } H (Y - A_1 - B,\ X - \Delta (A_1))\} \end{cases}$$

- Z2 : ensemble des parties de Y caractérisé par :

A2 = A $\cup$ {suivants de y dans Y - A - B - {y} qui couvrent des sommets dont ils sont les seuls prédécesseurs dans le graphe réduit H (Y-A-B-{y}, X - Δ (A))}

B2 = B $\cup$ {y} $\cup$ {sommets isolés dans le graphe réduit H (Y - A2 - B - {y}, X - Δ (A2))}

Alors, on montre facilement que :

Z1 $\cap$ Z2 = $\emptyset$ et que

Z1 $\cup$ Z2 $\subset$ Z, cette condition étant différente de l'axiome de conservation.

Cependant, on montre facilement que :

- Z contient toutes les couvertures minimales de X qui contiennent A et d'intersection vide avec B.
- Z1 $\cup$ Z2 contient toutes ces couvertures minimales.

Donc, on peut se contenter de cette relation puisqu'il n'y a pas "perte" de couvertures minimales pendant la séparation.

D'autre part $\hat{\Phi}$ (U) est fini, puisque $\hat{\Phi}$ (U) $\subset \mathcal{P}$ (U) qui est fini.

Le graphe $\mathcal{C} = (\mathcal{U}, \Phi)$ est donc, aussi, une arborescence dichotomique. Les feuilles sont telles que Y - A - B = $\emptyset$.

Examen d'un sommet s de $\mathcal{C}$:

On peut savoir si ce sommet ne contient pas de couvertures minimales par la condition nécessaire suivante : si le sommet s de $\mathcal{C}$, caractérisé par A et B, contient une solution minimale, alors :

$$(\forall y \in A)\ (\Delta (y) \not\subset \Delta (A - \{y\})).$$

D'autre part, il est nécessaire que le poids de A, P (A), soit inférieur ou égal à P_0.

Si l'une de ces deux conditions n'est pas vérifiée, le sommet s ne contient pas de solution.

Si ces deux propriétés sont vérifiées et si s est une feuille de l'arborescence caractérisée par A et B, alors, A est une couverture minimale de X

de poids inférieur ou égal à P_0 dans le graphe $H = (Y, X, \Delta)$ (d'après la proposition énoncée en 2.3.2.1.2. sur les couvertures minimales).

La procédure de résolution est alors sensiblement la même que précédemment, aux différences de programmation près.

Toutes les remarques qui ont été faites pour le précédent algorith P.S.E.S. restent valables :

- chaque partie de X est représentée par un vecteur booléen à n composantes,
- chaque partie de Y est représentée par un vecteur booléen à m composantes.
- Le graphe H (Y, X, Δ) peut être représenté par une matrice bool enne $R_{m \times n}$.

Nous aurons besoin :

- des procédures EMPILER et DEPILER (déjà vues , mais en rem plaçant N par M).
- des "pseudo fonctions-procédures" : PREMIER, Δ (déjà vues) et

PREDEC (U, V, W) : suivants de U (= $\emptyset$ ou U n'ayant qu'une compos non nulle) dans V qui couvrent des sommets dont ils sont les seuls prédécesseurs dans H (V, W) ; quand U = $\emptyset$, au lieu de "suivants de dans V" on lira : "éléments de V".

ISOLES (V, W) : "sommets isolés" dans le sous-graphe H (V, W).

- de la fonction-procédure P qui calcule le poids de tout vecteur booléen à m composantes.

Le prédicat $(\forall y \in A)(\Delta(y) \not\subset \Delta(A - \{y\}))$ sera conservé tel quel.


```
PROCEDURE COUV (R, M, N, P, P0) ; BOOLEEN TABLEAU R ;
    ENTIER M, N ; REEL PROCEDURE P ; REEL P0 ; VALEUR M, N ;
DEBUT COMMENTAIRE Cette procédure énumère les couvertures mini-
     males d'un graphe biparti (dont la matrice booléenne est R) dont le
     poids est inférieur ou égal à P0 ;
BOOLEEN TABLEAU PILE [1:3, 1:M, 1:M], Y, A, B, Ø, Y1, Z [1:M],
                                        X, DA [1:N] ;
ENTIER H, I, J ;
PROCEDURE EMPILER (U, V, W) ; BOOLEEN TABLEAU U, V, W ;
DEBUT COMMENTAIRE  Voir précédemment ... ;
  .
  .
  .
PROCEDURE DEPILER (U, V, W) ; BOOLEEN TABLEAU U, V, W ;
DEBUT COMMENTAIRE  Voir précédemment ... ;
  .
  .
  .
     autres déclarations :  Δ (A), PREMIER (A), PREDEC (U, V, W),
                            ISOLES (U, V, W), (∀ Z ∈ A) (Δ (Z) ⊄
                                                        Δ (A - Z))
  .
  .
  .
     .....
```

```
INIT :   POUR I:=1 PAS 1 JUSQUA M FAIRE
         DEBUT Y [I] := VRAI ; Ø [I] := FAUX FIN ;
         POUR I:= 1 PAS 1 JUSQUA N FAIRE X [I] := VRAI ;
         H:= 0 ;
         A := PREDEC (Ø, Y, X) ;
         B := ISOLES (Y - A, X - Δ (A)) ; Y1 := Y - A - B ;
         SI P (A) > P0 ALORS ALLERA EXIT ;
         SI Y1 = Ø ALORS DEBUT
              COMMENTAIRE A est une couverture minimale de poids
                          ≤ P0 et c'est la seule ;
              SORTIR (A) ; ALLERA EXIT FIN ;
         Z := PREMIER (Y1) ;
         EMPILER (A, B, Z) ;

SEPAR1 : A := A ∨ Z ;
         B := B ∨ ISOLES (Y - A - B, X - Δ (A)) ;

EXAM :   SI P (A) ≤ P0 ∧ (∀ Z ∈ A) (Δ (Z) ∉ Δ (A - Z))
         ALORS ALLERA MARQ SINON ALLERA DEPIL ;

MARQ :   Y1 = Y - A - B ;
         SI Y1 = Ø ALORS ALLERA SØRT
         SINON DEBUT Z:= PREMIER (Y1) ;
                     EMPILER (A, B, Z) ; ALLERA SEPAR1 FIN ;

SORT :   SØRTIR (A) ;

DEPIL :  SI H = 0 ALORS ALLERA EXIT
         SINON DEBUT DEPILER (A, B, Z) ;
         A = A ∨ PREDEC (Z, Y - A - B, X - Δ (A)) ;
         B = B ∨ Z ∨ ISOLES (Y - A - B, X - Δ (A)) ; ALLERA EXAM FIN

EXIT :   FIN procédure couv ;
```


Application à l'exemple précédent :

Les algorithmes précédents ont fourni l'ensemble des e. s. i. m. pour le graphe G :

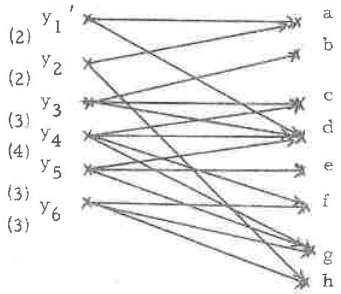
$Y = \{\{a, d\}, \{a, h\}, \{b, c, d\}, \{c, d, f, g\}, \{d, e, g\}, \{f, g, h\}\}$

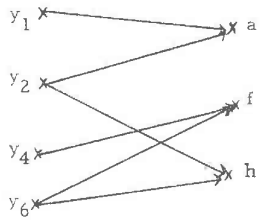
$Y = \{y_i\}$ $i = 1, \ldots 6$

$X = \{a, b, c, d, e, f, g, h\}$ $\quad Y \subset \mathcal{P}(X)$

Soit $H = (Y, X, \Delta)$ défini par

$y \in Y,\ x \in X \qquad y \Delta x \Leftrightarrow x \in y$ (figure 1)





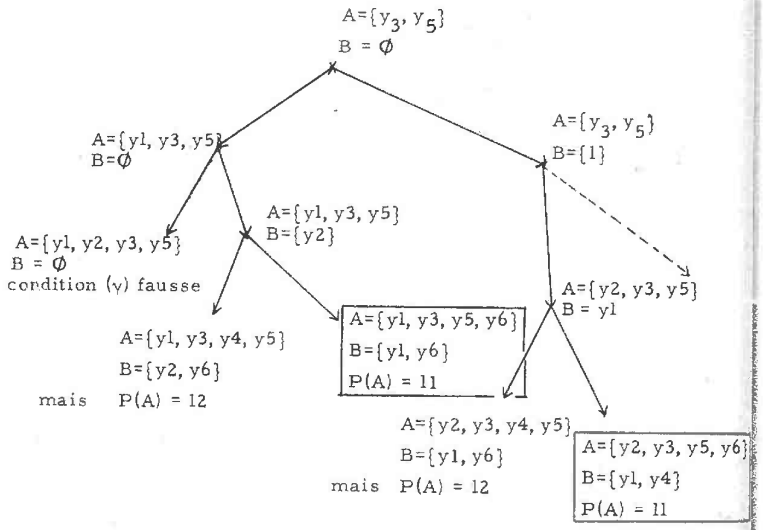
L'ordre est défini sur Y par l'ordre lexicographique (sur les indices).

Nous prendrons $P(y) = \text{card}(y)$ et $P_0 = 11$, $p(y_1) = p(y_2) = 2$;

$p(y_3) = p(y_5) = p(y_6) = 3$; $p(y_4) = 4$.

Après l'initialisation (INIT), le graphe H se réduit au sous-graphe de la figure 2 (puisque b et e admettent comme seul prédécesseur respectivement y_3 et y_5).

Une partie de l'arborescence associée à l'algorithme est la suivante :



En fait, il est très facile de constater que les deux couvertures trouvées sont les deux seules vérifiant la condition sur P_0.

Remarque : - Cet algorithme permet de déterminer la (ou les) couverture (s) de poids minimum du graphe $H = (Y, X, \Delta)$: il suffit d'initialiser P_0 à $+\infty$ au départ (en fait une valeur très grande) et de poser P_0 = poids de la couverture trouvée quand on détermine une couverture de poids inférieur ou égal à P_0.

- On peut aussi trouver toutes les couvertures minimales du graphe H en posant $P_0 = +\infty$.

2.4 Délivrance des diplômes

Pour chaque étudiant et pour l'année en cours figurent dans le fichier des étudiants (historique ou annuel) les diplômes préparés. Nous avons vu qu'il est possible de connaître les éléments auxquels l'étudiant a été reçu. Il est donc possible de délivrer automatiquement les diplômes en appliquant la procédure suivante : pour chaque diplôme préparé (nécessitant p éléments) :

> - si l'étudiant possède les q éléments nécessaires (relation R_1).
> et si l'étudiant possède (p-q) éléments n'appartenant pas aux éléments interdits pour ce diplôme (relation R_3),

alors on peut faire figurer ce diplôme parmi les diplômes obtenus par l'étudiant.

Nous avons déjà remarqué (§ 2. 2) que le partage des diplômes en niveaux pouvait faciliter la gestion : en particulier, parmi les éléments interdits pour un diplôme, ne pourront figurer alors que les éléments du même niveau.

2.5 Travaux divers

Ils ne sont pas essentiels à la gestion des étudiants telle que nous la concevons ici, mais ils sont souvent fort utiles à un secrétariat (dont ils pourront souvent apparaître comme le juste retour d'un travail considérable que ce secrétariat fournit pour la mise à jour du fichier historique).

Ils se présentent généralement sous forme de listes : citons pêle-mêle :

- procès-verbaux d'examens,
- listes électorales,
- liste des étudiants étrangers,
- liste des élèves de l'IPES,
- liste des étudiants salariés, etc...

chaque liste doit posséder ses caractéristiques particulières et les possibilités d'un programme "Editeur" sont alors bien venues. En tout état de cause, il ne s'agit pas de "sortir du papier pour du papier" ; il est nécessaire que chaque édition soit rigoureusement justifiée pour qu'on ne demande

plus à l'avenir de "lister le fichier des étudiants pour vérifier les statis-tiques" comme cela s'est produit.

2.6 Elaboration des statistiques

2.6.1 Types de statistiques et leur utilité :

Nous avons vu (chap. 1) que les deux grands types sont des statistiques sur les effectifs (inscrits, diplômés selon diverses catégorie et sur les flux d'étudiants dans le système d'éducation. Or, nous avons v que ce système peut se représenter comme un multigraphe assez comple les observations portent alors sur l'état des différents sommets du multi graphe et sur celui des arcs. Divers facteurs socio-économiques peuvent expliquer ces états. Des modèles mathématiques ont été conçus ([10]) p servir de support à des analyses de systèmes d'éducation. Ces analyses peuvent conduire à des prévisions d'effectifs, de nombre de professeurs, locaux, de coûts et de ressources, à définir des programmes d'enseigne-ment ... etc, si elles sont combinées à d'autres données sur la populatio et l'économie (voir [10]). Cette planification dépasse largement le cadre de la gestion des étudiants. Cependant, au niveau de l'enseignement supé-rieur dans une académie, on peut s'inspirer de ces méthodes pour parve-nir à des constatations intéressantes. (Mais, il n'est guère possible de penser, dans l'immédiat, à une prise de décision pour les grandes option au niveau académique).

exemple : considérons la chaîne de Markov (du chap. 1) représent l'avancement des étudiants dans un établissement. Nous supposerons que parmi les m états de l'espace des états, t états soient absorbants et m - t états soient non absorbants. (Un état est absorbant s'il n'existe aucune p babilité de le quitter : dans notre système, nous pouvons supposer, par exemple, que ces états absorbants sont les diverses études achevées qui obligent à quitter le système). Nous supposerons en outre que le proces de Markov est stationnaire. Alors, la matrice $P = (P_{ij})$ associée au pro-cessus peut se décomposer en 4 sous-matrices carrées (si l'on a pris soin de placer en tête les états absorbants).

$$P = \begin{array}{c|c} \overset{t}{I} & \overset{p}{O} \\ \hline R & Q \end{array}$$

où I : matrice-unité (t x t)

O : matrice nulle (p x p)

R : $(r_{ij})_{\substack{i = 1, \dots p \\ j = 1, \dots p}}$ telle que r_{ij} est le pourcentage d'étudiants dans l'état i qui quitteront le système après une année en ayant obtenu une instruction j.

Q : $(q_{ij})_{\substack{i = 1, \dots, p \\ j = 1, \dots, p}}$ telle que q_{ij} est le pourcentage d'étudiants dans l'état i qui seront l'année suivante dans l'état j.

Les éléments diagonaux de Q sont les coefficients de redoublement.

Nous avons vu (chap. 1) que, si le processus est stationnaire, l'élément $q_{ij}^{(n)}$ de la matrice Q^n est le pourcentage d'étudiants qui sont actuellement dans l'état i et qui seront n années plus tard dans l'état j.

Nous pouvons alors calculer le pourcentage des élèves qui se trouvent actuellement dans l'état i (non absorbant) et qui seront n années plus tard dans l'un quelconque des états du système (non absorbant) : on peut l'appeler taux de survie-scolaire :

$$q_i^{(n)} = \sum_{j=1}^{p} q_{ij}^{(n)}$$

c'est la somme des éléments de la ligne i de la matrice Q^n.

Définissons sur l'ensemble des applications linéaires (continues) de $\mathbb{R}^n$ dans $\mathbb{R}^n$ (considéré comme espace vectoriel de Banach sur $\mathbb{R}$ pour la norme $\|X\| = \sup_{i=1}^{n} |x_i|$) $\mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ la norme suivante :

Soit $u \in \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ associée à la matrice $A = (a_{ij})$

$$\|u\| = \sup_{i,j} |a_{ij}|$$

Si $\|u\| < 1$, on sait que $1 - u$ est inversible dans $\mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ et $(1 - u)^{-1} = \sum_{n \geq 0} u^n$, (voir par exemple [11]).

Or, Q est une matrice associée à une application linéaire continue de $\mathbb{R}^p$ dans $\mathbb{R}^p$.

Si tous les éléments de Q sont inférieurs à 1 (ce qui est vraisemblable), alors

$\sup_{i,j} |q_{ij}| < 1$ et la matrice $I_p - Q$ est inversible (où I_p est la matrice unité d'ordre p) et

$$V = (I_p - Q)^{-1} = \sum_{n \geq 0} Q^n \qquad \text{(avec } Q^0 = I_p\text{)}.$$

Il est alors facile de montrer les résultats suivants :

$V = (v_{ij})$ pour $i = 1, \ldots, p$ et $j = 1, \ldots, p$.

1) v_{ij} = temps moyen que passeront les étudiants dans l'état j s'ils sont actuellement dans l'état i (i et j non absorbants). En effet, ce temps est égal à la somme des pourcentages d'étudiants qui, partant de i, seront dans j après 0 (si i = j), 1, 2, 3 ans ...

donc $v_{ij} = \delta_{ij} + \sum_{n \geq 1} q_{ij}^{(n)}$ δ_{ij} : symbole de Kronecker

($q_{ij}^{(n)}$ est l'élément (i, j) de Q^n).

2) $\sum_{j=1}^{p} v_{ij}$ = nombre moyen d'années de scolarité restant à effectuer pour un étudiant actuellement dans l'état i (raisonnement analogue).

3) La proportion des étudiants, actuellement dans l'état i (non absorbant) qui seront diplômés avec l'instruction e (état absorbant) n années après est l'élément (i, e) de la matrice

$$Q^{n-1} R$$

4) La proportion des étudiants, actuellement dans l'état i (non absorbant) qui auront achevé les études e (état absorbant) dans n années est $\sum_{k=1}^{n} r_{ie}^{(k)}$ ou $r_{ie}^{(k)}$ est l'élément (i, e) de la matrice R^k.

5) La proportion des élèves actuellement dans l'état i (non absorbant) et qui quitteront le système d'éducation avec l'instruction e (état absorbant) est :

$$\sum_{k=1}^{+\infty} r_{ie}^{(k)}$$

(Si la matrice R est telle que $\sup_{i,j} |r_{ij}| < 1$, il est possible d'inverser $I_t - R$ et de faire un raisonnement analogue à celui fait pour Q).

Remarque : plutôt que de traiter toute l'information du fichier, on peut préférer travailler sur des échantillons plus faciles à manipuler ou sur des "cohortes" : une cohorte est un groupe d'étudiants pris dans le même état à un instant donné et dont on étudie la progression dans le système d'éducation au cours du temps. Cette dernière méthode est évidemment facilitée si l'on dispose d'un fichier historique complet, et c'est souvent de cette façon que l'on pourra dégager les facteurs sociologiques et économiques qui influent sur les coefficients de transfert.

2.6.2. Elaboration des statistiques :

Elles sont généralement demandées sous la forme de tableaux croisés et la méthode informatique pour les obtenir n'est pas différente s'il s'agit de tableaux d'effectifs ou de flux.

Il s'agit de dénombrer des sous-fichiers de F, ensemble des articles du fichier à interroger :

si $F = \bigcup_{i=1}^{n} A_i$, on associe à chaque A_i une propriété $\wp_i$.

Il faut parcourir tous les enregistrements x du fichier et si $\wp_i$ est vraie pour x, alors x appartient à A_i. Pour une étude détaillée des propriétés $\wp_i$ et des sous-fichiers qui leur sont associés, on lira les chapitres 3 et 4.

