50 376
1968
28.

n° d'ordre 167

THÈSES

50.376
1968
28

PRÉSENTÉES

A LA FACULTÉ DES SCIENCES DE L'UNIVERSITÉ DE LILLE

POUR OBTENIR

LE GRADE DE DOCTEUR ÈS-SCIENCES PHYSIQUES

PAR

Vincent Cordonnier
ingénieur ISEN
licencié ès-sciences physiques
diplomé d'études supérieures

PREMIÈRE THÈSE

ETUDE DE LA SIMULTANEITE EN MEMOIRE CENTRALE D'UN ORDINATEUR MODELE DE MEMOIRE PARTAGEE

DEUXIÈME THÈSE

Propositions données par la Faculté

BIBLIOTHÈQUE UNIVERSITAIRE Section SCIENCES LILLE

Soutenues en mai 1968 devant la Commission d'examen

MM.	P. BACCHUS	Président
	P. POUZET P. VIDAL	Examinateurs
	J. ARSAC	Membre invité

UNIVERSITE DE LILLE
FACULTE DES SCIENCES

-=-=-=-

DOYENS HONORAIRES

MM. H. LEFEBVRE, M. PARREAU

PROFESSEURS HONORAIRES

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

DOYEN

Monsieur DEFRETIN, Professeur de biologie et physiologie animales.

ASSESSEURS

MM. HEUBEL, Professeur de chimie.
LEBRUN, Professeur d'électronique.

PROFESSEURS

MM.	BACCHUS	Mathématiques Appliquées
	BEAUFILS	Chimie
	BONNEMAN	Chimie
	BECART	Physique
	BLOCH	Biologie et Physiologie Animales
	BONTE	Sciences de la Terre
	BOUGHON	Mathématiques Pures
	BOUISSET	Biologie et Physiologie Animales
	BOURIQUET	Biologie Végétale
	CELET	Sciences de la Terre
	CONSTANT	Electronique, Electrotechnique et Automatique
	CORSIN	Sciences de la Terre
	DECUYPER	Mathématiques Pures
	DEDECKER	Mathématiques Pures
	DEHORS	Electronique, Electrotechnique et Automatique
	DELATTRE	Sciences de la Terre

MM.	DELEAU	Sciences de la Terre
	DELHAYE	Chimie
	DESCOMBES	Mathématiques Pures
	DURCHON	Biologie et Physiologie Animales
	FOURET	Physique
	GABILLARD	Electronique, Electrotechnique et Automatic
	GLACET	Chimie
	GONTIER	Mathématiques Appliquées
	HEIM DE BALSAC	Biologie et Physiologie Animales
	HOCQUETTE	Biologie Végétale
	LEBEGUE	Botanique (Amiens)
	Mme LEBEGUE	Physique (Amiens)
	Melle LENOBLE	Physique
	LIEBAERT	Electronique, Electrotechnique et Automatique
	LINDER	Biologie Végétale
	LUCQUIN	Chimie
	MARION	Chimie (Amiens)
	MARTINOT-LAGARDE	Mathématiques Appliquées
	Melle MARQUET	Mathématiques Pures
	MENNESSIER	Géologie (Amiens)
	MONTARIOL	Chimie
	MONTREUIL	Chimie
	MORIAMEZ	Physique (Valenciennes)
	MOUVIER	Chimie (Saint-Quentin)
	PARREAU	Mathématiques Pures
	PEREZ	Physique
	PHAM MAU QUAN	Mathématiques Pures
	POUZET	Mathématiques Appliquées
	PROUVOST	Sciences de la Terre
	SAVARD	Chimie
	SCHILTZ	Physique
	SCHALLER	Biologie et Physiologie Animales
	Mme SCHWARTZ	Mathématiques Pures
	TILLIEU	Physique
	TRIDOT	Chimie
	VAZART	Botanique (Amiens)
	VIVIER	Biologie et Physiologie Animales (Amiens)

MM.	WATERLOT	Sciences de la Terre
	WERTHEIMER	Physique

MAITRES DE CONFERENCES

MM.	BELLET	Physique
	BENABOU	Mathématiques Pures
	BILLARD	Physique
	BOILLET	Physique
	BUI TRONG LIEU	Mathématiques Pures
	CHERRUAULT	Mathématiques Pures
	CHEVALIER	Mathématiques (Amiens)
	DERCOURT	Sciences de la Terre
	DEVRAINNE	Chimie (Calais)
	Mme DIXMIER	Mathématiques (Amiens)
	Mme DRAN	Chimie
	DUQUESNOY	Chimie (Amiens)
	GOUDMAND	Chimie
	GUILBAULT	Biologie et Physiologie Animales
	GUILLAUME	Biologie Végétale
	HENRY	Physique (Amiens)
	HERZ	Mathématiques Appliquées
	HEYMAN	Physique (Amiens)
	HUARD DE LA MARRE	Mathématiques Appliquées
	JOLY	Biologie et Physiologie Animales (Amiens)
	LABLACHE-COMBIER	Chimie
	LACOSTE	Biologie Végétale
	LAMBERT	Physique (Saint-Quentin)
	LANDAIS	Chimie
	LEHMANN	Mathématiques Pures
	Mme LEHMANN	Mathématiques Pures
	LOUCHEUX	Chimie
	MAES	Physique
	METTETAL	Zoologie (Amiens)
	MONTEL	Physique
	NGUYEN PHONG CHAU	Mathématiques (Saint-Quentin)
	PANET	Electronique, Electrotechnique et Automatique

PARSY	Mathématiques Pures
RACZY	Physique (Valenciennes)
SAADA	Physique
SEGARD	Chimie
TUDO	Chimie Minérale Appliquée (Amiens)
VAILLANT	Mathématiques Pures
VIDAL	Electronique, Electrotechnique et Automatique
Mme ZINN-JUSTIN	Mathématiques Pures

TABLE DES MATIERES

INTRODUCTION

CHAPITRE I - LES ELEMENTS DE DEFINITION DE LA MEMOIRE CENTRALE

A - Capacité de la mémoire centrale et format technologique.

B - Format logique de l'information.

- Les informations à représenter.
- Longueur optimum de l'unité d'information.

CHAPITRE II - ACCROISSEMENT DE LA VITESSE DE LA MEMOIRE : MEMOIRE PARTAGEE.

A - L'allongement des mots dans la mémoire.

B - Partage de la mémoire.

C - Evaluation du débit d'une mémoire partagée.

D - Organisation pratique d'une mémoire partagée.

- Commutation spaciale
- Commutation spacio-temporelle.

E - La mise en oeuvre des priorités.

CHAPITRE III - COMPORTEMENT DE LA MEMOIRE EN PRESENCE DES CIRCUITS QUI L'EXPLOITENT.

Référence à des matériels existant ou en projet.

A - Occupation de la mémoire pour un taux donné de demandes.

B - Solution particulière dans le cas du régime stationnaire.

C - Emploi d'une file d'attente.

D - Occupation de la mémoire associée à une file d'attente.

E - Temps de service.

CHAPITRE IV - COMPORTEMENT DE LA MEMOIRE EN REGIME TRANSITOIRE.

A - Calcul du comportement de la mémoire en régime transitoire.

B - Solution par l'emploi de variables discontinues.

C - Rôle de la file d'attente en régime transitoire.

- Cas de l'interruption de programme.
- Cas du branchement conditionnel

D - Temps perdu par le traitement.

CHAPITRE V - MODELE DE GESTION D'UNE MEMOIRE PARTAGEE.

A - Principes d'une organisation de recherche anticipée.

- La file d'attente de données.
- La file d'attente d'instructions.
- La file d'attente d'écriture.

B - Optimisation de la recherche anticipée.

- Caractèristiques communes aux files d'attente.
- Principe de l'organisation proposée.
- La mémoire à logique distribuée.
- Chaine des cellules disponibles.
- Chaine des priorités.

C - Format des données traitées.

D - Séquences de traitement des demandes.

CONCLUSION.

BIBLIOGRAPHIE.

INTRODUCTION

L'évolution technologique des systèmes de traitement de l'information se poursuit depuis quelques années dans deux directions complémentaires. Toutes deux visent à accroitre la vitesse à laquelle les traitements peuvent être poursuivis. L'accroissement de vitesse ne concerne pas seulement les unités centrales mais aussi et parallélement les moyens d'accés à la machine.

La première de ces deux directions est une tentative au niveau des composants élémentaires d'une machine que sont les circuits logiques et de stockage. On peut admettre facilement que le gain de vitesse d'un ensemble est proportionnel au gain réalisé sur les vitesses des composants, à condition, toutefois, que l'accroissement des performances élémentaires reste homogéne pour l'ensemble des circuits. Il semble que cette première direction soit, dans l'état actuel de la technologie, d'un intérêt limité en raison du coût lié à l'accroissement des performances et qui croit exponentiellement en fonction de la vitesse. Pour les circuits logiques, on situe autour d'une dizaine de nano-secondes, le temps de commutation qui permette de réaliser le meilleur compromis de prix à performances. On admet, de plus, que cet optimum ne variera guère dans les prochaines années, les efforts de recherche s'orientant plus vers l'intégration à grande échelle des circuits de traitement et de stockage que vers un gain de performances intrinséques de chacun d'eux.

La seconde direction permettant d'obtenir un gain en performances est la simultanéité des traitements. Il y a longtemps que les machines disposent du moyen de poursuivre simultanément le traitement interne et les échanges avec les organes périphériques. Il y a quelques années, la notion de simultanéité a été introduite dans l'unité centrale elle-même sous des formes diverses et plus ou moins efficaces. Ce qu'on appelle le multitraitement ne peut être l'apanage que d'ensembles importants ou trés spécialisés. L'évolution technologique citée plus haut va vraisemblablement permettre de doter des ensembles moyens, voire même modestes, de ces possibilités de traitements internes simultanés.

Les éléments de stockage et particulièrement la mémoire centrale qui se trouvent sollicités au début et à la fin de toute opération, doivent suivre une évolution comparable. Or un examen de l'évolution des circuits logiques d'une part, des circuits de stockage d'autre part, fait apparaitre une différence de plus en plus marquée entre leurs vitesses relatives.

Cet état de fait se trouve aggravé par l'accroissement des tâches qui sont confiées à la mémoire. L'apparition des simultanéités, leur gestion, le multi-traitement et le volume croissant des programmes de contrôle et d'ordonancement interne exigent des capacités croissantes et des débits de mémoire de plus en plus élevés.

Le but de ce travail est d'étudier dans quelles conditions il est possible d'améliorer les performances globales d'un organe de stockage en faisant fonctionner en simultanéité plusieurs éléments indépendants. L'étude concerne surtout la mémoire centrale d'un ordinateur qui est la plus sollicitée et des manières les plus diverses. Il est certain, néammoins, à condition de changer l'échelle des temps, que presque tous les résultats pourraient s'appliquer à un groupe de mémoires externes pour définir, par exemple, les conditions d'exploitation d'un fichier de trés grandes dimensions.

Ce qui, cependant, particularise l'application à la mémoire centrale, c'est que les choix, les décisions et les réglements de conflits, inhérents à toute gestion simultannée, ne peuvent étre pris en charge par le programme. La logique associée à un groupe de mémoires centrales fonctionnant en simultanéité doit étre totalement cablée ce qui conduit ~~conduit~~ à examiner en détail les diverses éventualités qui peuvent se présenter. Un modèle de gestion est proposé qui présente l'avantage d'une structure logique totalement unifiée. Ceci est rendu possible par l'emploi d'une mémoire supplémentaire de type associatif. Ses performances doivent étre nettement supérieures à celles des unités qu'elle contrôle mais elle ne réclame, en contre-partie, qu'une capacité très limitée.

CHAPITRE I

LES ELEMENTS DE DEFINITION DE LA MEMOIRE CENTRALE.

INTRODUCTION

Pour deux raisons, la définition des caractèristiques d'un système de traitement de l'information doit commencer par celles de sa mémoire.

La première résulte de la position privilégiée de la mémoire au centre de tous les échanges et de tous les traitements entrepris par la machine. Le choix des caractèristiques de la mémoire fixe, de manière presque difinitive, une grande partie des caractèristiques globales de l'ensemble : Le format des données et des instructions, les possibilités d'adressage, les modes d'accès des organes périphériques.

La seconde raison est d'ordre technologique. Actuellement la fonction de stockage est, relativement à la fréquence de son emploi, la plus lente des fonctions internes d'une machine. De plus, depuis quelques années, l'éccart des vitesses relatives se creuse: En 1959, par exemple, la machine I.B.M. 704 disposait d'une mémoire à tores de 11 micro-secondes de temps de cycle alors qu'il lui fallait 72 micro-secondes pour effectuer une addition; dans une machine récente du même constructeur disposant d'une mémoire de 750 nano-secondes de temps de cycle, l'addition ne réclame plus que 120 nano-secondes. L'équilibre ne peut se rétablir que par une étude logique de la structure de la machine dans laquelle la mémoire sera, en premier lieu, favorisée.

Les éléments sur lesquels portent les choix à faire sont liés les uns aux autres; tous doivent être susceptibles d'une évaluation du rapport: performances à prix.

A - CAPACITE DE LA MEMOIRE ET FORMAT TECHNOLOGIQUE.

La presque totalité des mémoires centrales qui équipent les ordinateurs actuels est représentée par des mémoires à tores magnétiques. L'évolution la plus prévisible se fera dans le sens d'un accroissement de la vitesse et de possibilités de lecture non destructive. La technologie qui, dans quelques années, a les plus grandes chances de supplanter les tores magnétiques est la mémoire à circuits actifs intégrés. L'organisation logique en sera identique.

Une mémoire est, en premier lieu, définie par sa capacité. Il est habituel de le faire en indiquant successivement le nombre de positions adressables et la longueur de chacune de ces positions. Si l'on suppose que la mémoire est utilisée au mieux, on exprimera plus simplement cette capacité par le nombre total de positions binaires dont elle dispose. Cette capacité C est le produit du nombre M de positions adressables par le nombre L de positions binaires des mots technologiques.

A priori, on cherchera à obtenir la capacité désirée avec un choix sur les valeurs de M et L tel que le prix unitaire de la cellule de mémoire soit le plus faible possible. Il faut ajouter comme élément du choix, la vitesse que l'on peut évaluer par le temps de cycle de la mémoire ou mieux, par le produit de ce temps de cycle par la longueur du mot traité.

L'obtention de la formule PRIX = F (M, L, T) permettrait de comparer avec précision les différentes solutions possibles.

Une telle tentative peut s'appuyer soit sur des considérations de prix de revient, soit sur les catalogues de prix fournis par les constructeurs. Ces deux approches sont complémentaires, la première pouvant fournir l'allure de la courbe représentative, la seconde fournissant la valeur réelle des coéfficients dans la détermination desquels entrent trop d'éléments partiels : Prix des composants, temps d'assemblage, ammortissement du matériel de fabrication, présence des services commerciaux et nécessité d'uniformiser et de rationaliser les prix.

Estimation à partir des composants;

Toutes les mémoires à tores magnétiques comportent:

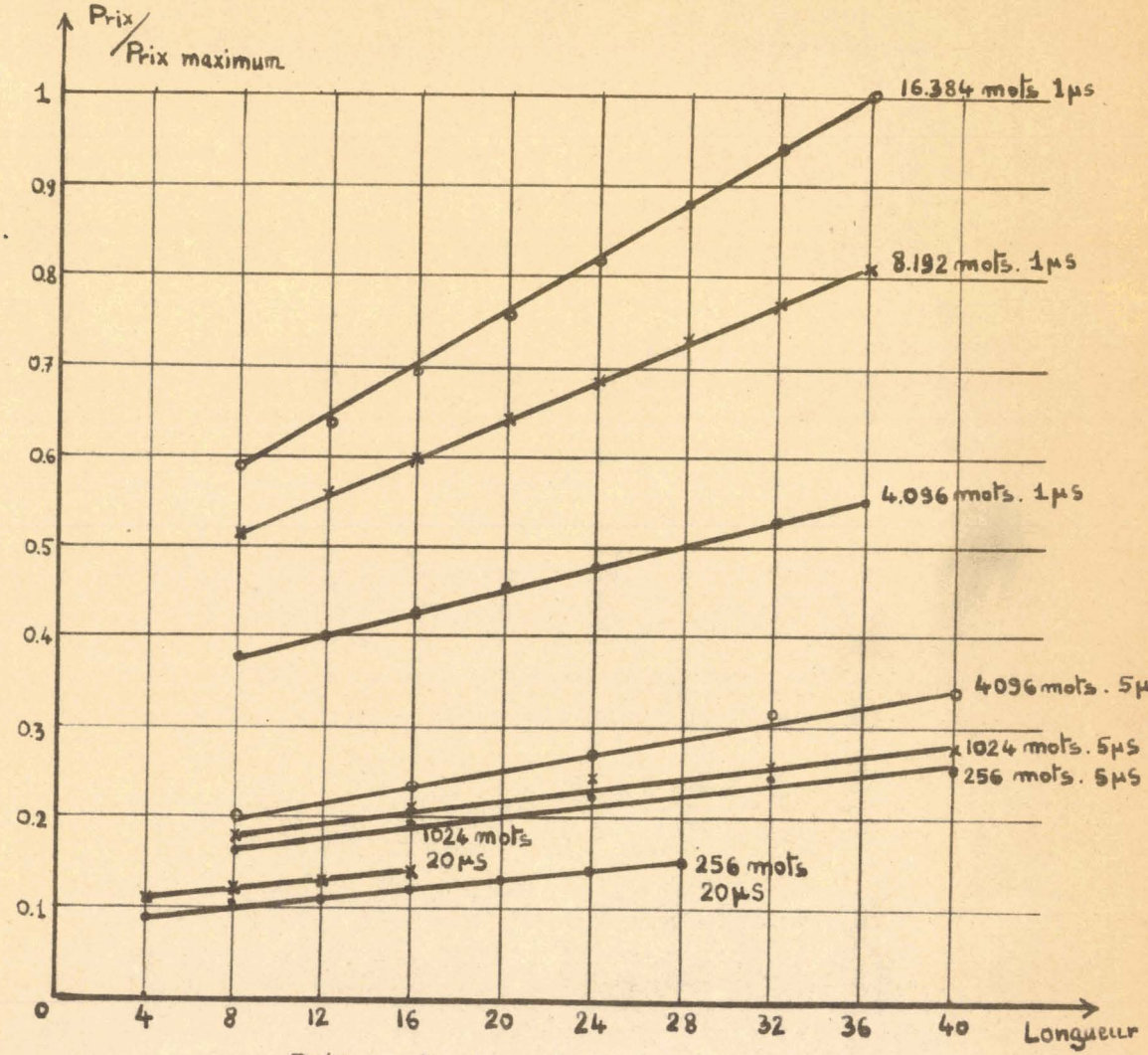
- L'empilage des plans de mémoire.
- Les circuits de sélection d'adresses.
- Les générateurs de courants pour la lecture, l'écriture et qui commandent les fils de sélection.
- Les circuits de lecture et d'inhibition.
- Les alimentations et les circuits annexes.
- Les circuits de rythme et les liaisons avec l'extérieur.

La loi la plus simple à déterminer est celle qui relie le prix à la longueur du mot, la vitesse et le nombre de mots restant constant. Le nombre de plans et de circuit de lecture et d'inhibition est égal à la longueur du mot. Les autres circuits sont pratiquement indépendants de cette longueur.

Estimation directe à partir des prix de vente;

On doit donc trouver une loi de la forme : $P = P_o + k.L$

Les courbes ci-dessous montrent que cette loi correspond bien à la réalité.



Prix en fonction de la longueur du mot.

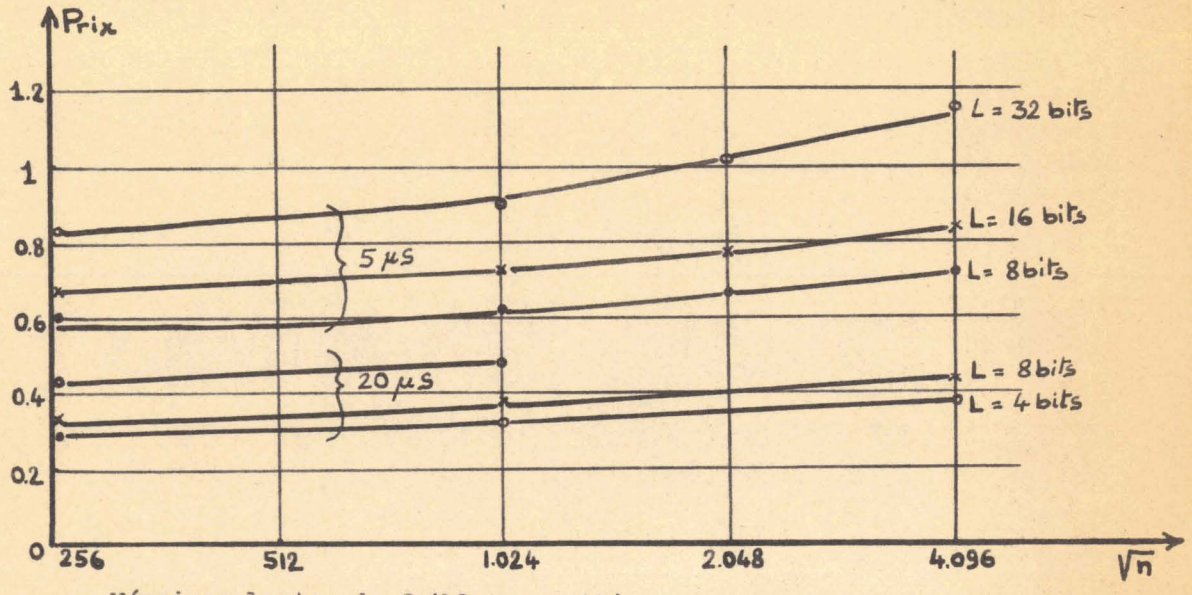
La loi qui permet d'évaluer le prix en fonction du nombre de mots adressables de la mémoire est plus complexe car, selon la technologie employée et la rapidité désirée, les solutions sont très diverses. Lorsque le nombre de mots passe de n à 4 n, le nombre de lignes de commande est doublé. En pratique, l'emploi de transformateurs de courants commandés par plusieurs primaires, de circuits d'aiguillage ou de regroupement à diodes, la division d'un plan de mémoire en deux ou quatre sous-groupes afin de conserver un bon rapport signal à bruit, font que cette estimation n'est qu'approximative.

Il faut, de plus remarquer que la variation ne peut être continue comme pour la longueur des mots. Il s'agit ordinairement d'une loi géométrique de raison 2 et dans un intervalle relativement restreint (1.024 à 32.768 mots).

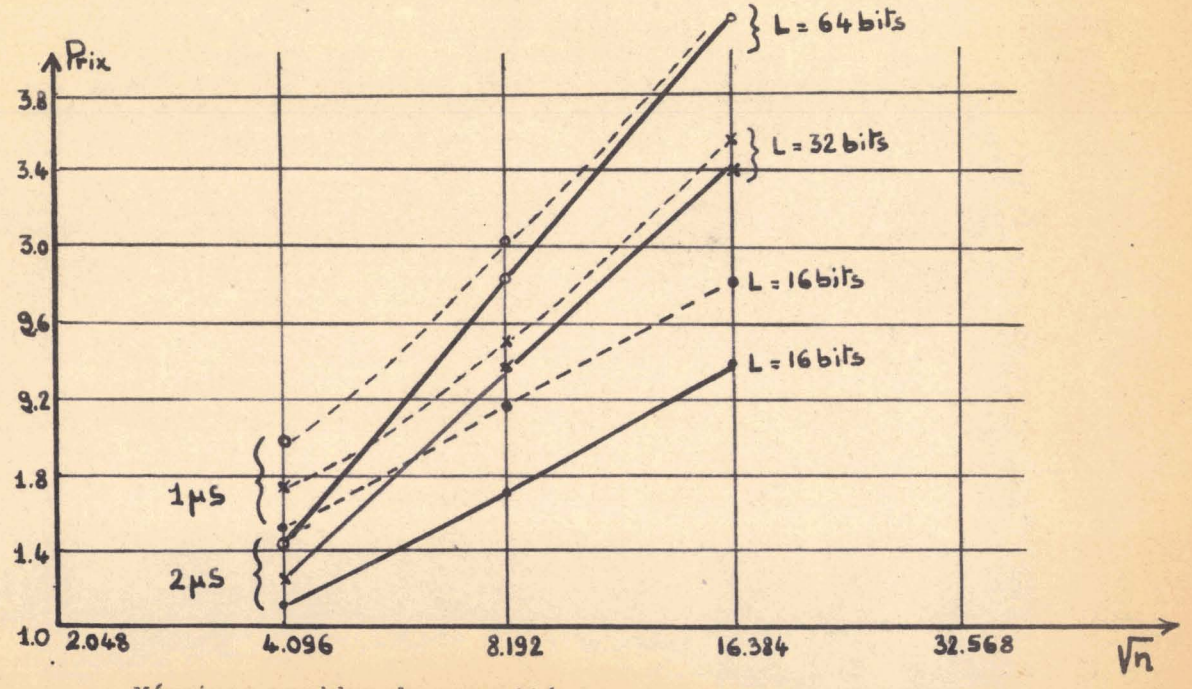
Si on pose cependant que le prix doit être de la forme :

$$P = P'_0 + k' \sqrt{M}$$

On obtient un résultat qui ne concorde que d'assez loin avec les valeurs réellement proposées.



Mémoires lentes de faible capacité



Mémoires rapides de capacité moyenne

Il est possible d'extraire de ces deux courbes déterminant le prix en fonction de la longueur et du prix en fonction du nombre des mots, un résultat plus concret en prenant comme paramètres la vitesse et la capacité exprimée par la quantité de positions binaires contenues dans la mémoire.

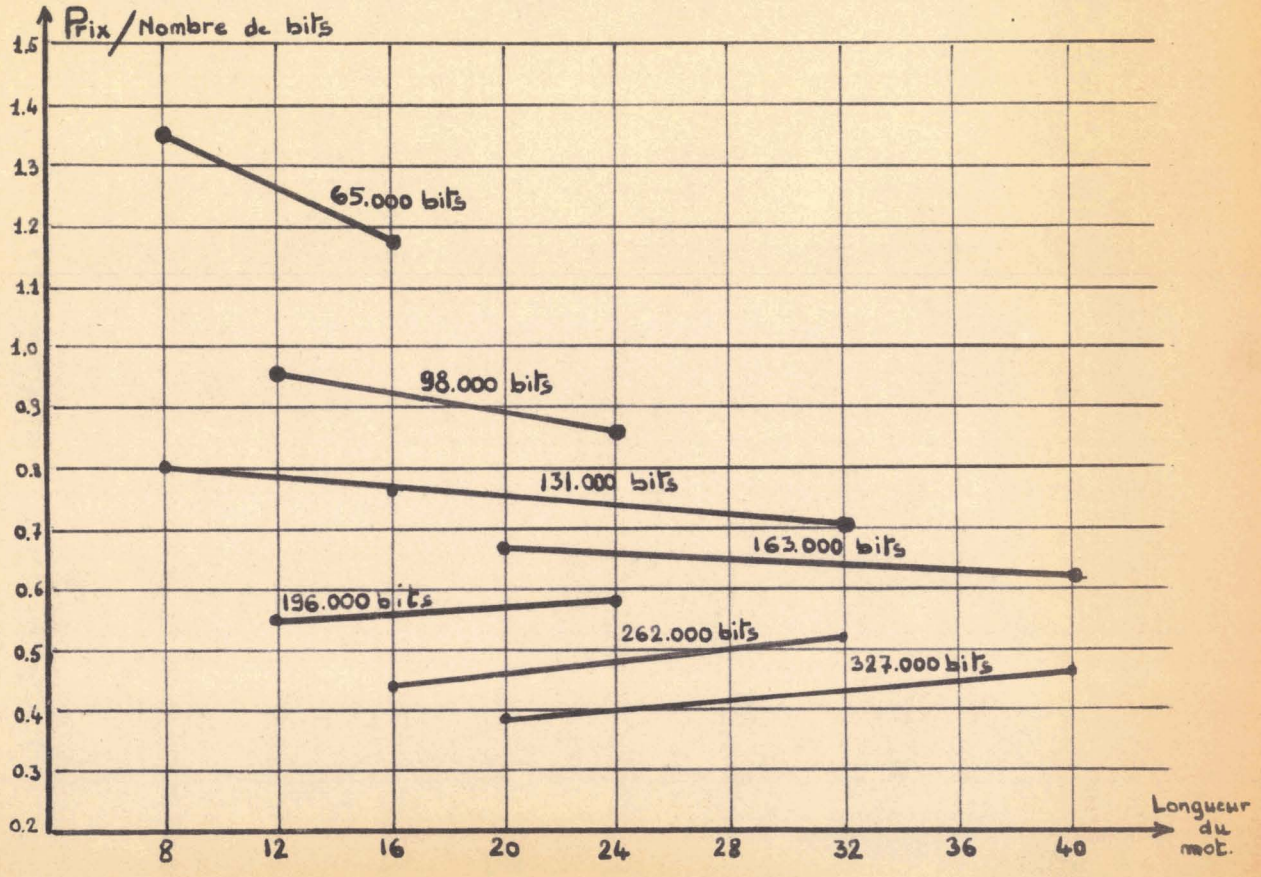
Pour une capacité donnée, la variable sera par exemple, la longueur du mot et la fonction le prix de la position binaire.

Pour des mémoires ne comportant qu'un petit nombre de mots, la nécessité de disposer d'alimentations, de circuits de dythmes dont le coût est à peu près indépendant de la capacité de la mémoire, doit faire apparaître l'intérêt d'allonger les mots.

Pour des mémoires de capacité plus importante, le coût de ces éléments devient proportionnellement plus faible. Comme le prix croit linéairement avec la longueur des mots mais simplement comme la racine carrée de leur nombre, il doit exister une capacité pour laquelle il y a exactement compensation.

Au delà de cette capacité, l'accroissement de la capacité sera plus avantageusement obtenu par une augmentation du nombre des mots.

La courbe ci-dessous fait effectivement apparaître ce fait. Elle ne peut cependant constituer seule, un premier stade indépendant des choix à opérer car elle ne représente que les éléments statiques du stockage. Les caractéristiques dynamiques de ce stockage sont directement liées à la longueur du mot.



B - FORMAT LOGIQUE DE L'INFORMATION.

Le support physique de l'information est presqu'universellement binaire. Cette propriété, imposée par la technologie, se prête, du reste, parfaitement au traitement automatique car elle ne laisse, à chaque instant et pour chaque information élémentaire, que l'alternative entre deux solutions. Une technologie capable de présenter plus de deux états serait certainement plus riche mais il n'est pas certain que les moyens théoriques actuellement disponibles permettent d'exploiter au mieux cette richesse.

Il existe une autre raison de choisir le binaire: Etant donné qu'en un endroit d'une machine doit prendre place une information pouvant présenter N valeurs différentes, le code binaire conduit à une redondance moyenne minimale et donc à la structure la plus économique.