A partir de ces tableaux, il est intéressant de pouvoir calculer des sommes marginales, des pourcentages, des sommes intermédiaires, des moyennes, des écarts-type etc...

Mais le problème de l'édition n'est pas négligeable car il est nécessaire que ces tableaux soient lisibles, donc accompagnés de libellés appropriés, pour chaque ligne et pour chaque colonne.

Certains constructeurs d'ordinateurs ou sociétés de service ont proposé des systèmes d'interrogation de fichiers : citons I. C. L.

avec "Find-2" [12], X. D. S. avec "Manage" [13], C. D. C. avec "Infol" [14], I. I. I. avec "NLT"... Ces packages sont parfois des systèmes généraux de traitement de fichiers : "Manage" et surtout "Infol" présentent des services de création et mise à jour de fichiers intéressants (quoique pas toujours très réalistes dans l'environnement informatique actuel). Ces systèmes possèdent tous les phases suivantes :

- création d'un "dictionnaire" : description du fichier à interroger (où l'on note d'ailleurs l'absence d'enregistrements structurés),
- sélection d'articles suivant un ou plusieurs critères (questions),
- édition de rapports avec mise en page, totaux intérmédiaires... L'édition de tableaux à double entrée est parfois possible (Find-2 et NLT) dans des cas particuliers.

C'est dans cette optique que l'on a examiné au chapitre 4 la possibilité de construire un langage orienté vers la structure de tableaux statistiques. Dans l'immédiat, nous nous sommes contentés d'un programme général écrit en COBOL (chap. 3).

2.7 Conclusion

Nous avons donné ici les applications de gestion interne que l'on peut attendre immédiatement d'un fichier historique des étudiants et certains modèles et outils qui peuvent contribuer à observer un système d'éducation et son évolution. Il reste que cette observation ne concerne que le système du point de vue des étudiants et que des modèles devront être pensés pour formaliser les autres systèmes (enseignants, administration) et leurs relations dans le cadre très général défini en introduction et, plus particulièrement, à l'intérieur de chaque université. On devrait pouvoir ainsi contrôler l'écart qui peut exister entre les objectifs fixés et ceux réellement atteints afin de modifier le système ou ses entrées (étudiants).

Nous avons déjà comparé le système d'éducation à un système de production. Il s'agirait donc d'effectuer certaines mesures sur ce système afin de pouvoir le réguler, la gestion des étudiants actuelle étant alors comparable à un "suivi de production". Cette comparaison peut sembler dangereuse en raison de l'importance du facteur humain dans cette étude,

mais il semble qu'une théorie des systèmes (voir par exemple J. MELESE [17]) bien comprise peut fournir un cadre conceptuel fondamental pour défricher ce domaine. Les outils qui servent à formaliser les processus administratifs sont à rechercher dans les théories mathématiques (par exemple : théorie des ensembles, graphes, algèbre de Boole ... comme nous l'avons vu tout au long du chapitre 2).

Ceux qui permettront la prévision sont à rechercher en statistiques (ici chaîne de Markov), recherche opérationnelle et simulation. Beaucoup de problèmes posés aux entreprises industrielles se retrouveront ici sous une forme très complexe, du fait de la taille considérable du système d'enseignement et de la prépondérance des facteurs humains.

CHAPITRE 3

REALISATION A COURT TERME (1970 - 1971)

Il est évident que la mise en place d'un système tel qu'il a été décrit au chapitre 2 nécessite un grand effort d'organisation administrative (refonte des procédures d'inscription et des imprimés, examen poussé des structures scolaires, information du personnel), d'analyse et de programmation et sans doute un matériel, un software évolués. Nous avons préféré, dans une première étape, restreindre les objectifs et mettre en œuvre un système de gestion plus modeste pouvant mettre en valeur quelques points particuliers et rendre immédiatement un certain nombre de services. Il est à prévoir que, en visant plus haut, nous aurions pu risquer de figer un système alors que les structures universitaires étaient en pleine évolution.

Le système de gestion que nous allons décrire maintenant est basé sur la gestion d'un fichier des étudiants annuel pour chaque établissement. Ce fichier est fort comparable au fichier qui devra prendre naissance après la mise à jour du fichier-historique ; c'est dire que l'on va retrouver bon nombre des applications dont nous avons parlé au chapitre 2, et qu'il n'y a pas de rupture insurmontable pour le passage d'un système à l'autre. La mise en place ne s'est pas faite sans quelques heurts entre services administratifs et informatique, mais elle a permis de faire prendre conscience aux uns et aux autres des problèmes que posent, dans beaucoup de domaines, une automatisation de procédures administratives.

Le chap. 3 s'intéresse d'abord à l'analyse du problème ; le résultat de l'analyse de cette application est décrite d'abord sous forme de niveaux, puis sous forme de modules, le langage utilisé étant aussi proche que possible de la langue naturelle, du moins dans les premières phases.

Nous rejoignons ici les études menées à Nancy par Monsieur le Professeur LEGRAS et dans le cadre du projet CIVA (Monsieur le Professeur PAIR et Monsieur J. C. DERNIAME [16]) dont on retrouvera les concepts du langage qu'il développe pour la description d'un résultat d'analyse :

expression sous forme de modules (réutilisables) avec tables de décision, instruction "pour chaque ...", etc... Nous nous sommes efforcé de ne pas faire apparaître trop de ruptures entre les procédures administratives et informatiques, considérant qu'elles font partie d'un même processus.

Cette description tente de réunir une approche fonctionnelle et organique des problèmes, avec les compromis inévitables que cela entraîne : par exemple, dans le niveau 3 (§ 2.2), nous avons écrit :

- INITIALISATION
- pour chaque enregistrement de 'inscriptions 3' faire :
  - Si numéro d'inscription = numéro - en - cours et cycle = cycle - en - cours faire RANGEMENT
  - sinon faire
    - TRAITEMENT
    - RANGEMENT
- TRAITEMENT

Ceci est évidemment une approche organique ; la solution fonctionnelle aurait été :

- INITIALISATION
- pour chaque (numéro - d'inscription || cycle) faire
  - RANGEMENTS
  - TRAITEMENT.

En effet, la première forme s'intéresse au support tandis que la 2è s'intéresse à l'entité étudiant (n° || cycle).

L'avantage de la première forme est évidemment la programmat plus facile dans un langage actuel courant (COBOL, PL/1), et une mainte nance aisée et, celui de la 2è,une compréhension plus facile pour le lect

On doit noter que ces considérations contraignantes de support disparaîtront dans le projet CIVA grâce à des services d'acquisition de données et d'édition en même temps que les problèmes d'avancement de fichiers seront réglés par l'instruction "pour chaque ..." associée à des modules de recherche d'enregistrements dans des fichiers (l'avancement d'un fichier conditionnant celui des autres).

Mais, malgré un souci de rigueur, il ne s'agit pas ici de programmation : par exemple, nous n'hésiterons pas à présenter plusieurs calc successifs avec un élément de tableau tab (i) alors qu'en programmation nous aurions d'abord rangé cet élément dans une mémoire de travail.

D'autre part, certains modules seront détaillés, mais nous ne décrirons en général que la logique des traitements, sans descendre à un niveau très fin qui deviendrait vite fastidieux (contrôles élémentaires, éditions, mouvements...).

La meilleure approche, dans une méthode d'analyse, consisterait sans doute en un découpage très fin des actions à accomplir en niveaux et en modules de façon fonctionnelle (sans souci précis de l'ordinateur), puis de regrouper ces modules en programmes, compte-tenu des contraintes technologiques et de la chronologie. Il en est de même pour l'information et le regroupement en fichiers. Ce que nous décrivons ici est en quelque sorte le résultat de ce regroupement .

En fait, le chapitre 3 s'intéresse essentiellement à l'aspect fonctionnel du système : nous arrêterons la description à la rencontre des premiers modules de chaque programme pour faciliter la compréhension de l'ensemble, mais le lecteur trouvera en annexe le détail de chaque module et des nouveaux modules créés.

La deuxième partie du chapitre 3 donnera un aperçu des problèmes posés par la mise en œuvre et fera le point sur l'exploitation du système.

3.1 Gestion des étudiants - Projet à court terme (réalisé en 1970-19
Analyse pour un établissement

niveau 1

1. Inscription annuelle des étudiants.
2. Création du fichier annuel.
3. Mise à jour périodique du fichier annuel.
4. Exploitations "type secrétariat".
5. Exploitations statistiques.

niveau 2

1. Inscription annuelle des étudiants

1.1 Remplissage du formulaire d'inscription par les étudiants.

1.2 Codification dans le secrétariat et vérification visuelle.

1.3 Saisie de l'information.

1.4 Mise à jour du fichier-historique (sur fiches cartonnées).

niveau 3

1.1 Remplissage du formulaire d'inscription

Ce formulaire a été déterminé par le secrétariat de l'établissement concerné, en accord avec l'I. U. C. A., compte tenu du modèle national proposé (pour respecter le contenu minimum) et de l'adaptation nécessaire à nos procédures de gestion. Ce modèle est mis à la disposition de l'étudiant dans le dossier d'inscription annuel. L'étudiant le remplit et le rapporte au secrétariat au moment de son inscription.

L'obligation est donc faite, actuellement, pour chaque étudiant, de fournir à nouveau tous les renseignements sur l'état-civil, les études antérieures et les caractéristiques socio-économiques (même si elles n'ont pas changé).

Dans cette optique, les formulaires d'inscription sont distincts pour les différents niveaux d'études (cycles). Quand un étudiant désire s'inscrire à deux niveaux (ce qui se produit parfois), il doit remplir deux bordereaux. On sait qu'un étudiant s'est inscrit à deux niveaux grâce à la rubrique "Etes-vous inscrit dans un autre cycle ?".

Remarque : La communication éventuelle (pour l'établissement d'un fichier historique, par exemple) entre deux fichiers annuels de deux années distinctes pourra être réalisée grâce au numéro national d'identification.

niveau 3

1.2 Codification de l'information par le secrétariat et vérification visuelle

La codification exige l'usage de notices de codification dont une partie est fournie par la Direction des Enseignements Supérieurs et l'autre établie par le secrétriat en accord avec l'I. U. C. A.

A chaque étudiant est affecté un numéro d'inscription (porté par un tampon-encreur s'incrémentant automatiquement de un pour chaque nouvel inscrit). Ce numéro est important car il caractèrise l'étudiant pour l'année : c'est sur lui que l'on effectue le regroupement de l'information lors de la création du fichier et il permettra une recherche facile dans les formulaires d'inscriptions (classés suivant ce numéro) après une éventuelle erreur.

Quand un étudiant s'inscrit à deux niveaux, on porte à la main ce numéro sur l'un des formulaires remplis, de sorte que un étudiant ne possède qu'un seul numéro.

La vérification consiste en une lecture de la fiche de l'étudiant et un contrôle de la validité des codes (par une personne différente du codifieur) ainsi qu'un contrôle de la validité de l'inscription pour ce qui concerne le droit, au vu des études antérieures, de s'inscrire aux diplômes et examens mentionnés.

A chaque code d'élément (numérique) est associée une lettre de contrôle qui devra être présente et correcte pour que l'information soit considérée comme exacte.

niveau 3

1.3 Saisie de l'information : perforation et vérification

Les formulaires d'inscription, servant de bordereaux de perforation, sont acheminés par fournées au service de perforation (I. U. C. A. ou Société de service). Cette saisie s'effectue actuellement sur cartes perforées, mais devant le volume de cartes important, il faudra envisager une saisie directe sur bande magnétique rapidement. Le bordereau donne lieu à 4 cartes sur lesquelles on porte le numéro d'inscription et un caractère permettant de les distinguer. La perforation peut s'effectuer de deux façons :

- toutes les cartes "à la fois" pour un étudiant
- toutes les cartes d'un même type pour tous les étudiant

L'usage d'une saisie directe devrait, à l'avenir, éviter ce morcellement de support qui entraîne souvent des erreurs.

Les cartes sont contrôlées par une vérificatrice.

Le fichier sur cartes ainsi créé est appelé fichier 'inscriptions'

niveau 3

1.4 Mise à jour du fichier-historique

Dans cette méthode, le fichier-historique est encore sur fiches cartonnées dans le secrétariat. Il est généralement classé par noms.

Pour chaque étudiant ancien, on extrait sa fiche cartonnée, on compare les renseignements qu'elle contient avec ceux de la fiche d'inscription et on les modifie si besoin est. On note soigneusement les diplômes et examens préparés, pour l'année, ainsi que le numéro d'inscription annuel de l'étudiant. On s'assure que l'étudiant possède les diplômes nécessaires aux inscriptions demandées.

Pour chaque étudiant nouveau dans l'établissement, il est nécessaire de créer une fiche cartonnée avec les diplômes précédemment acquis, soigneusement vérifiés grâce à des attestations d'études.

niveau 2

2. Création du fichier annuel.

2.1 Regrouprement de l'information pour chaque étudiant.

2.2 Création et vérification.

2.3 Rangement et copie du fichier.

niveau 3

2.1 Regroupement de l'information pour chaque étudiant

2.1.1. Carte à bande :

* Entrée : fichier 'inscriptions 1'.
* Sortie : fichier 'inscriptions 2'.
* Traitement : recopie.

2.1.2. Tri :

* Entrée : fichier 'inscriptions 2'.
* Sortie : fichier 'inscriptions 3'.
* Traitement : tri par 1. numéro d'inscription croissant
  2. cycle croissant
  3. type de carte.

niveau 3

2.2 Création et vérification

* Entrée : fichier 'inscriptions 3'.

  - tab-nat (250) : table des nationalités
  - tab-emp (30) : table des groupes d'emplois
  - tab-uer (37) : table des U. E. R.
  - lib (31) : table des libellés.

* Sortie : fichier 'établissement'
  fichier 'anomalies'.

* traitement

  2.2.1 faire : INITIALISATION.

  2.2.2 Pour chaque enregistrement de 'inscriptions 3' faire :

    Si numéro d'inscription = numéro en cours et
    cycle = cycle en cours
    faire : RANGEMENT
    sinon faire : TRAITEMENT / RANGEMENT

  2.2.3 faire TRAITEMENT.

* mémoires de travail :
  - numéro en cours :
  - cycle en cours.

Remarque :

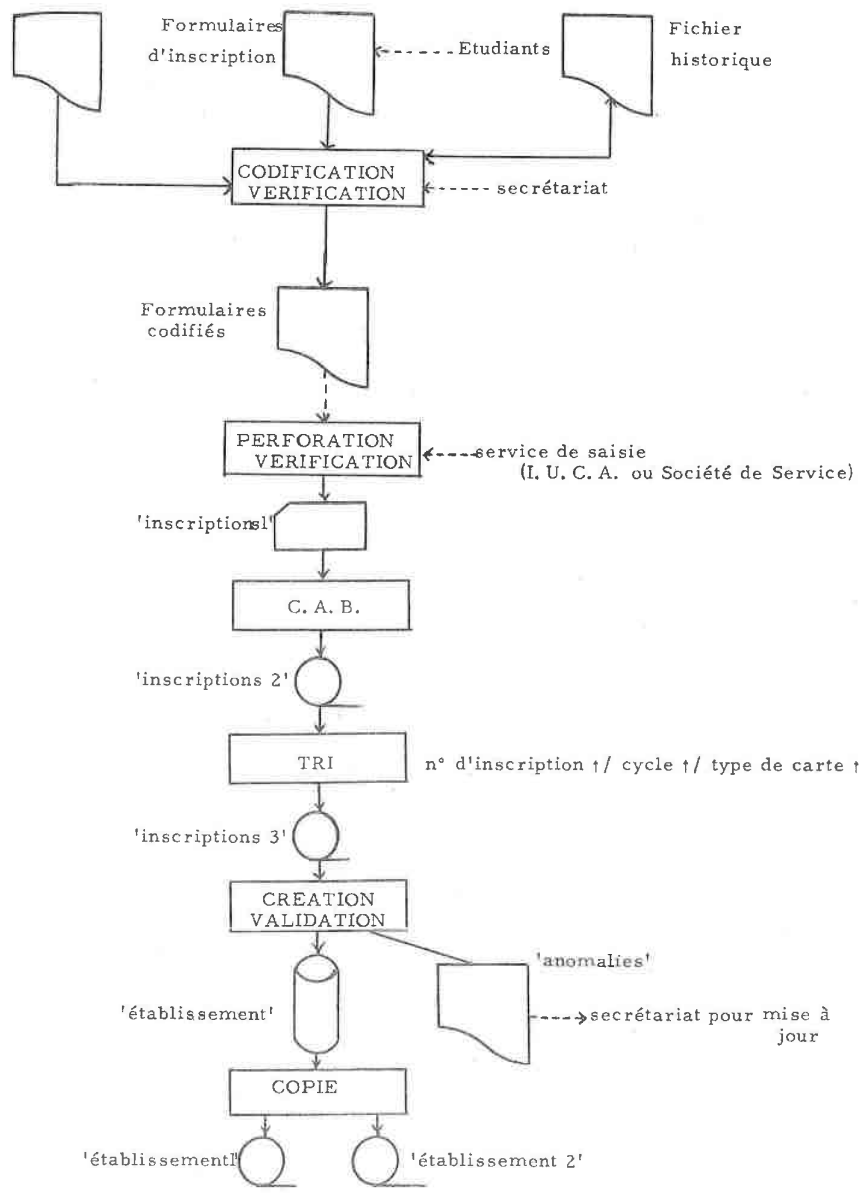
Le détail des modules INITIALISATION
RANGEMENT
TRAITEMENT est donné en annexe 1, ainsi que celui de leurs modules sous-jacents.

niveau 3

2.3 Rangement et Copie de fichier :

* Entrée : fichier 'établissement'.
* Sortie : fichier 'établissement1'.
  fichier 'établissement2'.
* traitement : copies.