Pour comparer deux sytèmes, il faut le faire sur la même valeur maximum de N : Pour comparer les systèmes à base 2 et 10, on a pris 1.024 et 1.000 . Pour comparer les systèmes à base 2 et 3 on a pris 2.048 et 2.187 .

Lorsque N varie de manière continue de 0 à la valeur maximum on trouve pour les trois systèmes, les redondances moyennes suivantes:

- Système à base 2 : 1,35
- Système à base 3 : 1,50
- Système à base 10: 1,82

En pratique, cet avantage du binaire est un peu illusoire de ce point de vue car la distribution des valeurs de N est loin d'être régulière et continue; d'autre part il n'est pas toujours possible d'exploiter pratiquement l'avantage théorique invoqué : La longueur des mots ne sera pas la longueur exactement utile mais un multiple entier d'une longueur de base ce qui accroit la redondance; la longueur de certains mots peut varier au cours du traitement et la longueur retenue en sera la valeur maximum.

On doit enfin remarquer que dans les algorithmes soumis au machines, le raisonnement s'appuie sur des choix entre deux solutions à partir de la présence ou de l'absence d'une condition unique. Il est rare que le nombre des issues soit supérieur à deux et il s'agit alors de cas particuliers provenant plus de la nature propre du problème que de principes généraux. Le binaire facilite ainsi la conception et surtout l'emploi des machines. On retrouve aussi un critère économique puisque tout aiguillage complexes peut se décrire comme un enchainement de dichotomies successives.

a) Les informations à représenter.

Une analyse faite par les logiciens d'IBM à propos de la machine STRETCH conduisait à distinguer cinq familles de données en fonction de leur taille, leur codage et leur emploi (1):

- Les quantités numériques exprimées en virgule fixe.
- Les adresses qui représentent un cas particulier de nombre en virgule fixe mais avec un format réduit.
- Les quantités numériques exprimées en virgule flottante.
- Les mots logiques tels que les codes d'opération, les masques et les mots chargés de décrire l'état de la machine et l'évolution du traitement en cours.
- Les informations dites d'édition qui se présentent comme des suites de caractères de longueur quelconque. Ces informations représentent les données qui apparaissent dans les échanges entre l'unité centrale et les organes d'accès.

Une information peut faire l'objet de traductions qui, en lui conservant son sens, la font passer d'une famille à une autre. C'est ainsi qu'un nombre est présenté au début du traitement comme une suite de caractères puis se trouve transformé en une expression binaire en virgule flottante ou en virgule fixe pour être enfin reconvertie en une suite de caractères avant d'être éditée.

On pourrait imaginer d'accroitre encore la variété des informations que peut gérer la machine en faisant figurer les nombres en précision multiple, les nombres complexes et, par exemple, les listes et les tableaux.

Pour que ces différents modes d'expression soient acceptés par la mémoire, donc dans le format fixe et unique imposé par la technologie et d'autre part par les circuits spécialisés, le logicien ne peut que découper un mot de mémoire ou, au contraire, en grouper plusieurs selon ses besoins. La tendance actuelle qui favorise le traitement en parallèle pour des question de rapidité, provoque un alongement des mots technologiques afin de permettre à la mémoire de fournir, en un seul cycle, les informations les plus volumineuses qu'elle puisse contenir.

L'accroissement du débit de la mémoire obtenu par des mots technologiques longs conduit à un diminution du rendement de l'élément de stockage, les informations les plus courtes n'occupant qu'une fraction de la place qui est mise à leur disposition. Le taux d'occupation utile de la mémoire sera donc plus faible.

Pour concilier la bonne occupation de la mémoire avec son débit, il faut employer des formats variables, obtenus par regroupement mais surtout par découpage du mot technologique en essayant d'organiser le stockage pour que les informations extraites de la mémoire soient toutes utiles.

Le meilleur emploi d'une zone de mémoire exigerait que chaque mot se voit attribuer le nombre exact de bits dont il a besoin. Si L représente la longueur maximum, les longueurs des différents formats seront des sous-multiples de L. Une information sera adressée par l'indication du mot technologique qui la contient, sa position dans ce mot et sa longueur. Le bénéfice obtenu ainsi dans la zone de stockage des données exige, en contre-partie, une augmentation de capacité dans la zone de définition, zone qui contient les adresses. Ainsi, par exemple, dans une mémoire de 1024 mots technologiques, chaque mot étant formé de 64 bits, il existe 65.536 positions binaires. Une adresse capable de désigner une position comporterait 16 bits. Si la longueur peut être comprise en 1 et 64 bits, il faut ajouter à l'adresse de la position initiale, une partie "longueur" qui réclamera 6 bits. Au total, la définition d'un mot exigera 22bits alors que la définition des mots technologiques de la mémoire n'en réclame que 10. Il faut donc trouver une solution de compromis pour laquelle on cherchera une optimisation simultannée de la zone des données et de la zone des adresses.

b) Recherche de la longueur optimum de l'unité d'information.

L'adressage se faisant en binaire, la découpe des mots technologiques doit, elle aussi, se faire selon cette numération de manière à établir une continuité entre les positions successives, condition indispensable à l'emploi du calcul d'adresses et de l'indexation. C'est aussi la seule méthode pour exprimer, sans redondance, la longueur du mot.

La place occupée par une information se compose de la zone qui lui est allouée pour le stockage arondie à un multiple entier de la longueur de base et de la zone qui sert à la définition de son emplacement ; cette seconde zone se trouve le plus souvent dans la partie réservée au programme.

On appelle L_1 le nombre de bits utilisés pour le stockage, L_2 le nombre de bits nécessaires à l'adressage et L_0 la longueur de l'information de base. Tout mot enregistré dans la mémoire sera astreint à occuper un nombre de positions binaires qui soit un multiple de L_0 .

On suppose que la distribution des longueurs est uniforme ce qui permet d'évaluer facilement la perte de place correspondant à un choix donné de L_0.

Si on prend la mémoire de 1.024 mots de 64 bits déjà citée plus haut en exemple, on suppose que la plus grande information qu'il soit possible de définir comporte effectivement 64 bits. Pour les longueurs L_0 comprises entre 1 et 64, on peut chiffrer la place totale nécessaire en mémoire pour stocker une information :

Valeur de L_0	Définition de l'adresse initiale	Définition de la longueur	Perte moyenne de place par mot	Nombre de bits utilisés
1 bit	16 bits	6 bits	0 bit	22 bits
2 bits	15 bits	5 bits	0,5 bit	20,5 bits
4 bits	14 bits	4 bits	1,5 bits	19,5 bits
8 bits	13 bits	3 bits	3,5 bits	19,5 bits
16 bits	12 bits	2 bits	7,5 bits	21,5 bits
32 bits	11 bits	1 bit	15,5 bits	27,5 bits
64 bits	10 bits	0 bit	31,5 bits	41,5 bits

Le minimum se situe donc entre les valeurs de L_0 de 4 et 8 bits. Si on prend comme unité indivisible d'information, un groupe de 4 ou 8 bits, tout mot pris en charge par la mémoire sera un multiple entier de l'une de ces deux quantités. On choisira de préférence la longueur de 8 bits pour deux raisons: La logique sera sensiblement plus simple dans ce cas puisque le découpage est moins poussé. Par ailleurs, la bonne exploitation de ces possibilités suppose que le programmeur ou le système de programmation soient capables de déterminer la longueur nécessaire pour chaque information; le travail de l'un et la complexité de l'autre seront réduits dans la mesure où l'analyse du format nécessaire à cette information ne requiert qu'une précision moindre.

On retrouve, par ce raisonnement l'octet, introduit il y a quelques années bien que les principales raisons invoquées, à cette époque, en sa faveur aient été d'un autre ordre.

Le raisonnement ne tient pas compte de la répartition des longueurs qui ont été implicitement supposées être toutes équiprobables. En pratique, la probabilité d'apparition des différentes longueurs est très variable. Il faut alors tenir compte de la place réclamée par chaque information en pondérant le résultat par la probabilité de rencontrer cette information.

On a repris le calcul précédent en considérant deux lois de distribution empruntées à une enquête faite par la revue CONTROL ENGINEERING, l'une pour les application scientifiques, l'autre pour la gestion.

Applications de gestion - Répartition des longueurs.

- Mots de 4 bits : caractères numériques..... 30%
- Mots de longueur comprise entre 4 et 8 bits... 30%
- Mots de longueur comprise entre 8 et 16 bits.. 20%
- Mots de longueur comprise entre 16 et 32 bits. 15%
- Mots de longueur comprise entre 32 et 64 bits.. 5%

Sur un total de 1.000 mots, en supposant que la répartition dans chaque tranche est uniforme, on trouve les occupations suivantes pour le stockage:

Valeur de L_0 / Tranche de longueur	1	2	4	8	16	32	64
4bits	1200	1200	1200	2400	4800	9600	19200
4-8 bits	1950	2100	2400	2400	4800	9600	19200
8-16 bits	2500	2600	2800	3200	3200	6400	12800
16-32 bits	3920	4000	4160	4480	5120	5120	10240
32-64 bits	1552	1568	1600	1664	1792	2048	2048
Longueur de la partie adresse	22	20	18	16	14	12	10
Zone correspondante d'adressage	22000	20000	18000	16000	14000	12000	10000
TOTAL	33122	31468	30160	30144	33712	44768	73448

Dans la mesure où les hypothèses faites sur la distribution des longueurs sont assez proches de la réalité, on retrouve les conclusions précédentes : La longueur de l'information unitaire doit être de 2, 4 ou 8 bits.

Application scientifique - Répartition des longueurs.

- Mots de 4 bits : caractères numériques 10%
- Mots de longueur comprise entre 4 et 8 bits..... 5%
- Mots de longueur comprise entre 8 et 16 bits.... 15%
- Mots de longueur comprise entre 16 et 32 bits... 45%
- Mots de longueur comprise entre 32 et 64 bits... 25%

Il semblerait utile d'introduire une tranche supplémentaire entre 32 et 48 bits, la longueur correspondante se prêtant bien à la représentation des nombres en virgule flottante en simple précision c'est à dire avec de 7 à 9 chiffres de mantisse. Il semble que les auteurs aient reporté cette catégorie d'informations dans la tranche comprise entre 16 et 32 bits ce qui est un minimum compte tenu des erreurs d'arondi qui se propagent au cours des calculs.

En prenant les mêmes bases de calcul, on peut établir le tableau suivant:

Valeur de L_0 / Tranche de longueur	1	2	4	8	16	32	64
4 bits	400	400	400	800	1600	3200	6400
4 - 8 bits	325	350	400	400	800	1600	3200
8 -16 bits	1875	1950	2100	2400	2400	4800	9600
16-32 bits	11025	11250	11700	12600	14400	14400	28800
32-64 bits	12125	12250	12500	13000	14000	16000	16000
Longueur de la partie adresse	22	20	18	16	14	12	10
Zone correspondante d'adressage	22000	20000	18000	16000	14000	12000	10000
TOTAL	47750	46200	45100	45200	47200	52000	74000

On trouve encore un optimum pour L_0 égal à 4 ou 8. On remarque cependant que la variation relative du nombre de bits nécessaires lorsque l'on s'éloigne de ces valeurs,est plus faible que dans le cas précédent. Il est possible, en particulier, de prendre une valeur de 16 bits ce qui n'apporte pas de pertes importantes de capacité tout en apportant une simplification aux circuits de décodage d'adresses.

Deux facteurs qu'il est difficile de chiffrer peuvent modifier les résultats obtenus. Ils agissent tous deux dans le même sens de telle sorte que la longueur la plus économique peut s'en trouver modifiée.

Le premier provient de la nécessité de conserver à une information, un format suffisant au cours de toutes les opérations qu'elle subira au cours du traitement; Les réajustements de longueur sont très difficiles à entreprendre en cours de calcul et faisant perdre du temps. Si le programmeur ou le système de programmation affectent à une donnée, au début du traitement, la longueur qui lui convient exactement, elle risque, par la suite de provoquer un dépassement de capacité. Ce dépassement est d'autant plus génant que la zone qui fait suite à la zone allouée à l'information peut être occupée. Chaque dépassement provoque un appel du programme superviseur qui devra décaler une partie importante des informations stockées ou allouer une nouvelle zone à l'information. Dans les deux cas, ce dépassement de capacité va provoquer une perte de temps telle qu'il est préférable de l'éviter en affectant systèmatiquement à une information, une place supérieure à celle qu'elle réclame effectivement.

Le second facteur provient du fait que toutes les positions contenant des informations ne sont pas nécessairement définies par une adresse individuelle dans la zone de désignation. L'adressage indirect, l'emploi de registres d'index qui permettent à l'aide d'une seule adresse explicite, d'accéder à un grand nombre d'informations disposées régulièrement dans la mémoire, permettent une réduction de la dimension de la zone de mémoire affectée à la désignation des informations.

Le premier point conduit à accroitre les dimensions de la zone de stockage tandis que le second permet la réduction de la zone de désignation. Il s'en suit un déplacement de la longueur optimale vers des valeurs plus élevées.

En conclusion, une longueur de 8 bits, pour l'information la plus élémentaire, répond le mieux au critère d'occupation posé. Si cependant on admet qu'il est nécessaire d'ajouter de 10 à 20% de bits supplémentaires à chaque information et que 20 à 30% des adresses effectives ne sont pas explicitées dans la zone de désignation, il est alors plus normal de prendre une longueur de 16 bits. On a considéré comme obligatoire de subdiviser le mot technologique en termes élémentaires dont le nombre soit une puissance de deux afin que , l'adressage étant fait en binaire, le calcul d'adresses conduise toujours à un nombre formé de deux parties indépendantes : le code de désignation du mot technologique d'une part et le code de désignation de la position et de la longueur d'autre part. Il n'est cependant pas nécessaire que la longueur de l'information de base soit une puissance de deux. La discussion entreprise n'a considéré que ces puissances de deux mais elle a permis de constater que la variation de la place occupée en mémoire pour L_0 variant de 1 à 16 ne dépasse pas 10% dans le cas le plus défavorable. Si pour des raisons particulières, le choix d'une longueur de base qui ne soit pas une puissance de deux s'imposait, on serait assuré, dans la limite des bornes indiquées, de ne pas s'éloigner beaucoup de l'optimum.

CHAPITRE I I

ACCROISSEMENT DE LA VITESSE DE LA MEMOIRE : MEMOIRE PARTAGEE .

INTRODUCTION

Dans presque toutes les machines actuelles, les performances de l'unité de traitement sont étroitement liées à celles de la mémoire. Cette situation résulte du fait que les vitesses des éléments logiques ont progressé plus rapidement que celles des éléments de stockage. La différence est accentuée par le fait que la charge des organes de stockage s'accroit en raison:

- De la généralisation des interruptions de programme.
- De l'emploi systématique des langages symboliques qui conduisent à des programmes en langage machine plus lourds et imposent pour un même travail, un nombre plus élevé de consultations de la mémoire.
- Des systèmes de programmation - Moniteurs et superviseurs - qui peuvent occuper près de 30% du temps de la machine et plus de la moitié de la capacité de la mémoire.

La gestion des interruptions comme celle des canaux d'entrées-sorties de même que les compilateurs et les divers programmes du système se caractérisent par le fait qu'ils n'emploient que des instructions très élémentaires mais font un grand emploi de la mémoire.

Devant cet état de choses, deux tendances se manifestent; leurs buts sont identiques bien qu'elles ne concernent pas les mêmes classes de matériels: Il s'agit toujours d'aligner les possibilités de la mémoire centrale sur celles de la logique de traitement.

a) A partir du temps de cycle de la mémoire, considéré comme donnée initiale, on réalise une machine avec une technologie lente ce qui en réduit le prix. On trouve, par exemple, des machines dans lesquelles le stockage se fait en parallèle et le traitement en série. Une telle option ne concerne que des machines de performances faibles ou moyennes.

b) En conservant au temps de cycle technologique des valeurs usuelles, il est possible d'accroitre le débit de l'information soit en allongeant les mots stockés soit en faisant travailler en simultanéité plusieurs blocs de mémoire c'est à dire en partageant la mémoire centrale.

En pratique, les performances et les circuits cablés correspondant à ces solutions ne se situent pas au même niveau. L'allongement des mots de la mémoire n'est pas un phènomème récent mais plutôt une évolution progressive suivant les progrès technologiques et la demande du marché pour des systèmes de plus en plus puissants. Le partage de la mémoire, au contraire, représente une modification brutale par rapport aux solutions précédentes et ne peut être envisagé que pour des systèmes de grosse puissance, actuellement du moins.

A - L'ALLONGEMENT DES MOTS DANS LA MEMOIRE.

On peut évaluer le débit d'une mémoire par le nombre d'informations binaires qui peuvent être pris en charge par une opération de lecture ou d'écriture pendant l'unité de temps. Selon cette définition, le débit est inversement proportionnel au temps de cycle de la mémoire mais il est aussi proportionnel à la longueur du mot technologique. Lorsque l'information traitée est de longueur variable, une distinction est nécessaire entre le débit utile de la mémoire correspondant aux seules informations traitées et le débit réel défini plus haut.

Dans les machines à mots de performances moyennes, le mot de mémoire correspond au mot des registres de traitement. A toute opération de lecture ou d'écriture correspond l'échange d'une information totalement utile. D'ordinaire, cette information constitue un tout pour laquel un seul cycle de mémoire est nécessaire.

Dans les machines à caractères, il faut plusieurs cycles de mémoire pour permettre l'échange d'une information complète. Pour cette raison, la mémoire à caractères est réservée aux petits systèmes. L'évolution entre les machines 1400 et les machines de la série 360 de I.B.M., par exemple, est marquée, entre autres choses, par le fait que la correspondance entre les adresses et les positions physiques de la mémoire ne s'établit pas du tout de la même manière. Dans les machines de la série 1400, les circuits de sélection, commandés par les poids faibles de l'adresse, sélectionnaient un mot de la mémoire. Ce mot pouvait comporter 8, 16 ou 32 bits. Le dernier chiffre de poids fort de l'adresse permettait alors la sélection d'un groupe unique de 8 bits dans le registre de sortie de la mémoire. Une telle solution, par le fait que deux caractères successifs étaient placés dans deux mots nécessairement différents ne permettait pas d'augmenter le débit. Dans les machines de la série 360, ce sont les poids forts qui désignent le mot technologique et les poids faibles qui désignent la position dans le mot. De la sorte, deux caractères d'adresses successives peuvent apparaitre dans le même cycle de lecture et le débit de la mémoire, à performances identiques, s'en trouve augmenté.

Suivant la même évolution que les machines à caractères, les machines à mots voient la longueur des formats technologiques augmenter en même temps que leur puissance de traitement.

Bien qu'il soit un peu prématuré de préjuger de l'accroissement de performances qu'apporte l'allongement du format des mots stockés, les formats étant très diversifiés selon la nature de l'information enregistrée, on peut essayer de chiffrer le bénéfice qui résulte du groupement d'informations indépendantes à l'intérieur d'un même mot technologique.

Il est utile d'extraire de la mémoire, en un même cycle, deux ou plusieurs informations indépendantes lorsque ces informations doivent faire l'objet d'un traitement séquentiel.

Parmi les informations séquentielles issues de la mémoire, on trouve en premier lieu, les programmes. Il peut s'agir aussi bien de programmes exprimés en langage source que de programmes en langage machine. La seule différence porte sur la longueur des groupes binaires correspondants. Une statistique faite sur un groupe de programmes écrits en langage machine pour l'ordinateur de l'ISEN fait apparaitre la proportion suivante de ruptures de séquences:

- Pour les programmes traduits par le compilateur, une instruction de saut pour 12 instructions.
- Pour le compilateur, une instruction de saut pour 8 instructions.

Dans une proportion approximative de 90 à 95% les programmes écrits en langage machine sont séquentiels. Les langages symboliques, sont, quant à eux, lus de manière presque exclusivement séquentielle. Leur traitement ne l'est pas pour autant puisque l'analyse de chaque caractère peut conduire à de nombreuses opérations de recherche, de rangement, de branchements et d'identification imbriquées les unes dans les autres par des instructions de sauts. L'analyse des programmes symboliques ne trouve son intèret dans l'allongement des mots de mémoire que si les instructions d'analyse de ces programmes sont très évoluées. C'est le cas de la machine B 5500 de BURROUGHS qui permet l'analyse de huit caractères contenus dans une position de mémoire en les appelant en séquence à la suite d'une seule opération de lecture. Il s'agit cependant d'un cas relativement exceptionnel. (2)

Une seconde catégorie d'informations traitées dans l'ordre où elles figurent dans la mémoire sont celles qui font l'objet d'opérations d'entrées-sorties.

Il arrive, enfin, surtout dans les travaux de gestion qu'une partie des données traitées soient disposées dans la mémoire dans l'ordre où elles ont à apparaitre.

Pour la recherche du programme, le nombre d'instructions que l'on peut faire figurer dans un mot de mémoire est compris entre 1 et 8. Le format d'une instruction peut varier entre 8 et 32 bits. Le premier cas représente une instruction sans adresse; le second autorise une forme très complète d'instruction avec une partie adresse importante et de nombreuses information complémentaires telles que l'indication d'un accumulateur, d'un registre d'index et d'un adressage indirect. Par ailleurs, la longueur du mot technologique ne peut guère dépasser une centaine de bits. Comme on a intérêt à diviser ce mot en groupes distincts, par une puissance de 2, il sera pratiquement possible de loger 1, 2, 4 ou, au plus 8 instructions par mot.

En admettant que la proportion des consultations de la mémoire réservées à la recherche du programme se situe aux environs de 25 à 30%, on peut chiffrer le gain de temps résultant du groupement d'instructions. Si on prend comme longueur moyenne d'une séquence sans branchement, la valeur de 10 instructions, on trouve:

-Pour deux instructions par mot,un gain de 4,5 cycles pour 30 cycles soit: 15%.

-Pour quatre instructions par mot, un gain de 6,75 cycles pour 30 cycles soit: 22;5%.

-Pour huit instructions par mot, un gain de 7,25 cycles pour 30 cycles soit: 24%.

Ce gain de temps est obtenu au prix d'un effort technologique acceptable: Il suffit, dans le premier cas, par exemple de dédoubler le registre d'ordres et de valider l'une ou l'autre des deux parties par un sélecteur à bascule.

Une solution de ce genre, adoptée sur la machine B 5500 permet le chargement simultané de quatre instructions; le format réduit des instructions de cette machine favorise une telle option.

Il est plus difficile d'évaluer au niveau des données, l'avantage que l'on peut retirer du groupement de plusieurs informations indépendantes à l'intérieur d'un seul mot de mémoire. Si ces informations font effectivement l'objet d'un traitement séquentiel, le bénéfice est du même ordre de grandeur que pour le programme. C'est le cas des listes, de tableaux, des piles et des suites de caractères, groupement que l'on rencontre assez souvent pour que se justifie certaines options technologiques basées sur le principe invoqué.

Le caractère séquentiel des informations traitées a semblé suffisament fréquent pour que, dans la machine de l'ISEN, il apparaisse un indicateur d'adressage séquentiel qui permet de supprimer la partie adresse de l'instruction soit, en moyenne quatre caractères.

B - PARTAGE DE LA MEMOIRE.

Une autre solution qui commence à se faire jour dans les ensembles importants, consiste à multiplier les blocs de mémoire en leur conférant la possibilité de fonctionner en simultanéité. Si, pour une technologie donnée, une mémoire unique peut atteindre un débit donné, plusieurs mémoires fonctionnant simultanément auront un débit incontestablement supérieur. Le gain réalisé est lié au nombre de blocs indépendants et à la gestion de leur simultanéité.

L' Adressage dans les blocs. En supposant que la mémoire, constituée de blocs indépendants, est entièrement adressable, une partie de l'adresse servira à définir le numéro du bloc tandis que l'autre partie définira la position dans le bloc. L'emploi obligatoire des méthodes de calcul d'adresses impose que ces deux parties soient exprimées dans un système de numération unique de telle sorte que toute indexation des adresses permette le chevauchement sur plusieurs blocs sans solution de continuité.

On peut alors se demander s'il faut affecter les poids forts ou les poids faibles de l'adresse à la définition des blocs. En d'autres termes, il s'agit de savoir si le balayage en séquence des adresses explorera la totalité d'un bloc avant de passer au suivant ou si, au contraire, il prendra une position successivement dans chacun des blocs.

Dans le cas où le bloc est défini par les poids forts de l'adresse, en supposant que chacun des utilisateurs travaille dans une zone de positions voisines les unes des autres, on spécialise, pour des séquences longues, chaque bloc en l'attribuant à un ou plusieurs demandeurs. Dans le cas, par exemple, de quatre blocs pouvant être mis à la disposition de quatre canaux demandeurs, on peut imaginer que chaque bloc travaille, pour un travail donné, avec l'un des canaux. Les blocs sont de capacité identique mais les besoins des canaux sont très différents de telle sorte que:

- Ou bien on assigne rigidement un bloc à un canal, tous les blocs ayant une capacité suffisante pour absorber le volume d'informations le plus élevé. On exploite alors très mal la capacité de la mémoire. Une structure de ce type existe dans la machine SABRAC (3) où deux blocs indépendants fonctionnent en simultanéité, l'un connecté à l'unité de traitement, l'autre connecté à une mémoire externe à tambour. Les connexions sont assurées par une bascule. Il ne semble pas que cette organisation ait eu un grand succès en raison de son manque de souplesse et des difficultés de programmation qu'elle entraine.

- Ou bien on autorise le chevauchement sur plusieurs blocs mais il arrive alors que pendant de longues périodes, deux canaux exploitent le même bloc tandis que d'autres blocs seront au repos. Les performances diminuent donc. Par ailleurs, le jeu des priorités entre les demandes issues des divers canaux risque de couper complètement l'accès à la mémoire au canal qui ne dispose que de la priorité la plus faible.

Si ce sont les poids faibles de l'adresse qui définissent le bloc sélectionné, l'attribution des blocs à un demandeur travaillant sur des adresses voisines se fera de manière totalement aléatoire et les conflits de priorité n'auront qu'un caractère très fugitif. Il se peut même que, sous certaines conditions, les circuits de priorité soient totalement supprimés et remplacés par un simple chainage des demandes et un verrouillage des blocs au travail. Cette seconde solution semble donc préférable. C'est celle qui a été adoptée dans la machine CDC 6600 de CONTROL DATA, sans que de leur propre aveu, les logiciens aient été capables de chiffrer exactement le bénefice obtenu.

C - EVALUATION DU DEBIT D'UNE MEMOIRE PARTAGEE.

Puisque l'attribution des blocs aux canaux demandeurs est aléatoire, l'évaluation du débit ne peut être que statistique. Cependant on préfère évaluer maintenant les performances d'une machine sur un long traitement que sur les temps d'exécution d'opérations élémentaires de telle sorte que la définition statistique du débit est aussi significative que celle du temps de cycle d'une mémoire centrale classique.

Soit une mémoire partagée en m blocs simultannés ayant même capacité et même temps de cycle T. Ces mémoires desservent n canaux demandeurs. On suppose que:

- Chaque canal dispose d'une voie de lecture et d'une voie d'écriture qui peuvent être satisfaites ensemble ou séparément sur un cycle complet du bloc concerné.
- Chaque canal dispose de plus, d'une voie adresse qui précise la position concernée.
- Les probabilités d'appel de tous les canaux sont du même ordre. Si certains canaux pouvaient se contenter de débits très inférieurs à ceux des autres, ils peuvent être groupés sur une seule voie d'accès à la mémoire.

On peut supposer, pour faciliter le calcul, que toutes les mémoires sont exactement synchrones et que l'ensemble des demandes qui apparaissent pendant un temps de cycle sont regroupées au début du cycle. Il est impossible, dans cette hypothèse, de déterminer le temps d'accès à la mémoire mais l'évaluation du débit n'est pas modifiée. Le nombre de demandes qui se présentent pendant le temps T n'est pas nécessairement égal au nombre de canaux. Il se peut qu'un canal présente plusieurs demandes tandis qu'un autre n'en présente, en moyenne, qu'une fraction.

Lorsqu'un cycle vient de se terminer, tous les blocs sont libres. Les demandes sont alors examinées dans l'ordre des priorités décroissantes. Soient D1, D2, D3, ... Dp, ces demandes.

- D1 a une probabilité 1 d'être satisfaite.
- La demande D2 dispose de m - 1 blocs et la probabilité de voir D2 satisfaite est:

$$P_2 = \frac{m-1}{m}$$

- La demande D3 aura la probabilité $P'_3 = \frac{m-2}{m}$ d'être satisfaite si D2 a été satisfaite. Dans le cas contraire, D3 aura la probabilité de satisfaction $P''_3 = \frac{m-1}{m}$

$$P_3 = P'_3 . P_2 + P''_3 . (1 - P_2)$$

$$P_3 = \frac{m-2}{m} \cdot \frac{m-1}{m} + \frac{1}{m} \cdot \frac{m-1}{m}$$

$$P_3 = \left(\frac{m-1}{m}\right)^2$$

- On trouve que la probabilité de satisfaction de la demande D4 est de la forme:

$$P4 = \left(\frac{m-1}{m}\right)^3$$

La loi de formation des probabilités successives est récurrente et la probabilité de satisfaction de la demande i s'obtient à partir de la probabilité de satisfaction de la demande d'ordre (i - 1):

$$P_{i-1} = \left(\frac{m-1}{m}\right)^{i-2}$$

- La probabilité P_i est la somme de deux termes représentant le succès ou l'échec de la demande de priorité immédiatement supérieure. Si cette dernière demande s'est soldée par un échec, la demande de rang i conserve la même probabilité puisque le nombre de blocs reste identique. Si cette demande a été un succès, le nombre des blocs disponibles a diminué de 1. On constate qu'il existe une probabilité non nulle de satisfaction pour une demande dont le rang serait supérieur au nombre de blocs.

$$P_i = P_{i-1}(1 - P_{i-1}) + P_{i-1}(P_{i-1} - 1/m)$$

$$P_i = P_{i-1}^2 - P_{i-1}^2 + P_{i-1} \cdot \frac{m-1}{m}$$

$$P_i = P_{i-1} \cdot \frac{m-1}{m} = \left(\frac{m-1}{m}\right)^{i-1}$$

Si, à l'instant initial, p demandes se sont présentées, la probabilité globale sur l'ensemble des réponses effectivement fournies sera la somme des probabilités affectées à chacune des demandes. Cette somme représente le débit de la mémoire:

$$D_p = \sum_{i=0}^{i=p-1} \left(\frac{m-1}{m}\right)^i$$

Expression que l'on peut encore mettre sous la forme:

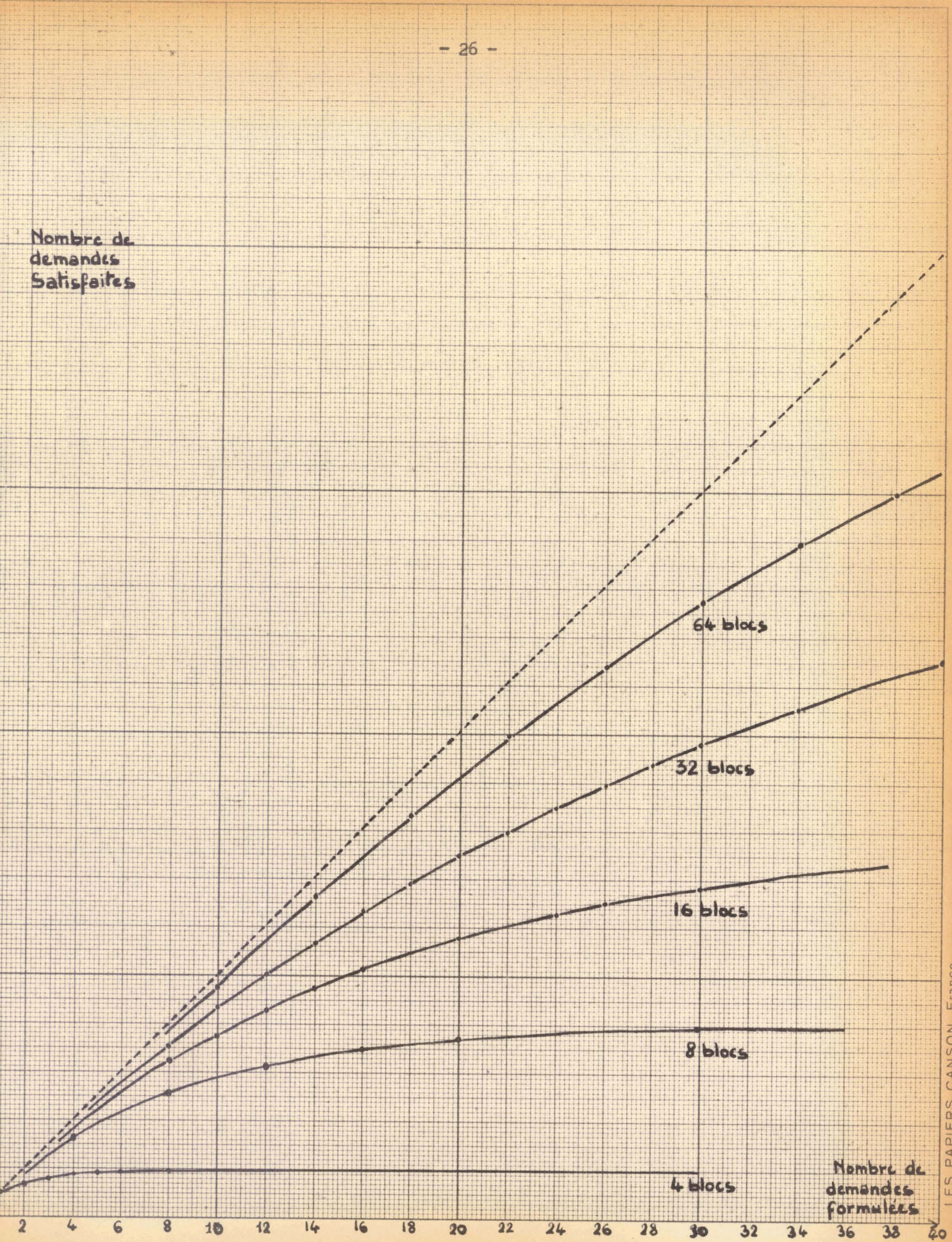
$$D_p = \frac{1 - \left(\frac{m-1}{m}\right)^{p+1}}{1 - \frac{m-1}{m}}$$

Lorsque p tend vers l'infini, le débit tend vers m, valeur maximum obtenue si tous les blocs sont mis au travail. Il est possible de présenter plus de demandes qu'il n'existe de blocs dans la mémoire; toute demande, quel que soit le nombre des demandes prioritaires desservies avant elle, conserve une chance d'être servie. Cette remarque concerne tout particulièrement les machines fonctionnant en temps partagé et capables de se mettre à la disposition d'un grand nombre de périphériques très lents. Pour ceux-ci, une attente pouvant se prolonger sur plusieurs centaines de cycles de la mémoire n'a guère d'importance. Ils peuvent donc se contenter d'une probabilité de satisfaction très faible.

Application. Si on admet que les poids faibles de l'adresse définissent le bloc, l'adressage étant binaire, le nombre de blocs indépendants doit être une puissance de deux. Le calcul du débit en fonction du nombre de blocs a été fait pour 2, 4, 8, 16, 32 et 64 blocs.

Le nombre de demandes, qui représente la variable varie de 1 jusqu'à une valeur telle que la probabilité de satisfaction soit inférieure à 0,01.

Les courbes de la page suivante résument ces résultats.



LES PAPIERS CANSON France

D - ORGANISATION PRATIQUE D'UNE MEMOIRE PARTAGEE.

S'il existe n canaux indépendants et m blocs de mémoire, il faut assurer à chaque canal la possibilité d'accéder à chacun des blocs ce qui réclame théoriquement n.p aiguillages. Chacun de ces aiguillages est triple puiqu'il doit commuter d'une part la partie adresse désignant la position dans le bloc et d'autre part les voies de lecture et d'écriture.

1) Commutation purement spaciale.

Un réseau d'interrupteurs permet, à tout instant de relier n'importe quel canal à n'importe quel bloc. La logique de priorité commande les portes placées du coté du canal tandis que la partie adresse qui désigne le bloc dans la demande commande les portes placées du coté des mémoires.

Cette solution devient rapidement très lourde: Par exemple, pour 8 blocs de 8.000 mots soit une capacité totale de 64.000 mots de 32 bits chacun, il faut, pour desservir 4 canaux:

- Pour les adresses (13 bits):13 x 8 x 4 soit 416 portes.
- Pour la voie d'écriture: 32 x 8 x 4 soit 1024 portes.
- Pour la voie de lecture: 32 x 8 x 4 soit 1024 portes.

Au total, cette solution réclame 2.464 portes ce qui est très lourd. Elle présente cependant l'avantage de permettre un fonctionnement totalement asynchrone.

2) Commutation spacio-temporelle.

On peut obtenir une économie significative de circuits par la mise en oeuvre d'une solution en apparence moins souple: La répartition non seulement spaciale mais aussi temporelle des connexions entre canaux et blocs. Chaque canal débouche par un premier réseau de portes sur trois unités d'échange, l'une pour les adresses, les deux autres pour la lecture et l'écriture. Ces unités d'échange sont, à leur tour reliées par un second réseau de portes aux blocs de la mémoire. Il suffit de n + p aiguillages triples. Dans le cas de la mémoire prise en exemple plus haut, il faut:

- Pour les portes du coté des canaux: (13 + 32 + 32).4 soit 308 portes.
- Du coté des blocs: (13 + 32 + 32).8 soit 616 portes.

Au total, cette solution ne réclame que 924 portes.

Cette économie sur les circuits d'aiguillage ôte, en contre-partie, la possibilité d'une simultanéité totale des blocs. A un instant donné, il ne peut s'établir qu'une seule connexion entre un canal et un bloc.

Pour que le débit ne soit pas diminué, l'unité d'échange doit être extrêmement rapide. Si on désire qu'elle ne puisse, en aucun cas, réduire ce débit, il faut qu'elle soit capable, pendant le temps T de cycle d'un bloc, de satisfaire chacun des autres. Si la mémoire comporte m blocs, le temps total d'intervention de l'unité d'échange sera au maximum égal à T/m. En fait, il faut faire la différence entre les opérations de lecture et les opérations d'écriture. En supposant que chaque bloc de la mémoire dispose de son propre registre d'entrée-sortie, une opération de lecture va se traduire par une demande suivie, après un certain délai, de l'acquisition de l'information demandée dans le registre d'entrée-sortie. Il est impossible de figer les portes de commutation pendant ce délai. Les ordres de commutation doivent donc être dédoublés, les demandes d'écriture provenant du canal en même temps que l'information à écrire et les demandes de lecture étant émises par les blocs lorsqu'ils ont obtenus l'information.

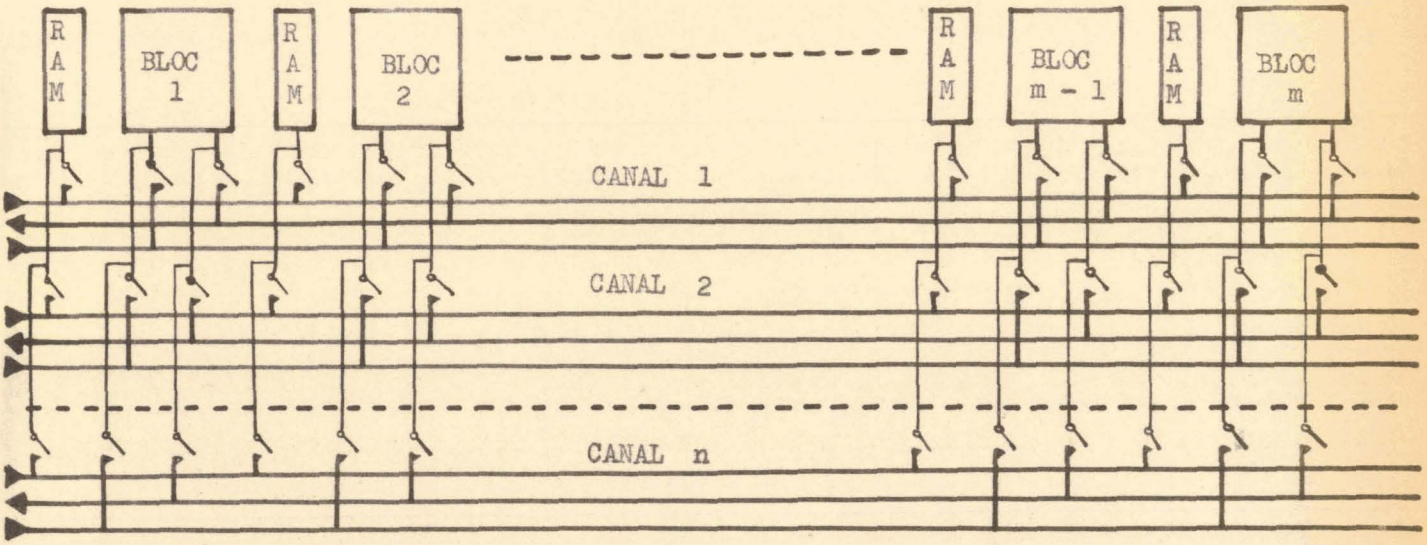
Le temps de cycle de l'unité d'échange, qui sera pris comme référence dans toute la suite, représente donc le temps de base de la mémoire partagée. Pendant un cycle on doit pouvoir faire les opérations suivantes:

- Choisir une demande provenant de l'un des canaux, déterminer s'il s'agit d'une lecture ou d'une écriture.
- Dans le cas d'une demande d'écriture, décoder l'adresse du bloc, examiner s'il est libre et dans l'affirmative transférer l'adresse et l'information vers le bloc.
- Dans le cas d'une demande de lecture, procéder aux mêmes opérations, sans transfert d'information.
- Lorsqu'un bloc signale la fin d'une opération de lecture, prendre le contenu du registre d'entrée-sortie de ce bloc et le transférer vers le canal demandeur.

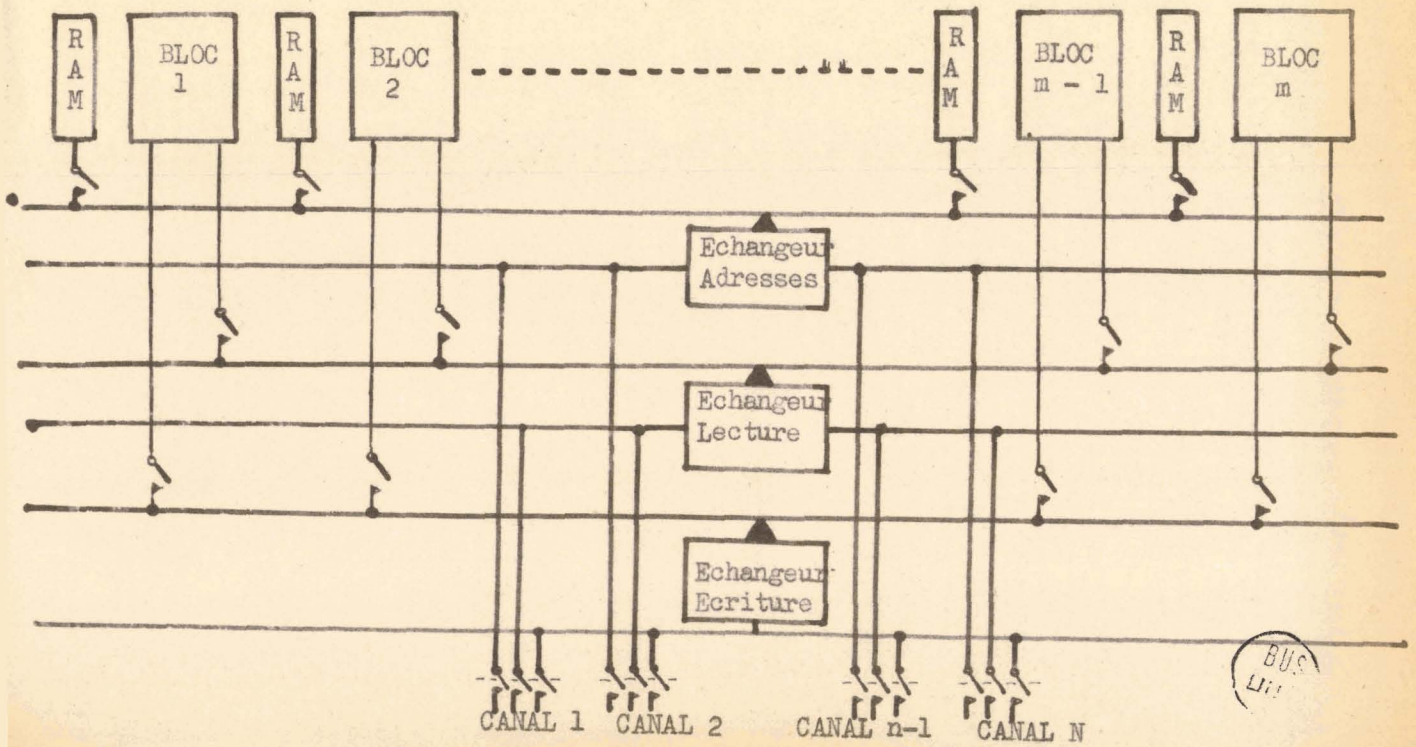
Chaque bloc au travail, que ce soit pour une lecture ou pour une écriture réclame l'intervention de l'unité d'échange une fois en début de cycle. Les blocs qui poursuivent une opération de lecture réclament son intervention une fois de plus au cours de leur cycle. En admettant qu'il existe autant d'ordres de lecture que d'ordres d'écriture, il faut réduire le temps de cycle élémentaire à $T/(1{,}5.m)$. En pratique, il est très râre que les m blocs de la mémoire soient tous au travail et on peut tolérer un cycle un peu plus long. Par la suite, on prendra T/m comme durée du cycle élémentaire.

On remarquera que cette organisation évite les conflits pour la seconde demande liée à une lecture puisqu'une seule opération ne peut être lancée, une seule demande peut se présenter pour le transfert vers le canal.

COMMUTATION PUREMENT SPACIALE ENTRE BLOCS ET CANAUX



COMMUTATION SPACIO-TEMPORELLE ENTRE BLOCS ET CANAUX



La comparaison des seules voies d'aiguillage ne permet pas de justifier complètement un choix. La seconde solution qui semble, de ce seul point de vue, plus économique, exige la réalisation d'un registre d'information pour chaque bloc de la mémoire. Ce registre n'est pas obligatoire dans la solution purement spaciale où on peut admettre que le canal fournit à la mémoire, aussi longtemps qu'elle en a besoin, les informations nécessaires.

Une estimation des circuits nécessaires tant pour la commutation que pour le stockage intermédiaire a été faite pour diverses configurations de mémoire:

A) Mémoire de 16.000 mots de 16 bits partagée en 4 blocs;2 canaux.
B) Mémoire de 32.000 mots de 24 bits partagée en 8 blocs;4 canaux.
C) Mémoire de 64.000 mots de 32 bits partagée en 8 blocs;4 canaux.
D) Mémoire de 128.000 mots de 64 bits partagée en 32 blocs et 8 canaux.

L'estimation a été faite en éléments logiques NAND; une bascule est comptée comme deux éléments logiques. L'emploi de circuits intégrés semble le plus normal. Dans ce cas, il est possible d'éviter la réalisation des circuits de stockage pour lesquels des éléments intégrés à grande échelle fournissent une solution immédiate, sous forme de mémoires complètement cablées.

Les tableaux de la page suivante détaillent les besoins de chaque solution pour les quatre configurations proposées. Dans des mémoires de capacité moyenne, on trouve que la solution purement spaciale est plus économique. Au niveau de la configuration B), la solution I peut encore être retenue car elle présente des avantages qui compensent l'exédent de circuits nécessaires. Lorsque la mémoire présente une capacité importante et que la longueur des mots est élevée, il est pratiquement obligatoire de choisir la solution II.

SOLUTION I

	Adresses	voies d'entrée	voies de sortie	stockage	Logique de choix	TOTAL
MEMOIRE A	144	192	160	16	16	528
MEMOIRE B	490	960	864	64	48	2.426
MEMOIRE C	520	1.280	1.152	64	48	3.064
MEMOIRE D	3.456	18.434	16.898	512	320	39.620

SOLUTION II

	adresses		voie d'entrée		voie sortie		Logique	
	aiguillage	stockage	aiguillage	stockage	aiguill.	stock	de choix	TOTAL
MEMOIRE A	72	120	96	160	96	32	32	608
MEMOIRE B	144	216	288	432	288	48	102	1.518
MEMOIRE C	156	234	384	576	384	64	102	1.900
MEMOIRE D	480	792	2.560	4.224	2.560	128	832	11.576

E - LA MISE EN OEUVRE DES PRIORITES.

Lorsqu'un bloc au travail fait l'objet d'une demande de la part d'un canal autre que celui qu'il desservit, il est impossible d'interrompre son cycle, même si la nouvelle demande jouit d'un niveau de priorité supérieur. On ne peut, en effet, autoriser des appels en cascade d'un même bloc avec des traitements partiellement satisfaits qui se trouveraient imbriqués les uns dans les autres. Si même la technologie le permettait, il faudrait alors mémoriser les éléments nécessaires à la reprise du cycle interrompu dans une mémoire nettement plus rapide que celle dont elle permet la gestion. Or cette mémoire devrait disposer d'une capacité importante puisqu'il faut y stocker: L'adresse désignée dans le bloc interrompu, l'information qu'il fallait y déposer s'il s'agissait d'une écriture, les numéros du bloc et du canal intéressés et une information indiquant l'état d'avancement du cycle au moment de l'interruption. Dans le cas le plus défavorable, s'il existe n canaux, il faut pouvoir stocker n - 1 fois ces informations. En ce sens, le jeu des priorités est plus simple que celui qui, par exemple, régit le travail des organes périphériques.

On pose donc, comme principe initial, que lorsqu'un bloc est mis au travail, il poursuit son cycle quels que soient les événements extérieurs. Outre que, comme cela vient d'être montré, il est difficile de procéder autrement, le fait que les demandes de concours sont permanentes et que les temps d'occupation d'un bloc sont du même ordre de grandeur que les temps les plus élémentaires du traitement, conduit à adopter une logique de choix aussi simple que possible.

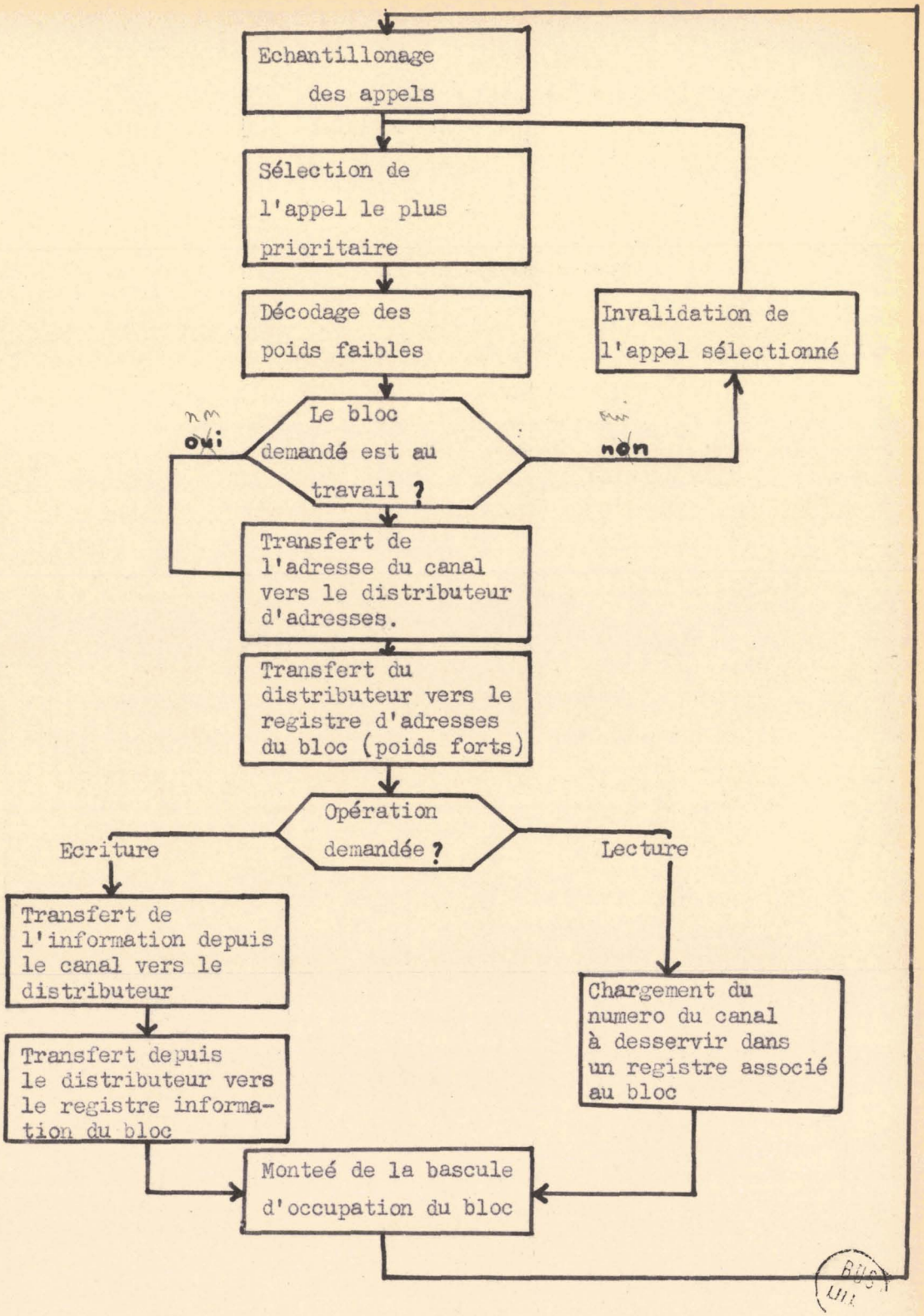
Deux organisations sont possibles selon que l'on prévoit l'existence ou non d'une file d'attente.

1) Mémoire sans file d'attente.

On suppose qu'un canal maintient sa demande tant qu'il n'a pas obtenu satisfaction et qu'il ne peut formuler une autre demande, même sur un bloc différent pendant ce temps. Le stockage temporaire des demandes se fait dans le canal lui-même et il n'existe qu'une seule source de demandes ce qui conduit à une logique de choix unique.

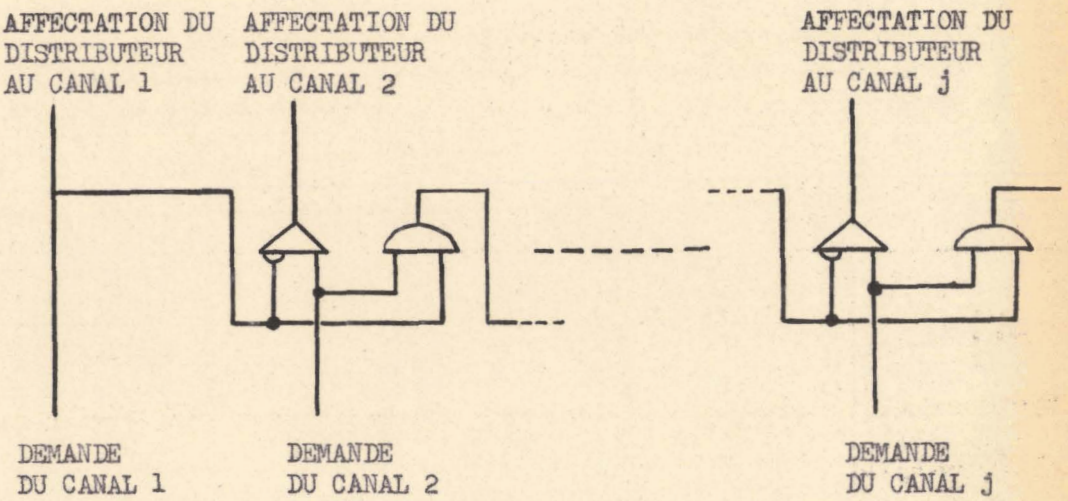
L'organigramme de la page suivante indique la suite des opérations élémentaires à entreprendre. La première boucle qui permet de choisir la demande à traiter doit être très rapide puisqu'elle peut être parcourue m fois. Il est possible de faire un pré-traitement des demandes qui désignent un bloc occupé afin de les invalider à priori.

DIAGRAMME DES OPERATIONS DE TRAITEMENT DES APPELS



Si l'organisation des priorités se trouve simplifiée par le fait que l'examen des demandes est assuré une fois pour toutes et sans appel au début de chaque cycle élémentaire, elle reste néammoins délicate à réaliser dans la pratique en raison de la grande vitesse de décision requise. La décision d'affecter un bloc de mémoire à l'un des canaux demandeurs ne doit requérir qu'une fraction du temps de cycle.

La solution par circuits logiques statiques, telle qu'elle pourrait être réalisée selon le schéma de la figure ci-dessous, ne peut convenir car les retards apportés lorsque la chaine commute, permettraient à une demande non prioritaire d'être satisfaite en même temps qu'une autre de priorité supérieure.



Sens des priorités décroissantes.

Il faut alors figer, à un instant donné, les appels présents et ne faire porter la sélection que sur eux. Si un nouvel appel se présente pendant que le choix s'effectue, il ne peut plus être pris en compte car il exigerait une reprise complète de l'opération. On peut obtenir ce résultat en échantillonnant les demandes externes, au début du cycle élémentaire et en opérant le choix sur les seules demandes échantillonnées.

La priorité peut s'obtenir facilement à l'aide d'une série de retards d'autant plus longs que le niveau de priorité est plus bas. Dès qu'une demande a vu s'écouler le temps de retard qui lui est assigné, elle bloque les autres. Pour éviter de faire plus d'une fois la boucle de sélection, on peut n'autoriser l'échantillonnage que pour les demandes qui désignent des blocs libres. Cette disposition exige l'analyse de la partie adresse désignant le bloc, en amont des circuits de sélection; il faut donc prévoir autant de circuits qu'il existe de canaux.

2) Mémoire avec file d'attente.

Les canaux qui ne peuvent accéder immédiatement au bloc dont ils ont besoin, doivent se débarasser de la demande qui les occupe avant de pouvoir en formuler une nouvelle. Dans le cas où les échanges entre la mémoire et les canaux ne sont pas obligatoirement séquentiels, on peut autoriser ce mode de fonctionnement en créant une mémoire tampon supplémentaire. Cette mémoire est la file d'attente. Les échanges non séquentiels concernent en premier lieu deux types d'informations: Les informations échangées avec les organes périphériques qui sont regroupés sur un canal d'accès à la mémoire et les informations qui doivent être inscrites dans la mémoire. Il se peut que les données à traiter et mêmes les instructions ne soient pas non plus appellées en séquence, soit qu'il existe plusieurs unités de traitement fonctionnant en parallèle,soit que des files d'anticipation assurent ultérieurement la mise en ordre des lectures demandées à la mémoire.

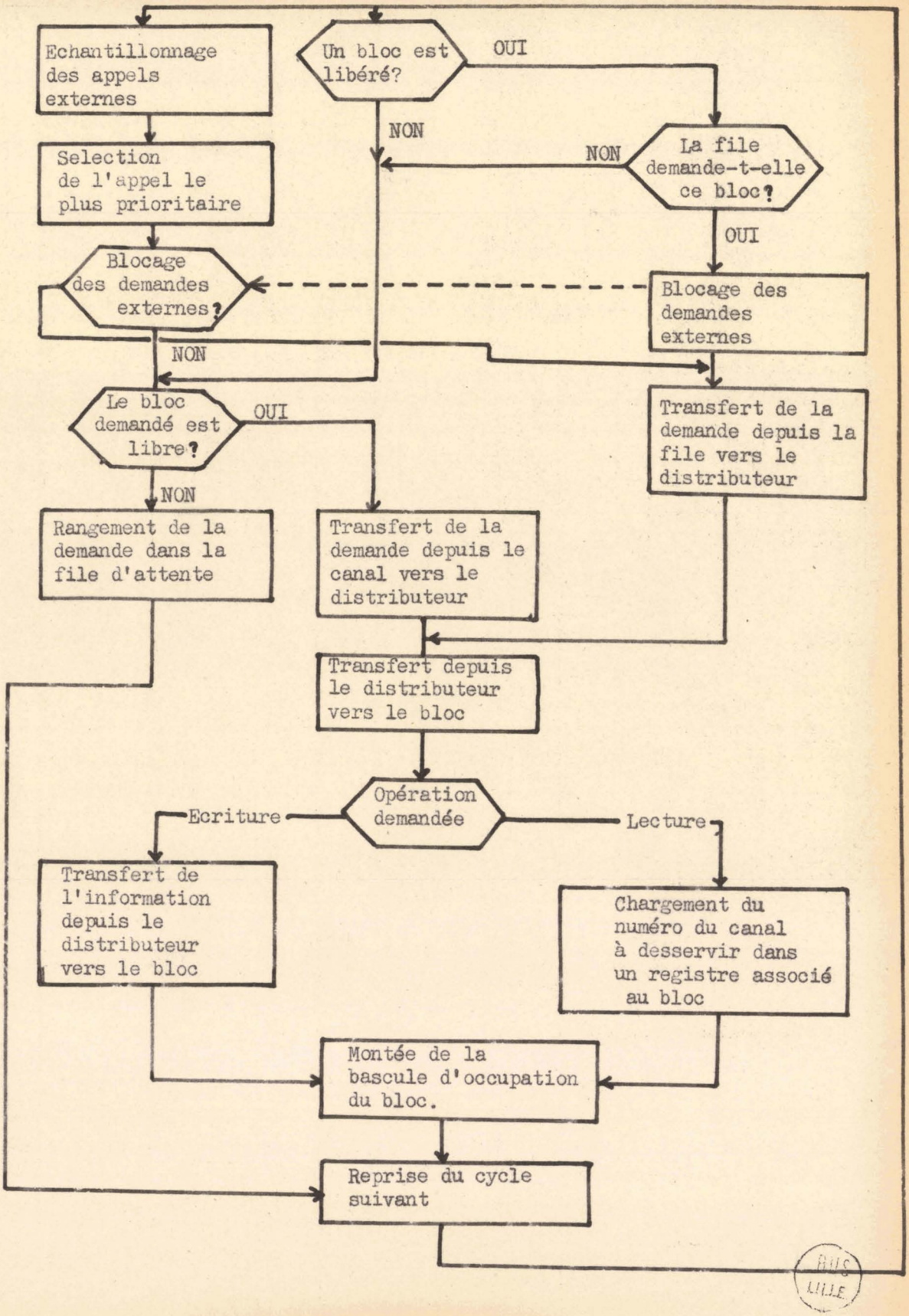
La mémoire tampon est organisée en file d'attente car, ici encore, pour des questions de vitesse, la règle qui préside au choix des demandes doit être simple: La priorité d'accés à un bloc donné est accordée à la demande la plus ancienne désignant ce bloc. De plus, cette même règle d'ancienneté confère toujours la priorité à la file d'attente sur les demandes directes en provenance des canaux.