Chaîne de création du fichier annuel : (sur C. I. I. 10070) :



niveau 2

3. Mise à jour périodique du fichier-annuel

3.1 Recherche et correction des formulaires dans le secrétariat.

3.2 Nouveaux inscrits et suppressions, corrections spéciales.

3.3 Saisie des modifications.

3.4 Mise à jour du fichier-historique.

3.5 Archivage des bordereaux.

3.6 Mise à jour du fichier 'établissement'.

Remarque : les types admis pour la mise à jour sont :

- adjonction d'un enregistrement (nouvel inscrit).
- modification de l'information concernant une carte
- suppression d'un enregistrement
- modification d'une zone bien déterminée d'un enregistrement caractérisée par :
  - le caractère de départ
  - la longueur de la zone à modifier.

niveau 3

3.1 Recherche et correction (éventuelle) des bordereaux :

* entrée : - fichier 'anomalies'.
  - fiches d'inscriptions.
  - tables de codage.
* sortie : - fichier 'modifications'.
* traitement :

pour chaque ligne du fichier 'anomalies' faire :

- recherche (séquentielle ou dichotomique) du bordereau portant le même numéro d'inscription et le même cycle.
- Si l'erreur n'est pas une erreur de perforation.
  Si on peut la corriger (à l'aide des tables de codage éventuellement) : on barre l'erreur et on écrit au-dessus le nouveau code (au stylo rouge)
  sinon on convoque l'étudiant concerné et on corrige l'erreur de la même façon.
- Indiquer la perforation à effectuer : recopier la mention figurant dans la ligne d''anomalies' :
  'perforer toutes les cartes' ou
  'perforer la carte n°i avec 'M' en colonne 80'.

niveau 3

3.2 Nouveaux inscrits et suppressions : corrections spéciales

* entrée : - fiches d'inscription
  - tables de codage
* sortie : - fichier 'modifications'
* traitement :

- pour chaque nouvel inscrit faire (1-1), (1-2), (1-4) et
  rangement de sa fiche dans 'modifications' avec la mention : 'perforer toutes les cartes'.
- pour chaque suppression faire :
  - recherche du (ou des) bordereau (x) correspondant à l'étudiant considéré.
  - rangement de la (ou des) fiche (s) avec la mention : 'perforer une carte avec 'S' en colonne 80' dans 'modifications'.
- pour chaque correction spéciale faire : (3-1) avec la mention 'correction spéciale'.
  Chacune de ces corrections fait l'objet d'une ligne sur un bordereau de programmation habituel (avec position du caractère de départ de la modification, longueur, information et '*' en colonne 80)

niveau 3

3.3 Saisie des modifications

* entrée : fiches 'modifications'
* sortie : fichier 'modifications 1'.
* traitement :

pour chaque bordereau effectuer l'opération mentionnée :

- perforation de toutes les cartes
ou - perforation d'une carte avec 'M' en colonne 80.
ou - perforation d'une carte avec 'S' en colonne 80.
ou - perforation d'une carte pour corrections spéciales (avec '*' en colonne 80).

niveau 3

3.4 Mise à jour du fichier-historique :

* entrée : - fichier-historique
  - fichier 'modifications'
* sortie : - fichier-historique
* traitement : le même que (1-4) compte-tenu des modifications.

niveau 3

3.5 Archivage des bordereaux d'inscription :

* entrée : - fiches d'inscription
  - fichier 'modifications'
* sortie : - fiches d'inscription archivées
  - fichier 'suppressions'
* traitement :

interclassement des fiches d'inscription et du fichier 'modifications' à l'exclusion des suppressions et corrections spéciales qui vont respectivement dans le fichier 'suppressions'

niveau 3

3.6 Mise à jour du fichier 'établissement'

3.6.1 Regroupement des modifications pour un étudiant.

3.6.2 Copie du fichier 'établissement'.

3.6.3 Mise à jour proprement dite et Vérification du fichier 'établissement'.

3.6.4 Rangement et copie du fichier 'établissement'.

Remarque : nous allons retrouver de nombreuses phases de la Création (2) et de nombreux modules.

niveau 4

3.6.1 Regroupement des modifications pour un étudiant :

3.6.1.2 Carte à bande

* entrée : fichier 'modifications 1'.
* sortie : fichier 'modifications 2'.
* traitement : copie.

3.6.1.2 Tri

* entrée : fichier 'modifications 2'.
* sortie : fichier 'modifications 3'.
* traitement : tri par 1. numéro d'inscription croissant
  2. cycle croissant
  3. type de modification dans l'ordre : '␣'<'M'<'*'<'S'
  4. type de carte croissant.

3.6.2 Copie du fichier 'établissement' :

* entrée : fichier 'établissement 1'
* sortie : fichier 'établissement'
* traitement : copie.

niveau 4

3.6.3 Mise à jour proprement dite et Vérification

* entrée : - fichier 'modification 3'
  - fichier 'établissement'
  - tables de codification.

* sortie : fichier 'anomalies'

* traitement :

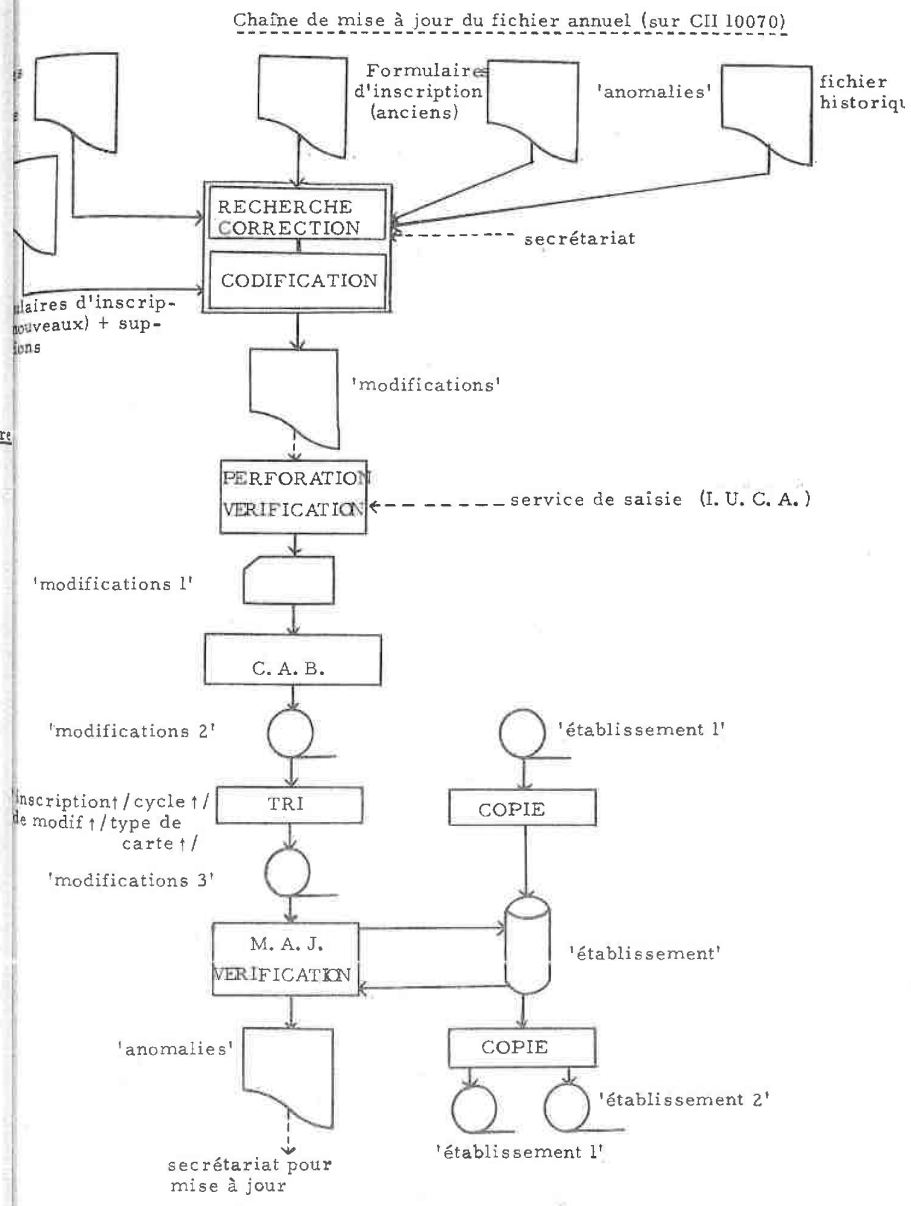
  - faire : INITIALISATION.
  - pour chaque enregistrement de 'modifications 3' faire
    - Si numéro d'inscription = numéro en cours et cycle = cycle en cours faire : TRAIT
    - sinon faire : TRAITEMENT / TRAIT.
  - faire : TRAITEMENT.

* mémoires de travail :

  - numéro en cours
  - cycle en cours.

Remarques : - INITIALISATION et TRAITEMENT ont déjà été détaillés précédemment (annexe 1), le détail du module TRAIT se trouvant en annexe 2, ainsi que ses modules sous-jacents.

- Un deuxième type de mise à jour en accès séquentiel pour les fichiers ancien et mouvement) a été prévu pour les modifications volumineuses.



niveau 2

4. Exploitations "type secrétariat"

4.1 Procédure de "remontée" des résultats d'examens dans le fichier-annuel.

4.2 Construction du graphe des conflits.

4.3 Etablissement de listes diverses.

niveau 3

4.1 Procédure de remontée des résultats d'examens dans le fichier annuel (pour une session).

4.1.1 Création du fichier des 'étudiants/examens'.

4.1.2 Tri du fichier créé.

4.1.3 Création du fichier 'éléments'.

4.1.4 Edition de procès-verbaux d'examens.

4.1.5 Notification des résultats d'examens.

4.1.6 Mise à jour du fichier 'étudiants/examens'.

4.1.7 Mise à jour du fichier-annuel.

4.1.8 Mise à jour du fichier-historique.

niveau 4

4.1.1 Création du fichier 'étudiants/examens'

* Entrée : fichier 'établissement 1'
* Sortie : fichier 'étudiants/examens 1'.
* traitement :

- Pour chaque enregistrement de 'établissement 1' fai
  - pour chaque examen (i) (i = 1, 2, ..., 12) faire :
    - Si examen (i) ≠ (blanc ou 0) faire
      - CREATION (d'un enregistrement de 'étudiants/examen
      - écriture de cet enregistrement.

4.1.2 Tri du fichier 'étudiants/examens'

* entrée : fichier 'étudiants/examens 1'.
* sortie : fichier 'étudiants/examens 2'.
* traitement : tri par - 1 - élément
  2 - nom - prénom.

Remarque : le module CREATION est sensiblement distinct d'un établissement à l'autre car on désire posséder des renseigneme différents. Mais nous n'entrerons pas, ici, dans le détail.

niveau 4

4.1.3 Création du fichier 'éléments'

* entrée : fichier 'éléments 1'
* sortie : fichier 'éléments 2'.
* traitement : C. A. D.

Remarque : C'est la montée de la table des éléments pour l'établissement considéré dans un fichier en accès direct.

niveau 4

4.1.4 Edition des procès-verbaux d'examens

* entrée : fichier 'étudiants/examens 2'
* sortie : procès-verbaux.
* traitement :

- faire INITIALISATION
- pour chaque enregistrement de 'étudiant/examens 2' faire :
  - si examen ≠ examen-en- cours faire
    - EDITION - FIN
    - EDITION - ENTETE
  - faire : EDITION - LIGNE.

Remarque : nous ne détaillerons pas INITIALISATION, EDITION - FIN, EDITION - ENTETE, EDITION - LIGNE qui sont des traitements simples. Il y a édition d'un total de contrôle à la fin de l'édition concernant chaque examen. Chaque procès-verbal comprend au maximum 25 lignes (entête non compris).

niveau 4

4.1.5 Notification des résultats d'examens :

* entrée : procès-verbaux
* sortie : procès-verbaux codifiés.
* traitement : pour chaque examen

pour chaque procès-verbal
pour chaque ligne faire
INSCRIPTION des résultats
CODIFICATION des résultats.

pour chaque examen
pour chaque étudiant non régulièrement inscrit, mais passant tout de même l'examen :
CREATION d'une ligne supplémentaire sur le dernier procès-verbal de l'examen (ou sur un procès-verbal supplémentaire).

Remarque : la codification consiste seulement à inscrire pour chaque étudiant et pour chaque examen un caractère dans une case bien déterminée au bout de la ligne.

niveau 4

4.1.6 Mise à jour du fichier 'étudiants/examens 2'

4.1.6.1 Perforation des résultats d'examens : création de 'résultats 1'.

4.1.6.2 Tri de 'résultats 1' pour donner 'résultats 2'.

4.1.6.3 Mise à jour du fichier 'étudiants/examens 2'.

niveau 5

4.1.6.1 Perforation des résultats d'examens

* entrée : procès-verbaux
* sortie : 'résultats 1'
* traitement : pour chaque procès-verbal perforer une carte.
pour chaque ligne supplémentaire perforer une carte.

Remarque : la saisie des résultats est donc une opération très simple car, sur la même carte, on fait figurer 25 résultats (sauf pour les lignes supplémentaires peu nombreuses).

4.1.6.2 Tri

* entrée : fichier 'résultats 1'
* sortie : fichier 'résultats 2'
* traitement : tri : arguments : 1. numéro d'élément
2. numéro du procès-verbal.

Remarque : les procès-verbaux d'un même examen sont numérotés à partir de 01 en ajoutant 1 à chaque procès-verbal. Les cartes supplémentaires portent le numéro 99 (jamais atteint par un procès-verbal).

niveau 5

4.1.6.3 Mise à jour du fichier 'étudiants/examens'

* entrée : - fichier 'résultats 2'
  - fichier 'étudiants/examens 2'
  - session d'examens, année.
* sortie : fichier 'étudiants/examens 3'.
* traitement :

  - faire : INITIALISATION
  - pour chaque enregistrement de 'résultats 2' faire :
    - TRAIT
  - faire VERIF-NOMBRE (f, m) : vérification du total de contrôle à la fin des procès - verbaux de chaque examen.

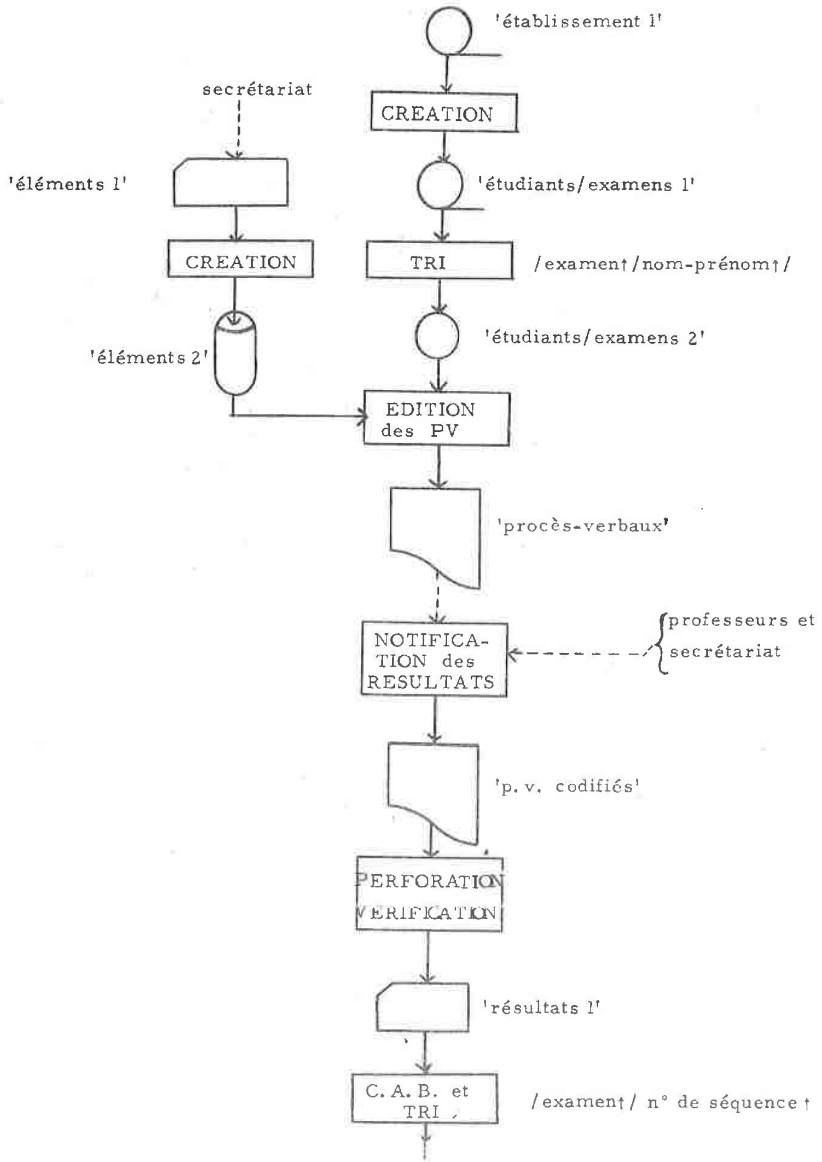
* mémoires de travail :

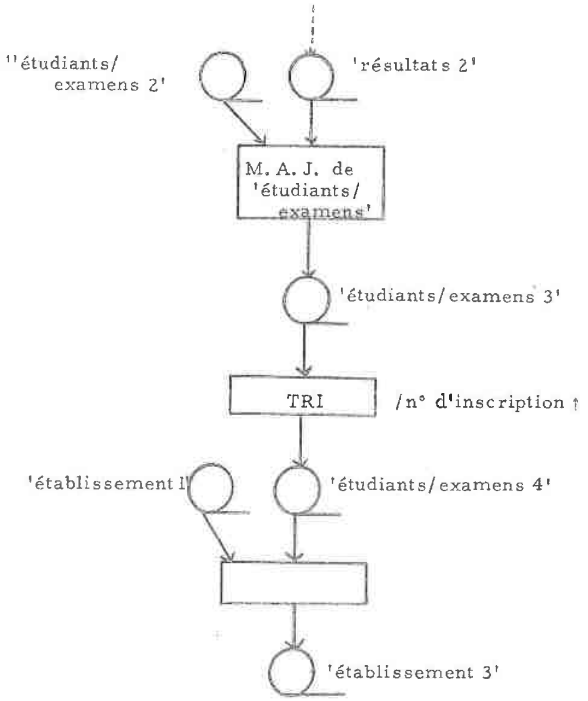
  f : indique que VERIF-NOMBRE a déjà été fait pour l'examen considéré (1) sinon (0)

  m : indique que le résultat trouvé pour la vérification est exact (1 sinon 0).

Remarque : le fichier 'étudiants/examens 3' devient 'étudiants/examens 1' pour la session suivante. Les modules TRAIT et VERIF-NOMBRE sont détaillés en annexe 3, ainsi que leurs modules sous-jacents.

Chaîne de remontée des résultats d'examens dans le fichier annuel (CII





niveau 4

4.1.7. Mise à jour du fichier 'établissement 1'

4.1.7.1 Tri de 'étudiants/examens 3' pour créer 'étudiants/examens 4'

4.1.7.2 Mise à jour du fichier 'établissement 1'.

niveau 5

4.1.7.1 Tri de 'étudiants/examens 3'

* entrée : fichier 'étudiants/examens 3"
* sorties: fichier 'étudiants/examens 4'
* traitement : tri ; argument : numéro d'inscription.

niveau 5

4.1.7.2 Mise à jour du fichier annuel par les résultats d'exame

* entrée : - fichier 'établissement 1'
  - fichier 'étudiants/examens 4'
* sortie : - fichier 'établissement 3'
* traitement :

- faire : INITIALISATION.
- pour chaque enregistrement de 'étudiants/examens 4' non supplémentaire faire :
  - si numéro dans 'étudiants/examens 4' = numéro dans 'établissement 1' faire TRAIT
  - sinon
    - FIN - TRAIT
    - TRAIT
- écrire le dernier enregistrement de 'établissement 4'.

Remarque : le détail des modules TRAIT et FIN - TRAIT est donné en annexe 4 ainsi que celui de leurs modules sous-jacents.

niveau 4

4.1.8 Mise à jour du fichier historique par les méthodes d'examens

* entrée : -procès-verbaux
  - fichier historique
* sortie : - fichier historique

- pour chaque procès-verbal faire
  - pour chaque ligne du procès-verbal faire
    - recherche de la fiche de l'étudiant concerné
    - mise à jour de la fiche par le résultat.

niveau 3

4.2 Construction du graphe des conflits

* entrée : - fichier 'éléments 1'
  - fichier 'établissement 1'
* sortie : - tableau booléen (représentant le graphe des conflits).
* traitement :

1 - Construction de la table tabex.
2 - Construction du tableau tab.
3 - Edition de tab.

* mémoires :
  - tabex (n): table des éléments pris en compte pour les conflits triée par des éléments croissants.
  - tab : tableau booléen.

Remarque : Nous appellerons n le nombre des éléments des tabex.

niveau 4

4.2.1 Construction de la table tabex :

- i = 1
- pour chaque enregistrement de 'éléments 1' pris en compte pour les conflits faire :
  - tabex (i) = élément de éléments 1'
  - i = i + 1.

niveau 4

4.2.2 Construction du tableau tab :

- pour chaque élément tab (i)(i = 1, . . . n) faire tab (i) = 0.
- numéro-en-cours = 0.
- pour chaque enregistrement de 'établissement1' faire :
  - si numéro d'inscription = numéro-en-cours faire
    - RANGEMENT
    - TRAITEMENT
  - sinon faire
    - RANGEMENT.
- faire TRAITEMENT.

* mémoire de travail :
  numéro-en-cours.

Remarques : Il est en effet nécessaire d'effectuer ce traitement puisque un étudiant peut posséder 2 enregistrements.

. On notera l'analogie avec la création du fichier annuel : c'est bien la même logique de traitement.

. Le détail des modules TRAITEMENT et RANGEMENT est donné en annexe 5 ainsi que celui de leurs modules sous-jacents.

niveau 4

4.2.3 Edition du tableau tab :

pour chaque feuille (page) (page = 1, ..., E (n/60)) faire :

- pour chaque tabex (i) (i = 1, ..., n) faire :
  - zone = tabex (i)
  - pour chaque position (k) (k = 1, ..., 60) faire :
    - j = (page-1) x 60 + k
    - si k > n, fin pour
    - faire : REMPL-LIGNE
    - édition de la ligne.

Remarque : - Si le tableau est trop grand pour être contenu dans une page (ce qui est généralement le cas), il doit être édité en "tranches verticales" (de 60 éléments par ligne sur 10070).

- La ligne d'édition est composée de :
  - zone : élément correspondant à la ligne
  - tabed (60) : tableau qui reçoit la partie de la ligne de tab correspondant à i.
- Le module REMPL-LIGNE est détaillé en annexe 5.

niveau 3

4.3 Edition de listes diverses

La procédure d'édition est toujours la même:

1. Création d'un sous-fichier d''établissement' (par sélection et calcul).
2. Tri du sous-fichier.
3. Edition du sous-fichier trié.

Nous ne détaillerons pas plus ces traitements qui ne présentent pas un grand intérêt. Remarquons simplement que c'est généralement de cette manière que procéde un éditeur (Manage, Find ...).

niveau 2

5. Exploitations statistiques.

5. 1 Exploitation statistique sur les inscrits (fichier annuel).

5. 2 Exploitation statistique sur les résultats d'examens (fichier 'étudiants/examens 3').

Commentaire :

En fait, ces statistiques ne sont pas différentes quant à leur élaboration et nous ne distinguons pas entre eux les traitements servant à les obtenir.

Elles ont pu être obtenues dans leur totalité grâce à des programmes en cobol présentant tous la même charpente standard et nous verrons que les données et modules adjoints à ce "squelette" de programme sont véritablement le minimum que l'on puisse donner pour décrire les tableaux statistiques à obtenir. Cette charpente fonctionne à l'aide du système de façon analogue à un "interprète" comme nous allons le montrer.

Pour que ces tableaux apparaissent clairement à l'édition, il est nécessaire d'écrire un entête par tableau, des libellés pour les lignes et pour les colonnes. Il est souvent utile de calculer des sommes marginales (ou intermédiaires).

L'avantage de ce système est qu'il est adaptable sur toutes les machines, puisque sa partie essentielle est en fait un générateur de programme Cobol standard. (générateur existant dans la plupart des compilateurs Cobol : ordres COPY et INCLUDE).

niveau 3

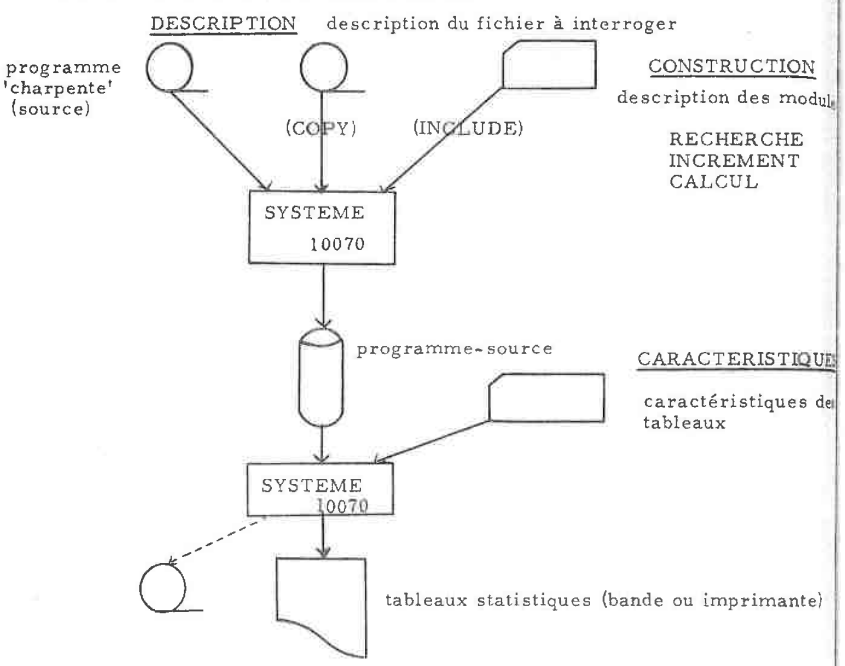
Description du programme standard d'élaboration de statistiques

Nous disposons donc d'une structure de programme Cobol standard (charpente) dans laquelle nous "injectons" des modules complémentaires (DESCRIPTION (du fichier à interroger) et CONSTRUCTION (des tableaux)) pour générer un programme Cobol d'interrogation. Ce dernier est alors compilé et exécuté avec des données sur les tableaux à construire : module CARACTERISTIQUES. Le programme "charpente" peut être mis en bibliothéque (sur bande ou disque) de même que les modules DESCRIPTION et CONSTRUCTION : ceux-ci sont facilement inclus dans le programme "charpente" par les "instructions" COPY et INCLUDE (sur 10070).

Le programme "charpente" comprend un certain nombre de déclarations
- fichier d'édition, tableaux divers, variables élémentaires...
- et un module EDITION des tableaux obtenus. Ce module EDITION met à leur place les divers libellés et calcule automatiquement des sommes marginales sur les lignes et les colonnes, avec la possibilité de ne pas prendre en compte, dans ces sommes marginales, certaines lignes ou colonnes dont le numéro est donné dans CARACTERISTIQUES (et qui correspondent en général à des sommes intermédiaires).

Nous décrirons successivement ces divers modules (annexe 6) et nous donnerons un exemple d'utilisation complète (en annexe 6) portant sur l'élaboration de 2 tableaux à partir d'un fichier d'étudiants.

Programme d'élaboration de statistiques



L'écriture des modules DESCRIPTION, CARACTERISTIQUES et CONSTRUCTION constitue véritablement le minimum d'informations à fournir pour obtenir des tableaux. Elle peut alors s'interpréter comme l'écriture d'un programme, ce qui conduit naturellement aux considérations du chapitre 4. Ce sont les seuls modules à écrire quand on désire des tableaux statistiques. Pour le détail de la réalisation et pour un exemple, se reporter en annexe 6.

On trouvera en annexe 7 une liste des fichiers utilisés et leurs caractéristiques principales, une liste des traitements effectués avec mention de leurs documents d'entrée et de sortie, un formulaire d'inscription, une description des fichiers principaux, une fiche historique et un procès-verbal d'examens.

3.2 Mise en œuvre du système en 1970-1971

Nous ferons d'abord un bref rappel historique de l'exploitation, puis nous parlerons de l'étude préalable et enfin de la réalisation effective.

3.2.1 Historique de l'exploitation :

Quand le ministère de l'Education Nationale décida en 1964 de lancer la gestion des étudiants, l'exploitation en fut confiée à un programmeur disposant d'un petit matériel mécanographique : une trieuse (IBM), une interclasseuse (IBM), une tabulatrice (IBM) et, épisodiquement, une reproductrice (IBM). La gestion fut d'abord effectuée pour les étudiants de sciences et de lettres. Les résultats des travaux se présentaient alors sous forme de tabulations obtenues à partir de fichier sur cartes triés. Mis à part les inconvénients inhérents à la mécanographie (fractionnement de l'information, non intégration et durée des traitements, "bourrages", ...) le système a fonctionné parfaitement, essentiellement pour :

- éditer des listes simples ;
- éditer des statistiques (sous forme de tabulations élémentaires).

Le volume d'informations traité s'est ensuite accru considérablement avec les étudiants en médecine, de manière à occuper le responsable de la gestion de façon continue pendant toute l'année.

Parallèlement, des essais ont été faits sur les ordinateurs CAE 510 et GAMMA 60 (Bull), pour tenter d'utiliser les facilités des bandes magnétiques et des programmes enregistrés écrits en assembleur ou Algol linguistique, avec plus au moins de bonheur. Ce système a fonctionné jusqu'en 1970, date d'arrivée de l'ordinateur CII 10070 à l'I.U.C.A.

3.2.2 Etude préalable : 1969 - 1970

Celle-ci s'est réalisée en plusieurs étapes :

- analyse de l'existant : étude des circuits de documents, des documents et des traitements mécanographiques.
- Définition des premiers objectifs par des visites dans les secrétariats, interviews des responsables...

- Elaboration d'une solution pour les problèmes immédiats, analyse et programmation.
- En mars 1970, l'ordinateur CII 10070 est disponible pour les premiers essais :

  les nôtres sont effectués sur des fichiers fictifs et des jeux d'essai d'une centaine d'enregistrements par un programmeur de l'I. U. C. A. et deux stagiaires de l'IUT d'informatique : les applications testées sont :

  - création et mise à jour du fichier ;
  - quelques problèmes d'édition de statistiques ;
  - édition de listes élémentaires.

- Après ces essais, une réunion générale est tenue en avril avec des représentants de tous les établissements de Nancy pour définir les modalités de mise en place du nouveau système : la définition des bordereaux de saisie de l'information est sans aucun doute la phase la plus délicate, car il s'agit de standardiser les informations prises en compte au maximum pour éviter de fractionner les traitements, mais tout en gardant une certaine souplesse d'adaptation : 30 formulaires distincts ont été établis.
- L'analyse et la programmation du système se poursuivent jusqu'en octobre 1970.

3.2.3 Exploitation en 1970 - 1971 :

Tous les programmes qui sont décrits dans le chapitre "réalisation à court terme" ont été écrits et mis en exploitation en 1970-1971.

- Création et mise à jour :

En fait, les programmes de création et de mise à jour présentés ont été écrits en Cobol et testés à la fin de l'année scolaire 1970-1971 ; les programmes utilisés réellement avaient été écrits en langage symbolique 10070 (META-SYMBOL) et ne comportaient pas tous les tests de contrôle élémentaires prévus maintenant essentiellement pour une question de temps de réalisation. Mais la logique utilisée en est la même, de même que les contrôles fondamentaux. D'autre part, le regroupement de l'information pour un étudiant s'effectuait préalablement sur le numéro d'identification qui

était souvent source d'erreurs. Aussi, dans la version actuelle des programmes, le regroupement s'effectue sur un numéro d'inscription (beaucoup plus facile à manipuler) qui existait de toute façon dans la plupart des établissements. Ce numéro permet en outre une recherche plus facile dans le fichier des bordereaux de perforation classés par numéros d'inscription, alors que la connaissance du nom seulement nécessitait d'abord une recherche de la fiche historique de l'individu sur laquelle était inscrit le numéro de la fiche d'inscription.

Les fichiers d'étudiants de lettres et sciences 1970-1971 ont été créés, et les premières mises à jour effectuées, fin novembre. Les autres l'ont été vers le 15 décembre (médecine, dentaire, pharmacie, droit et sciences économiques). On doit noter que certains fichiers ont été mis à jour (pour les examens de lettres en particulier) jusqu'en mai !

Listes :

Les premières listes d'étudiants par examens ont été éditées pour sciences vers le 1er décembre et, pour lettres, vers Noël. Ces listes comportaient de nombreuses erreurs provenant d'une mauvaise codification des éléments préparés par les étudiants. Ceci a nécessité une nouvelle et profonde mise à jour du fichier et un deuxième tirage des listes. C'est la raison pour laquelle, cette année, nous avons introduit, à la fin de chaque élément, une lettre de contrôle permettant la vérification immédiate. Les listes électorales ont été réalisées pour sciences et lettres avant Noël sans trop de problèmes. D'autres listes ont été réalisées sur tous les fichiers, mais d'importance moindre.

Conflits :

Les tableaux booléens du graphe des conflits ont été édités pour sciences (environ 100 éléments à examiner) et pour lettres (250 éléments, 1 heure d'ordinateur, tableau de 4 m^2) ce qui laisse supposer quelques difficultés pour la mise en œuvre du problème complet (voir chap. 2).

- Gestion des examens :

Celle-ci a été expérimentée en 1970-1971 sur sciences uniquement. Un modèle de procès-verbal pré-imprimé a été établi en accord avec le secrétariat. Les procès-verbaux ont été édités en 3 exemplaires (après une dernière mise à jour concernant les examens préparès) et transmis aux secrétariats qui en ont envoyé un double à chaque professeur. Les professeurs ont reçu ces documents avec plus ou moins de surprise (il reste un grand problème d'information) et n'ont pas toujours su les remplir et pu les renvoyer à temps.

Les résultats des examens ont été recueillis en grande partie, mais encore trop partiellement. La mise à jour du fichier étudiants/examens s'est effectuée correctement et les procès-verbaux de la 2ème session ont pu être édités en juillet. Il reste à recueillir les résultats définitifs pour terminer la mise à jour du fichier annuel.

On peut donc remarquer que, dans ce système, les problèmes essentiels sont dûs à la saisie des données (étudiants ou professeurs) et à la vérification.

- Statistiques :

Deux séries d'édition de statistiques ont été réalisées : la première en décembre destinée à la Direction des Enseignements supérieurs et la 2ème en avril destinée au Service Central des Statistiques.

Nos conceptions sur l'élaboration des tableaux statistiques ont beaucoup évolué en cours d'année :

notre première idée a été de considérer deux types de statistiques correspondant à deux méthodes d'élaboration : la 1ère consistait à construire un tableau à 2 dimensions tel que nous l'effectuons à peu près actuellement ; la 2ème consistait à trier préalablement le fichier (ou un sous-fichier) suivant un certain indicatif pour effectuer ensuite l'édition d'un tableau horizontal unidimensionnel à chaque "rupture" de la valeur de cet indicatif.

Exemple : si l'on désirait ventiler les étudiants par nationalités (environ 150) et par cycle, on triait d'abord le fichier par nationalité et, pour chaque nationalité, on ventilait par cycle dans un tableau à une dimension qui était édité.

Il s'est avéré, à l'usage, que ce deuxième type prenait un temps prohitif (essentiellement à cause du tri), et que, dans tous les cas, on devait se ramener à un tableau du premier type, beaucoup plus rapide à établir.

Nous avons conçu alors une structure de programme standard avec des modules de DESCRIPTION du fichier à interroger et d'EDITION d'un tableau standard en bibliothèque. Pour chaque tableau, on écrivait donc un programme dans lequel on ne se préoccupait pas de ces deux parties mais où figuraient les modules CONSTRUCTION et CARACTERISTIQUES du tableau. La première série de statistiques (10 tableaux distincts) a été réalisée de cette manière.

Nous désirions cependant alléger encore la tâche du programmeur et la charge machine en ayant la possibilité d'éditer facilement plusieurs tableaux après une seule lecture du fichier. Nous avons alors inversé nos conceptions : nous possédons cette fois un programme standard où est inclus le module EDITION et nous fournissons les modules DESCRIPTION, CONSTRUCTION et les CARACTERISTIQUES des tableaux à chaque passage (voir paragraphe 3.1). Il a suffi, pour pouvoir éditer plusieurs tableaux dans le même passage, de considérer un seul tableau à 3 dimensions et de le "surdimensionner".

Cette nouvelle conception nous a grandement facilité la tâche : la deuxième série de tableaux (30 tableaux différents environ, et 80 tableaux à construire sur l'ensemble des fichiers) a été programmée et mise en exploitation en 15 jours par deux personnes. Pour un fichier de 5 000 enregistrements, il faut environ 5 mn pour éditer sur 10070,5 tableaux de 15 x 12 éléments (avec libellés pour chaque ligne et colonne, et sommes marginales). On doit noter que les tableaux édités ainsi peuvent être de tailles différentes.

CHAPITRE 4

ELEMENTS POUR LA CONSTRUCTION D'UN LANGAGE D'OBTENTION DE TABLEAUX STATISTIQUES

4.1 Introduction

Ce langage veut réduire la tâche d'un utilisateur qui désire établir, à partir d'un fichier trié de façon quelconque, une série de tableaux statistiques à une ou deux dimensions. Des calculs sont permis sur les lignes ou colonnes de façon à obtenir des sommes marginales, pourcentages, moyennes etc... Le programmeur n'a plus alors à préciser, par le langage, que :

- la description du fichier en entrée ;
- les caractéristiques indispensables à
  - la construction de ses tableaux : procédure pour l'entrée dans une ligne ou une colonne, condition de construction ;
  - l'édition des tableaux : essentiellement les libellés pour les lignes et les colonnes pour une édition claire.

Il nous a semblé souhaitable d'associer aux procédures d'entrée dans les lignes (ou colonnes) les libellés devant les accompagner, de façon à éviter les comptages fastidieux (le nième libéllé devant accompagner la nième ligne), d'où la création du type "dimension", terme déjà introduit dans le chapitre précédent.

4.1.1 Notion de "dimension" :

Soient :

E : ensemble des articles du fichier à interroger.

L : ensemble des mots finis non vides d'un alphabet A muni de l'opération concaténation $\|$.

$\wp(E)$: ensemble des parties de E.

Définition : une dimension sur E, par rapport à L est une suite d'éléments de $L \times \wp(E)$.

Ceci revient donc à définir une suite de sous-fichiers du fichier en entrée et à associer à chacun un mot de L (ensemble de "libellés"). Un élément de E appartient à un ou plusieurs éléments de cette suite ou à aucun. Ces sous-fichiers ne sont pas nécessairement d'intersection vide pour laisser plus de liberté.

4.1.2 Opérations sur les dimensions

Soient dim1 et dim2 deux dimensions sur E par rapport à L :

$\dim 1 = (\ell_i^1,\ F_i^1)_{i=1,\ldots,m}$ $\quad \ell_i^1 \in L$ et $F_i^1 \subset E$

$\dim 2 = (\ell_j^2,\ F_j^2)_{j=1,\ldots,n}$ $\quad \ell_j^2 \in L$ et $F_j^2 \subset E$

- concaténation : $\|$:

dim1 $\|$ dim2 est la suite

$(\ell_1^1,\ F_1^1),\ \ldots,\ (\ell_i^1,\ F_i^1),\ \ldots,\ (\ell_m^1,\ F_m^1),\ m\ (\ell_1^2,\ F_1^2),\ \ldots,\ (\ell_j^2,\ F_j^2),\ \ldots$

$(\ell_n^2,\ F_n^2)$.

c'est la concaténation habituelle des suites.