Il serait possible d'établir des règles de priorité plus adaptées aux necéssités de chaque canal mais il convient à nouveau de remarquer que le caractère volontairement aléatoire des occupations ne garantit jamais à un canal qu'une priorité relative sur les autres et, en aucun cas,une priorité absolue sur les travaux en cours. Dans ces conditions, il est préférable d'assurer autrement la réduction du temps de réponse, en particulier par l'emploi de files d'anticipation.

L'organigramme de la page suivante indique le déroulement d'un cycle élémentaire. On suppose qu'une logique, qui peut être statique, fournit pour chaque bloc, la demande la plus ancienne le concernant.

La gestion de la file d'attente peut être simplifiée par la remarque suivante: Si une demande se trouve placée dans la file d'attente, c'est qu'au moment où elle s'est présentée, le bloc qu'elle désignait se trouvait occupé. L'interrogation de la file peut donc n'avoir lieu que lorsque un bloc termine son cycle et se trouve libéré.

ORGANIGRAMME DE GESTION DES PRIORITES AVEC UNE FILE D'ATTENTE.



BUS LILLE

CHAPITRE I I I

COMPORTEMENT DE LA MEMOIRE EN PRESENCE DES CIRCUITS QUI L'EXPLOITENT.

INTRODUCTION

Les calculs précédents ont permis d'évaluer le débit de la mémoire en fonction du nombre de blocs, du temps de cycle de chacun d'eux et du nombre de demandes présentées pendant de temps de cycle. On a supposé que toutes les demandes étaient groupées à l'instant initial et qu'alors étaient réglés, pour un cycle complet, tous les conflits de priorité. Un tel modèle, particulièrement simple ne peut donner une idée précise de ce qui se passe dans la réalité où les demandes de concours présentées à la mémoire et les mises en route de chacun des blocs doivent se chevaucher pour réduire les temps d'attente. Le résultat obtenu pour le débit reste valable mais la seule connaissance de ce débit est insuffisante.

Une autre information est nécessaire: Le temps d'attente pour l'obtention d'une information définie par son adresse et, dans le cas de l'écriture, le temps qu'il faudra conserver cette information avant qu'elle ne prenne place dans la mémoire. Dans une mémoire partagée, ce temps d'attente est plus long que dans une mémoire unique puisqu'au temps de cycle propre de la mémoire, il faut ajouter un temps d'attente si le bloc demandé n'est pas libre. Pour certaines configurations, ce temps d'attente peut être bien supérieur au temps de cycle technologique.

Il faut uitliser un modèle dynamique de la mémoire c'est à dire capable de décrire son évolution instantannée. Dans un premier temps, on considèrera le fonctionnement de la mémoire en présence d'une distribution dans le temps des demandes dont on tentera de justifier le choix. On pourra en déduire le taux d'utilisation des blocs, le taux des demandes immédiatement acceptées et celui des demandes refusées. Pour ces dernières, une file d'attente sera nécessaire et sa longueur peut faire l'objet d'une étude statistique fournissant le taux d'occupation de chacune de ses positions.

La file d'attente se comporte alors comme un demandeur nouveau, prioritaire sur les autres et il se crée une nouvelle distribution des travaux dans la mémoire. Le débit de la mémoire est lié à la longueur de cette file d'attente et on peut essayer de trouver la longueur optimum.

On peut enfin essayer de chiffrer le temps qui s'écoule entre le moment où une demande est présentée à la mémoire depuis l'extérieur et le moment où cette demande est totalement satisfaite. Comme ce temps est différent des temps qui caractérisent habituellement le fonctionnement d'une mémoire (temps d'accès, temps de cycle), on l'appelera le temps de service.

- REFERENCES A DES MATERIELS EXISTANTS OU EN PROJET.

La multiplication ou, comme l'on dit parfois, le multiplexage de la mémoire est devenu une nécessité pour des machines puissantes disposant de circuits de décisions et de calculs performants et dans lesquelles ces circuits fonctionnent en simultanéité. Les matériels existant sont peu nombreux ; certains sont encore à l'état de projets et il est souvent difficile d'apprécier leurs performances.

La première machine qui ait disposé d'une mémoire partagée est la 6600 de CONTROL DATA. De l'aveu même de ses concepteurs, les performances de la machine n'avaient pu être précisées à priori. Le but, d'ailleurs parfaitement atteint, était de réaliser un ensemble plus puissant que tout autre matériel existant à l'époque et pour lequel il existait un marché. La 6600 comporte une mémoire de 131.000 mots divisée en 32 blocs indépendants. Quatre canaux indépendants y ont accès. Les risques de conflits sont donc faibles. Leur règlement est assuré simplement sur le principe de l'antériorité : la demande la plus ancienne est la première servie.

A l'époque où la 6600 était présentée sur le marché, l'Université de Colombia (4) entreprenait l'étude d'une petite machine, très rapide à diodes tunnel. Cette machine devait suppléer par sa vitesse au petit nombre de ses circuits de traitement : la fréquence d'horloge était de 250 Mégacycles, c'est-à-dire plus de dix fois supérieure à celle de la 6600. Les concepteurs se heurtaient au problème de la mémoire et ont proposé de la diviser en 4 blocs indépendants. L'étude du fonctionnement de l'ensemble a été faite par simulation avec les données suivantes :

- Temps de cycle de la mémoire : Une micro-seconde.
- Distribution périodique des demandes ; chaque demande suivant la précédente demande satisfaite de 0,25 micro-seconde. L'absence de file d'attente dans le programme de simulation impose un traitement rigoureusement séquentiel des demandes.

Le résultat obtenu est un débit moyen de 2,2 mots par micro-seconde.

Il est aisé de contrôler ce résultat par un calcul : le débit moyen de la mémoire est égal au nombre moyen de blocs occupés. Soit X ce nombre. Il suffit d'écrire que pour X, la probabilité de libération d'un bloc à chaque instant où se produit une demande est égale à la probabilité de réoccupation de l'un des blocs libres :

$$\text{Prob. de libération} = \frac{X}{m} = \frac{X}{4} = \text{Prob. de reprise} = \left(\frac{m-X}{m}\right).\left(\frac{m-X}{m}\right) + \frac{X}{m}\left(\frac{m-X+1}{m}\right)$$

On trouve X = 2,28 mots par micro-seconde.

On verra plus loin que cette configuration, en présence d'une distribution aléatoire et non plus périodique des demandes fournit un débit de 2 mots par micro-seconde. Il est normal qu'une distribution périodique des demandes fournisse un meilleur résultat.

L'étude la plus récente sur la question de la mémoire partagée semble être celle qui a été faite à propos de la machine IBM 360 modèle 91 (5),(6), capable non seulement de fonctionner en multi-programmation, mais aussi d'effectuer des traitements concourants. De tels objectifs rendaient nécessaires l'emploi d'une mémoire partagée. Dans les diverses versions proposées de la machine, le nombre de blocs peut être de 16, 8 ou 4 ; dans ce dernier cas, les circuits de traitement, qui sont identiques pour toutes les versions, semblent ne pas pouvoir se satisfaire du débit limité de la mémoire. Cette mémoire est divisée en deux parties : La mémoire principale et son extension. Toutes deux sont capables, dans leur version la plus performante d'un débit théorique de 172 megabyte par seconde. Ces chiffres supposent que les 16 blocs de chaque mémoire sont tous au travail, ce qui est très improbable.

L'étude de la mémoire et de ses circuits associée a été conduite par simulation. La donnée initiale étant le cycle élémentaire de l'unité centrale (60 nanosecondes), la simulation a permis d'indiquer pour diverses vitesses technologiques de la mémoire, le débit et le temps d'accès.

Pour la vitesse retenue (temps de cycle complet de 750 nanosecondes), et pour un taux de demandes maximum, (270 megabytes par seconde soit 30.10^6 mots de 72 bits par seconde, soit encore 22,5 mots par temps de cycle), les débits sont de

- 3,1 mots pour 4 blocs indépendants,
- 6 mots pour 8 blocs,
- 9,2 mots pour 16 blocs,
- 12,3 mots pour 32 blocs.

En dehors de ces résultats, la description reste presque entièrement qualitative et certains choix ne sont guère explicités.

A - OCCUPATION DE LA MEMOIRE POUR UN TAUX DONNE DE DEMANDES.

Le système formé de la mémoire et des canaux qui y ont accès, constitue un ensemble où l'élément déterminant du fonctionnement est l'apparition d'une demande de concours à la mémoire.

Pour définir le comportement de cette mémoire, il faut connaitre la loi de distribution des demandes dans le temps.

On prendra comme référence de temps, la durée T du cycle technologique d'un bloc et on fera l'hypothèse suivante:

La distribution des demandes dans le temps est aléatoire et suit un processus de POISSON. Autrement dit, la probabilité pour que j demandes se présentent pendant le temps t, est donnée par l'expression:

$$P_j(t) = \frac{(k.t)^j \cdot e^{-k.t}}{j!}$$

k s'appelle le taux moyen de demandes.

Le choix de ce modèle de distribution aléatoire des demandes peut se justifier par les remarques suivantes:

- La probabilité $P_j(t)$ ne dépend pas de l'instant initial choisi mais seulement de l'intervalle de temps t considéré.
- Une mémoire partagée ne trouve son intérêt que si elle dessert un nombre relativement élevé d'organes indépendants. Dans ces conditions, les demandes de chaque organe se présentent indépendamment de celles des autres ce qui contribue à rendre leur apparition aléatoire.
- La probabilité pour que deux demandes se présentent en même temps est nulle. Ceci résulte du fait que le système qui gère la mémoire ne peut traiter qu'une demande à la fois. Si deux demandes sont simultannées, le jeu des priorités accordera le passage à la plus prioritaire tandis que la prise en charge de l'autre ne sera faite qu'au cycle suivant.

En prenant T comme unité de temps, on appelera dt le temps de cycle du système de gestion et d'attribution des blocs. On suppose que dt est très petit vis à vis de T.

Par la suite, on assimilera dt à une différentielle.

La probabilité pour qu'une demande se présente pendant l'intervalle de temps dt est : k. dt

$$P_1 (dt) = k . dt$$

Les probabilités pour que 2, 3, ... n demandes se présentent pendant l'intervalle de temps dt sont:

$$P_2 (dt) = P_3 (dt) = ... = P_n (dt) = 0$$

Il en résulte que la probabilité pour qu'aucune demande ne se présente pendant le temps dt est:

$$P_0 (dt) = 1 - P_1 (dt) = 1 - k . dt$$

Lorsqu'une demande se présente, elle fait l'objet d'un contrôle pour déterminer si le bloc qu'elle réclame est disponible ou non. Si, à l'instant de la demande, x blocs sont occupés, si la mémoire comporte m blocs:

- La probabilité de satisfaction est: $\frac{m - x}{m}$
- La probabilité de refus est: $\frac{x}{m}$

Enfin, toujours pendant le temps dt, la probabilité pour qu'un bloc occupé préalablement se trouve libéré est:

$$\sum_{1}^{m} \frac{x}{m} . \frac{dt}{T} = \frac{x . dt}{T}$$

Il est alors possible d'exprimer la probabilité pour que à un instant donné, x blocs de la mémoire soient occupés: Supposons qu'à l'instant t, la probabilité pour que x blocs soient occupés s'écrive $P_x (t)$; on peut évaluer la probabilité pour que x blocs soient occupés à l'instant t + dt.

La probabilité $P_x (t + dt)$ résulte de diverses possibilités:

a) - A l'instant t, x blocs étaient occupés.
   - Aucune demande ne s'est présentée ou celles qui se sont présentées n'ont pu être satisfaites.
   - Aucun bloc n'a été libéré pendant le temps dt.

b) - A l'instant t, x - 1 blocs étaient occupés.
   - Une demande s'est présentée et a été satisfaite.
   - Aucun bloc n'a été libéré.

c) - A l'instant t, x + 1 blocs étaient occupés.
   - Aucune demande ne s'est présentée ou celles qui se sont présentées n'ont pu être satisfaites.
   - Un bloc a été libéré.

d) - A l'instant t, x blocs étaient occupés.
   - Une demande s'est présentée et a été satisfaite.
   - Un bloc a été libéré.

On remarquera que la libération simultannée de plusieurs blocs est impossible puisque le processus des libérations est l'image, retardée d'un temps T, des attributions aux demandes et que, selon l'hypothèse faite plus haut, il ne peut se faire qu'une seule attribution par cycle dt.

Il vient alors:

$$P_x(t+dt) = P_x(t).\left[(1-k.dt)+k.dt.\frac{x}{m}\right].\left(1-\frac{x.dt}{T}\right) \quad (a)$$

$$+ P_{x-1}(t).\left[k.dt\ \frac{m-x}{m}\right].\left(1-\frac{x.dt}{T}\right) \quad (b)$$

$$+ P_{x+1}(t).\left[(1-k.dt)+k.dt\,\frac{x}{m}\right].\frac{x.dt}{T} \quad (c)$$

$$+ P_x(t).\left[k.dt\ \frac{m-x}{m}\ .\ \frac{x.dt}{m}\right] \quad (d)$$

En effectuant les produits:

$$P_x(t+dt) =$$

$$P_x(t).\left[1-\frac{x.dt}{T}-\frac{m-x}{m}k.dt+\frac{m-x}{m}k\frac{x}{T}dt^2+\frac{m-x}{m}k\frac{x}{T}dt^2\right]$$

$$+P_{x-1}(t).\left[\frac{m-x}{m}k.dt-\frac{m-x}{m}k\frac{x}{T}dt^2\right]$$

$$+P_{X+1}(t).\left[\frac{x}{T}dt-\frac{m-x}{m}k\frac{x}{T}dt^2\right]$$

Si on néglige les termes en dt^2 en considérant qu'ils sont faibles vis à vis des termes en dt, il vient:

$$P_x(t+dt) = P_x(t).\left[1-\frac{x}{T}dt-\frac{m-x}{m}k.dt\right]+P_{x-1}(t).\left[\frac{m-x}{m}k.dt\right]+P_{x+1}(t).\left[\frac{x}{T}dt\right]$$

expression que l'on peut encore écrire sous la forme:

$$\frac{P_x(t+dt)-P_x(t)}{dt} =$$

$$-\left[\frac{x}{T}+\frac{m-x}{m}k\right].P_x(t)+\left[\frac{m-x}{m}k\right].P_{x-1}(t)+\left[\frac{x}{T}\right].P_{x+1}(t)$$

On aboutit enfin au système d'équations différentielles (1) en considérant que le terme dt est effectivement une différentielle.

$$P'_x = -\left[\frac{x}{T}+k\frac{m-x}{m}\right].P_x+\left[k\frac{m-x}{m}\right].P_{x-1}+\left[\frac{x}{T}\right].P_{x+1} \qquad (1)$$

$$x = 1,\ 2,\ 3,\ \ldots\ldots\ m-1$$

Il existe, pour une mémoire de m blocs, m + 1 états possibles d'occupation. Les équations différentielles obtenues ne conviennent pas aux deux cas extrêmes où la mémoire est, soit totalement disponible, soit totalement saturée. L'évolution ne peut se faire alors que dans une seule direction.

Cas de la mémoire totalement disponible :

La probabilité $P_0\ (t + dt)$ résulte de deux cas possibles.

a) A l'instant t, tous les blocs sont libres.
Aucune demande ne se produit pendant l'intervalle de temps dt.

b) Un seul bloc est occupé à l'instant t.
Une libération se produit.
Aucune demande ne se présente pendant l'intervalle dt.

Il n'est pas nécessaire de tenir compte, ici, de la probabilité de satisfaction des demandes puisque toutes seraient satisfaites.

$$P_0\ (t + dt) = P_0\ (t) \cdot (1 - k.dt) + P_1\ (t) \cdot (1 - k.dt)\ \frac{dt}{T}$$

$$\frac{P_0\ (t + dt) - P_0\ (t)}{dt} = - k.\ P_0\ (t) + \frac{1}{T} . P_1\ (t)$$

et, en faisant tendre dt vers zéro:

$$\boxed{P'_0 = - k.\ P_0 + \frac{1}{T} \cdot P_1} \qquad (2)$$

En toute rigueur, il convient de préciser exactement le fonctionnement de la logique de contrôle de la mémoire pour écrire les équations. On a supposé jusqu'à présent qu'un bloc de mémoire pouvait, dans le même temps dt être libéré et repris par une nouvelle demande. Il est plus vraisemblable, en fait qu'au cours d'un même cycle dt, un bloc ne peut pas faire à la fois l'objet d'une libération et d'une remise au travail. Pour des raisons de simplicité, on reportera la prise en charge de la demande sur le cycle suivant et on doit considérer que si x blocs étaient occupés à l'instant t, seuls les m - x blocs restant sont disponibles pour répondre à d'éventuelles demandes, même si un nouveau bloc est libéré au cours du cycle.

Dans la suite des calculs, on prendra cette solution technologique comme hypothèse pour l'établissement des équations. Si l'équation donnant $P_0\ (t)$ n'est pas modifiée, il convient d'apporter une correction à l'équation générale.

Dans la nouvelle hypothèse les probabilités deviennent:

	Satisfaction	refus	libération
P_x	$\frac{m-x}{m}$	$\frac{x}{m}$	$\frac{x}{T}$
P_{x+1}	$\frac{m-x-1}{m}$	$\frac{x+1}{m}$	$\frac{x+1}{T}$
P_{x-1}	$\frac{m-x+1}{m}$	$\frac{x-1}{m}$	$\frac{x-1}{T}$

On trouve alors pour l'équation différentielle (1) l'expression suivante: **(1')**

$$P'_x = -\left[\frac{x}{T} + \frac{m-x}{m}\,k\right] \cdot P_x + \left[k\,\frac{m-x+1}{m}\right] \cdot P_{x-1} + \left[\frac{x+1}{T}\right] \cdot P_{x+1}$$

Dès que le nombre des blocs est relativement élevé, les deux expressions sont sensiblement équivalentes.

Cas de la mémoire totalement occupée:

Le nombre de blocs occupés est m; compte tenu de l'hypothèse précédente, aucune nouvelle demande ne peut être acceptée, même si l'un des blocs est libéré pendant le temps dt.

La probabilité $P_m\,(\,t + dt\,)$ est la somme des termes suivants :

a) Probabilité pour que m blocs soient occupés;
Aucune libération ne se produit.

b) Probabilité pour qu'il y ait m - 1 blocs occupés;
Probabilité pour que se présente une demande concernant le bloc libre.
Aucune libération ne se produit.

$$P_m\,(\,t + dt\,) = P_m\,(\,t\,).\left[1 - \frac{m.dt}{T}\right] + P_{m-1}(\,t\,).\,\frac{k.dt}{m}.\left[1 - \frac{m-1}{T}\right].\,dt$$

soit encore, en passant à la limite:

$$P'_m = -\,\frac{m}{T} \cdot P_m + \frac{k}{m} \cdot P_{m-1} \qquad (3)$$

L'ensemble des équations (1'), (2), et (3), soit, au total m + 1 équations définit complètement le comportement de la mémoire. On va établir, dans le cas particulier du régime stationnaire, une solution particulière du système.

B - SOLUTION PARTICULIERE DANS LE CAS DU REGIME STATIONNAIRE.

Si l'on considère que les échanges entre la mémoire et les circuits utilisateurs sont établis depuis assez longtemps pour que le regime continu soit atteint, les probabilités P_x sont indépendantes du temps et donc leurs dérivées par rapport au temps sont nulles.

$$P'_0 = P'_1 = \ldots\ldots = P'_m = 0$$

L'équation générale (1) se réduit alors à :

$$-\frac{x}{T} + \frac{m-x}{m} k \cdot P_x + \frac{m-x+1}{m} k \cdot P_{x-1} + \frac{x+1}{T} \cdot P_{x+1} = 0$$

soit encore:

$$\frac{x+1}{T} \cdot P_{x+1} = \frac{x}{T} + \frac{m-x}{m} k \cdot P_x - \frac{m-x+1}{m} k \cdot P_{x-1}$$

Le même raisonnement s'applique aux équations (2) et (3) et on obtient en définitive, le système complèt suivant :

$$P_1 = k \cdot T \cdot P_0$$

$$P_{x+1} = \left[\frac{T}{x+1} \cdot \frac{x}{T} + k \cdot \frac{m-x}{m}\right] P_x + \left[\frac{T}{x+1} k \frac{m-x+1}{m}\right] \cdot P_{x-1}$$

pour $x = 1, 2, \ldots\ldots m-1$.

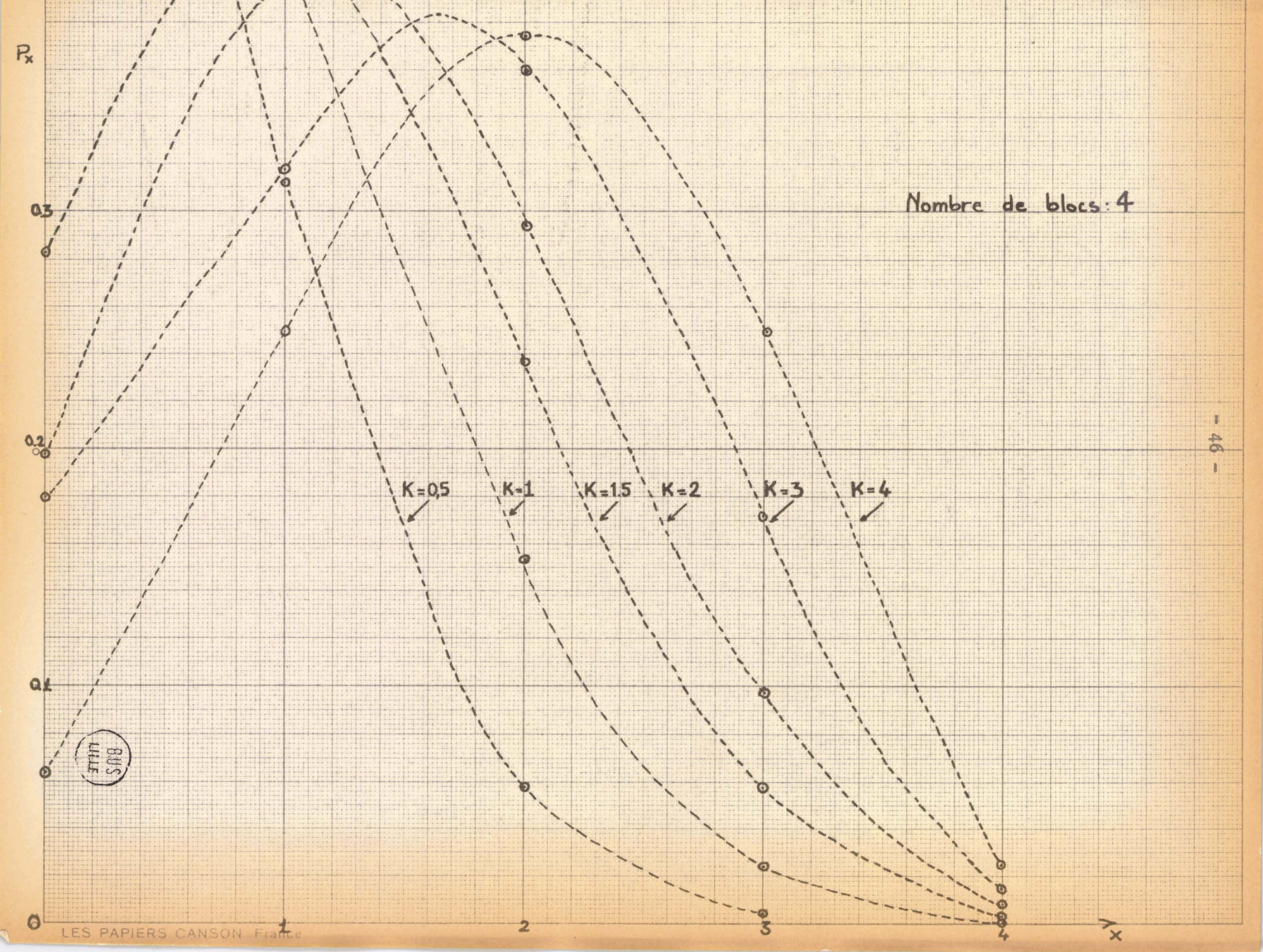
$$P_m = \frac{k \cdot T}{m^2} \cdot P_{m-1}$$

Le calcul direct de l'expression de P_x en fonction de P_0 et des autres paramètres caractèristiques de la mémoire, conduit à des expressions très longues et il a semblé préférable de procéder plutôt au calcul des valeurs numériques correspondant aux cas les plus usuels.

En partant de $P_0 = 1$, en prenant T comme unité de temps, on trouve P_1 qui vaut k; la connaissance de ces deux premiers termes permet le calcul de P_x pour x compris entre 0 et m-1. La formule (3) fournit P_m.

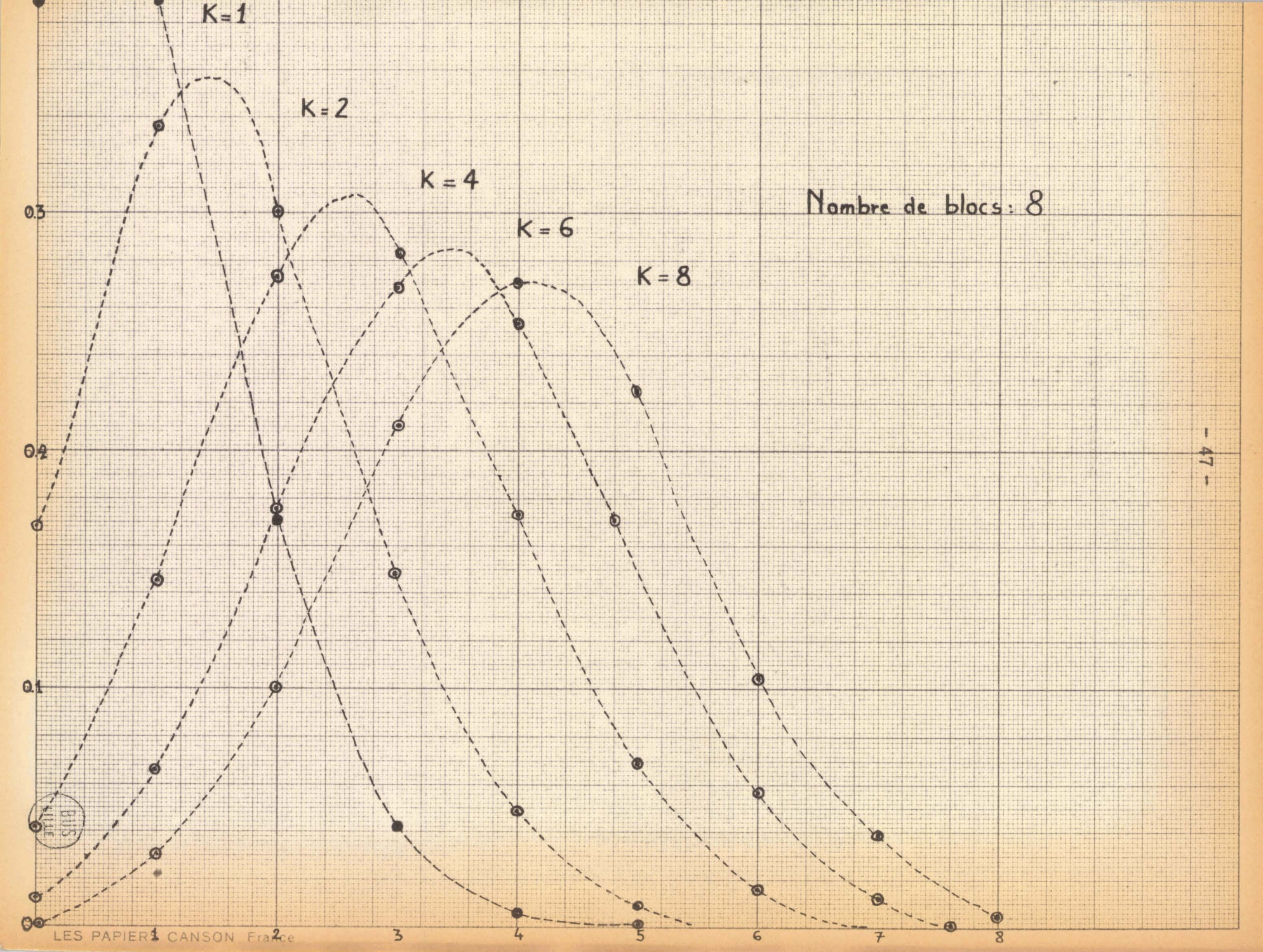
Les courbes des pages suivantes résument les résultats numériques obtenus pour m = 4, 8, 16 et 32 avec k/m = 0,25, 0,50, 0,75 et 1.

On a porté en abcisse le nombre de blocs occupés et en ordonnée, la probabilité de rencontrer ce nombre.