- multiplication :

dim1 $*$ dim2 est la suite $(\ell_i^1 \| \ell_j^2,\ F_i^1 \cap F_j^2)_{\substack{i=1,\ldots,m \\ j=1,\ldots,m}}$

l'indice j variant "plus vite" que l'indice i (l'ordre est important).

4.1.3 Tableau statistique :

Il existe deux grandes classes de statistiques sur un fichier : celles qui concernent le nombre d'articles et celles qui s'intéressent aux valeurs annulées de certaines zones numériques dans le fichier.

Nous nous préoccupons ici de ces deux classes :

- dans la première, il s'agit de compter le nombre d'articles de différents sous-fichiers du fichier initial.
- Dans la seconde, nous cumulons les valeurs d'une même zone numérique dans les articles de différents sous-fichiers.

Nous pouvons alors définir ces sous-fichiers par des dimensions, qui permettent, en outre, de préparer l'édition (par les libellés).

Soit $T[i]_{i=1,\ldots,n}$ un tableau statistique et $\dim = (\ell_i,\ F_i)_{i=1,\ldots n}$ sa dimension associée.

- 1er cas : T [i] = card (F_i) quel que soit i = 1, ..., n
- 2ème cas : T [i] = $\sum_{\text{éléments de } F_i}$ (valeur de x) quel que soit i = 1, .., n,

x étant le nom d'un champ numérique pour tous les articles du fichier.

Ces tableaux sont supposés ici à une dimension. Cependant, il est très fréquent que l'on ait à travailler, en statistiques, sur des tableaux à 2 dimensions (à "double entrée").

Ceci nous conduit alors à définir des tableaux à double entrée par deux dimensions. Dans le langage, nous trouverons les expressions :

COMPTER {REEL / ENTIER} { T [dim1, dim2] / T [., dim] / T [dim, .] } pour le 1er cas

SOMMER {REEL / ENTIER} (x) { T [dim1, dim2] / T [., dim] / T [dim, .] } pour le 2è cas

Les expressions T [.,dim] et T [dim, .] sont introduites pour construire des tableaux unidimensionnels respectivement horizontaux et verticaux.

Les tableaux peuvent être entiers ou réels et rien n'interdit d'effectuer des opérations sur les tableaux, comme nous le verrons.

Les dimensions sont obtenues par application des opérations $\|$ et $*$ sur des dimensions élémentaires.

4.1.4 Exemple pratique de dimensions et de tableau statistique :

Soit un fichier d'étudiants pour lequel on désire obtenir le tableau suivant : nombre des étudiants par âge, par sexe, par cycle, par année d'études dans le cycle.

		CYCLE1		CYCLE2		CYCLE3	
		AN1	AN2	AN1	AN2	AN1	AN2
AGE = 16	H						
	F						
AGE = 17	H						
	F						
AGE = 27	H						
	F						
AGE < 16	H						
	F						
AGE > 27	H						
	F						

Définissons alors les dimensions :

1. DIM-AGE, 'AGE=' AGE, POUR AGE = 16 PAS 1 JUSQUA 28 ;
   AGE est le nom d'une zone de l'article du fichier d'entrée.
   DIM-AGE est l'identificateur de la dimension.
   Celle-ci définit 12 sous-fichiers, correspondant aux 12 classes d'âge désignées dans le POUR... A chacun de ces sous-fichiers est associé le libellé 'AGE=' AGE.
   Exemple : un étudiant de 18 ans est compté dans le 3ième sous-fichier auquel est associé le libellé : AGE = 18.

2. DIM-CYCLE, 'CYCLE' CYC, POUR CYC = 1 PAS 1 JUSQUA 3 ;
   Cette dimension est du même type que la précédente.
   Exemple : un étudiant de 2ème cycle est compté dans le 2ème sous-fichier auquel est associé le libellé : CYCLE 2.

3. DIM-AGE-SUP, 'AGE < 16', AGE < 16
   , 'AGE > 27', AGE > 27 ;

c'est un 2ème type de dimension : un article appartient à un sous-fichier si l'expression booléenne correspondant à ce sous-fichier et calculée sur les valeurs de cet article est vraie. Ici, nous avons deux expressions booléennes AGE < 16 et AGE > 28 précédées des libellés associés aux sous-fichiers qu'elles définissent.
Si un étudiant a 30 ans, il est compté dans le 2ème sous-fichier auquel est associé le libellé : AGE > 28.

4. DIM-SEXE, 'H', SEXE = '1'
   , 'F', RESTE ;
   RESTE remplace ici une expression booléenne : elle définit le sous-fichier des articles ne vérifiant aucune des expressions booléennes précédentes.

5. DIM-AN, 'AN1', ANNEE = 1
   , 'AN2', ANNEE > 1 ;
   cette définition est du même type que les deux précédentes.

Remarque : nous verrons que les dimensions 1 et 2 appartiennent au type CALC (calcul) et les 3 suivantes au type BOOL (booléen).

Le tableau statistique ci-dessus peut alors être construit par l'instruction :

SOMMER ENTIER TAB [[DIM-AGE ⓘ DIM-AGE-SUP]
ⓧ DIM-SEXE, DIM-CYCLE ⓧ DIM-AN] ;

4.2 Essai de description du langage

Comme dans tout programme ou package d'interrogation, nous retrouverons ici les phases de :

- description du fichier à interroger
- questions
- calculs et éditions (pour s'en convaincre, voir par exemple [14]).

En fait, les questions sont ici une suite de déclarations de dimensions et de spécifications de constructions de tableaux :

< programme > ::= < dcl-fichier > < suite-dcl-constr >
< calculs-éditions > STOP

< suite-dcl-constr > ::= < suite-dcl-constr > < dcl-constr > | < dcl-constr >.

4.2.1 Description du fichier à interroger :

Cette description commence avec le mot réservé FICHIER suivi d'un identificateur : cet identificateur est le nom affecté au fichier dans le programme et servant d'interface avec le système (par l'intermédiaire du "Data Control Block" (D. C. B.)). Cette description se poursuit sous la forme d'une arborescence (type Cobol ou PL/1) définie à l'aide de niveaux successifs. Une zone peut être redéfinie par une autre (avec un découpage différent) comme dans ces langages.

Les seuls types élémentaires permis sont ici : entier, chaîne de caractères, réel décimal (format "fixed" de PL/1). La composition de ces types élémentaires se fait sous forme de structures ou de tableaux (de structures ou de types élémentaires). Les noms qualifiés (type PL/1) sont permis comme identificateurs de champs pour les articles.

<dcl-fichier> ::= FICHIER < ident >, < structure > ;

< structure > ::= 01 < identif >, < suite-de-2è niv > | 01 < ident >
< type-élém >

par abus de langage pour < n >

< suite-de-n^{è} niv > ::= <suite-de-n^{è}niv>, <élém-de-n^{è}niv> | <élém-de-n^{è}niv>

<élém-de-n^{è}niv> ::= <n> <identifi> <type> | <n> < identifi >, < suite-de(n+1)^{è}niv>
| <n> <identifi> <occur>, < suite de (n+1)^{è} niv>

< identifi > ::= < identif > | < ident > REDEFINES < ident >
< identif > ::= < ident > | FILLER
< occur > ::= (< entier >)
<type-élém > ::= CAR < occur > | ENT < occur > | DEC (< entier >. < entier >)

Remarque : Il est nécessaire de limiter le nombre des niveaux permis. (pour la compilation).

- Rien ne s'oppose à la création de nouveaux types élémentaires : nombres en virgule flottante, chaîne de bits...

4.2.2 Les questions : déclarations de dimensions et construction des tableaux.

Nous avons décidé de mêler les déclarations de dimensions (suites de L x $\mathcal{P}$ (E)) et les instructions de construction de tableaux, pensant qu'il est plus agréable de définir au fur et à mesure les dimensions dont a besoin pour ces constructions (étant entendu que les déclarations de dimensions nécessaires à une construction sont faites avant celle-ci).

< dcl-const > ::= < dcl-dimensions > | < constructions >

4.2.2.1 Déclarations de dimensions :

Les déclarations sont de plusieurs types (nous en avons retenu quatre). Chaque groupe de déclarations de dimensions commence par le mot reservé DIMENSION.

< dcl-dimensions > ::= DIMENSION < liste-dim > ;
< liste-dim > ::= < liste-dim > ; < dcl-dim > | < dcl-dim >
< dcldim > ::= < dcldim 1 > | < dcldim 2 > | < dcldim 3 > | < dcldim 4 >

- dimensions du type "booléen" :

la suite de sous-fichiers du fichier initial est définie par une suite d'expressions booléennes que l'on doit calculer sur les valeurs de chaque article du fichier (une expression booléenne étant associée à chaque sous-fichier). Un groupe de déclarations de ce type commence par le mot clé BOOL.

< dcldim1 > ::= BOOL < listedim1 >
< listedim1 > ::= < listedim1 > ; < dim1 > | < dim1 >
< dim1 > ::= < ident >, < ldim1 >
< ldim1 > ::= < ldim1 >, < ligdim1 > | < ligdim1 >
< ligdim1 > ::= < libellé >, < EB > | < libéllé >, TOTAL |
< libellé >, RESTE | < libellé >,

Une expression booléenne définissant un sous-fichier suit toujours le libellé qui lui est associé. A la place de l'une d'elles, on peut trouver RESTE ou TOTAL ou rien : RESTE définit le sous-fichier des articles ne satisfaisant à aucune des expressions booléennes précédentes (dans la dimension) ; TOTAL définit le sous-ensemble de E (sous-fichier) réunion des sous-ensembles (sous-fichiers) correspondant aux expressions booléennes précédentes, RESTE compris ; s'il n'y a rien après < libellé >, ceci définit le sous-fichier vide. Mais on pourra utiliser les éléments de tableau correspondants pour des calculs. (sommes intermédiaires).

Exemple : BOOL ANN , 'AN1', AN = 1
, 'AN2', AN = 2 OU PR = 'V307'
, 'SOM', TOTAL
, 'RES', RESTE ;

Si le fichier interrogé possède 100 articles, 30 articles étant tels que : AN = 1 et 40 articles tels que : AN = 2 OU PR = 'V307', le sous-fichier défini par TOTAL (et portant le nom SØM) possède 70 articles et le sous-fichier défini par RESTE (de nom RES) en possède 30.

- dimension du type "calcul" :

Soit une suite de nombres réels $(a_i)_{i = 1, \ldots n+1}$ tels que $a_{i+1} - a_i = a_i - a_{i-1}$ pour $i = 2, \ldots, n$.

Soit x le nom d'une zone de chaque article du fichier initial.

Si la condition à vérifier pour qu'un article du fichier appartienne au $i^{ème}$ sous-fichier est que la valeur de x appartienne au $i^{è}$ intervalle $[a_i, a_{i+1}[$, on peut utiliser une description de dimension du type "calcul". Les déclarations du type "calcul" commencent par le mot clé CALC.

< dcldim 2 > ::= CALC < listedim 2 >
< listedim 2 > ::= < listedim 2 > ; < dim 2 > | < dim 2 >
< dim 2 > ::= < dime 2> | < dime 2>, RESTE | < dime 2>, TOTAL |
< dime 2>, RESTE, TOTAL | < dime 2>, TOTAL, RESTE
< dime 2>::= < ident >, < libellé >,
POUR < nomqual > = < nbre > PAS < nbre > JUSQUA <nbre>

< nomqual > ::= < nomqual > . < vari > | < vari >
< vari > ::= < ident > | < ident > (< entier >)

RESTE et TOTAL ont la même signification que précédemment, mais il ne peuvent intervenir qu'à la fin de la définition de la suite "régulière" des sous-fichiers ; les libéllés associés sont RESTE et TOTAL.

Le libéllé associé à chaque sous-fichier se compose du libellé (décrit juste avant POUR...) concatené à la borne inférieure de l'intervalle correspondant.

Exemple :

CALC D-AGE, 'AGE=', POUR AGE = 16 PAS 1 JUSQUA 28, TOTAL ;

Cette dimension définit 12 classes d'âge (16 à 17 non compris jusque 27 à 28 non compris) et un sous-fichier comprenant tous les articles pour lesquels 16 ≤ AGE < 28. A la $2^{è}$ classe par exemple, est associé le libellé : AGE = 17.

- Dimension du type "recherche lexicographique" :

Soit une table de valeurs $(t_i)_{i = 1, \ldots, n}$ et x le nom d'une zone de chaque article du fichier initial.

Si la condition à vérifier pour qu'un article du fichier appartienne au $i^{è}$ sous-fichier est que la valeur de x soit égale à t_i, on peut utiliser une description de dimension du type "recherche lexicographique" (après le mot-clé LEXI). A l'élément t_i de la description de la table est associé le libellé correspondant.

< dcldim 3 > ::= LEXI < listedim 3 >
< listedim 3 > ::= < listedim 3 > ; < dim 3 > | < dim 3 >
< dim 3 > ::= < dime 3 > | < dime 3 >, RESTE | < dime 3 >, TOTAL |
< dime 3 >, RESTE, TOTAL | < dime 3 >, TOTAL, REST
< dime 3 > ::= < ident >, < nomqual > = < table >
< table > ::= < table >, < elem > | < elem >
< elem > ::= < libellé > < constante >
< constante > ::= < nbre > | < libellé >

Les éléments de la table peuvent être des chaînes de caractères, des entiers ou des réels, mais sont nécessairement du même type.

La recherche dans la table pour chaque article peut, par exemple, s'effectuer par une méthode dichotomique, sans pour cela que la table soit donnée triée sur les valeurs de ses éléments (le compilateur effectuant lui-même le tri avant toute recherche). TOTAL et RESTE ont le même effet que précédemment.

Exemple : LEXI NATION, EC.NAT (1) = 'ALLEM' 100,
'BELGI' 104,
'ANGLE' 102,
'DANEM' 110 ;

Si un article est tel que EC. NAT (1) a la valeur 102, cet article appartient au 3^{e} sous-fichier et le libellé correspondant est : ANGLE.

- Dimensions du type "rupture"

Soit x le nom d'une zone des articles du fichier. Soient a et b deux articles du fichier : la relation binaire R : a R b ⇔ x a la même valeur pour a et b définit une partition du fichier à interroger en sous-fichiers. Si l'on s'intéresse à la suite des n sous-fichiers correspondant aux n valeurs de x les plus petites ou les plus grandes, ou peut utiliser une dimension du type "rupture".

< dcldim 4 > ::= RUPT < listedim 4 >
< listedim 4 > ::= < listedim 4 > ; < sens > < dim 4 > | <sens> < dim 4 >
< sens > ::= GRAND | PETIT
< dim 4 > ::= < dime 4 > | < dime 4 >, RESTE | < dime 4 >, TOTAL |
< dime 4 >, RESTE, TOTAL | < dime 4 >, TOTAL, REST
< dime 4 > ::= < ident >, < entier >, < nomqual >

exemple : RUPT PETIT DIM1, 8, AGE, TOTAL ;

Cette dimension DIM1 définit les 8 sous-fichiers correspondant aux 8 classes des âges les plus petits.

TOTAL et RESTE ont la même signification que précédemment. Le libellé associé à chaque sous-fichier est la valeur de la zone consultée.

Remarque : on imposera que les libellés aient même taille pour une dimension donnée, mais ceci n'est pas nécessaire pour deux dimensions distinctes. Pour déterminer la largeur d'une colonne de tableau, on prendra la borne supérieure de la longueur du libellé correspondant et de la dimension de la case du tableau (déterminée par < format > : voir 4. 2. 3).

4. 2. 2. 2 Construction des tableaux :

Il s'agit ici de définir les dimensions avec lesquelles on désire effectuer les entrées dans les lignes ou colonnes des tableaux. Cependant, on peut désirer adjoindre des conditions supplémentaires globales pour la construction d'un tableau : c'est le but de l'instruction SI < EB > ALORS < II > [SINON < II >], l'expression entre crochets étant optionnelle. < EB > est une expression booléenne calculée successivement sur les valeurs de chaque article du fichier, < II > est une instruction ou une liste d'instructions conditionnelles ou inconditionnelles; pour éviter les ambiguïtés dûes aux instructions conditionnelles imbriquées, toute suite d'instructions incluse dans une instruction conditionnelle encadrée par DEBUT et FIN. Si < EB > est vraie pour un article du fichier, cet article participe aux constructions indiquées après ALORS sinon à celles indiquées après SINON (ou à aucune si SINON... n'existe pas).

Ces constructions sont de deux types, comme nous le faisions remarquer précédemment : comptage ou sommation.

Les instructions de constructions commencent avec le mot réservé CONSTRUCTION.

< constructions > ::= CONSTRUCTION < listeconstr > ;
< listeconstr > ::= < listeconstr > ; < constr > | < constr >
< constr > ::= < comptage > | < sommation > | < ICC >
< ICC > ::= SI < EB > ALORS < II > SINON < II > |
SI < EB > ALORS < II >

< II > ::= < comptage > | < sommation > | DEBUT < listeconstr> FIN
< comptage > ::= COMPTER < typetab > < ident > < arbre >
< sommation > ::= SOMMER < typetab > (< nomqual >) < ident > <arbre
< typetab > ::= ENTIER | REEL
< arbre > ::= [< express >, < express >] | [., <express>] | [<express
< express > ::= < express > ⓘ < terme > | < terme >
< terme > ::= < terme > ⊛ < facteur > | < facteur >
< facteur > ::= [< express >] | < ident >

voir exemple 4.1.1 ou 4.3.

4.2.3 Calculs et éditions

Dans les paragraphes précédents, nous avons montré comment construire des tableaux statistiques simples. Il faut maintenant les éditer. On doit remarquer que tout est prêt pour l'édition puisque les dimensions associent à chaque sous-fichier un libellé : la présentation des libéllés doit suivre la structure arborescente des questions pour éviter les redondances inutiles et clarifier la présentation (voir exemple 4.3). L'édition des tableaux est réalisée par l'instruction EDITER (< format >) < liste d'édition >. Ce format permet de préciser la taille de chaque case du tableau contenant un nombre et la place du point décimal. Cette instruction permet aussi l'édition de libellés à n'importe quel moment de l'édition. Le saut en haut de page et le saut de lignes sont autorisés eux aussi par les instructions SAUT-DE-PAGE et SAUT-DE < entier > LIGNES. La hauteur de la case de tableau est standard ainsi que le cadrage des nombres ou libellés dans chaque case.

Pour permettre les calculs habituels en statistique élémentaire (sommes intermédiaires, moyennes, pourcentages... etc) nous avons prévu des opérations arithmétiques sur les tableaux ou leurs éléments. Le résultat d'un calcul est affecté à un tableau, une ligne ou une colonne, ou

à un élément. Les opérations mixtes (réel-entier) sont autorisées.

Une instruction POUR permet de boucler sur une liste d'instructions.

La forme générale des instructions est inspirée de PL/1 (qu'il serait sans doute possible de reprendre presque entièrement, déclarations et procédures comprises.