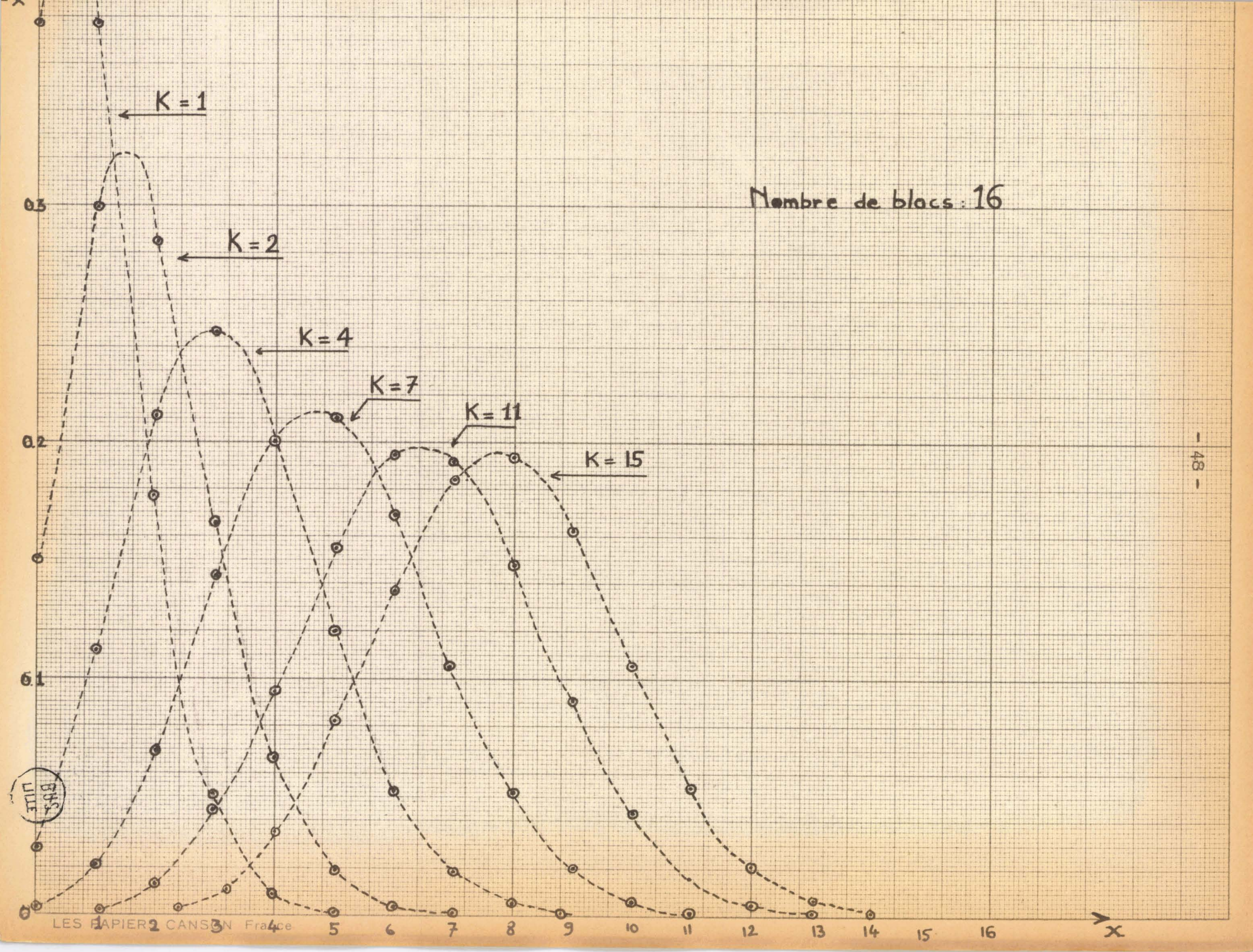


LILLE SUB

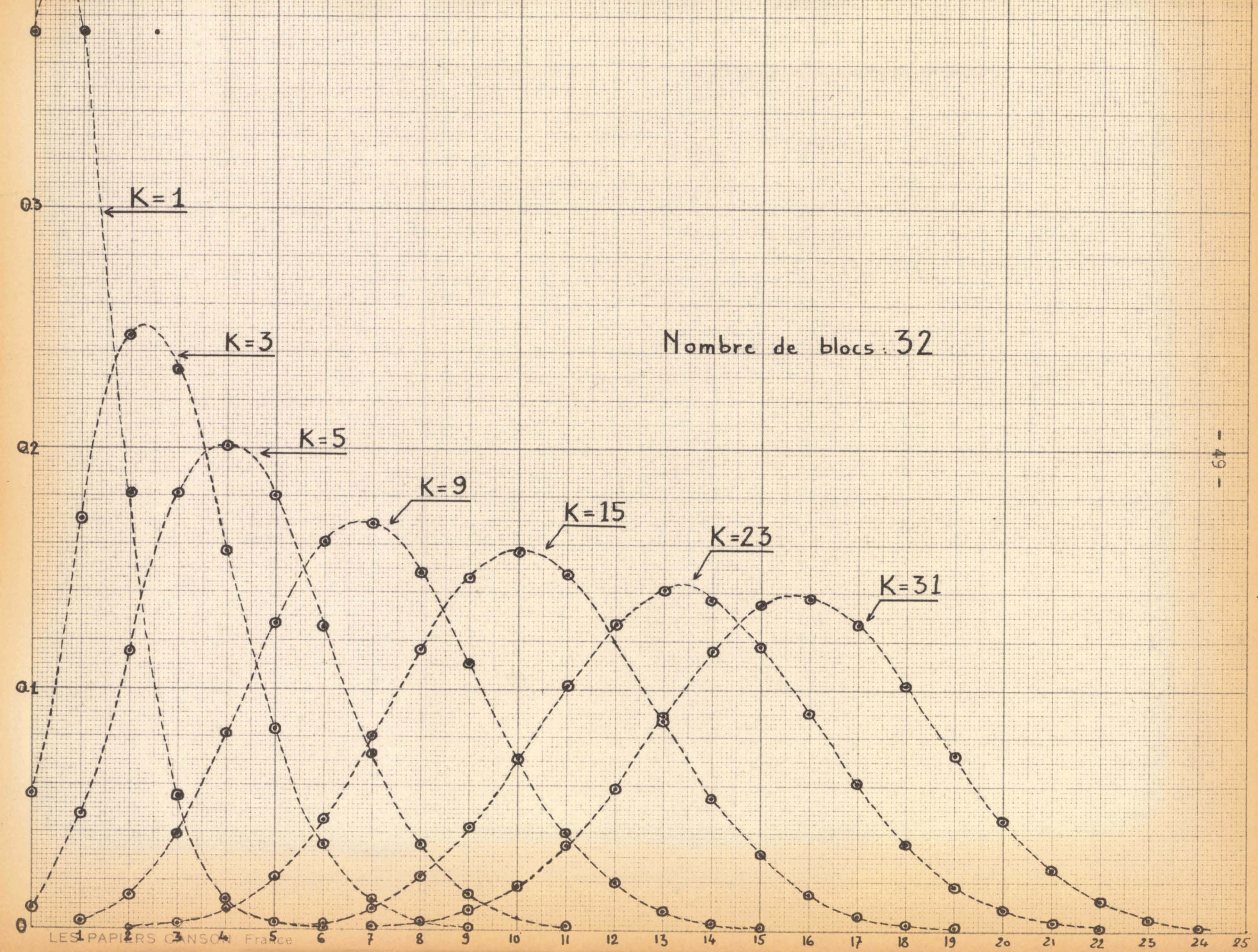
LES PAPIERS CANSON France



LES PAPIERS CANSON France



LES PAPIERS CANSON France



LES PAPIERS CANSON France

C - EMPLOI D'UNE FILE D'ATTENTE.

Les calculs précédents n'ont pas tenu compte du fait que les demandes rejetées devaient être représentées ultérieurement, aucune des demandes présentées ne pouvant être oubliée. Si une demande n'est acceptée qu'un certain temps après qu'elle soit apparue, il faut la stocker en dehors du canal normal d'accès aux mémoires pour ne pas bloquer les demandes suivantes ou pour ne pas la perdre si ces dernières viennent prendre sa place.

Il est donc absolument nécessaire d'adjoindre une file d'attente où seront stockées provisoirement les demandes non immédiatement satisfaites.

1) Longueur de la file d'attente.

Les grandeurs qui caractérisent la mémoire et les circuits qui l'exploitent étant fixées, il reste à déterminer la longueur de la file d'attente capable d'absorber les irrégularités du débit imposé à la mémoire. Une première idée de la valeur qu'il faut lui donner peut s'obtenir en écrivant une condition d'équilibre, en régime stationnaire :

Le nombre moyen des entrées dans la file doit être égal au nombre moyen des sorties.

- Il y a une demande stockée dans la file d'attente chaque fois qu'une demande se présente et ne peut être immédiatement staisfaite, le bloc désigné étant occupé.

- Une demande est extraite de la file d'attente lorsque, un bloc étant libéré, il fait l'objet d'un appel antérieur stocké dans la file.

La probabilité d'entrée est donc la probabilité pour qu'une demande se présente, multipliée par la probabilité pour que cette demande ne puisse être satisfaite :

$$P_E = (k.dt) . (\frac{x}{m})$$

La probabilité de sortie est la probabilité pour que, dans le même intervalle de temps dt, un bloc soit libéré, multipliée par la probabilité pour que ce bloc fasse l'objet d'une demande dans la file d'attente.

$$P_S = (\frac{X}{T} . dt) . H(y)$$

en posant : y, nombre de demandes dans la file et H (y) la probabilité pour que l'une de ces demandes en attente concerne le bloc qui vient d'être libéré. (p43)

S'il existe y demandes dans la file, la probabilité pour que l'une d'elles réclame un bloc qui vient d'être libéré est :

- La probabilité pour que la première le réclame, plus ....
- La probabilité pour que, la première ne le réclamant pas, la seconde le fasse, plus ....
- .....
- La probabilité pour que les (y - 1) premières demandes ne le réclamant pas, la dernière le fasse.

$$H(y) = \frac{1}{x} + \left(1 - \frac{1}{x}\right).\frac{1}{x} + \left(1 - \left(\frac{1}{x} + \left(1 - \frac{1}{x}\right).\frac{1}{x}\right)\right)\frac{1}{x} + \ldots$$

$$H(y) = \frac{1}{x}.\left(1 + \frac{x-1}{x} + \left(\frac{x-1}{x}\right)^2 + \left(\frac{x-1}{x}\right)^3 + \ldots + \left(\frac{x-1}{x}\right)^{y-1}\right)$$

x représente le nombre de positions occupées à l'instant considéré; ces positions étant les seules que puisse réclamer la file d'attente.

$$H(y) = \frac{1}{x}\left(1 + R + R^2 + R^3 + \ldots + R^{y-1}\right) \quad \text{avec} \quad R = \frac{x-1}{x}$$

Comme on cherche à déterminer une valeur moyenne de la longueur y de la file, on peut prendre pour x, taux d'occupation de la mémoire, la valeur moyenne k. La condition d'équilibre entre les entrées et les sorties peut donc s'écrire :

$$P_E = P_S = \frac{x}{m} \cdot k \cdot dt = x \cdot H(y) \cdot dt$$

$$\text{or} \quad H(y) = \frac{1}{k}\left[\frac{1 - \left(\frac{k-1}{k}\right)^y}{1 - \frac{k-1}{k}}\right] = 1 - \left(\frac{k-1}{k}\right)^y$$

La première condition pour que la relation soit possible, s'obtient sans développer H(y) en remarquant que cette quantité étant inférieure à 1, le taux de demandes k doit être inférieur à m.

La condition d'équilibre peut s'écrire :

$$\frac{k}{m} = 1 - \left(\frac{k-1}{k}\right)^y$$

$$\text{soit :} \quad y.\text{Log}\left(\frac{k-1}{k}\right) = \text{Log}\left(\frac{m-k}{m}\right)$$

$$\boxed{y = \frac{\text{Log}\left(\frac{m-k}{m}\right)}{\text{Log}\left(\frac{k-1}{k}\right)}}$$

Cette valeur de la longueur moyenne de la file d'attente a été obtenue en imposant l'égalité des débits de demandes à l'entrée et à la sortie. Ces débits ont, eux mêmes été assimilés à leur valeur moyenne.

De la sorte, le calcul ne fournit pas une indication suffisante sur le comportement de la file; le résultat est cependant utile car à partir d'un calcul simple, il fournit un ordre de grandeur de la dimensinn à prévoir pour cette file.

Le calcul numérique de la longueur moyenne a été fait pour 4, 8, 16 et 32 blocs indépendants. Lorsque le taux de demandes tend vers m, nombre de blocs, la longueur de la file tend vers l'infini ce qui était prévisible.

Il a semblé utile, de manière à comparer les performances des quatre groupement étudiés, de les faire figurer simultanément sur un même graphique. On exprime alors la longueur réduite (rapport de la longueur moyenne au nombre de blocs) en fonction du taux réduit de demandes (rapport du nombre de demandes au nombre de blocs)

On constate qu'une mémoire plus largement divisée présente, relativement à ses dimensions, des performances sensiblement moins bonnes.

Il est possible d'obtenir une expression un peu plus simple de la longueur de la file en procédant à un développement limité des logarithmes:

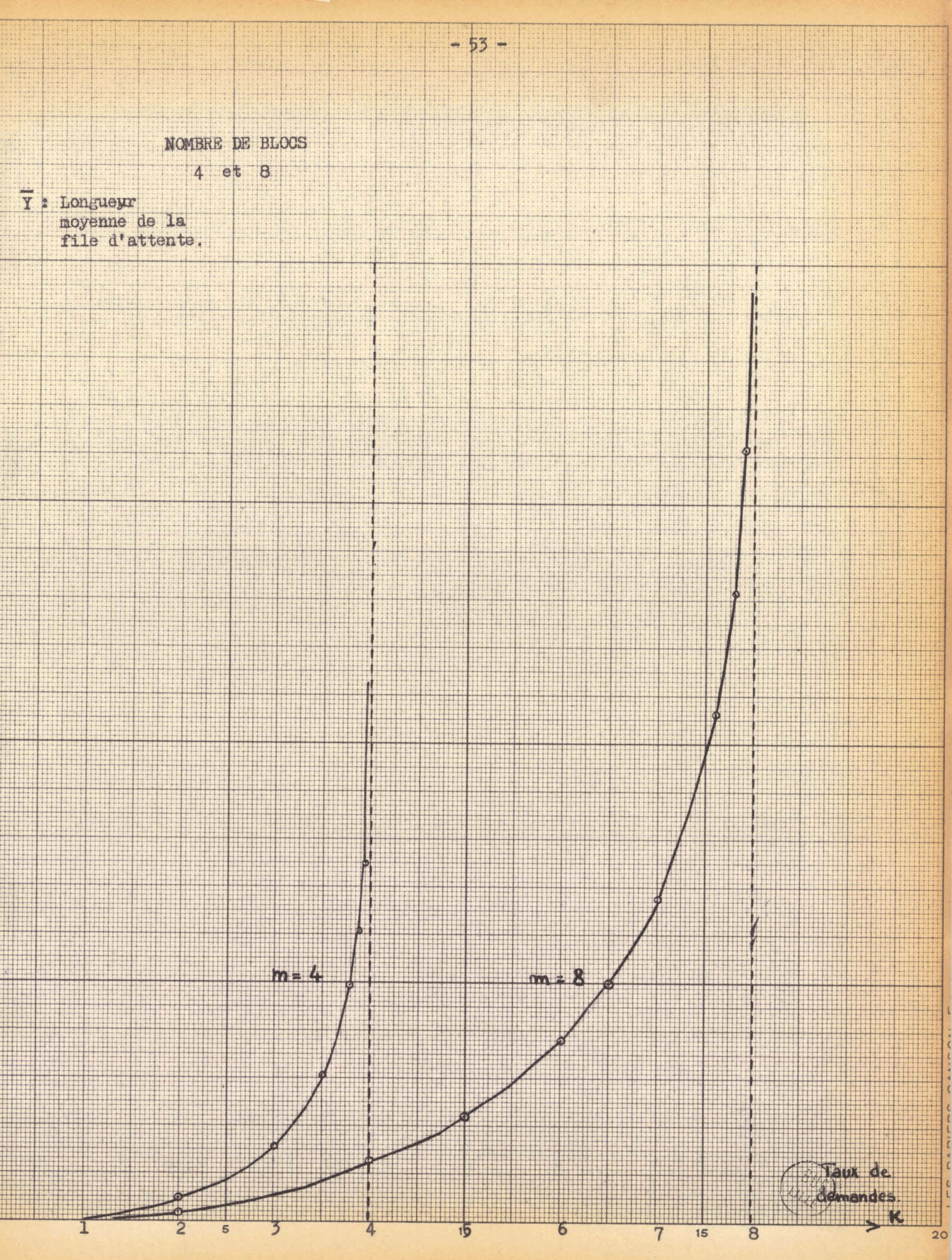
$$\text{Log}\,(1+x) = x - \frac{x^3}{3} + \ldots\ldots$$

$$y = \frac{\text{Log}\,(1-\frac{\tau k}{m})}{\text{Log}\,(1-\frac{1}{k})} = \frac{-\frac{k\tau}{m} + \frac{(\tau k)^3}{3\,m^3}}{-\frac{1}{k} + \frac{1}{3\,k^3}} = \frac{\tau\frac{k}{m}}{\frac{1}{k}} = \tau\frac{k^2}{m}$$

Cette approximation est valable pour de faibles valeurs du rapport k/m or c'est souvent dans ces conditions que l'on exploitera les mémoires partagées pour des raisons qui seront vues par la suite.

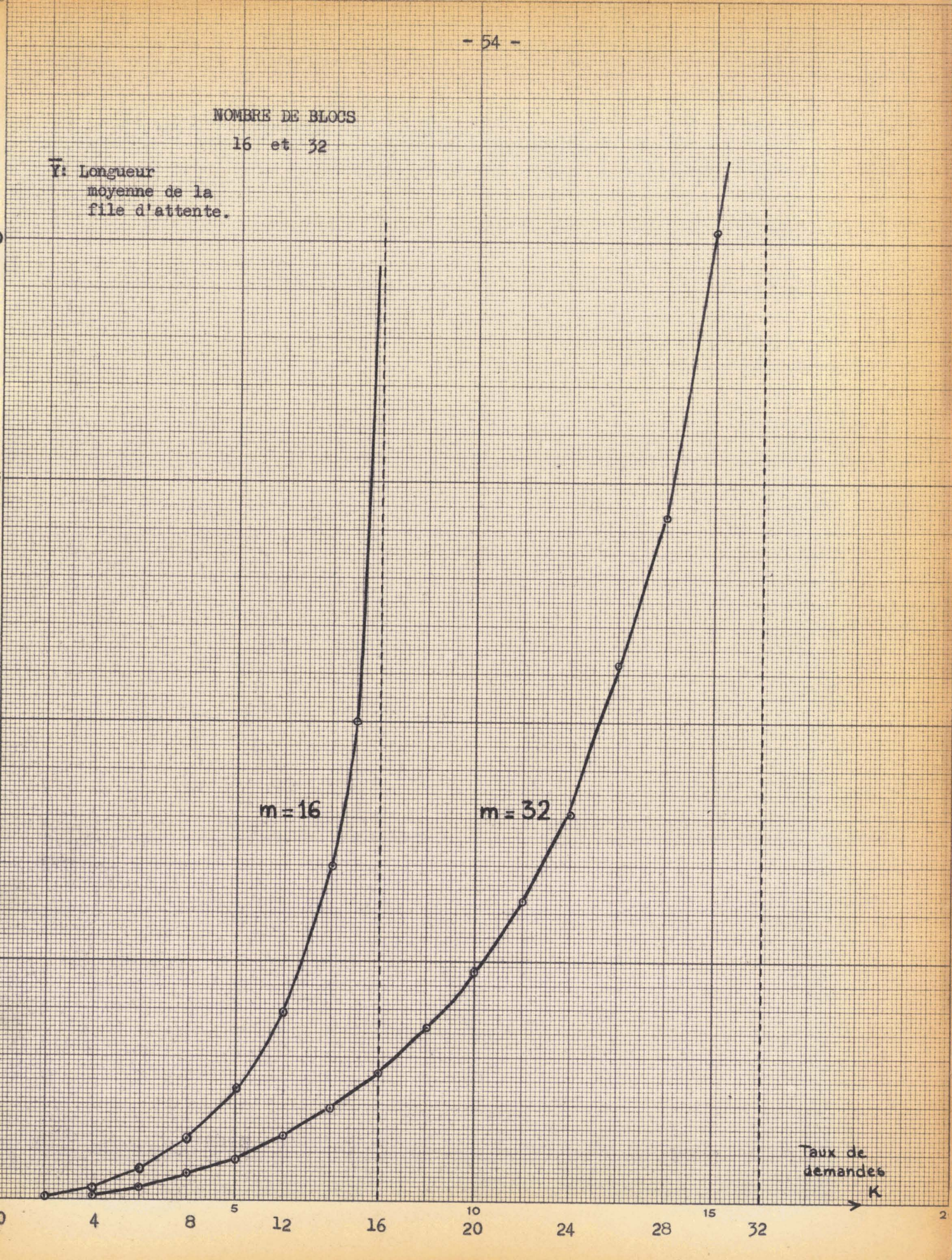
NOMBRE DE BLOCS
4 et 8

$\overline{Y}$: Longueur moyenne de la file d'attente.



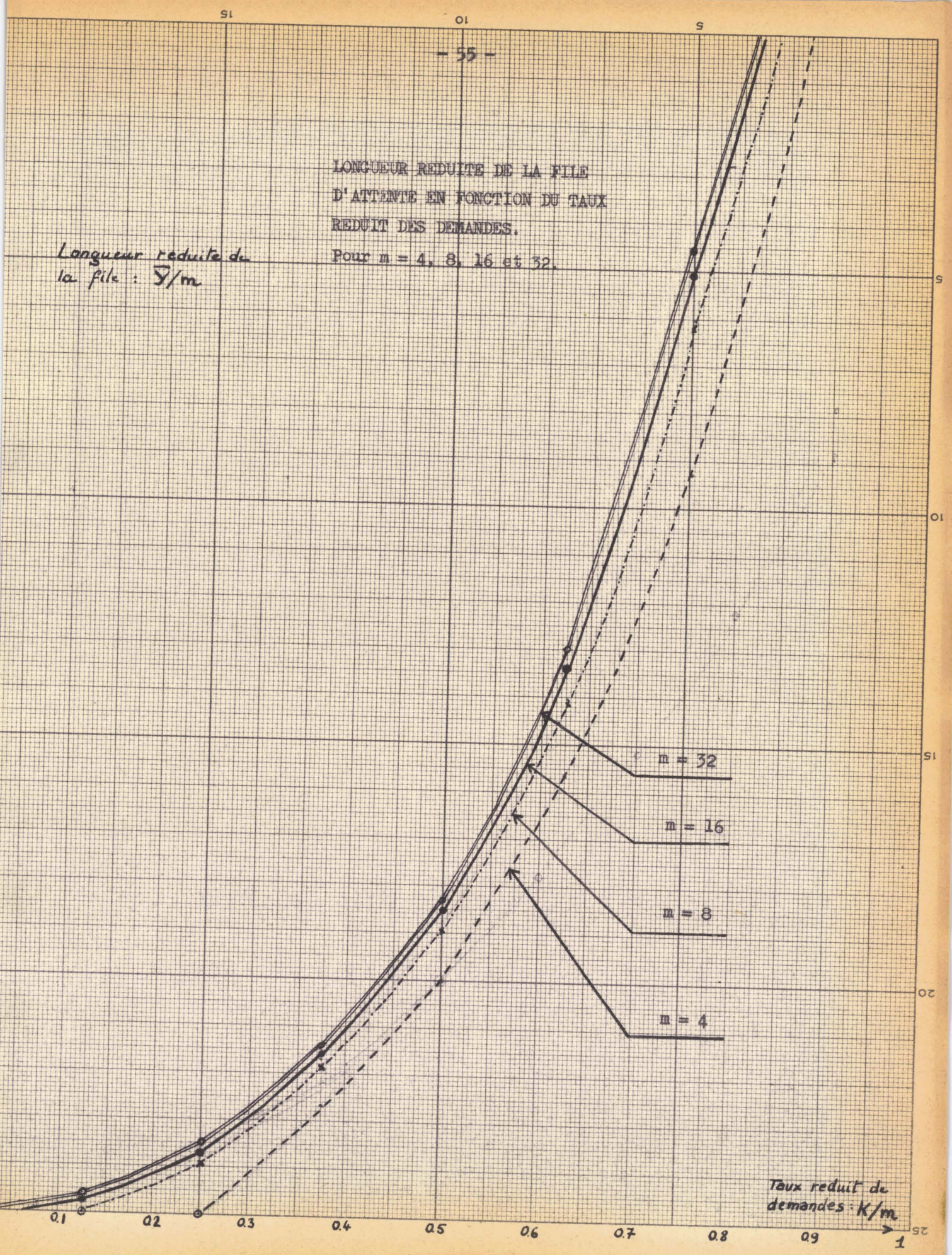
NOMBRE DE BLOCS
16 et 32

$\overline{Y}$: Longueur moyenne de la file d'attente.



LONGUEUR REDUITE DE LA FILE D'ATTENTE EN FONCTION DU TAUX REDUIT DES DEMANDES.

Pour m = 4, 8, 16 et 32.



La valeur de la longueur moyenne de la file, $\overline{y}$, obtenue par le calcul précédent ne peut avoir qu'un rôle indicatif. Si on prend effectivement cette valeur comme longueur de la file, celle-ci se trouvera saturée pendant la moitié du temps et son intérêt sera considérablement réduit.

Pour déterminer la longueur réelle de la file, il faut connaitre la loi de distribution de la longueur occupée et choisir une valeur telle que la probabilité de saturation devienne très faible.

Soit Q_y (t), la probabilité pour que, à un instant donné, la longueur de la file soit y c'est à dire qu'elle contienne y demandes non satisfaites. On va évaluer la probabilité pour qu'à l'instant suivant, t + dt, la file contienne y demandes : Q_y (t + dt)

La probabilité Q_y (t + dt) résulte des possibilités suivantes:

a) A l'instant t, la file contenait y demandes.
Il n'y a pas eu d'entrée.
Il n'y a pas eu de sortie.

b) A l'instant t, la file contenait y demandes.
Il y a eu une entrée.
Il y a eu une sortie.

c) A l'instant t, la file contenait y + 1 demandes.
Il n'y a pas eu d'entrée.
Il y a eu une sortie.

d) A l'instant t, la file contenait y - 1 demandes.
Il y a eu une entrée.
Il n'y a pas eu de sortie.

Il vient alors:

$$
\begin{aligned}
Q_y(t+dt) = \; & Q_y(t) \cdot \left[1 - \frac{x}{m} k.dt\right] \cdot \left[1 - \frac{x}{T} H(y) \cdot dt\right] \\
& + Q_y(t) \cdot \left[\frac{x}{m} \cdot k.dt\right] \cdot \left[\frac{x}{T} \cdot H(y) \cdot dt\right] \\
& + Q_{y+1}(t) \cdot \left[1 - \frac{x}{m} \cdot K.dt\right] \cdot \left[\frac{x}{T} \cdot H(y+1) \cdot dt\right] \\
& + Q_{y-1}(t) \cdot \left[\frac{x}{m} .k.dt\right] \cdot \left[1 - \frac{x}{T} \cdot H(y-1) \cdot dt\right]
\end{aligned}
$$

En prenant T comme unité de temps et en négligeant les termes en dt^2:

$$Q_y(t+dt) = Q_y(t).(1 - x.H(y).dt - \frac{k}{m}.x.dt) + Q_{y+1}(t).x.H(y+1).dt + Q_{y-1}(t).\frac{k}{m}.x.dt$$

Le passage à la limite donne:

$$Q'_y(t) = -Q_y(t).(\frac{x}{T}.H(y) + \frac{kx}{m}) + Q_{y+1}(t).\frac{x}{T}.H(y+1) + Q_{y-1}(t)\frac{kx}{m}$$

Dans le cas du régime stationnaire:

$$0 = -Q_y.(H(y) + \frac{k}{m}) + Q_{y+1}.H(y+1) + Q_{y-1}.\frac{k}{m}$$

Les fonctions Q sont alors indépendantes du temps et on remarque que x a disparu de l'expression ce qui signifie que la loi de distribution des occupations dans la file d'attente ne dépend que des paramétres fixes de la mémoire : Taux de demandes et nombre de blocs, directement et aussi par l'intermédiaire de la fonction H (y).

La relation définitive s'écrit:

$$\boxed{Q_{y+1} = \frac{1}{H(y+1)}.Q_y.(H(y) + \frac{k}{m}) - \frac{1}{H(y+1)}.Q_{y-1}.\frac{k}{m}}$$

Dans le cas particulier où la file est vide :

$$Q_0(t+dt) = Q_0(t).(1 - \frac{x}{m}k.dt) + Q_1(t)\frac{x}{T}.\frac{1}{m}.dt$$

$$Q'_0(t) = -Q_0(t).\frac{x}{m}.k + Q_1(t).\frac{x}{T}.\frac{1}{m}$$

et, dans le cas du régime stationnaire, en prenant $H(0) = 0$ et $H(1) = 1/m$:

$$\boxed{Q_1 = k.Q_0}$$

On suppose que la longueur de la file est suffisante pour que la probabilité de la saturer soit très faible. Si néammoins cette éventualité se présentait, la méthode de calcul employée jusqu'à présent ne serait plus valable. La saturation de la file entrainerait un blocage des demandes et le taux de demandes tomberait provisoirement à une valeur nulle. Le calcul ne peut se faire qu'en tenant compte du comportement de la mémoire.

Calcul numérique du taux de remplissage.

Le calcul des probabilités d'occupation de la file d'attente se poursuit de la même manière que pour les taux d'occupation de la mémoire. En partant de $Q_0 = 1$, on peut déterminer les valeurs de Q_1, Q_2, etc. Le calcul a été poursuivi jusqu'à ce que la probabilité de trouver le taux de remplissage correspondant devienne négligeable.

Les configurations comportant 4, 8, 16 et 32 blocs ont été considérées pour des taux de demandes correspondant à des valeurs de k/m, taux réduit de demandes, de 0,125 - 0,250 - 0,375 - 0,500 - 0,625 - 0,750 et 0,875.

On remarque que pour k/m = 1, limite théorique du débit:

$$Q_{y+1} = \frac{1}{H(y+1)} \cdot Q_y \cdot (H(y) + 1) - \frac{1}{H(y+1)} \cdot Q_{y-1}$$

Lorsque y augmente H(y+1) tend vers H(y) et donc Q(y+1) tend vers Q(y) ce qui conduit à prévoir une file d'attente de longueur infinie.

Les résultats numériques ont été portés sur les courbes des pages suivantes. On trouve en abcisse la longueur de la file et en ordonnée, la probabilité d'occuper effectivement la longueur correspondante. Le taux de demandes est employé comme paramètre.

Remarque 1

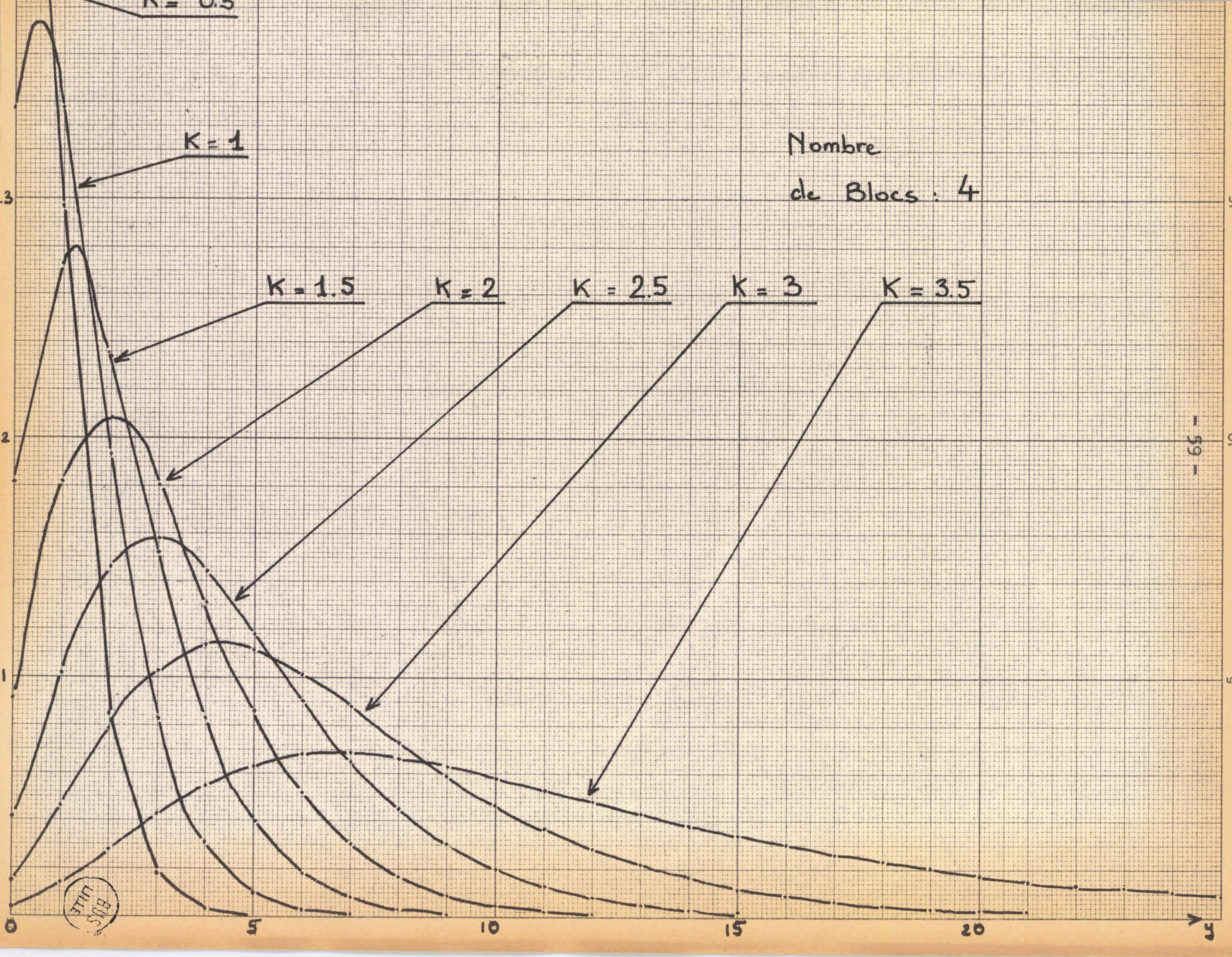
Certaines valeurs numériques qui ont été reportées ne présentent aucun intérêt en raison de la longueur de la file qu'elles supposent. On constate, par exemple, que, pour m = 32 et K = 28, la probabilité pour que la longueur de la file dépasse 65 est approximativement 0,5. Il faudrait une longueur supérieure à 200 pour que la probabilité de dépassement devienne inférieure à 0,01. Une telle longueur n'apporterait aucun bénéfice parce que son contenu serait invalidé lors de chaque rupture de séquence.

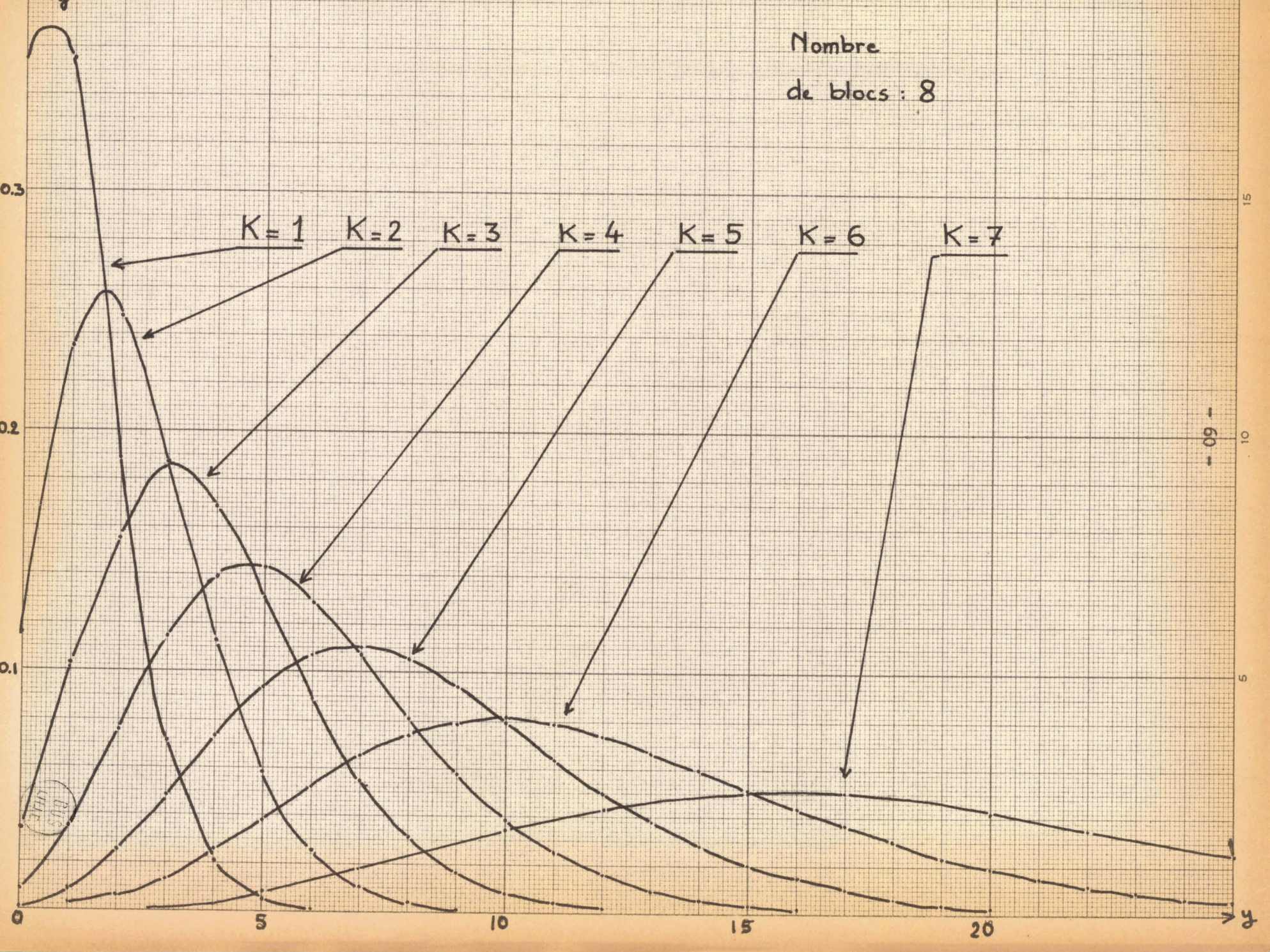
Remarque 2

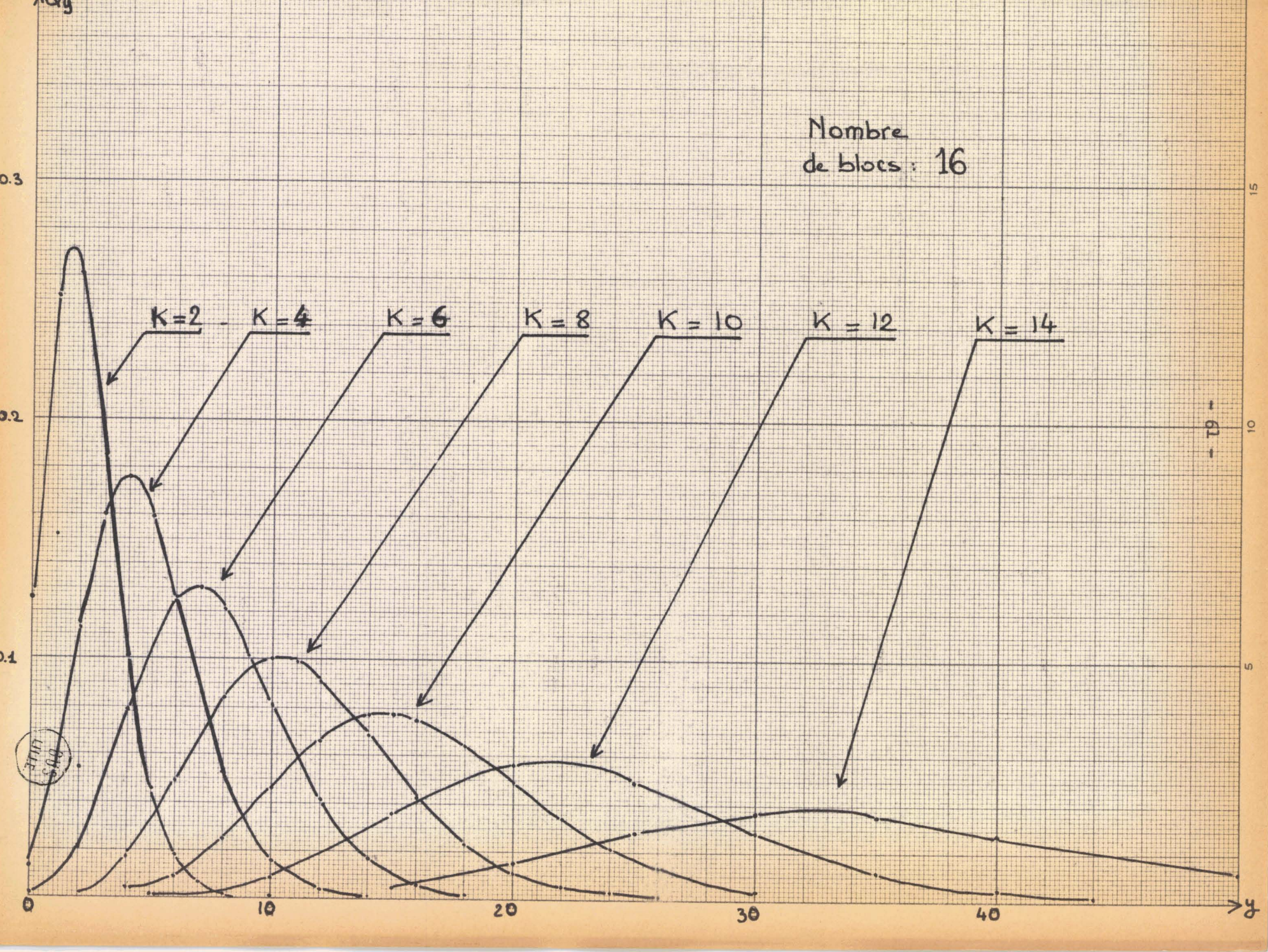
Le taux d'occupation de la file d'attente pourrait aussi s'exprimer en déterminant la probabilité pour que le nombre de positions occupées soit inférieur ou égal à une valeur donnée. Si on appelle Q'_y cette probabilité:

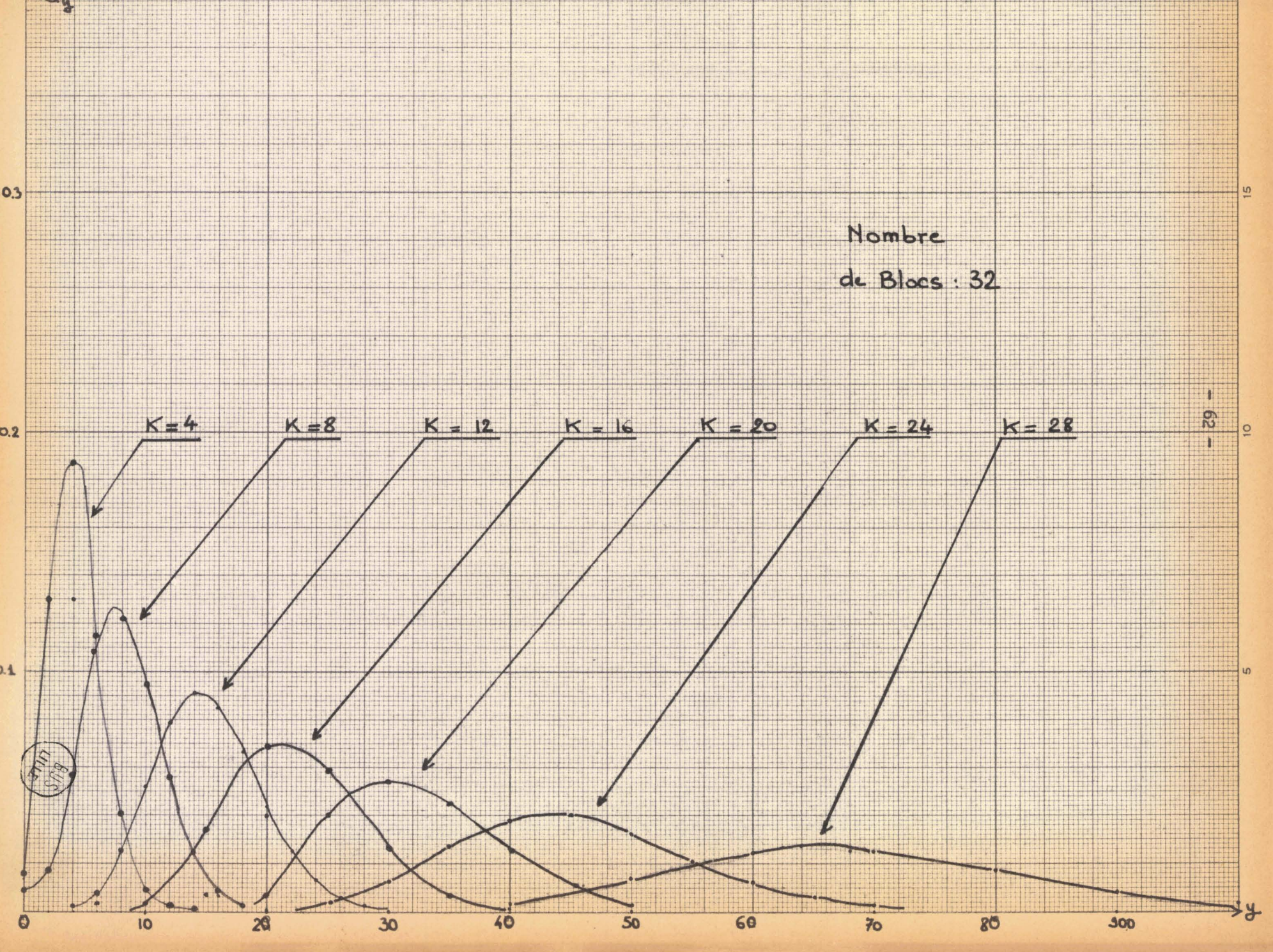
$$Q'_y = \sum_{j=o}^{j=y} Q_j$$

La probabilité Q'_y sera plus significative dans le choix de la longueur maximum de la file. On a conservé la probabilité Q_y car c'est ce terme qui intervient dans les calculs ultérieurs.


K = 0.5
K = 1
Nombre
de Blocs : 4
K = 1.5
K = 2
K = 2.5
K = 3
K = 3.5
0
5
10
15
20
y
.3
2
1
15
10
5
- 99 -


Nombre
de blocs : 8
K = 1
K = 2
K = 3
K = 4
K = 5
K = 6
K = 7
0.3
0.2
0.1
0
5
10
15
20
- 60 -


Nombre
de blocs : 16
K=2
K=4
K=6
K=8
K=10
K=12
K=14
0.3
0.2
0.1
0
10
20
30
40
y
15
10
5
- 61 -

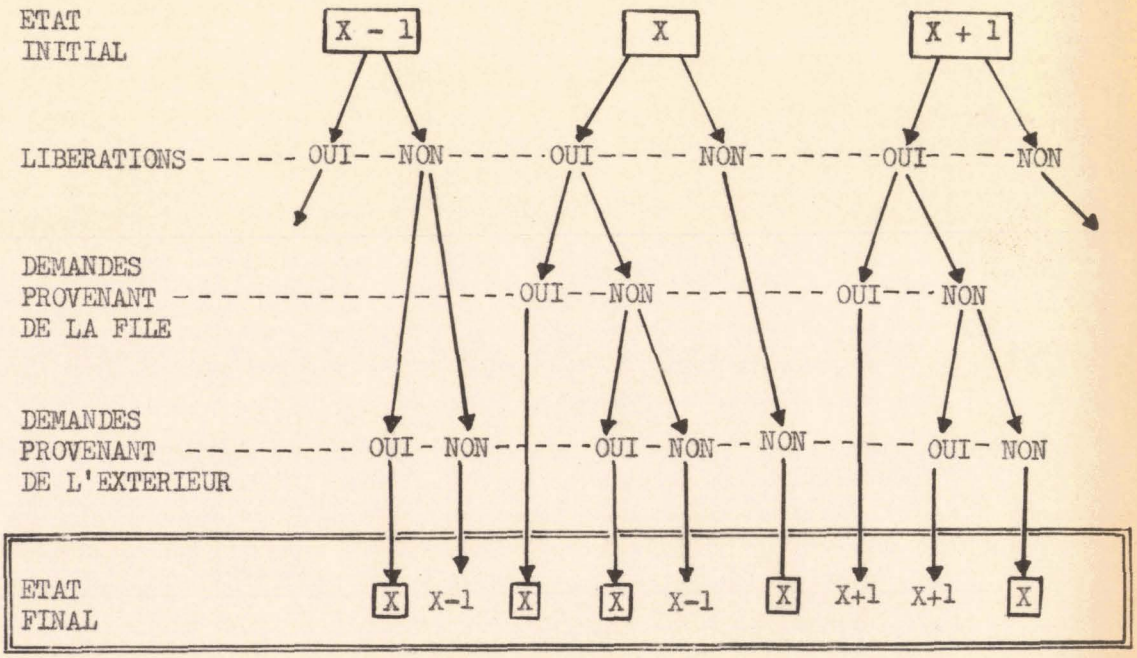

Nombre
de Blocs : 32
K = 4
K = 8
K = 12
K = 16
K = 20
K = 24
K = 28
0.3
0.2
0.1
15
10
5
0
10
20
30
40
50
60
70
80
90
dy

D - OCCUPATION DE LA MEMOIRE ASSOCIEE A UNE FILE D'ATTENTE.

Lorsqu'une file d'attente peut recevoir les demandes issues des circuits utilisateurs et qui ne peuvent être immédiatement satisfaites, on doit tenir compte de deux sources distinctes de demandes : La voie directe et la file d'attente. Il est possible de tenir compte, dans les calculs, de priorités relatives entre les demandes apportées par la voie directe et celles qui proviennent de la file d'attente. Il est logique d'accorder aux demandes de la file, la priorité sur les demandes directes puisqu'elles sont plus anciennes. Dans la file elle-même, les demandes sont rangées dans leur ordre d'arrivée et servies dans cet ordre.

Le principe du calcul reste identique à celui des calculs précédents. On remarquera cependant que la file d'attente ne doit être consultée que si une libération se produit; en effet, les demandes stockées dans cette file ne peuvent concerner que des blocs occupés.

L'organigramme ci- dessous décrit les différentes évolutions par lesquelles, à l'instant t + dt, x blocs de la mémoire sont occupés à partir de l'état à l'instant t et des modifications survenues pendant l'intervalle de temps dt.



Si l'on désirait accorder à certains canaux demandeurs une priorité sur la file d'attente, il faudrait distinguer plusieurs familles de demandes directes et placer la consultation des canaux prioritaires avant celle de la file. Il faut pourtant s'assurer qu'il n'existe pas dans la file des demandes provenant d'un canal prioritaire ou admettre que l'organisation des canaux est séquentielle et qu'une demande ne peut se présenter que lorsque la précédente a été satisfaite.

En conservant la solution présentée sur l'organigramme précédent, on peut évaluer la probabilité d'avoir x blocs occupés à l'instant t + dt :

$$P_x(t+dt) =$$

$$P_{x-1}(t) \cdot \left(1 - \frac{x-1}{T}\,dt\right) \cdot \left(k\,\frac{m-x+1}{m}\,dt\right)$$

$$+ P_x(t) \cdot \left(\frac{x}{T}\,dt\right) \cdot \left(H(y) + k\,\frac{m-x}{m}\,dt - k\,\frac{m-x}{m}\,H(y)\,dt\right)$$

$$+ P_x(t) \cdot \left(1 - \frac{x}{T}\,dt\right) \cdot \left(1 - k\,\frac{m-x}{m}\,dt\right)$$

$$+ P_{x+1}(t) \cdot \left(\frac{x+1}{T}\,dt\right) \cdot \left(1 - H(y)\right) \cdot \left(1 - k\,\frac{m-x+1}{m}\,dt\right)$$

En effectuant les produits et en négligeant les termes en dt^2:

$$P_x(t+dt) =$$

$$P_{x-1}(t) \cdot \left(k\,\frac{m-x+1}{m}\,dt\right)$$

$$+ P_x(t) \cdot \left(1 - \frac{x}{t}\,dt - k\,\frac{m-x}{m}\,dt + \frac{x}{T}\,H(y)\,dt\right)$$

$$+ P_{x+1}(t) \cdot \left(\frac{x+1}{T}\,dt - \frac{x+1}{T}\,H(y)\,dt\right)$$

On en déduit:

$$\frac{P_x(t+dt) - P_x(t)}{dt} =$$

$$P_x(t).\left(\frac{x}{T}(H(y) - 1) - k\,\frac{m-x}{m}\right) + P_{x-1}(t).k\,\frac{m-x+1}{m} + P_{X+1}(t)\frac{x+1}{T}(1 - H(y))$$

Si on fait tendre dt vers zéro et si on prend T comme unité de temps:

$$P'_x(t) = P_x(t)x(H(y)-1) - k\,\frac{m-x}{m}\Big) + P_{x-1}(t)k\,\frac{m-x+1}{m} + P_{x+1}(t)\,\frac{x+1}{m}\,(1 - H(y))$$

équation différentielle valable pour x compris entre 1 et m-1.

En prenant, comme précédemment, l'hypothèse du regime stationnaire, on substitue aux équations différentielles le système d'équations:

$$(x+1).(1-H(y)).P_{x+1} = P_x(x.(1-H(y) + k\frac{m-x}{m}) - P_{x-1}.k\frac{m-x+1}{m}$$

$$x = 1, 2, 3, \ldots\ldots m-1 .$$

Cas de la mémoire totalement libre:

$$P_0(t+dt) = P_0(t).(1-k.dt) + P_1(t).\frac{dt}{T}\ (1-H(y)).(1-k.dt)$$

$$P_0(t+dt) = P_0(t).(1-k.dt) + P_1(t).(1-H(y))\frac{dt}{T}$$

$$\frac{P_0(t+dt) - P_0(t)}{dt} = P_0(t).k + P_1(t).(1-H(y)).\frac{1}{T}$$

et, en se plaçant dans le cas du régime stationnaire, il vient :

$$P_1 = \frac{k\,P_0}{1-H(y)}$$

Cas de la mémoire totalement occupée:

$$P_m(t+dt) = P_{m-1}(t).(1-\frac{m-1}{T}dt).\frac{k}{m}dt + P_m(t).(1-\frac{m}{T}dt)$$

$$+ P_m(t).(H(y) + (1-H(y))\frac{k}{m}dt)\frac{m}{T}dt$$

$$P_m(t+dt) = P_{m-1}(t)\frac{k}{m}dt + P_m(t).(1-\frac{m}{T}(H(y)-1))$$

$$\frac{P_m(t+dt) - P_m(t)}{dt} = -P_m(t)\frac{m}{T}(1-H(y)) + P_{m-1}(t)\frac{k}{m}$$

et, dans le cas du régime stationnaire:

$$P_m = \frac{k\ P_{m-1}}{m^2\,(1-H(y))}$$

En définitive, le système complet s'exprime à l'aide des m + 1 équations suivantes:

$$P_1 = k\,P_0 / (1-H(y))$$

$$P_{x+1} = \frac{1}{(x+1).(1-H(y))}\ P_x(x.(1-H(y)) + k\frac{m-x}{m} - P_{x-1}\,k\frac{m-x+1}{m}$$

$$P_m = k.P_{m-1} /(m^2\,(1-H(y))$$

En fait, ce ne sont pas m +1 équations qu'il faut traiter mais, si s représente le nombre de positions de la file d'attente, il existe pour chaque taux d'occupation de la file, un système de m+1 équations. La probabilité de voir x blocs occupés est la somme pondérée des termes P_x exprimés plus haut pour y allant de 0 à s.

On posera $P_{x,y}$ la probabilité d'avoir x blocs occupés dans la mémoire sachant que y positions de la file sont occupées.

$$P_{x,y} = \frac{1}{x.(1 - H(y))} \; P_{x-1,y}^{(x-1)} (1 - H(y)) + k \frac{m-x-1}{m} - P_{x-2,y} \; k \frac{m - x}{m}$$

$$X = 1, 2, 3, \ldots m-1 \quad \text{et} \quad y = o, 1, 2, \ldots s$$

En appelant $\overline{P}_x$, la probabilité totale d'avoir x blocs occupés, on trouve:

$$\overline{P}_x = \sum_{y=0}^{y=s} P_{x,y} \cdot Q_y$$

La longueur y de la file n'apparait alors plus explicitement dans l'expression bien que la longueur maximum de cette file, s, modifie la distribution des probabilités.

P_x s'exprime uniquement en fonction des trois paramètres suivants:

- m : Nombre de blocs de la mémoire;
- s : Nombre maximum de positions de la file d'attente;
- k : Taux de demandes.

Le débit de la mémoire, directement lié au taux d'occupation a pour valeur:

$$D = \sum_{x=0}^{x=m} x. \overline{P}_x$$

$$D = \sum_{x=0}^{x=m} x \; . \sum_{y=0}^{y=s} P_{x,y} \cdot Q_y$$

Ce débit s'exprime en mots traitées dans des opérations de lecture ou d'écriture par temps de cycle technologique T .

Expression approchée des probabilités de distribution.

Afin de permettre une simplification des calculs on peut faire l'hypothèse suivante : La file d'attente est assez longue pour que la probabilité de la saturer soit très faible. Le nombre moyen des demandes qu'elle contient est alors la quantité $\overline{y}$ calculée précédemment.

La probabilité $\overline{P}_x$, prise plus haut comme la moyenne pondérée par Q_y des probabilités $P_{x,y}$ sera maintenant assimilée à la probabilité $P_{x,y}$ avec, pour y la valeur moyenne $\overline{y}$.

$$\overline{P}_x = P_{x,\overline{y}}$$

Or, pour $y = \overline{y}$, la fonction $H(y)$ prend la valeur $H(\overline{y}) = k / m$.

On retrouve alors un systéme unique de m+1 équations:

$$P_1 = k \cdot P_0 / (1 - k / m)$$

$$P_x = \frac{1}{x.(1 - k/m)} P_{x-1} \left((x - 1).(1 - k/m) + k \frac{m-x-1}{m}\right) - P_{x-2} \cdot k \frac{m-x}{m}$$

$$P_m = k \cdot P_{m-1} / (m^2 (1 - k/m))$$

Ce système unique conduit à des calculs bien plus simples. Les résultats obtenus ne sont valables que dans la mésure où la fonction calculée pour la valeur moyenne de y est égale à la moyenne des fonctions calculées pour y variant de 0 à s.

Il existe certains moyens indirects de vérifier, à postériori, la validité de cette hypothèse simplificatrice:

- La valeur maximale de P_x doit étre obtenue pour $x = k$.
- La valeur moyenne des blocs occupés doit étre égale à k puisque, la file n'étant jamais saturée, le débit global doit étre égal au taux de demandes.

On trouvera plus loin, dans l'étude des régimes transitoires, une seconde méthode de détermination des taux d'occupation qui, elle est parfaitement rigoureuse. La comparaison des résultats obtenus par l'une et l'autre de ces deux méthodes permet de justifier, dans les configurations étudiées, la simplification proposée.

Calcul numérique du taux d'occupation de la mémoire.

a) Calcul rigoureux.

Pour m = 4, le calcul a été poursuivi à partir des formules exactes qui expriment $\overline{P}_x$.

Les valeurs de la fonction H(y) sont portées dans le tableau ci-contre. Pour chacune de ces valeurs, on calcule la suite récurrente des probabilités d'occupation de la mémoire. Ces valeurs sont alors pondérées par la probabilité pour que la file se trouve dans l'état considéré. La somme de ces différents termes représente la probabilité cherchée.

y	H(y)
0	0,000000
1	0,250000
2	0,437500
3	0,578125
4	0,683594
5	0,762695
6	0,822021
7	0,866516
8	0,899887

Dans les cas où la probabilité de saturer la file n'est pas négligeable, on constate que la valeur moyenne des occupations ne correspond plus à la valeur moyenne des demandes. La saturation de la file d'attente l'empêche de tenir son rôle et il faut bloquer les demandes.

Le calcul a été fait, dans le cas de m = 4 pour des valeurs de s égales à 1, 2, et 4 positions, les taux de demandes prenant les valeurs 1, 2, 3 et 4.

Les courbes correspondant à s = 1 ont été reportées sur le graphique de la page suivante (courbes -.-.-.-.); pour k = 1, on constate un faible recul par rapport à la courbe résultant de l'expression simplifiée. Pour k = 3, l'effet de saturation de la file réduit le débit réel dans la proportion des deux tiers.