< calcule-éditions > ::= EDITION < liste d'instructions >
< liste d'instructions > ::= < liste d'instructions > ; < instruction > |
< instruction >
< instruction > ::= < calcul > | < écriture > | < IF >

- Instructions de calcul :

< calcul > ::= < varind > := < EA >
< varind > ::= < ident > < contrôle >
< contrôle > ::= [< liste d'indices >]
< liste d'indices > ::= < indice >, < indice > | < indice >
< indice >, . | ., < indice >
< indice > ::= < ident > | < entier > | < EA1 >

Le résultat d'une expression arithmétique est toujours affecté à une variable contrôlée :

T [I, J] ou T [J] pour un élément de tableau (respectivement à 2 ou 1 dimensions),

T [I, .] et [., J] pour (respectivement) une ligne et une colonne de tableau. Les identificateurs figurant comme indices représentant des entiers et ne sont autorisés que dans des instructions POUR : les expressions arithmétiques < EA1 > sont calculées sur des indices de tableau et des entiers : le résultat est nécessairement entier.

Les expressions arithmétiques < EA > sont calculées sur des éléments de tableaux, lignes ou colonnes, tableaux complets, constantes entières ou réelles avec les mêmes règles qu'en PL/1.

- instructions d'édition :

< écriture > ::= EDITER (< format >) < liste d'édition > |
EDITER < libellé > | SAUT-DE-PAGE |
SAUT-DE < indice > LIGNES

< format > ::= < entier > | < entier > . < entier >

< liste d'édition > ::= < liste édition >, < ident > | < ident >

L'indice utilisé dans l'instruction de saut de ligne permet de préciser le nombre de lignes à passer ; ce n'est un identificateur entier que dans le cas où cette instruction figure dans une boucle POUR.

Remarque : - si la taille d'une ligne de tableau dépasse la longueur de la feuille d'imprimante, le tableau doit être "débité en tranches verticales", avec rappel des libellés de chaque ligne dans chaque tranche.

- instruction POUR :

< IP > ::= POUR < ident > := < entier > PAS < entier > JUSQUA
< entier > FAIRE < instruction 1 >

< instruction 1 > ::= < calcul > | < écriture > | DEBUT
< liste d'instructions > FIN

Il s'agit ici de boucler sur des indices de tableau donc, les bornes et le pas de variation sont entiers. L'instruction à exécuter est élémentaire ou c'est une liste d'instructions élémentaires encadrée par DEBUT et FIN.

Conclusion :

Nous ne détaillerons pas < ident >, < entier >, < nbre >, < libellé >, < EA >, < EB >, < EA1 >, dont les dérivations sont généralement simples et sans intérêt ici où il ne s'agit que d'un essai de description. Signalons simplement qu'il n'y a aucun inconvénient à s'inspirer d'un langage existant (en particulier pour les expressions arithmétiques < EA >, < EA1 > et les expressions booléennes < EB >) : nous avons à plusieurs reprises

mentionné PL/1. Rien ne s'oppose d'ailleurs à une extension du langage présenté ici par des idées empruntées à un langage existant où à une extension d'un langage existant par de nouveaux types et instructions empruntés ici. Nous avons voulu seulement donner ici les concepts minima permettant d'aborder sans trop de peine ces problèmes, souvent fastidieux, d'édition de tableaux statistiques à partir d'un fichier de données.

4.3 Exemple d'utilisation du langage

Construire et éditer ce tableau ventilant les inscriptions d'étudiants pour un établissement scolaire par années d'étude, selon que l'étudiant en est à sa première inscription ou redouble, selon que son admission à ce niveau est normale ou obtenue par équivalence, selon qu'il est français ou étranger. Ce tableau présente de nombreuses sommes intermédiaires.

* ANALYSE DES INSCRIPTIONS EN PHARMACIE *

		NORMAL			EQUIVA			FRANC	ETRAN
		FRAN	ETRA	TOTAL	FRAN	ETRA	TOTAL		
ANNEE1	INSCR1								
	INSCR2								
ANNEE2	INSCR1								
	INSCR2								
ANNEE3	INSCR1								
	INSCR2								
ANNEE4	INSCR1								
	INSCR2								
ANNEE5	INSCR1								
	INSCR2								
AUTRES	INSCR1								
	INSCR2								
TOTAL	INSCR1								
	INSCR2								

Calculer ensuite le pourcentage des français et étrangers dans chaque catégorie (normale ou équivalence) et éditer le tableau correspondant :

Le programme s'écrit ainsi :

```
FICHIER ETUDIANTS,
    01 ENR,
       02 FILLER   CAR (50),
       02 NAT      CAR (3),
       02 FILLER   CAR (22),
       02 ANNEE    ENT (1),
```



```
       02 ACCES    CAR (1);
       02 FILLER   CAR (5),
       02 INSCR    ENT (1),
       02 FILLER   CAR (150) ;
DIMENSION
    CALC D-ANNEE, 'ANNEE', POUR ANNEE = 1 PAS 1
                  JUSQUA 6, RESTE, TOTAL ;
    BOOL D-INSC,  'INSCR1', INSCR = 1,
                  'INSCR2', INSCR > 1 ;
         D-NAT,   'FRANC', NAT = '100',
                  'ETRAN', NAT ≠ '100',
                  'TOTAL', TOTAL ;
         D-ACCES, 'NORMAL', ACCES = 'N',
                  'EQUIVA', RESTE ;
         D-SOM,   'FRANC',,
                  'ETRAN', ;
CONSTRUCTION
    COMPTER REEL REPAR [D-ANNEE ⊛ D-INSC, D-ACCES ⊛
                        D-NAT ⓘ D-SOM] ;
EDITION
    POUR I:= 1 PAS 3 JUSQUA 4 FAIRE
        DEBUT REPAR [., 7] := REPAR [., 7] + REPAR [., I] ;
              REPAR [., 8] := REPAR [., 8] + REPAR [., I+1] ;
        FIN ;
    SAUT-DE-PAGE ;
    EDITER 'ANALYSE DES INSCRIPTIONS EN PHARMACIE' ;
    SAUT-DE 3 LIGNES ;
    EDITER (6) REPAR ;
    POUR I:= 1 PAS 1 JUSQUA 2 FAIRE
        DEBUT REPAR [., I] := REPAR [., I] * 100/REPAR [., 3] ;
              REPAR [., I+3]:= REPAR [., I+3] * 100/REPAR [., 6]
        FIN ;
    SAUT-DE-PAGE ;
    EDITER 'ANALYSE % DES INSCRIPTIONS EN PHARMACIE' ;
    SAUT-DE 3 LIGNES ;
    EDITER (2. 2) REPAR ;
    STOP
```

ANNEXES

ANNEXE 1

Modules de la Création du Fichier Annuel

niveau 4

- INITIALISATION

  * traitement :

    - numéro-en-cours = 0
    - compteur-cartes = 0
    - cycle-en-cours = 0
    - pour chaque élément tab (i) (i=1, 4) faire : tab (i) = 0.

  * mémoires de travail :

    tab (4).

niveau 4

- RANGEMENT

  * traitement :

    Si compteur-cartes < 5 faire :

    - compteur-cartes = compteur-cartes + 1
    - Si compteur-cartes ≤ 4 faire :
      - tab (compteur-cartes) = code-carte
      - Mouvement de l'enregistrement lu à sa place dans l'enregistrement de 'établissement'.

    sinon faire : ERREUR1.

niveau 4

TRAITEMENT

* traitement :

- Si compteur-cartes = 4 faire :
  - Si (tab (i) = i pour i = 1, 2, 3, 4) faire :
    - VALIDATION
    - Si code-validation ≠ 1 écrire l'enregistrement du fichier 'établissement'.
  - sinon faire : ERREUR 2
- sinon faire : ERREUR 3.
- cycle-en-cours = cycle
- numéro-en-cours = numéro d'inscription
- compteur-cartes = 0
- pour chaque élément de tab (i=1,,4) faire : tab (i) = 0
- code-validation = 0.

* mémoire de travail :

- code-validation.

niveau 5

VALIDATION

* traitement

Conditions	Actions	
	numéro d'Erreur	code-validation =
numéro d'inscription non numérique	3	1
numéro-ident non numérique ou sexe ∉ {1, 2} ou mois ∉ {01, ..., 12, 30}	4	1
RECH (Nat, tab-nat, 250) = 0	5	
emploi = 1 et RECH (emp, tab-emp, 20) = 0	6	
RECH (prof, tab-emp, 20) = 0	7	
situation-famille ∉ {1,2, 3, 4}	8	
nombre-enfants non (numérique ou blanc)	9	

pour chaque examen (i) (i = 1, ..., 12) faire :

- Si examen (i) ≠ (blanc ou nul) faire :
  - Si examen (i) numérique faire :
    - CALCUL-CLE (clé)
    - Si clé ≠ clef (i) faire : ERREUR 31; code-validation = 1
  - sinon faire : ERREUR 31; code-validation = 1.

* mémoires de travail :

- clé
- tab-nat, tab-emp sont des tables de codes.

niveau 6

RECH (x, tab, n) résultat d'une recherche dichotomique de x dans tab (n) :

si x est dans tab : RECH = 1

sinon RECH = 0.

Ceci peut s'exprimer au moyen de la fonction récursive :

REC (x, tab, a, b)

```
Si a = b alors
        Si x = tab (b) alors REC = 1
                       sinon REC = 0 ;
    sinon  c = E ( (a + b) / 2 ) ;
           si x > tab (b) alors REC = REC (x, tab, c+1, b)
                          sinon REC = REC (x, tab, a, c) ;
```

en effet :

RECH (x, tab, n) = REC (x, tab, 1, n)

CALCUL-CLE (clé)

* traitement : clé = lettre [Reste de (examen (i) / 22) + 1]

* mémoires de travail :

. lettre (22) : table des lettres sauf : I, O, Q, Z.

niveau 6

ERREUR i

* traitement : Ecrire dans 'anomalies' :

- numéro d'inscription
- nom (si possible)
- numéro d'identification (si possible)
- lib (i).

* mémoires de travail :

- lib (31) : table des libellés d'erreur.

Remarque : dans chaque libellé d'erreur figure l'action à accomplir pour la correction (en plus du type de l'erreur) :

'perforer toutes les cartes'

'perforer la carte n°i avec M en colonne 80'.

ANNEXE 2

Modules de la Mise à Jour du Fichier Annuel

niveau 5

TRAIT

- Si type-modif = '␣' faire RANGEMENT

sinon [Si type-modif ε {S, M, *} faire
[Si RECHERCHE (numéro d'inscription‖cycle)=1
faire MODIF
sinon faire ERREUR 32.
sinon faire ERREUR 33.

Remarque : RANGEMENT a déjà été détaillé précédemment, ainsi que ERREUR i (annexe 1).

niveau 6

RECHERCHE (a) = 1 si l'enregistrement de clé a existe dans 'établissement', 0 sinon.

MODIF

type modif = S	0		
type modif = M		0	
type modif = *			0
MODIF 1	1		
MODIF 2		1	
MODIF 3			1

niveau 7

MODIF 1

- effacer l'enregistrement lu (pendant RECHERCHE)
- faire : ERREUR 34 (en fait pour signaler le bon effacement).

MODIF 2

- suivant le type de l'enregistrement de 'Modifications 3' modification de la partie correspondante de l'enregistrement de 'établissement'.
- faire : VALIDATION
- si code-validation = 0, réécriture de cet enregistrement de 'établissement'.

MODIF 3

- Si dep et long sont "corrects" faire :
  - pour i = 1, 2, ..., long faire.
    table (i+dep-1) = list (i)
  - faire : VALIDATION
  - si code-validation = 0, réécriture de l'enregistrement de 'établissement'

sinon faire : ERREUR 35.

* mémoires de travail:

- table : tableau ayant autant d'éléments (de 1 caractère) que de caractères dans l'enregistrement de 'établissement'.
- list (long) : tableau contenant l'information de modification.

Remarque : le module VALIDATION a déjà été détaillé antérieurement (annexe 1).

ANNEXE 3

Modules de la remontée des résultats d'examens

niveau 6

TRAIT

- CALCUL-CLE (clef) ;
- Si clef ≠ clé faire ERREUR 6 sinon faire :
- Si examen ≠ examen-en-cours faire :
  - m = 0
  - VERIF-NOMBRE (f, m) (pour l'examen précédent)
  - examen-en-cours = examen
  - f = 0 ; Ind = 0 ; k = 0
  - RECHERCHE (k) (recherche du premier enregistrement de étudiants/examens 2 portant le numéro : examen).
  - si k = 0 faire
    - ERREUR 1
    - f = 1
    - fin pour chaque
  - nombre = 0.
- Si séquence = séquence-en-cours faire :
  - séquence-en-cours = séquence-en-cours + 1
  - T-CHAINE (traitement standard d'une carte).

sinon faire

- Si séquence = 99 faire
  - VERIF-NOMBRE (f, m)
  - si m = 0 faire MOUVEMENT.
- sinon faire ERREUR 2.

* mémoires de travail :

Ind : indique que la somme est calculée (1 sinon 0)

k : indique que l'on a trouvé dans 'étudiants/examens2' le premier enregistrement correspondant à examen (1 sinon 0).

nbré : nombre d'étudiants comptés pour cet examen.

séquence-en-cours : n° de séquence de procès-verbal (pour vérification de séquence).

niveau 7

T-CHAINE (traitement d'une carte non supplémentaire)

- pour chaque car (i) (i = 1, 2, ..., 25) faire
  - si car (i) ≠ '*' faire
    - si examen = examen-en-cours, faire :
      - Résultat (session) = car (i)
      - écriture de l'enregistrement de 'étudiants/examens'
      - lecture d'un nouvel enregistrement de 'étudiants/examens 2'
        (si fin faire : F- EX = 1)
      - nbre = nbre + 1.
      - si car (i) ∉ {A, B, C, T, P, R} faire : ERREUR 3
    - sinon faire : ERREUR 4, fin du traitement T-CHAINE.
  - sinon faire : CONST-SOMME, Ind = 1
- si Ind = 0 et car (26) = '*' faire : CONST-SOMME, Ind = 1

* mémoire de travail : F-EX = 1 si fin du fichier 'étudiants/examens 3', 0 sinon.

RECHERCHE (k)

- Si F-Ex = 1 faire k = 0 sinon :
  - pour chaque enregistrement de 'étudiants/examens 2'
  - faire : Si examen = examen-de-étudiants/examens 2'
    faire k = 1
    sinon écrire l'enregistrement de 'étudiants/examens 3'.

niveau 6

VERIF-NOMBRE (f, m)

- Si f ≠ 1
  - Si somme = nbre faire : m = 0
  - sinon faire : m = 1, ERREUR 5
- f = 1

CONST-SOMME

Somme = car (i+1) || car (i + 2) || car (i + 3)
(|| : opération concaténation)

MOUVEMENT

- Création d'un enregistrement supplémentaire de 'étudiants/examens 3'.
- VAL = 0
- faire : VALIDATION (VAL)
- Si VAL = 1 faire :
  écriture de cet enregistrement.

* mémoire de travail : VAL = 1 si l'enregistrement est correct, 0 sinon.

ERREUR i

- écrire examen, lib (i), séquence-en-cours.

* mémoire auxiliaire : lib (6) : tableau des libellés d'erreur.

ANNEXE 4

Modules de la remontée des résultats d'examens

niveau 6

TRAIT

- faire : RECH (i) : recherche de l'examen i coorespondant
- mise à jour de examen (i) dans l'enregistrement de 'établissement 1' par session et résultat.

FIN-TRAIT

- écriture de l'enregistrement de 'établissement 3'
- faire : RECHERCHE.

INITIALISATION

- lecture de 'établissement 1'.

---

niveau 7

RECH (i)

pour chaque examen (k) (k = 1, 2, . . . , 12) faire :

si examen (k) = examen faire : i = k

fin pour chaque

RECHERCHE

Lecture d'un enregistrement de 'établissement 1'.

ANNEXE 5

Modules de la construction du graphe des conflits

niveau 5

RANGEMENT

- pour chaque élément examen (i) (i = 1, 2, . . . , 12)
  - ≠ (0 ou blanc) faire :
    - RECHDIC (p)
    - si p ≠ 0 faire
      - m = m + 1
      - Ind (m) = p.

* mémoires de travail :

- Ind (24) : tableau des indices des éléments préparés (trouvés par recherche dichotomique RECHDIC dans tabex).
- p : indice trouvé après RECHDIC ou 0 si examen (i) n'appartient pas à tabex.
- m : nombre d'éléments dans le tableau Ind.

RECHDIC (p)

peut s'exprimer par RECl (examen (i), tabex, 1, n) dérivée de REC (x, tab, a, b) :

Si a = b alors
- si x = tab (b) alors RECl = b
  - sinon RECl = 0 ;

sinon
- c = E ($\frac{a + b}{2}$)
- si x > tab (b) alors RECl = RECl (x, tab, c+1, b)
  - sinon RECl = RECl (x, tab, a, c) ;

niveau 5

TRAITEMENT

Deux traitements selon que tab est défini comme un tableau à 2 dimensions (tab 2) ou à une seule (tab1) :

1. Tableau à deux dimensions tab 2 (n, n) :

- Si m ≠ (0 ou 1) faire :
  - pour chaque élément Ind (i) (i = 1, ..., m) faire :
    - pour j = i + 1, ..., m faire :
      - Si Ind (i) < Ind (j) faire : tab2 (Ind(i), Ind(j)) = 1
      - sinon faire : tab2 (Ind (j), Ind (i)) = 1.
- m = 0.

Commentaire : On ne calcule que la partie du tableau figurant sous la diagonale principale (non comprise). L'autre partie est obtenue par symétrie avant l'édition si on le désire.

niveau 5

TRAITEMENT (suite)

2. Tableau à une dimension tab 1 (n(n-1)/2)

- Si m ≠ (0 ou 1) faire :
  - pour chaque élément Ind (i) (i = 1, ..., m) faire :
    - pour j = i + 1, ..., m faire :
      - tab1 ((Ind (i)-1) (Ind (i)-2)/2 + Ind (j)) = 1
- m = 0.

Commentaire : Le tableau tab1 comporte n (n-1)/2 éléments (ceux qui sont situés sous la diagonale principale).

Au couple (p, q) d'éléments, on fait correspondre l'élément de tableau tab1 ((p-1) (p-2)/2+ q).

niveau 5

REMPL-LIGNE

tableau à 2 dimensions :

- si i < j faire : tabed (k) = tab2 (i, j)
- sinon faire : tabed (k) = tab2 (j, i).

tableau à une dimension :

- si i = j faire : tabed (k) = 0
- sinon faire : tabed (k) = tab1 ((i-1) (i-2)/2 + j).

ANNEXE 6

Modules du programme d'édition de statistiques et exemple

1. DESCRIPTION (du fichier à interroger) ; à fournir par le programmeur)

Cette description est écrite en Cobol standard et rangée dans un fichier sur bande ou disque. Elle est appelée à figurer dans le programme-source d'interrogation par un ordre COPY.