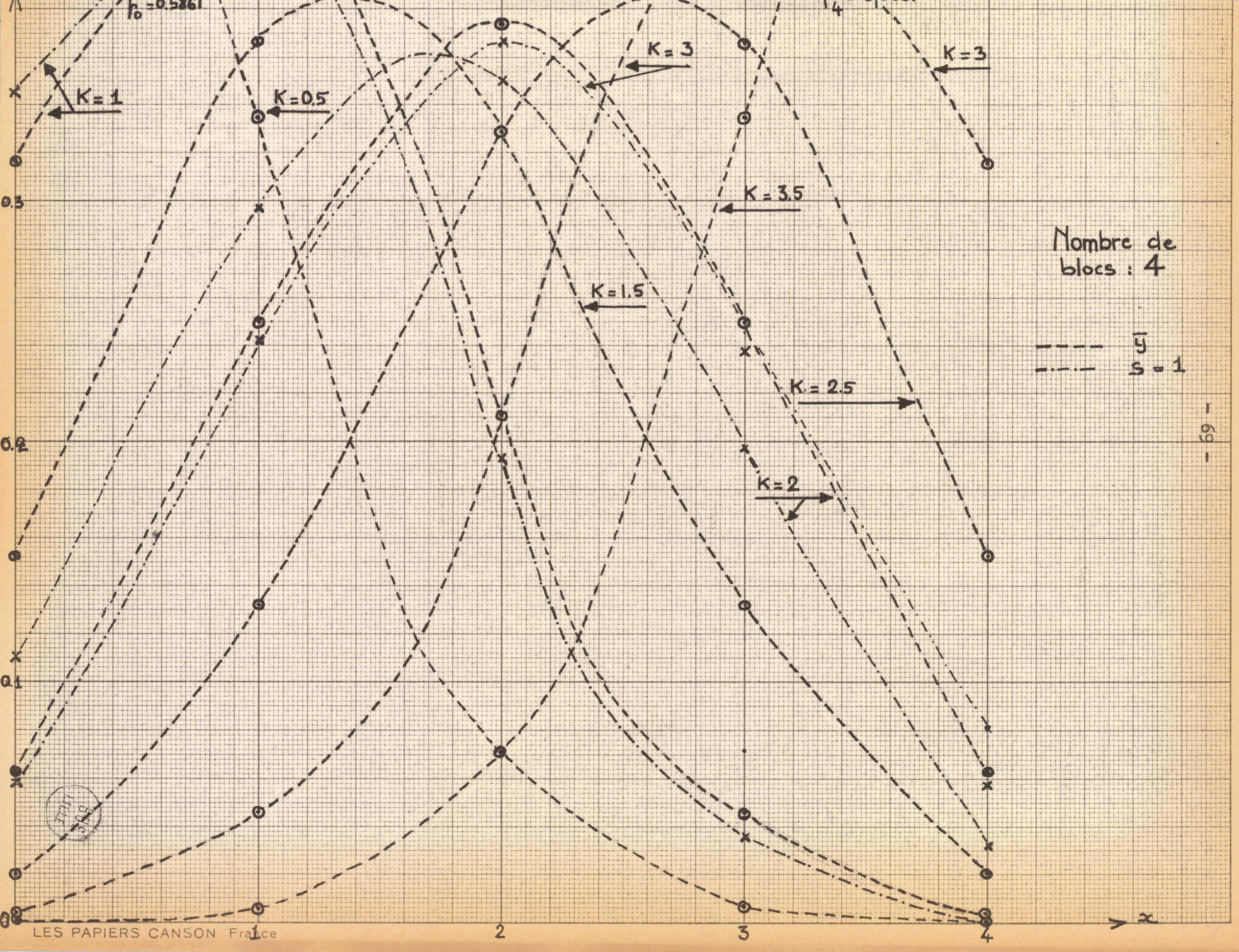
Le calcul exact de débits de la mémoire pour m=4 et s=1 donne:

- Pour k = 1 débit réel o,9317
- Pour k = 2 débit réel 1,7328
- Pour k = 3 débit réel 1,9508

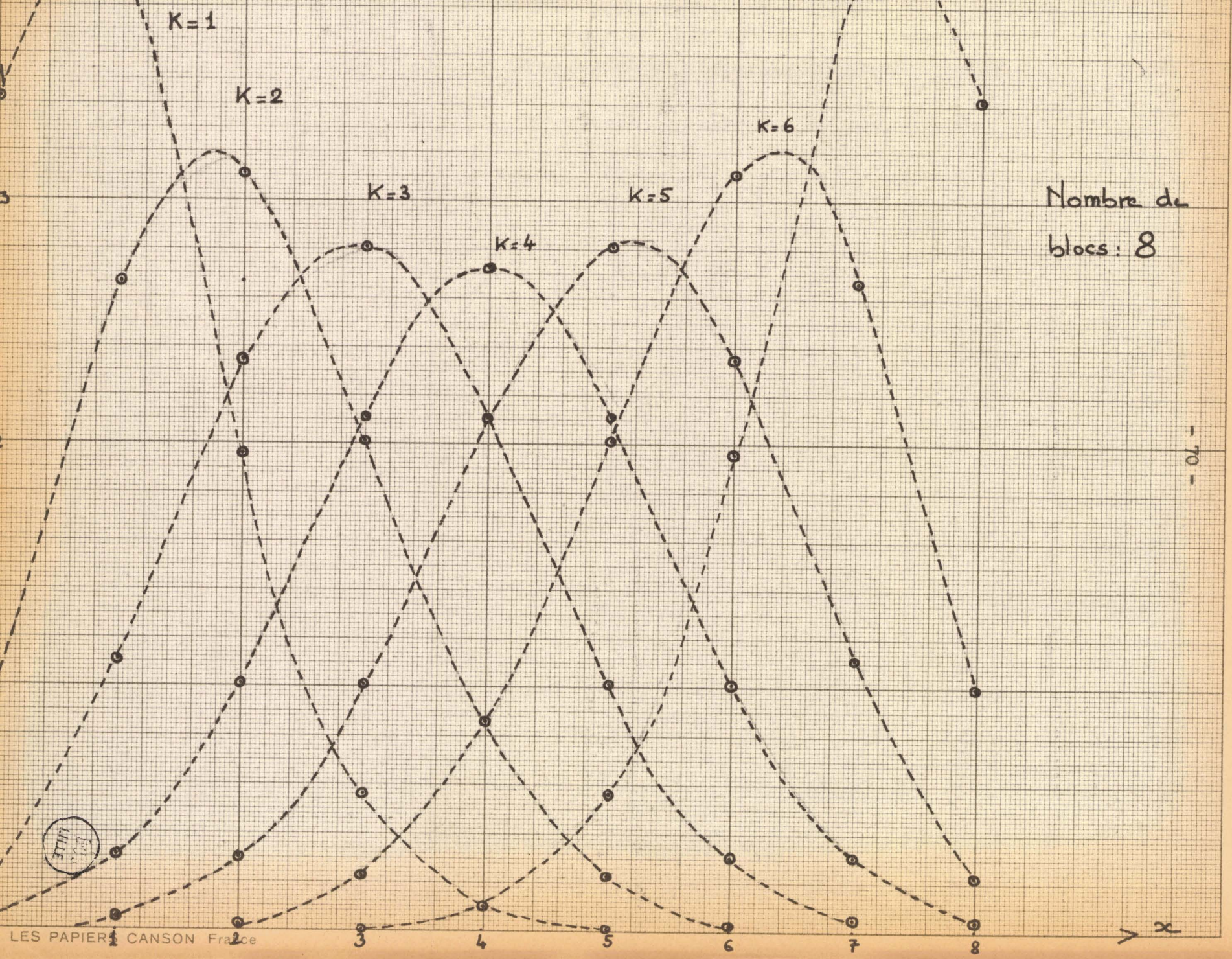
b) Toutes les autres courbes des graphiques qui suivent, résultent de l'emploi de l'expression simplifié et donc de l'hypothèse que la file n'a qu'une probabilité très faible d'être saturée.

Pour m = 4, 8, 16 et 32, on trouve en abcisse le nombre de blocs occupés de la mémoire et en ordonnée, la probabilité de trouver l'occupation correspondante. En paramètre sont portées les valeurs du taux de demandes, k .

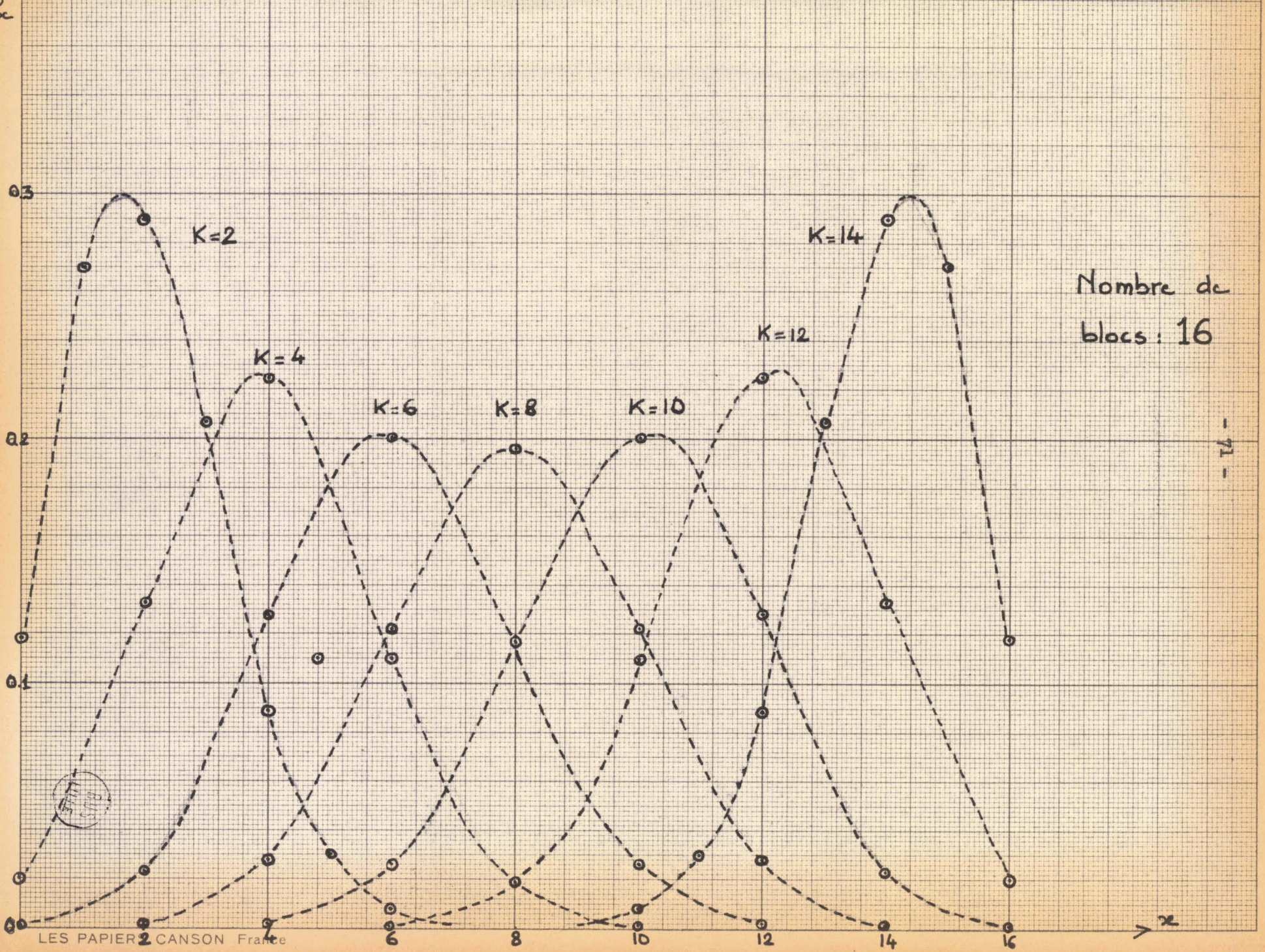
Sauf dans le cas où la longueur limitée de la file d'attente entraine un recul de la courbe, on constate une symétrie des courbes par rapport à la verticale définie par m/2 . La détermination approximative de la valeur moyenne par intégration graphique justifie la seconde hypothèse qui est à la base du calcul.



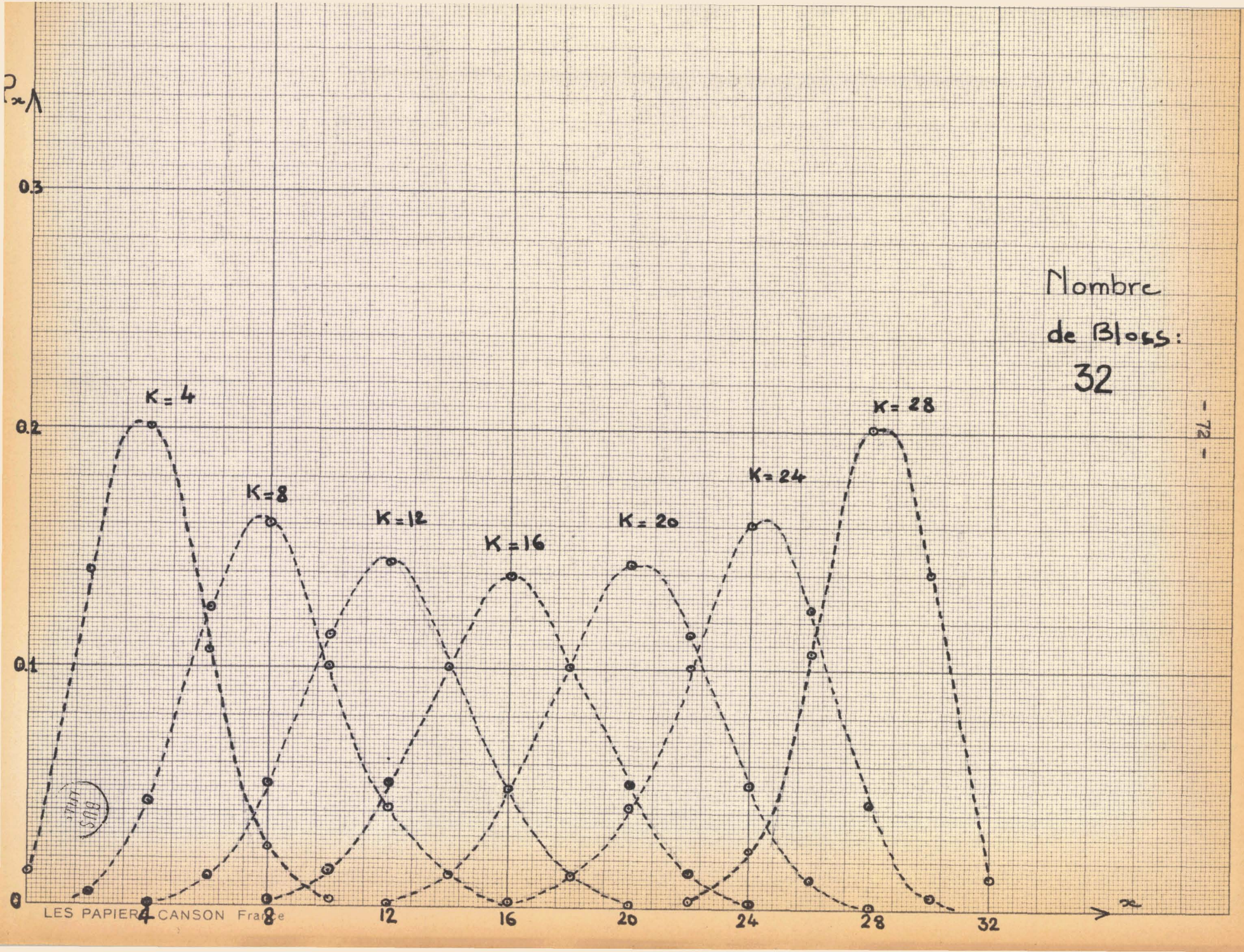
LES PAPIERS CANSON France



LES PAPIERS CANSON France



LES PAPIERS CANSON France



LES PAPIERS CANSON France

E - TEMPS DE SERVICE

Lorsqu'une demande se présente à la mémoire, il est impossible de déterminer au bout de combien de temps la mémoire aura satisfait au travail que lui définit cette demande. Il importe ici de bien préciser la différence qui existe entre les ordres de lecture et d'écriture. Lorsqu'une demande de lecture est présentée, le délai qui s'écoule entre l'instant où elle se présente et celui où elle est satisfaite correspond à une attente du demandeur ; au contraire, une demande d'écriture permet la libération du demandeur, une fois que tous les éléments du travail ont été confiés à la mémoire. Cette différence n'est en fait qu'apparente car, si une demande se présente pour une information qui attend d'être écrite dans la mémoire, cette demande devra être différée, tant que l'écriture ne sera pas terminée.

On ne considérera donc qu'une seule catégorie de demandes.

Le temps de service qui est bien plus lié à l'organisation du système qu'à la technologie de la mémoire peut être défini ainsi :

- C'est le temps qui s'écoule entre l'instant où la demande est formulée et l'instant où la mémoire désignée par la demande a terminé son cycle.

Ce temps de service est évidemment variable et l'on ne peut définir que des valeurs statistiques mais il est évident qu'il ne peut varier qu'entre deux limites :

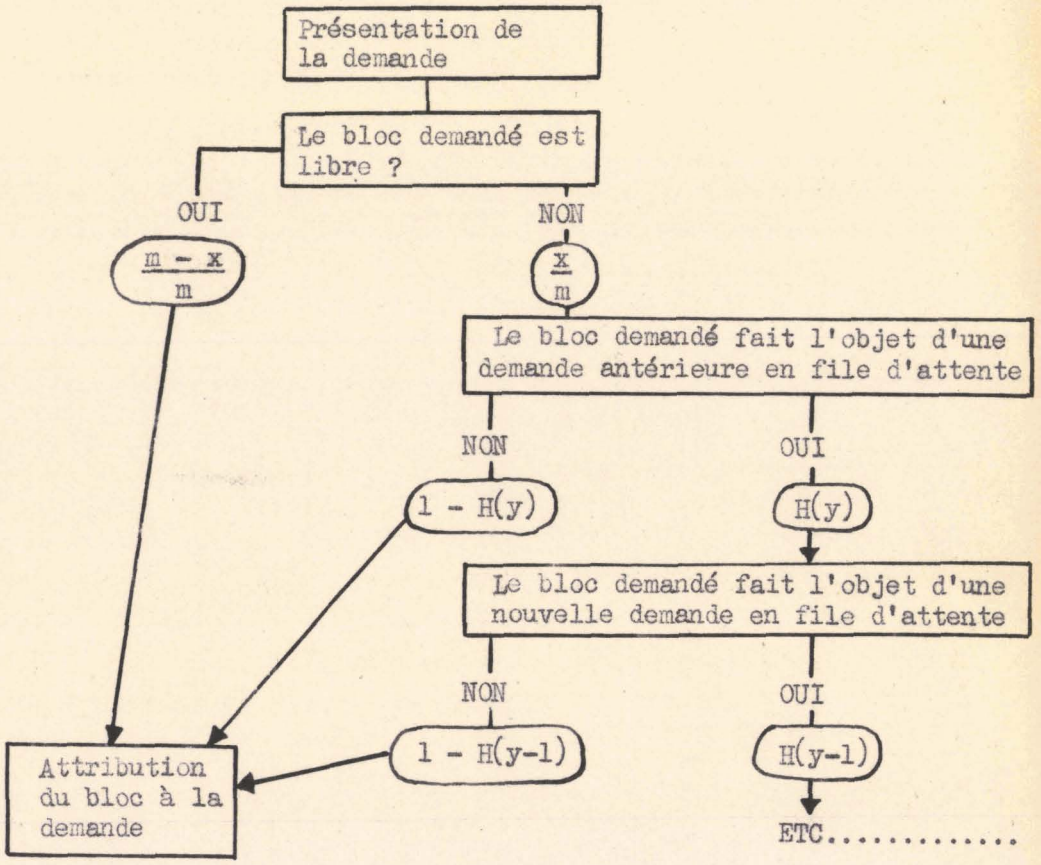
- Si le bloc demandé est libre, le temps de service est le temps de cycle de la mémoire. C'est la valeur la plus faible.
- Si le bloc désigné est occupé et fait l'objet de toutes les demandes placées dans la file d'attente et précédent la demande étudiée et si cette file d'attente est saturée, le temps de service sera de l'ordre de s T, s étant la longueur de la file et T le temps de cycle.

Le temps moyen d'attente sera la moyenne des temps possibles compris entre ces deux extrêmes, chacun d'eux étant affecté d'un coefficient représentant la probabilité qui lui correspond.

Plus que le temps moyen de service, il est utile de connaître la probabilité pour que ce temps soit inférieur à un certain délai fixé à partir des besoins des circuits utilisateurs. L'accroissement du débit de la mémoire, on l'a vu, ne correspond pas à un accroissement du temps de réponse. Il faut donc procéder à une recherche anticipée de l'information. Si l'on désire disposer

d'une mémoire efficace, l'information extraite de la mémoire doit être surement et immédiatement accessible à l'utilisateur au moment où il en a besoin. C'est la mesure du temps de service qui indique l'avance à prévoir pour la recherche des informations dans la mémoire.

Un organigramme des différentes éventualités qui peuvent se présenter à la suite de la formulation d'une demande est plus explicite qu'une longue description. Les termes figurant sur les flèches représentent les probabilités.



Il est possible d'affecter à chacun des chemins un temps de service :

Chemin suivi	temps de service
Le bloc demandé est libre	T
Le bloc demandé est occupé mais ne fait l'objet d'aucune demande en file d'attente	T + T/2
Le bloc demandé est occupé et fait l'objet d'une seule demande en file d'attente.	T + T/2 + T
.....	
Le bloc demandé est occupé et fait l'objet de j demandes en files d'attente	T + T/2 + j.T

La valeur T/2 ajoutée au temps de service correspond au délai nécessaire pour qu'un bloc au travail se trouve libéré. Il s'agit d'une valeur moyenne et, contrairement à tous les autres résultats obtenus jusqu'à présent, toutes les valeurs intermédiaires ont un sens.

Le temps de service s'exprime comme une fonction de x, nombre de blocs occupés et y, nombre de demandes dans la file d'attente. En partant des lois de distribution déjà établies pour x et y appliquées à la configuration étudiée, on peut définir une valeur moyenne pour chacun des temps d'attente.

La valeur moyenne de x est k puisque toutes les demandes qui se présentent sont satisfaites. La probabilité de satisfaction immédiate est donc :

$$P_T = \frac{m - k}{m}$$

Dans les autres cas, la longueur de la file d'attente intervient de telle sorte que, les équations définissant H(y) n'étant pas linéaires par rapport à y, il faut calculer le temps d'attente pour chaque distribution possible dans la file d'attente et prendre la moyenne des temps pondérés par la probabilité que la file se trouve dans l'état considéré.

On appelle P_j la probabilité pour que j demandes précédent, dans la file d'attente, la demande qui se présente. La file d'attente contient s positions. On appelle P_{ji} la probabilité partielle pour que, la file d'attente contenant au total i demandes, j d'entre elles concernent la demande qui se présente.

$$P_{ji} = H(i).H(i-1). \ldots\ldots .H(i - j + 1). (1 - H(i - j))$$

Si Q_i représente la probabilité pour que la file contienne i demandes :

$$P_j = \frac{\sum_{i=0}^{i=s} P_{ji} \cdot Q_i}{s + 1}$$

Si on considère que la file d'attente contient, en moyenne, $\bar{y}$ demandes, on peut obtenir une expression simplifiée :

$$P_j = H(\bar{y}). H(\bar{y} - 1). \ldots\ldots . H(\bar{y} - j + 1).(1 - H(\bar{y} - j)$$

Cette formule simplifiée a été employée pour déterminer le temps de service dans les configurations étudiées jusqu'à présent. A partir des résultats numériques obtenus, on peut définir la "profondeur" de la recherche anticipée en choisissant un temps d'avance tel que la probabilité pour une demande de ne pas être servie au bout de ce temps soit inférieure à une valeur fixée à priori.

Calcul numérique du temps de service.

En prenant toujours les mêmes configurations de 4, 8, 16 et 32 blocs et en considérant, à nouveau, que la longueur de la file est suffisante pour que sa saturation soit très improbable, on peut évaluer numériquement, pour divers taux de demandes, la distribution des temps de service.

L'unité de temps portée en abcisse est la durée T du cycle technologique d'un bloc. En ordonnée ont été portées les probabilités correspondantes aux différentes valeurs des temps de service.

Dans chaque cas, le dernier paramétre est le taux de demandes ou, ce qui revient au même, la longueur moyenne de la file d'attente puisque ces deux quantités sont liées.

Le tableau ci-dessous donne pour chaque cas la valeur du temps de service moyen:

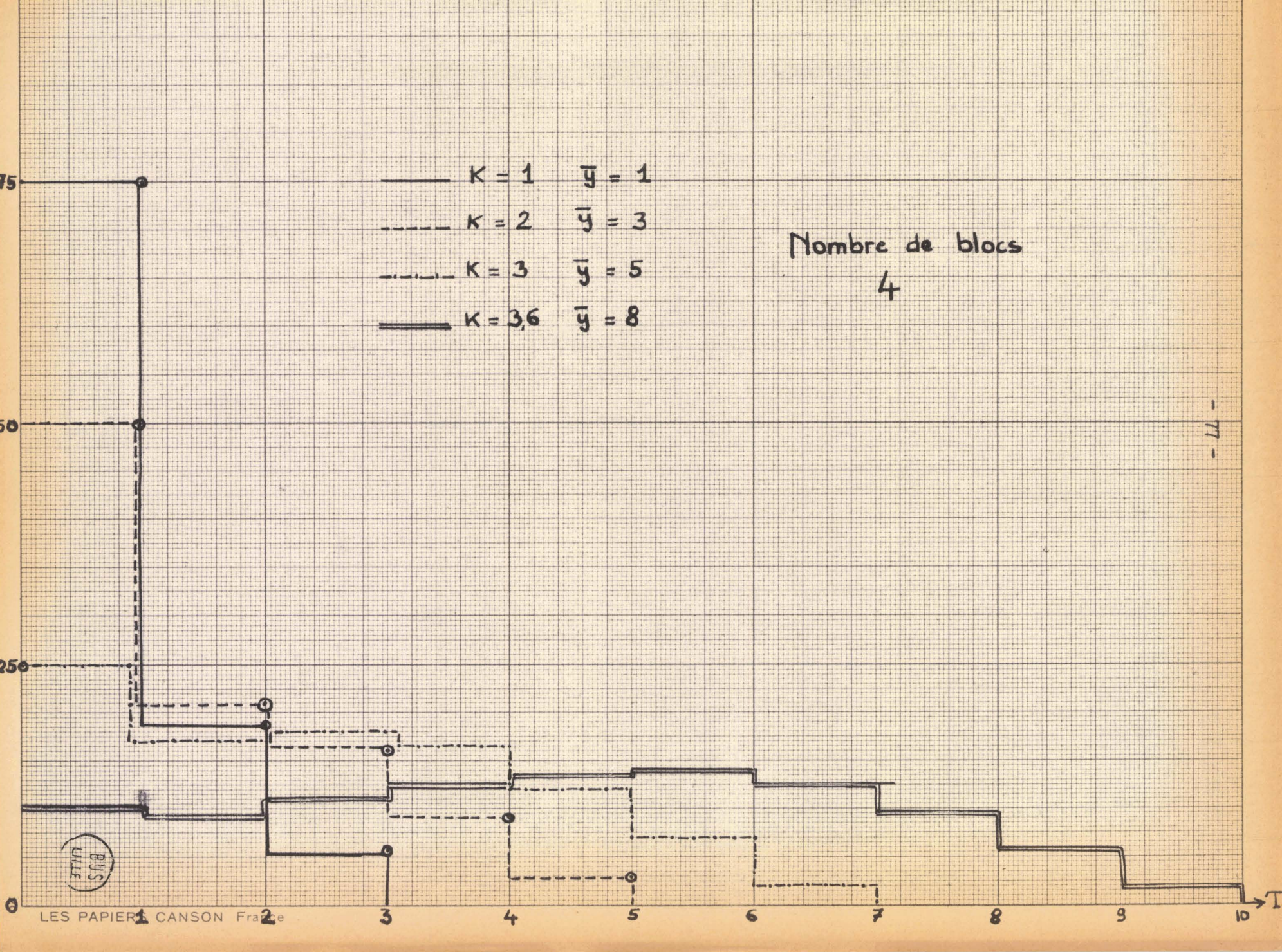
$$\overline{T} = \sum_{j=o}^{j=n} T_j . P(T_j)$$

Théoriquement n doit tendre vers l'infini. On limite la somme aux termes pour lesquels $P(T_j)$ est supérieur à 0,001.