2. CARACTERISTIQUES (des tableaux) ; à fournir comme données du programme compilé

Ce module fournit au programme compilé un certain nombre de données utiles sur les tableaux (par l'intermédiaire d'un fichier sur cartes):

- nombre des tableaux à construire pour ce passage : dans les 2 premières colonnes de la première carte.
- pour chaque tableau, on fournit deux types de cartes :
  - la première comporte :
    - col. 1 - 2 : nombre de lignes du tableau
    - col. 4 - 5 : nombre de colonnes du tableau
    - col. 7 - 56 : entête général
    - col. 58 - 64 : libellé figurant dans la première case en haut à gauche
    - col. 65 à 72 : numéros (4 au maximum sur 2 caractères) des colonnes ne figurant pas dans les sommes horizontales marginales.
    - col. 73 à 80 : numéros des lignes ne figurant pas dans les sommes verticales marginales.
  - les cartes suivantes donnent les libellés figurant pour chaque ligne et pour chaque colonne.

niveau 4

3. CONSTRUCTION (des tableaux) ; à fournir par le programmeur

- Pour chaque enregistrement du fichier à interroger faire :
  - pour chaque tableau tab faire :
    - RECHERCHE (du numéro I de la ligne et du numéro J de la colonne correspondant à l'enregistrement traité).
    - INCREMENT (incrémentation de l'élément (I, J) du tableau).
- Pour chaque tableau tab faire CALCUL (tab).

Commentaire : Chaque tableau est défini par deux "dimensions", une pour les lignes, l'autre pour les colonnes.

Une dimension est une procédure permettant d'associer à un enregistrement un entier positif qui sera en général à un indice de ligne ou de colonne d'un tableau ; une dimension permet donc de créer des sous-fichiers (ici : d'intersection vide) du fichier à interroger et d'associer à chacun un numéro.

Ce module CONSTRUCTION, écrit en Cobol et rangé sur bande ou disque, est inclus dans la "charpente" avant compilation au moyen de l'ordre INCLUDE (CII - 10070).

niveau 5

RECHERCHE

Elle consiste à déterminer les numéros de ligne et de colonne correspondant à l'enregistrement traité ; elle s'effectue de deux façons principales :

- par évaluation d'expressions booléennes successives b_k ; pour les lignes par exemple

  (k = 1, 2, . . . , nombre de lignes) :

  si b_k (enregistrement) = vrai alors I = k.

  (Ces expressions peuvent être imbriquées)

- par recherche d'un champ c de l'enregistrement dans une table de code table (n), par exemple par dichotomie :

  pour les lignes :

  I = RECl (c, table, 1, n).

Il existe bien d'autres manières d'associer à une enregistrement un entier positif ou nul, mais celles-ci sont les deux plus courantes dans nos applications. On pourrait, par exemple, définir une partition du fichier à interroger grâce à la "rupture" de la valeur d'un champ de l'enregistrement :

soit $F = \{x_1, \ldots, x_n\}$ l'ensemble des enregistrements du fichier, et c_i un champ de x_i :

si $c_{i+1} = c_i$ $x_{i+1} \in$ classe (x_i)

sinon $x_{i+1} \in$ nouvelle classe : classe (x_{i+1}).

Cette méthode est rencontrée dans les problèmes de tabulations qu'un programme éditeur est capable de résoudre. (voir chap. 4).

niveau 5

INCREMENT

A chaque enregistrement et à chaque tableau tab sont associés deux indices I et J (correspondant à deux dimensions). Le traitement peut alors s'écrire, en général :

si b = 1 et I ≠ 0 et J ≠ 0 faire tab (I, J) = tab (I, J) + 1

où b est une expression booléenne calculée à partir des valeurs des champs de l'enregistrement en cours de traitement.

Remarque : On peut, bien sûr, imaginer des procédures plus complexes d'entrée dans un tableau : par exemple combiner 3 dimensions, pour se ramener au cas de 2 dimensions (tableau à double entrée). Ces méthodes sont développées au chapitre 4 (§ opérations sur les dimensions).

CALCUL (tab)

Ce module permet d'effectuer des calculs sur les éléments de tab avant édition : par exemple sommes intermédiaires, pourcentages...

4. EDITION (des tableaux) ; dans la charpente (module standard)

pour chaque tableau tab faire : EDITION (tab).

Ce module EDITION est le même pour tous les tableaux ; il appartient donc à la "charpente". Il permet d'éditer, pour chaque tableau :

- l'entête général
- des "cases" délimitées par des lignes et colonnes d'astérisques dans lesquelles il range :
  - les libellés des lignes ou des colonnes,
  - les éléments de tab,
  - les sommes marginales,
  - un total général (en bas à droite).

Exemple :

A l'aide d'un fichier d'étudiants, on désire éditer 2 tableaux (avec leurs sommes marginales) :

1. Tableau ventilant par cycles (1er, 2ème, 3ème et préparations diverses), et par sexes les nationalités des étrangers européens.

2. Tableau ventilant par cycles et par sexes les dates de naissance (de 1954 à 1942 et < 1942) de tous les étudiants.

Module DESCRIPTION du fichier à interroger
(dans le fichier F-ETAT)

```
01 ENR.
   02 FILLER PIC XX.
   02 CYC    PIC 9.
   02 FILLER PIC X(11).
   02 SEX    PIC X.
   02 ANN    PIC 99.
   02 FILLER PIC X(41).
   02 NAT    PIC XXX.
   02 FILLER PIC X(161).
```

Module CONSTRUCTION (dans le fichier F-TEST)

```
TEST. EXIT.
TESTV1.
    MOVE CYC TO J.
    MULTIPLY DEUX BY J.
    IF SEX = '1' SUBTRACT UN FROM J.
TESTH1.
    MOVE UN TO A.
    MOVE 33 TO B.
RECH.
    IF A NAT = B  ADD A B GIVING C
                  DIVIDE 2 INTO C
          IF  NAT > LIB (1, C)
                  MOVE C TO A
                  ADD UN A GO TO RECH
          ELSE    MOVE C TO B
                  GO TO RECH
    ELSE IF  NAT = LIB (1, A)
                  MOVE A TO I
          ELSE GO TO TESTH2.
INC1.
    ADD UN TAB (1, I, J).
TESTH2.
    IF ANN < 42 OR > 54 MOVE 14 TO I
    ELSE  SUBTRACT ANN FROM 55 GIVING I.
INC2.
    ADD UN TAB (2, I, J).
    GO TO LECTURE.
```

RECHERCHE (par calcul sur CYC (cycle)) de la colonne J de chaque tableau

RECHERCHE (dichotomique dans la table des nationalités) de la ligne I du 1er tableau.

INCREMENT (pour le 1er tableau)

RECHERCHE (par calcul sur ANN) de la ligne I du 2ème tableau

INCREMENT (pour le 2ème tableau)

```
C21 08/07  C8881 SOURCE PROGRAM AND DIAGNOSTIC LISTING  (après GENERATION)

00000          C8881 LS,39
00001                IDENTIFICATION DIVISION.
00002                PROGRAM-ID. STATS.
00003                ENVIRONMENT DIVISION.
00004                CONFIGURATION SECTION.
00005                SOURCE-COMPUTER. CII-10070.
00006                OBJECT-COMPUTER. CII-10070.
00007                INPUT-OUTPUT SECTION.
00008                FILE-CONTROL.
00009                    SELECT F-IMPR ASSIGN TO PRINTER.
00010                    SELECT F-ETAB ASSIGN TO MAGNETIC-TAPE.
00011                    SELECT F-CART ASSIGN TO CARD-READER.
00012                DATA DIVISION.
00013                FILE SECTION.
00014                FD F-ETAB
00015                    LABEL RECORD STANDARD
00016                    DATA RECORD ENR.
00017                01 ENR COPY F-ETAB.
00017.00001          01 ENR.
00017.00002              02 FILLER PIC XX.
00017.00003              02 SYC    PIC 9.
00017.00004              02 FILLER PIC X(111).
00017.00005              02 SEX    PIC X.
00017.00006              02 ANN    PIC 99.
00017.00007              02 FILLER PIC X(611).
00017.00008              02 NAT    PIC XXX.
00017.00009              02 FILLER PIC X(116)1.
00018                FD F-IMPR
00019                    LABEL RECORD OMITTED
00020                    DATA RECORD ENR-IMPR.
00021                01 ENR-IMPR.
00022                    02 CAR PIC X.
00023                    02 FILLER PIC XX.
00024                    02 ASTE PIC X.
00025                    02 ENR-IMPR1.
00026                        03 FLEM OCCURS 14.
00027                        04 ESPACE PIC X(7).
00028                        04 ESPACE-REDEFINES ESPACE PIC Z(6)9.
00029                        04 ASTER PIC X.
00030                FD F-CART
00031                    LABEL RECORD OMITTED
00032                    DATA RECORDS CARTE1 CARTE2 CARTE3.
00033                01 CARTE1.
00034                    02 NB PIC 9.
00035                    02 FILLER PIC X(79).
00036                01 CARTE2.
00037                    02 AL PIC 99.
00038                    02 FILLER PIC X.
00039                    02 NL PIC 99.
00040                    02 FILLER PIC X.
00041                    02 LIBE PIC X(50).
00042                    02 FILLER PIC X.
00043                    02 BASE PIC X(7).
00044                    02 INTERM.
00045                      03 INTERMED OCCURS 8.
00046                        04 FILLER PIC XX.
00047                      03 INT1 REDEFINES INTERM OCCURS 8.
00048                        03 FILLER PIC 99.
00049                01 CARTE3.
00050                    02 LI PIC X(7).
00051                    02 FILLER PIC X(73).
00052                WORKING-STORAGE SECTION.
00053                77  V PIC 99.
00054                77  W PIC 99.
00055                77  Z  COMP.
00056                77  TOT   COMP VALUE 0.
00057                77  J1    COMP.
00058                77  NC2   COMP.
00059                77  UN COMP VALUE 1.
00060                77  DEUX COMP VALUE 2.
00061                77  SOM COMP.
00062                77  I   COMP.
00063                77  J   COMP.
00064                77  NC  COMP.
00065                77  NC  COMP.
00066                77  X   COMP.
00067                77  NB  COMP.
00068                77  TRUE COMP.
00069                77 A COMP.
```

les MODULES-DESCRIPTION et CONSTRUCTION sont les seuls à fournir pour compléter la charpente.

DESCRIPTION

```
00070                  77 B COMP.
00071                  77 C COMP.
00072                  01 FILLER.
00073                      02 NVA OCCURS 6.
00074                         03 N COMP.
00075                         03 A COMP.
00076                  01 FILLER.
00077                      02 NVAP OCCURS 6.
00078                         03 ENT PIC X(50).
00079                         03 CASE1 PIC X(7).
00080                         03 LIA   PIC X(7) OCCURS 41.
00081                         03 INTE.
00082                           04 INT OCCURS 8.
00083                             05 FILLER PIC 99.
00084                  01 FILLER.
00085                      02 T1 OCCURS 6.
00086                         03 T2 OCCURS 33.
00087                           04 TAB COMP OCCURS 8.
00088                  01 FILLER.
00089                      02 OSMV COMP OCCURS 8.
00090                  PROCEDURE DIVISION.
00091                  DEBUT.
00092                      OPEN OUTPUT F-IMPR INPUT F-ETAB F-CART.
00093                      READ F-CART AT END GO TO FR.
00094                      MOVE NB TO NA.
00095                      PERFORM INIT THRU FIN-INIT VARYING K FROM UN BY UN
00096                        UNTIL K > NB.
00097                  LECTURE.
00098                      READ F-ETAB AT END GO TO PRE-EDIT.
00099                  TEST. INCLUDE F-TEST.
00099.00001            TEST. EXIT.
00099.00002            TEST1.
00099.00003                MOVE CYC TO J.
00099.00004                MULTIPLY DEUX BY J.
00099.00005                IF SEX = 'I' SUBTRACT UN FROM J.
00099.00006            TEST11.
00099.00007                MOVE UN TO A.
00099.00008                MOVE 33 TO B.
00099.00009            RECH.
00099.00010                IF A NOT = B  ADD A B GIVING C
00099.00011                              DIVIDE 2 INTO C
00099.00012                   IF  NAT > LIA (1, C)
00099.00013                               MOVE C TO A
00099.00014                               ADD UN A GO TO RECH
00099.00015                   ELSE        MOVE C TO B
00099.00016                               GO TO RECH
00099.00017                ELSE IF  NAT = LIA (1, A)
00099.00018                               MOVE A TO I
00099.00019                     ELSE GO TO TEST2.
00099.00020            INC1.
00099.00021                ADD UN TAB (1, I, J).
00099.00022            TEST2.
00099.00023                IF ANN < 62 OR > 66 MOVE 14 TO I
00099.00024                ELSE  SUBTRACT ANN FROM 66 GIVING I.
00099.00025            INC2.
00099.00026                ADD UN TAB (2, I, J).
00099.00027                GO TO LECTURE.
00099.00028            PRE-EDIT. EXIT.
00100                  EDITION.
00101                      PERFORM ED0 THRU ED5 VARYING K FROM UN BY UN UNTIL K > NB.
00102                  FR. CLOSE F-CART F-IMPR F-ETAB.
00103                      STOP RUN.
00104                  ED0.
00105                      MOVE N (K) TO NC.
00106                      MOVE N (K) TO NC. ADD DEUX NC GIVING NC2.
00107                      MOVE SPACES TO ENR-IMPR.
00108                      MOVE '1' TO CAR.
00109                      MOVE ENT (K) TO ENR-IMPR1.
00110                      WRITE ENR-IMPR.
00111                      MOVE SPACES TO ENR-IMPR.
00112                      WRITE ENR-IMPR AFTER ADVANCING 3 LINES.
00113                      MOVE SPACES TO CAR.
00114
00115                  ED1.
00116                      PERFORM ED16 VARYING J FROM UN BY UN UNTIL J > NC2.
00117                      MOVE '0' TO ASTE.
00118                      WRITE ENR-IMPR.
00119                  ED2.
00120                      PERFORM ED12 VARYING J FROM UN BY UN UNTIL J > NC2.
00121                      WRITE ENR-IMPR.
```

CONSTRUCTION

```
00128                 ED4.
00129                     PERFORM ED2.
00130                     PERFORM ED1.
00131                 ED5.
00132                     PERFORM ED10 THRU F-ED10 VARYING I FROM UN BY UN
00133                         UNTIL I > NC.
00134                 ED6. PERFORM ED2.
00135                     MOVE 'SOMV' TO ESPACE (UN).
00136                     PERFORM ED11 VARYING J FROM UN BY UN UNTIL J > NC.
00137                     MOVE TOT TO ESPACA (NC2)
00138                     MOVE 0 TO TOT.
00139                     WRITE ENR-IMPR.
00140                     PERFORM ED2.
00141                     PERFORM ED1.
00142                 ED7. EXIT.
00143                 INIT.
00144                     READ F-CART AT END GO TO FR.
00145                     MOVE ML TO M (K) MC.
00146                     MOVE NL TO N (K) NC.
00147                     MOVE LIRE TO ENT (K).
00148                     MOVE CASE TO CASE1 (K).
00149                     MOVE UN TO Q.
00150                 RETOUR.
00151                     IF INTERMED (Q) = SPACES MOVE Q TO INT1 (Q).
00152                     ADD UN Q.
00153                     IF Q < 9 GO TO RETOUR.
00154                     MOVE INTERM TO INTE (K).
00155                     PERFORM IN1 VARYING I FROM UN BY UN UNTIL I > MC
00156                                 AFTER J FROM UN BY UN UNTIL J > NC.
00157                     ADD NC TO MC.
00158                     PERFORM IN2 VARYING I FROM UN BY UN UNTIL I > MC.
00159                 FIN-INIT. EXIT.
00160                 IN1.
00161                     MOVE 0 TO TAB (K, I, J).
00162                 IN2.
00163                     READ F-CART AT END GO TO FR.
00164                     MOVE LI TO LIB (K, I).
00165                 ED10.
00166                     MOVE ALL '=' TO ELEM (J).
00167                 ED12.
00168                     MOVE SPACES TO ESPACE (J).
00169                 ED9.
00170                     ADD J MC GIVING J1.
00171                         SUBTRACT UN FROM J1.
00172                     MOVE LIB (K, J1) TO ESPACE (J).
00173                 ED13.
00174                     MOVE 0 TO SOMV (J).
00175                 ED10.
00176                     PERFORM ED2.
00177                     MOVE LIB (K, I) TO ESPACE (UN).
00178                     MOVE 0 TO SOM.
00179                     PERFORM ED8 THRU 77 VARYING J FROM UN BY UN UNTIL J > NC.
00180                     MOVE SOM TO ESPACA (NC2).
00181                     WRITE ENR-IMPR.
00182                     PERFORM ED2.
00183                     PERFORM ED1.
00184                 F-ED10. EXIT.
00185                 ED8.
00186                     ADD UN J GIVING J1.
00187                     MOVE TAB (K, I, J1) TO TRUC ESPACA (J1).
00188                     MOVE J TO W MOVE I TO V.
00189                     IF = = INT (K, 1) OR INT (K, 2) OR INT (K, 3) OR INT (K, 5)
00190                         GO TO YY.
00191                     ADD TRUC SOM.
00192                 YY. IF V = INT (K, 5) OR INT (K, 6) OR INT (K, 7) OR INT (K, 8)
00193                         GO TO 77.
00194                     ADD TRUC SOMV (J).
00195                     IF = = INT (K, 1) OR INT (K, 2) OR INT (K, 3) OR INT (K, 5)
00196                         GO TO 77.
00197                     ADD TRUC TOT.
00198                 ZZ.  EXIT.
00199                 ED11.
00200                     ADD UN J GIVING J1.
00201                     MOVE SOMV (J) TO ESPACA (J1) .

00 NUMBER OF DIAGNOSTIC MESSAGES
```

Module CARACTERISTIQUES des tableaux
(données du programme d'interrogation)

```
2
33,0R, «SCIENCES» NAT. DES EUROPEENS ETRAN./CYCLE / SEXE»NAT/CYC
101
102
103
104
105
110
111
112
113
114
115
121
122
123
125
126
127
128
129
130
131
132
134
135
136
137
138
139
140
141
142
143
144
CYC1=-
CYC1=F
CYC2=-
CYC2=F
CYC3=-
CYC3=F
PR=01=-
PR=01=F
14,0R, «SCIENCES» ANNEES DE NAISSANCE /CYCLE /SEXE      »DATE/CY
 1954
 1953
 1952
 1951
 1950
 1949
 1948
 1947
 1946
 1945
 1944
 1943
 1942
<1942
 CYC1=H
 CYC1=F
 CYC2=H
 CYC2=F
 CYC3=H
 CYC3=F
PR=01=H
PR=01=F
```

Nombre de tableaux à éditer
Nombre de lignes, de colonnes et entête du 1er tableau

libellés pour les lignes du 1er tableau : codes INSEE des nationalités européennes (sauf française)

libellés pour les colonnes du 1er tableau

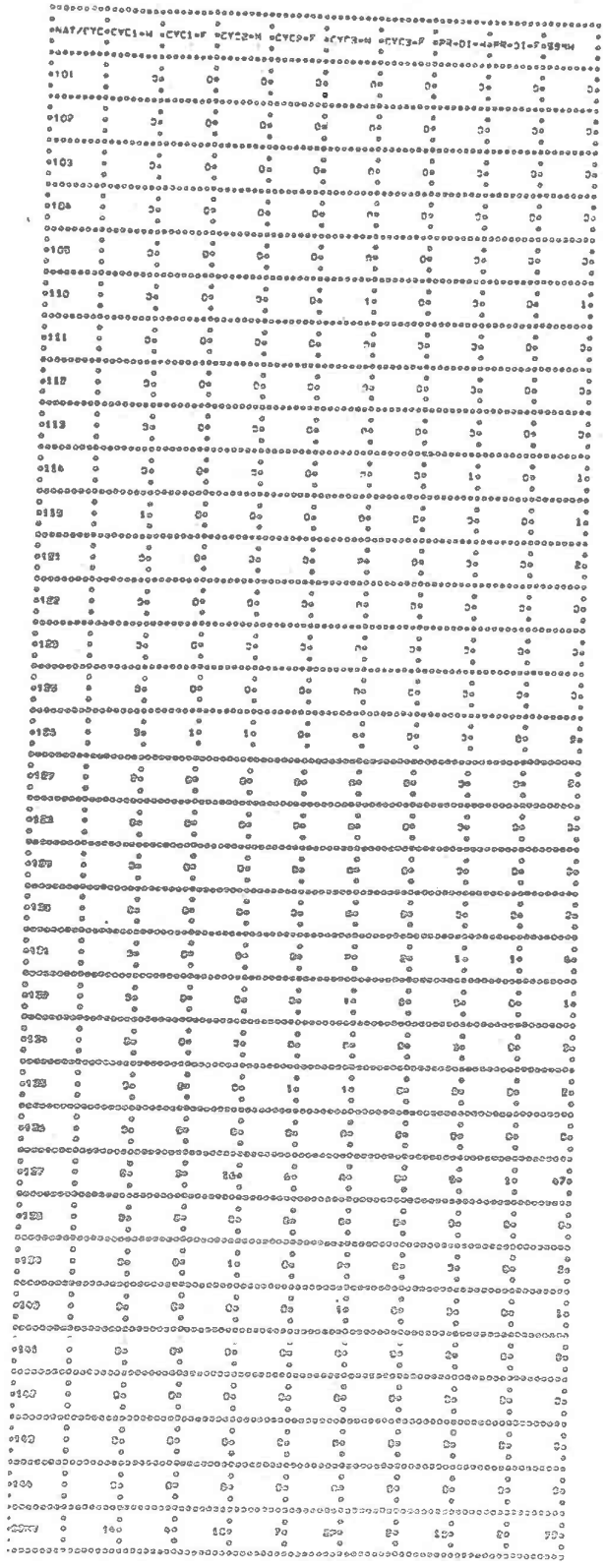
Nombre de lignes, de colonnes et entête du 2ème tableau

libellés pour les lignes du 2ème tableau : années de naissance

libellés pour les colonnes du 2ème tableau

RESULTATS :

«REFERENCE» NAT. DES EUROPEENS ETRAN./CYCLE / SEXE



SCIENCES ANNEES DE NAISSANCE /CYCLE /SEXE

ANNEE/CYC	CYC1-HO	CYC1-FE	CYC2-HO	CYC2-FE	CYC3-HO	CYC3-FE	PRO1-HO	PRO1-FE	SSMH
1954	0	1	1	0	0	0	0	0	2
1953	15	31	0	0	0	0	5	4	55
1952	124	119	1	0	0	0	20	9	273
1951	198	113	17	14	0	0	33	2	377
1950	210	83	83	43	1	0	46	7	473
1949	149	51	159	84	14	0	63	27	551
1948	73	25	166	76	56	8	77	32	511
1947	34	0	157	58	108	20	75	37	494
1946	0	0	89	35	131	15	60	25	373
1945	5	2	47	20	103	12	17	10	209
1944	4	2	38	16	82	12	12	7	175
1943	3	2	24	7	58	18	15	1	126
1942	3	0	19	8	40	7	9	3	89
<1942	5	1	54	18	284	37	8	6	415
TOTAL	893	466	884	383	877	129	449	170	4125

ANNEXE 7

- Liste des fichiers pour la gestion des étudiants
- Liste des traitements.
- Description des principaux fichiers.
- Description des principaux documents.

Liste des Fichiers pour la Gestion des Etudiants

Nom Fichier	Support	Accès (clé)	Fichier de la même classe	Arg. de Tri
établissement 1	B	S(numéro ‖ cycle)		numéro↑ | cycle ↑
établissement 2	B	d°	établissement 1	d°
établissement 3	B	d°	d°	d°
établissement	D	D(numéro ‖ cycle)	d°	d°
inscriptions 1	C	S		d°
inscriptions 2	B	S	inscriptions 1	d°
inscriptions 3	B	S	inscriptions 1	numéro | cycle | type
modifications 1	C	S		
modifications 2	B	S	modifications 1	
modifications 3	B	S	d°	numéro↑ | cycle↑ | maj↑ | type↑
éléments 1	C	S		élément
éléments 2	D	D (élément)	éléments 1	
étudiants/examens 1	B	S		numéro | cycle
étudiants/examens 2	B	S	étudiants/examens 1	élément | nom | prénoms
étudiants/examens 3	B	S	d°	d°
étudiants/examens 4	B	S	d°	numéro | cycle
résultats 1	C	S		
résultats 2	B	S	résultats 1	éléments | n° séquence

Liste des Traitements pour la Gestion des Etudiants

TRAITEMENTS	DOCUMENTS			FICHIERS				
	entrée	sortie	nbre	Entrée	Sortie	intermédiaire	Support	Accès
Création du fichier annuel	formulaires		5000	inscriptions1			C	S
	fiches historiques	anomalies	5000			inscriptions 2	B	S
			10000			inscriptions 3	B	S
		fiches historiques	11500			établissement	D	D
					établissement 1		B	S (clés)
					établissement 2		B	S (clés)
Mise à jour du fichier annuel	formulaires anomalies		500	modifications1			C	S
			500	établissement1			B	S (clés)
		anomalies	50			modifications 2	B	S
	fiches historiques					modifications 3	B	S
			500			établissement	D	D
		fiches historiques	500		établissement 1		B	S (clés)
					établissement 2		B	S (clés)
Remontée des résultats d'examens				établissement1			B	S
				éléments 1			C	S
						éléments 2	D	D
		procès-verbaux	2000			étudiants/ex1	B	S
				résultats 1		étudiants/ex2	B	S
							C	S
						résultats 2	B	S
						étudiants/ex3	B	S
						étudiants/ex4	B	S
					établissement 1		B	S (clés)
graphe des conflits		tableau		établissement1			B	S
listes diverses		liste		établissement1			B	S
						sous-fichier 1	B	S
						sous-fichier 2	B	S
Statistiques		tableaux	30	établissement1		modules	B	S
				description		DESCRIPTION		
						CONSTRUCTION		

DESCRIPTION DE FICHIER

APPLICATION : GESTION-ETUDIANTS				FICHIER : 'établissementl' SCIENCES
CODE U. T. : CREATION fichier-annuel				LABEL : N° GEN :
Séquentiel à clé (cycle || n° insee)				SUPPORT bande RET :
DE	A	REF. SYMBOL.	IMAGE	ANALYSE RUBRIQUE
1	2	ETAB	XX	code-établissement
3	3	CYCLE	9	cycle
4	7	NØ-INSC	9(4)	numéro d'inscription
8	21	NØ-NAT	9(13)X	numéro national d'identification
22	31	NOM	X(20)	nom
32	41	PNOM	X(10)	prénom
42	44	NAT	999	nationalité (nat)
45	45	SF	9	situation-famille
46	46	NENF	9	nombre-enfants
47	48	DØ	99	domicile
49	51	CØM	999	commune
52	52	SM	9	situation-militaire
53	53	LØ	9	logement
54	54	EMPL	9	emploi
55	56	EMP	99	profession-étudiant (emp)
57	57	DEM	9	demande-bourse
58	58	TAUX	X	taux de bourse
59	59	SBAC	X	série du bac
60	60	MENT	X	mention du bac

DESCRIPTION DE FICHIER

APPLICATION : GESTION-ETUDIANTS				FICHIER : 'établissement1' SCIENCES
CODE U. T.: CREATION fichier-annuel				LABEL : N° GEN :
				SUPPORT : bande RET :
DE	A	REF. SYMBOL.	IMAGE	ANALYSE RUBRIQUE
61	62	ANBAC	99	année-bac
63	64	DEPBAC	99	département-bac
65	73	DET	9(7)X9	dernier-établissement fréquenté
73	74	AD	99	année d'études (dans ce DET)
75	76	ES	99	études suivies (dans ce DET)
77	78	TI	99	titre le plus élevé obtenu
79	80	AT	99	année d'obtention (de TI)
81	82	AO	99	académie d'obtention (de TI)
83	84	SP	999	spécialité (de TI)
85	87	FILLER	XXX	
89	87	AET	9(7)X9	autre établissement fréquenté
88	89	AES	99	études suivies (dans AET)
90	90	IN	9	nombre d'années d'inscription dans ce cycle
91	92	AN1	99	année de 1ère inscription en 1er cycle
93	93	NI	9	niveau de 1ère inscription en 1er cycle
94	95	AN2	99	année de 2ème inscription en 2è cycle
96	97	AN3	99	année de 3ème inscription en 3è cycle
98	112	(UER,ES) 3 fois	99999 3 fois	UER, études suivies (3 fois)
113	127	ZSPEC	X(15)	zone-spécifique (dont diplômes préparés)

DESCRIPTION DE FICHIER

PAGE 0 3

APPLICATION : GESTION-ETUDIANTS				FICHIER : 'établissement l' SCIENCES	
CODE U. T. : CREATION fichier-annuel				LABEL : N° GEN :	
				SUPPORT: bande RET :	
DE	A	REF. SYMBOL.	IMAGE	ANALYSE RUBRIQUE	
128	209	EXAM	9(4)	élément préparé	12 fois
		RES1	9	résultat à la 1ère session	
		RES2	9	résultat à la 2ème session	
		RES-F	9	résultat final	
210		COL	9	collège électoral	

DESCRIPTION DE FICHIER

APPLICATION : GESTION-ETUDIANTS				FICHIER : 'éléments 1'
CODE U. T. : CREATION 'éléments 1'				LABEL : N° GEN :
				SUPPORT: cartes RET :
DE	A	REF. SYMBOL.	IMAGE	ANALYSE RUBRIQUE
1	2	ETAB	XX	code-établissement
3	3	CYCLE	9	cycle
4	7	EXAM	9(4)	code-examen
8	8	CLEF	A	clef de l'examen
9	11	UER	999	UER d'appartenance de l'examen
12	80	LIB	X (69)	libellé en clair

DESCRIPTION DE FICHIER

PAGE 0 1

APPLICATION : GESTION-ETUDIANTS				FICHIER : 'étudiants/examens 1'
CODE U. T. CREATION 'étudiants/examens'				LABEL : N° GEN :
séquentiel				SUPPORT : bande RET :
DE	A	REF. SYMBOL.	IMAGE	ANALYSE RUBRIQUE
1	2	ETAB	XX	code - établissement
3	3	CYCLE	9	cycle
4	7	EXAM	9(4)	examen
8	11	NØ-INS	9(4)	numéro d'inscription
12	25	NØ-NAT	9(13)X	numéro national d'identification
26	26	SF	9	situation de famille
27	46	NOM	X(20)	nom
47	56	PNOM	X(10)	prénom
57	62	FILLER	X(6)	divers
63	65	UER	XXX	UER principale
66	66	RES1	X	résultat 1ère session
67	67	RES2	X	résultat 2ème session
68	68	RESF	X	résultat final

FICHE HISTORIQUE (Groupe d' UER 'Sciences')

N° d'identification

Sexe M F

NOM :

Prénoms :

Date et lieu de naissance :

Nationalité :

Profession des Parents :

Adresses successives :

Situation Militaire :

FAMILLE C M V D

Nbre d'enfants :

Universités précédentes :

Baccalauréat : Année et lieu :

Brevet de Technicien : Année et lieu :

Equivalence :

Autres Etablts fréquentés :

EXAMENS

CERTIFICATS	ECHECS							Obtention	Mention

CHE D'ANALYSE ORGANIQUE
cipline dominante : SCIENCES 2e CYCLE

ANNEE UNIVERSITAIRE 1971-1972

UNIVERSITE DE NANCY I

CARTE 0 5 2 1 | 0 5 4 0 1 0 3 U

dentification à 13 chiffres (1 par case)
al et universitaire

Né(e) le | 1 9 | à

Département ou pays

our les femmes, nom de jeune fille)
juscules 1 lettre par case

Prénoms en majuscules
1 lettre par case

Pour les femmes mariées ou veuves, NOM DU MARI
En majuscules 1 lettre par case

NALITE

Cochez d'une croix la case qui correspond à votre situation ☒

Ne rien inscrire dans ce cadre

ATION DE FAMILLE
Célibataire ☐ 2. Veuf (ve) ☐ 3. Marié(e) ☐ 4. Divorcé(e) ☐ Nbre d'enfants

N

SSE COMPLETE DES PARENTS ou du TUTEUR (arrondissement pour les grandes villes).

SF NE

SSE COMPLETE DE L'ETUDIANT(E) pour l'année scolaire en cours

DO C

ATION MILITAIRE (pour les étudiants français seulement)
corporable ☐ 2. Sous les drapeaux ☐ 3. Exempté ☐ 4. Service accompli ☐

MI

E DE LOGEMENT ENVISAGE
z vos parents ☐ 2. Chambre chez un particulier ☐ 3. Interne dans un lycée ☐ 4. En foyer d'étudiants (privé) ☐
idence versitaire ☐ 6. Logement de fonction ☐ 7. Appartement personnel ☐ 8. Autre ☐

LO

VOUS CETTE ANNEE
MPLOI RETRIBUE ?
☐ 2. NON ☐
A temps complet ☐ A temps partiel ☐
Lequel ?

E1 PRO SA

E2

C

-VOUS UN EMPLOI RETRIBUE L'AN DERNIER ?
1 ☐ NON 2 ☐

ETIEZ-VOUS SOUS LES DRAPEAUX ?
OUI 3 ☐ NON 4 ☐

AC PRO SA

us êtes marié(e) conjoint est-il étudiant ? 1. OUI ☐ 2. NON ☐ Profession Conj.

père est-il en activité oui non Retraité oui non Décédé oui non
sion

RENSEIGNEMENTS SUR LA SCOLARITE

	Série	Mention	Année 19	Département d'obtention
lauréat				
lauréat de technicien				
titre équivalent au bac du 2e permettant accès à l'université				

Code Carte 0 5 2 2

S R A D

ous fait une demande de bourse ? OUI ☐ 0. NON ☐
s'agit-il d'un renouvellement ? 1. OUI ☐ 2. NON ☐
e refusée 0 ☐ Si la bourse est acceptée indiquez le taux de la bourse Indiquez le taux (de 1 à 7) ☐

quel établissement avez-vous été inscrit pour la dernière fois ?
les lycées et établissements privés, en préciser le niveau
nales : 01 ; classes préparatoires : 02 ; techniciens supérieurs et autres : 03)
sement : Département :
scolaire : 19 -19 Dernier niveau d'études suivies :

DET R

AD ES

titre d'accès au 2e cycle est :
☐ Section
alence ou dispense
us possédez la licence
alité
Année d'obtention 19
Académie d'obtention
Année d'obtention 19
Académie d'obtention

TI AT AO

TI AT AO SP

vous inscrit dans un autre établissement (université, école privée...)
☐ 2. NON ☐ Lequel ?
ier les études que vous y suivez (code n° 8)

AET R

ES

IN

AN

ésente année universitaire est votre
ière ou ième année d'inscription en 2e cycle
universitaire de 1re inscription en 1er cycle 19 -19
de 1re inscription en 1er cycle 1re année ou Propédeutique ☐ 2e année ☐
universitaire de 1re inscription en 2e cycle 19 -19

NI

AN

TSV

'inscription

quelle (1) UER vous inscrivez-vous à cette université en 1971-72
sez pour chaque UER les études que vous y suivez (code n°8)

vous élève de l'IPES ? OUI NON

ii, année d'entrée à l'IPES 19 -19

vous inscrit à un autre cycle - indiquez lequel (1, 3 ou 4) (4 : préparations diverses)

'ARATION DEMANDEE :

laîtrise Titre :

ES ou Unités

4)

5)

6)

certifié sincère et véritable

Signature :

Code carte 0 5 2 3

UER ES
UER ES
UER ES

AL O
IP
AN MT O
AC O
DI SP

Code Carte 0 5 2 4

Visa du préposé au contrôle

DÉCOMPTE DES DROITS

Insc.	Bib.	Disp.	S.S.	T.P.		Mutuelle			Total

P. PARADIS - LUNÉVILLE

Code carte 0 5 2 3

UER
UER
UER

ES
ES
ES

AL 0
IP
AN
AC
DI

MT 0
0

SP

Code Carte 0 5 2 4

Visa du préposé au contrôle

		Total

UNIVERSITÉ de

PROCÈS-VERBAL de la session de 197

cycle

UER :

l'ordre :

ÉPREUVES de

Cadre réservé à l'administration

N° d'identification de l'étudiant	Nom et Prénoms	

1 10

11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35

A NANCY, le

Le Président du Jury,

S.A. - 471

BIBLIOGRAPHIE

[1] O. C. D. E.
Méthodes et besoins statistiques pour la planification de l'enseignement (Paris 1967).

[2] KARLIN S.
A first course in stochastic process - (Stanford University Academic Press - New-York and London)

[3] BERGE C.
Théorie des graphes et ses applications - (Dunod - 1968)

[4] CLERMONT J. C.
Note technique RCHNT/11 (C. I. R. O. 1970).

[5] MAGHOUT K.
Applications de l'algèbre de Boole à la théorie des graphes. (Cahiers du Centre d'Etudes de R. O. Bruxelles 1963).

[6] KAUFMANN A.
Introduction à la combinatorique en vue des applications - (Dunod - 1968).

[7] HALL A. D.
Scheduling University Course examinations by computer (Communication A. C. M. - avril 1967).

[8] ROY B.
Algèbre moderne et théorie des graphes - (Dunod - 1970).

[9] HERVE P.
Les procédures arborescentes en exploration combinatoire - (Shell Berre Recherche et Développement - Paris 1967)

[10] O. C. D. E.
Modèles mathématiques pour la planification de l'enseignement (Paris 1969).

[11] CARTAN H.
Calcul différentiel - (Hermann - 1969).

[12] I. C. L. : F. I. N. D.
(File Interrogation of 1900 Data) - Manuel de présentation.

[13] X. D. S : MANAGE
Reference manual for S. D. S. Sigma 5/7 computer.

[14] CONTROL DATA : I. N. F. O. L.

(Information Oriented Language) manuel d'utilisation.

[15] Mc GEE (I. B. M.)

Generalized File Processing : (article paru en 1969 dans "Annual Review in Automatic Programming" Pergamon - Press).

[16] DERNIAME J. C. et son équipe

Séminaires 1970-1971 de l'I. U. C. A. sur le projet CIVA et présentation à l'école d'été (ALES Juillet 1971).

[17] MELESE J.

La gestion par les systèmes - (Editions Hommes et Techniques 1970).

NOM DE L'ETUDIANT : DUFOURD Jean François

Nature de la thèse : Doctorat de Spécialité en Mathématiques Appliquées

Vu, Approuvé

et permis d'imprimer

NANCY, le 19 Novembre 1971

Le Président du Conseil de l'Université de NANCY I

J.R. HELLUY