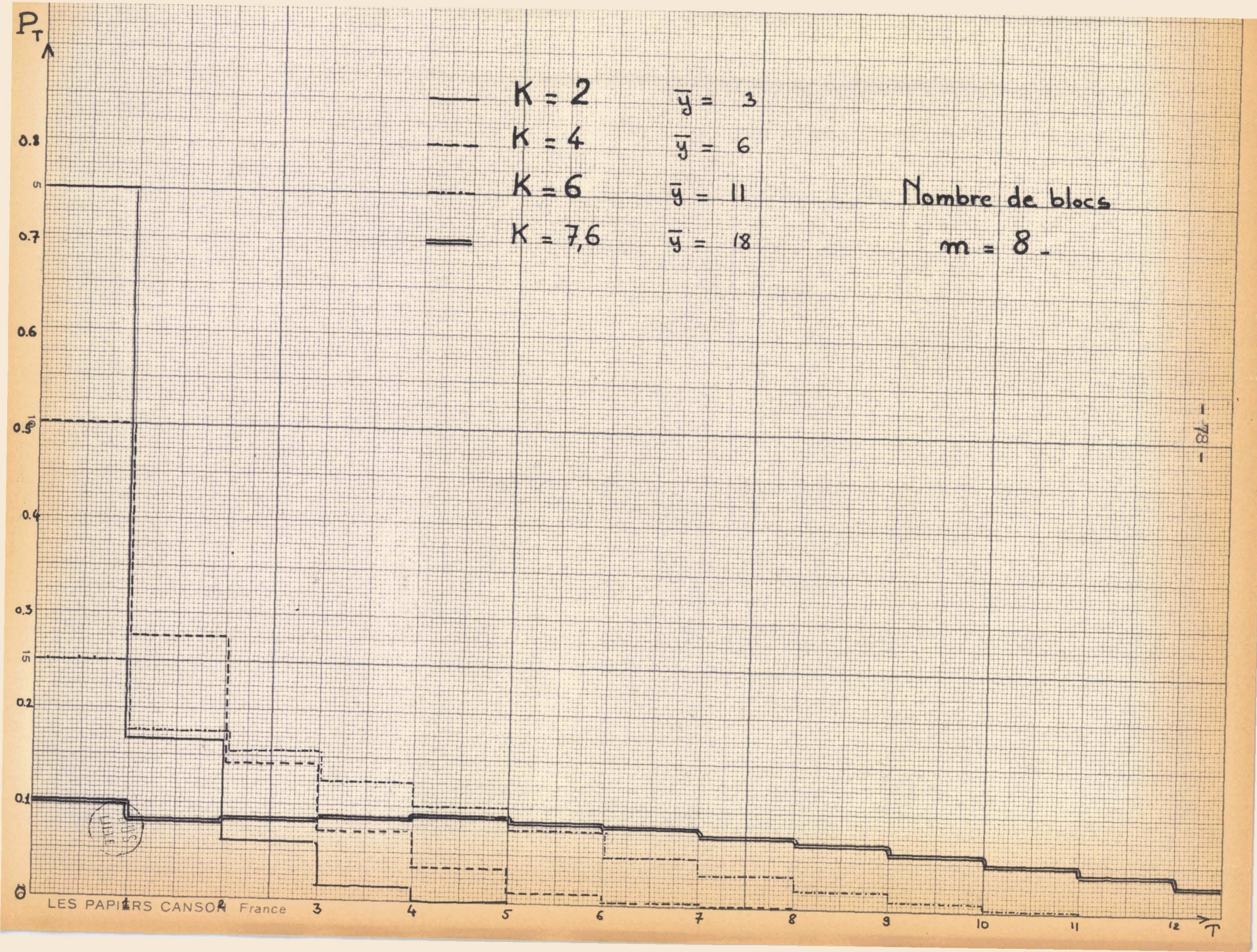
NOMBRE DE BLOCS

	k = 1	k = 2	k = 3	k = 3,6
m = 4	0,814	1,405	2,671	4,514
	K = 2	k = 4	k = 6	k = 7,6
m = 8	0,828	1,478	2,965	6,055
	k = 4	k = 8	k = 12	
m = 16	0,839	1,504	3,147	
	k = 8	k = 16	k = 24	
m = 32	0,854	1,532	3,248	

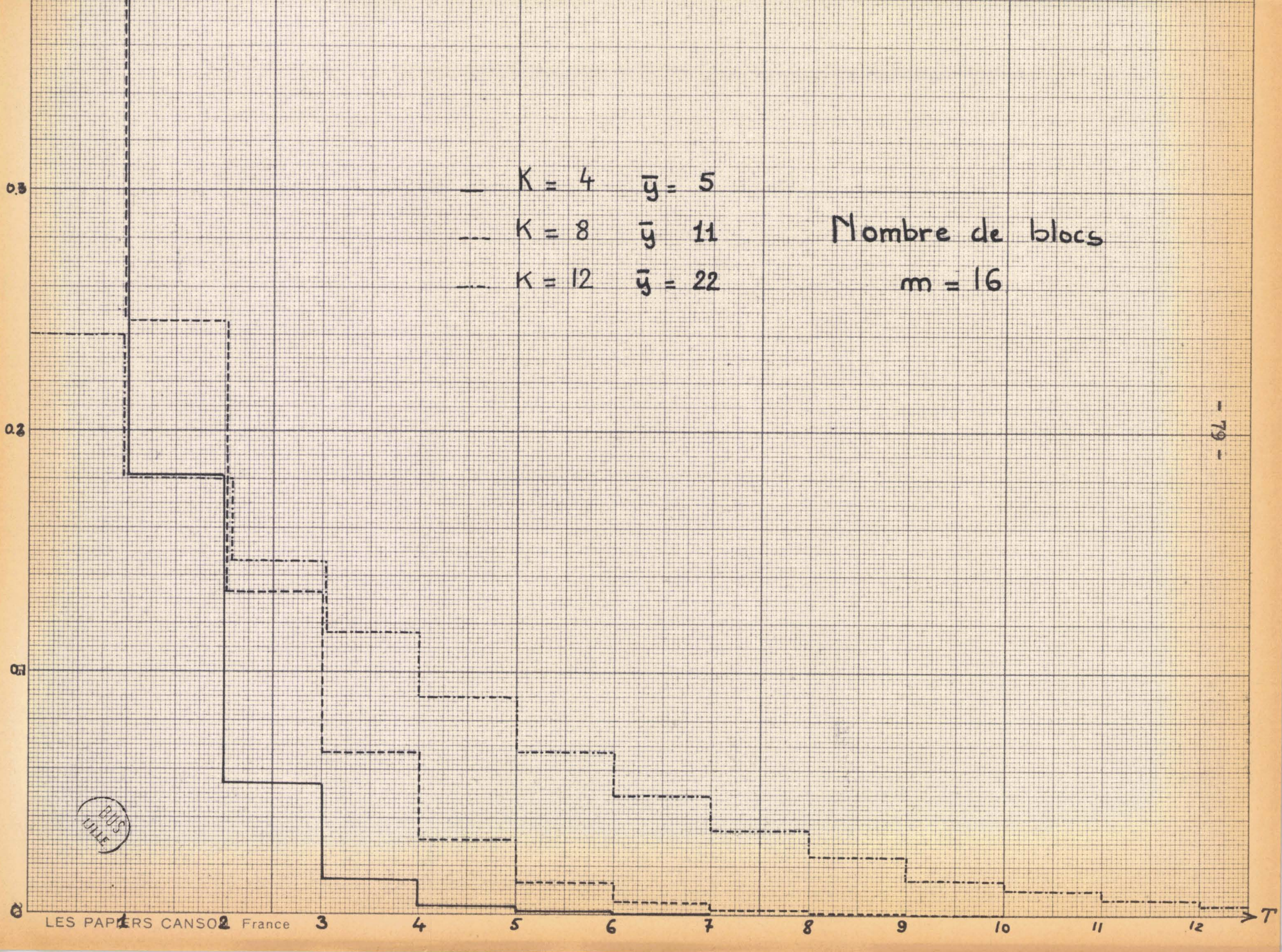
L'accroissement du temps de service entraine la nécessité d'augmenter la profondeur d'anticipation ce qui représente en soi une difficulté mais ce qui peut devenir une cause de diminution du débit utile dans un système où les ruptures de séquence sont nombreuses.



LES PAPIERS CANSON France

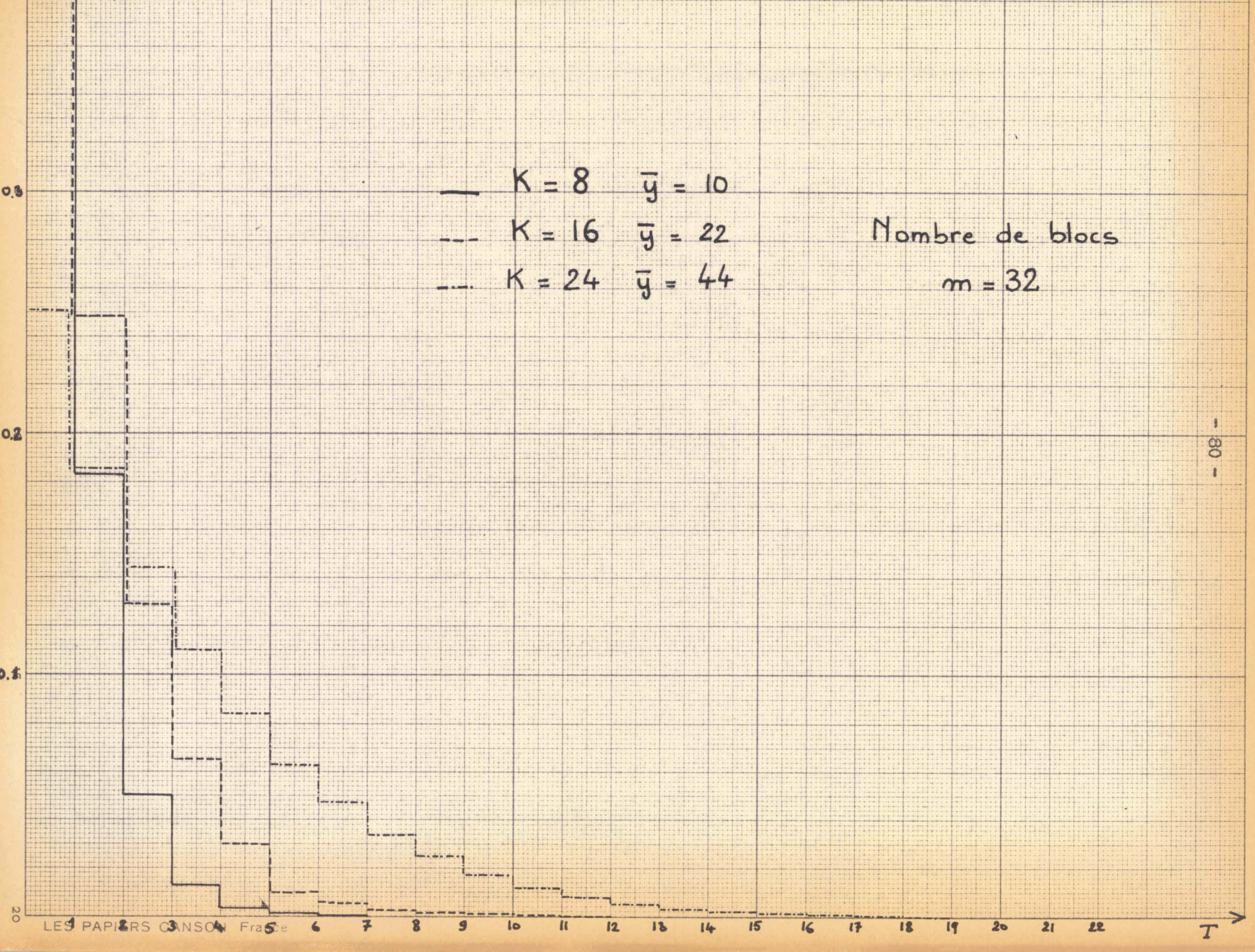


LES PAPIERS CANSON France



LES PAPIERS CANSO France

BUS LILLE



LES PAPIERS CANSON France

L'ensemble des résultats obtenus permet de définir les premiers éléments du choix d'un système de stockage.

Les termes que l'on suppose fixés par ailleurs sont:

- La capacité totale de la mémoire.
- Le temps de cycle de la technologie employée.
- Le taux moyen de demandes en provenance des organes directement connectés à la mémoire.

Imaginons, par exemple que ce taux moyen de demandes soit de 4. Cette valeur figure dans les quatre configurations étudiées et l'on peut donc opter pour chacune d'elles.

Nombre de blocs	occupation de la mémoire	Longueur de la file d'attente Valeur moyenne		temps de service
4	100 %	infinie		infini
8	50 %	5,2	14 (1)	2,478
16	25 %	4,5	12	1,839
32	12,5 %	3,9	9	1,452

(1) Le terme porté dans cette colonne représente la longueur maximum de la file en imposant que la probabilité de saturation de celle-ci soit inférieure à 1 % /

Dans le cas présent, si les capacités le permettent, on choisira de scinder la mémoire en 8 blocs en remarquant que la longueur supérieure de la file est largement compensée par la simplification des circuits d'aiguillage.

Il peut arriver que le problème se pose en d'autres termes et, en particulier, que l'on ait le choix entre plusieurs technologies de mémoire. La question se pose alors ainsi: Faut-il grouper un nombre élevé de blocs lents ou un nombre plus restreint de mémoires plus rapides. Il n'est pas possible de raisonner de manière aussi rigoureuse mais on peut chiffrer le choix que l'on fait à l'intérieur de deux limites: A débit donné, on ne peut réduire le nombre de blocs que par une augmentation de la longueur de la file d'attente, longueur qui risque de devinir prohibitive; à l'inverse, si on opte pour des éléments lents en grand nombre, le temps de service ne peut être inférieur au temps de cycle et chaque fois que l'anticipation des demandes ne pourra se faire, il en résultera une attente pour le canal demandeur.

CHAPITRE I V

COMPORTEMENT DE LA MEMOIRE EN REGIME TRANSITOIRE.

INTRODUCTION

Dans toute l'étude précédente, on a fait l'hypothèse que le régime de fonctionnement de la mémoire était stationnaire. Ceci s'est exprimé, dans les calculs par le fait que la distribution des probabilités était fixes et donc que les dérivées par rapport au temps de ces mêmes probabilités étaient nulles. Du point de vue de l'exploitation de la mémoire, cette hypothèse revient à supposer le programme uniquement séquentiel.

Il existe deux grandes familles d'évènements qui peuvent détruire le caractère séquentiel du programme :

- Les instructions de branchement, principalement lorsqu'elles sont conditionnelles et que le choix qui sera opéré par les circuits ne peut être prévu.
- Les interruptions de programme d'origine externe dont l'apparition est fortuite et indépendante du programme en cours de traitement.

Dans les deux cas, les demandes en cours ou déjà satisfaites et non encore exploitées doivent être annulées ou stockées provisoirement pour faire place à de nouvelles demandes liées au programme lancé.

Au moment où se présente la rupture de séquence, l'état de la mémoire et de ses circuits associés peut être défini par les valeurs statistiques suivantes :

- Le nombre moyen de blocs occupés est $\bar{x}$.
- La longueur de la file d'attente a sa longueur moyenne $\bar{y}$.
- La profondeur d'anticipation du programme est $\bar{p}$; par profondeur d'anticipation, on entend le décalage, compté en nombre d'instructions, entre l'instruction en cours et celle qui est totalement disponible dans la file d'attente d'instructions.

La rupture de séquence invalide :

- Tout ou partie du contenu de la file d'attente.
- Toutes les instructions non encore traitées.
- Tout ou partie des travaux en cours dans les blocs au travail.

L'invalidation de toutes les informations inutiles contenues dans les registres extérieurs à la mémoire s'obtient par une simple remise à zéro. Il faut aussi repositionner les indicateurs de niveau ou d'occupation des files d'attente. Il est par contre, impossible d'agir immédiatement sur les $\bar{x}$ blocs au travail. Pour des raisons technologiques, (lecture destructive) il est nécessaire de les laisser terminer leur cycle.

L'absence d'informations obtenues par anticipation, le fait qu'un certain nombre de blocs poursuivent un travail devenu inutile entraînent une diminution des performances de la mémoire. On fera l'étude du comportement de la mémoire en régime transitoire en généralisant la méthode appliquée au régime stationnaire ; cette méthode ne pouvant fournir le résultat cherché, on tentera de décrire ce comportement par un modèle probabiliste quantifié.

A - CALCUL DU COMPORTEMENT DE LA MEMOIRE EN REGIME TRANSITOIRE.

Il est normal de reprendre, pour les appliquer aux régimes transitoires, les résultats de l'étude effectuée en régime stationnaire. Au cours de la définition des lois régissant le fonctionnement de la mémoire, sont apparues des équations différentielles de la forme :

$$P'_j(t) = A.P_j(t) + B.P_{j-1}(t) + C.P_{j+1}(t)$$

On avait alors posé que, les probabilités étant fixes, la dérivée était nulle, ce qui conduisait à un système d'équations linéaires. Les m+1 équations associées à la condition $\sum P_j = 1$, fournissaient directement la distribution cherchée.

Il suffit alors de prendre en compte la dérivée de P_j et, en partant de conditions initiales connues, d'intégrer le système différentiel obtenu.

En fait, on va constater sur un exemple que l'analyse différentielle ne peut conduire à un résultat car elle est incapable de tenir compte de l'aspect nécessairement échantilloné du traitement des demandes.

Hypothèses initiales ;

- La mémoire est totalement disponible.
- Le taux de demandes reste constant.
- La file d'attente n'intervient pas.
- Dans l'intervalle (o, T) qui sera étudié, il ne peut y avoir de libération puisqu'aucune des demandes prises en compte n'a été présentée avant l'instant O.

Si $P_x\ (t + dt)$ représente la probabilité d'avoir x blocs au travail à l'instant t, on peut écrire :

$P_x\ (t + dt)$ = $P_{x-1}\ (t)$. Probabilité d'une demande. Probabilité de satisfaction de cette demande.

+ $P_x\ (t)$. Probabilité de non demande ou Probabilité d'une demande. Probabilité de non satisfaction.

$$P_x\ (t + dt) = P_{x-1}\ (t).\left(1 - \frac{x-1}{m}\right).\ k\ dt$$

$$+ P_x\ (t).\left(1 - k\ dt + \frac{x}{m}\ .\ k\ dt\right)$$

Ce qui conduit à l'équation différentielle :

$$P'_x\ (t) = k.\quad P_x\ (t)\ \frac{x-m}{m} \quad + \quad P_{x-1}\ (t)\ \frac{m-x+1}{m}$$

La solution d'une telle équation est :

$$P_x\ (t) = e^{-k\frac{m-x}{m}t}\ .\int_0^t e^{k\frac{m-x}{m}t}\ .\ P_{x-1}\ (t)\ dt + C.\ e^{-k\frac{m-x}{m}t}$$

Si on connait P_{x-1}, on peut calculer P_x . C est une constante d'intégration qui peut être déduite de l'examen des conditions initiales.

Pour faire le calcul, il faut connaître $P_0\ (t)$, c'est-à-dire la loi de variation de probabilité pour qu'aucun bloc de la mémoire ne soit au travail. On peut, compte tenu des hypothèses initiales que la mémoire n'était pas occupée, assurer que la première demande sera certainement prise encompte et donc la fonction P_0 devient nulle pour toute valeur de t différente de zéro.

On trouve alors :

$$P_1\ (t) = C_1.\ e^{-k\frac{m-1}{m}t}$$

On constate qu'alors la probabilité de n'avoir qu'un seul bloc occupé tend vers zéro et non pas vers la valeur obtenue par l'étude du régime stationnaire qui est finie. Ceci provient du fait que la validité des équations n'est assurée que dans l'intervalle (0 , T).

La probabilité P_2 se déduit de P_1 ; elle est de la forme :

$$P_2(t) = C_2 . e^{-k\frac{m-2}{m}t} + \int_0^t e^{k\frac{m-2}{m}t} . C_1 . e^{-k\frac{m-1}{m}t} . dt$$

$$P_2 = e^{-k\frac{m-2}{m}t} . \left(C_2 - C_1 \frac{m}{k} e^{\frac{-k}{m}t} \right)$$

Le terme exponentiel principal de la probabilité P_j est :

$$e^{-k\frac{m-j}{m}t}$$

Ce terme permet de connaître la vitesse d'évolution de la probabilité P_j mais pour déterminer complètement la loi, il faut aussi calculer les constantes d'intégration. Ceci ne peut se faire que par l'examen des conditions initiales, c'est-à-dire l'état d'occupation de la mémoire à l'instant choisi comme origine des temps. Si ce choix est exact pour P_0 et pour P_1, il ne l'est plus pour les autres termes car la probabilité d'occupation de j blocs reste identiquement nulle pendant les j-1 premiers cycles mineurs de l'unité de contrôle.

Il faudrait donc décaler l'origine des temps d'une valeur différente pour chacun des termes à définir. Or ce décalage, qui ne peut être que fini, si on veut qu'il ait un sens, représente l'élément différentiel choisi au départ .

La méthode ne peut donc pas s'appliquer aux régimes transitoires.

B - SOLUTION PAR L'EMPLOI DE VARIABLES DISCONTINUES.

L'emploi d'équations différentielles où l'élément dt représente une fraction non négligeable de la durée du phénomène étudié est évidemment sujet à caution. A l'intervalle de temps dt, il faut substituer la durée du cycle de la logique de gestion de la mémoire. Au cours de chacun de ces cycles, une seule demande peut être traitée, ce qui revient à dire qu'il ne peut se produire qu'une seule libération et une seule mise au travail de l'un des blocs.

L'analyse de l'évolution des termes caractéristiques de la mémoire ne peut alors se faire qu'en pas à pas, la valeur de ces termes se déduisant par des relations probabilistes de leur valeur au cours du cycle précédent.

A l'instant t_i, on suppose que la mémoire comporte j blocs occupés. On prendra comme valeur du temps de cycle de l'unité de contrôle, une durée Δt

égale à T/m. Le choix de cette valeur n'est pas critique ; il indique que si m demandes se présentent au cours d'un cycle technologique et s'il n'y a pas de conflit, ces demandes seront toutes prises en compte. Pour des valeurs de m assez peu élevées, il correspond aux possibilités de la logique associée ; dans le cas où la valeur de m est plus élevée, une valeur plus élevée elle aussi, conviendrait car alors le taux relatif de demandes étant faible, la probabilité pour que ces demandes se présentent en un groupe serré devient très faible.

La probabilité de libération d'un bloc est égale à :

$$\frac{\Delta t}{T} \cdot j = \frac{j}{T} \cdot \frac{T}{m} = \frac{j}{m}$$

La probabilité de demandes extérieures est :

$$k \cdot \frac{\Delta t}{T} = k \cdot \frac{T}{m} \cdot \frac{1}{T} = \frac{k}{m}$$

La probabilité de satisfaction d'une demande est égale à :

$$\frac{m - j}{m}$$

On dira que la mémoire est dans l'état j si j de ses blocs sont occupés. Au cours d'un cycle Δ t, si la mémoire se trouvait initialement dans l'état j, elle ne peut parvenir qu'à l'un des trois états : j - 1, j ou j + 1. Il est possible d'évaluer la probabilité pour que, partant de l'état j, elle se trouve dans chacun de ces trois états finaux.

On notera Pj, k la probabilité de transition de l'état j dans l'état k au cours d'un cycle unique.

Les termes Pj, j+1, Pj,j et Pj,J-1 se définissent alors ainsi :

- Pj, j+1 Probabilité pour que : il y ait une demande,
  elle soit satisfaite,
  il n'y ait pas de libération.

- Pj,j Probabilité pour que : il y ait une demande,
  elle soit satisfaite,
  il y ait une libération.
  ou
  il n'y ait pas de demande
  il n'y ait pas de libération.
  ou
  il y ait une demande,
  elle ne soit pas satisfaite,
  il n'y ait pas de libération.

- $P_{j,\,j-1}$ Probabilité pour que : il n'y ait pas de demande,
il y ait une libération
ou
il y ait une demande,
elle ne soit pas satisfaite,
il y ait une libération.

$$P_{j,\,j+1} = \left(\frac{k}{m} \cdot \frac{m-j}{m}\right) \cdot \left(1 - \frac{j}{m}\right)$$

$$P_{j,\,j} = \left(\frac{k}{m} \cdot \frac{m-j}{m}\right) \cdot \frac{j}{m} + \left(\frac{m-k}{m} + \frac{k}{m} \cdot \frac{j}{m}\right) \cdot \frac{m-j}{m}$$

$$P_{j,j-1} = \left(\frac{m-k}{m} + \frac{k}{m} \cdot \frac{j}{m}\right) \cdot \frac{j}{m}$$

Formules que l'on peut encore écrire en effectuant :

$$P_{j,j+1} = \frac{k}{m^3} \cdot (m-j)^2$$

$$P_{j,j} = \frac{m-j}{m^3} \cdot (2kj + m(m-k))$$

$$P_{j,j-1} = \frac{j}{m^3} \cdot (m^2 + k.(j-m))$$

Dans le cas où la mémoire est vide ou saturée, les probabilités de transition ont une expression un peu simplifiée :

Cas de la mémoire vide :

La probabilité pour que la demande qui se présente soit satisfaite est 1.

$$P_{0,0} = \frac{m-k}{m}$$

$$P_{0,1} = \frac{k}{m}$$

Ces deux probabilités ne dépendent que du taux de demandes.

1) Cas de la mémoire saturée :

La probabilité pour que la mémoire reste saturée s'exprime par la condition pour qu'il n'y ait pas de libération ou que, s'il y a une libération, une demande se présente qui concerne le bloc libéré. L'hypothèse faite sur la période Δt exclue la première possibilité.

La probabilité pour qu'il n'y ait plus que m-1 blocs au travail, s'exprime

par la condition qu'il n'y ait pas de demande ou que celle qui se présente ne concerne pas l'unique bloc qui doit obligatoirement se trouver libéré.

$$P_{m,m} = \left(1 - \frac{m}{m}\right) + \frac{m}{m} \cdot \frac{k}{m} \cdot \frac{1}{m} = \frac{k}{m^2}$$

$$P_{m,m-1} = \frac{m}{m} \cdot \left(1 - \frac{k}{m} + \frac{k}{m} \cdot \frac{m-1}{m}\right) = \frac{m^2 - k}{m^2}$$

On peut alors écrire la matrice de Markoff représentant la totalité des probabilités de transition :

$$A = \begin{pmatrix} m^2\frac{m-k}{m^3} & m^2\frac{k}{m^3} & 0 & \cdots & 0 \\ \frac{1}{m^3}(m^2-k(m-1)) & \frac{m-1}{m^3}(2k+m(m-k)) & \frac{k}{m^3}(m-1)^2 & \cdots & 0 \\ 0 & \frac{2}{m^3}(m^2+k(2-m)) & \frac{m-2}{m^3}(4k+m(m-k)) & \cdots & 0 \\ 0 & 0 & \frac{3}{m^3}(m^2+k(3-m)) & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 \cdots & \frac{1}{m^3}(2k(m-1)+m(m-k)) & 0 \\ 0 & 0 & 0 \cdots & m\frac{m^2-k}{m^3} & m\frac{k}{m^3} \end{pmatrix}$$

Seules la diagonale principale et ses deux diagonales voisines sont différentes de zéro.
Le coefficient $1/m^3$ peut être sorti de la matrice puisqu'il apparaît dans tous les termes.

L'étude du comportement de la mémoire en régime transitoire peut se poursuivre en appliquant la transformation définie par la matrice de Markoff à un vecteur initial connu. Ce vecteur à m+1 dimensions représente l'état dans lequel se trouve la mémoire lors de la reprise du nouveau programme.

Chaque cycle se déduit du précédent par le produit du vecteur par la matrice et donc, à l'instant t_i, au bout de i cycles, le vecteur qui représente la distribution des probabilités d'occupation des différents blocs peut s'écrire :

$$\left[E_i\right] = \left[A\right]^i \cdot \left[E_0\right] .$$

2) Cas de la mémoire totalement libre.

Dans le cas où la technologie permet d'interrompre le travail des blocs au cours de son exécution si ce travail est inutile, tous les blocs se trouveront au repos dès l'instant initial.

Le vecteur initial s'écrira alors :

$$E_0 = (1, 0, 0, \ldots\ldots 0)$$

Lorsque i tend vers l'infini, le vecteur E_i doit tendre vers la distribution des probabilités obtenue par l'étude des cas stationnaires. Afin de chiffrer la manière dont évolue l'occupation de la mémoire entre l'instant initial et l'instant où le régime stationnaire est atteint, il suffit de calculer, pour chaque valeur de i, la transformée du vecteur E_0 par la matrice A^i ; la loi de variation de chacune des composantes du vecteur indiquera le comportement de la mémoire lors de ce régime transitoire.

Si on constate que le régime stationnaire s'établit rapidement, c'est-à-dire en un temps de l'ordre de T, durée du cycle technologique, il n'est pas nécessaire de faire intervenir la file d'attente qui se contente d'enregistrer les demandes refusées mais qui ne peut les présenter utilement avant que le premier bloc mis au travail ait terminé son cycle complet.

Si, au contraire, le temps d'établissement porte sur plusieurs cycles consécutifs, il faudra faire intervenir la file d'attente dans les calculs. Il est néanmoins prévisible que, dans le cas présent, le rôle de la file d'attente dans la reprise d'un fonctionnement normal, est réduit car elle n'intervient de manière prépondérante que lorsqu'elle est fortement chargée ce qui signifie que de nombreuses demandes ont été refusées donc que la plus grande partie des blocs est occupée ; c'est qu'alors la mémoire a repris un débit élevé proche de la valeur stationnaire.

Afin de chiffrer le rôle du coefficient k, taux de demandes, on a effectué le calcul des diverses puissances de la matrice A dans le cas d'une mémoire partagée en quatre blocs. Le calcul a été poursuivi jusqu'à ce que les termes apparaissant sur chaque colonne se stabilisant autour de leur valeur d'équilibre.

Les matrices de transition pour k = 1, 2 et 3 s'écrivent :

$$A=\frac{1}{64}\begin{bmatrix}48 & 16 & 0 & 0 & 0\\ 13 & 42 & 9 & 0 & 0\\ 0 & 28 & 32 & 4 & 0\\ 0 & 0 & 45 & 18 & 1\\ 0 & 0 & 0 & 60 & 4\end{bmatrix} \qquad A=\frac{1}{64}\begin{bmatrix}32 & 32 & 0 & 0 & 0\\ 10 & 36 & 18 & 0 & 0\\ 0 & 24 & 32 & 8 & 0\\ 0 & 0 & 42 & 20 & 2\\ 0 & 0 & 0 & 56 & 8\end{bmatrix} \qquad A=\frac{1}{64}\begin{bmatrix}16 & 48 & 0 & 0 & 0\\ 7 & 30 & 27 & 0 & 0\\ 0 & 20 & 32 & 12 & 0\\ 0 & 0 & 39 & 22 & 3\\ 0 & 0 & 0 & 52 & 12\end{bmatrix}$$

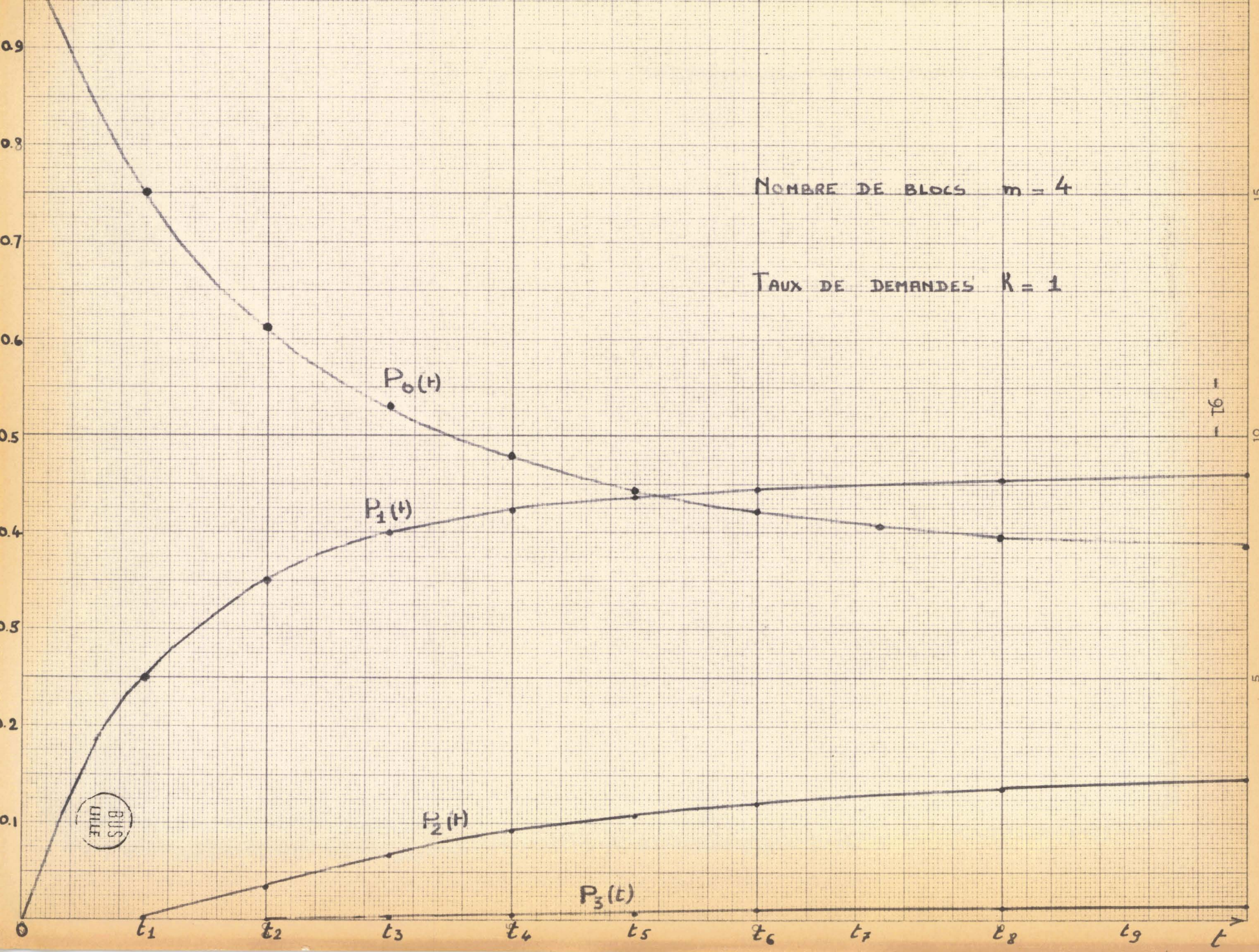
Le calcul a été fait pour toutes les puissances de A jusqu'à la puissance 10 ; pour cette valeur, les probabilités deviennent pratiquement stationnaires et indépendantes du choix du vecteur origine.

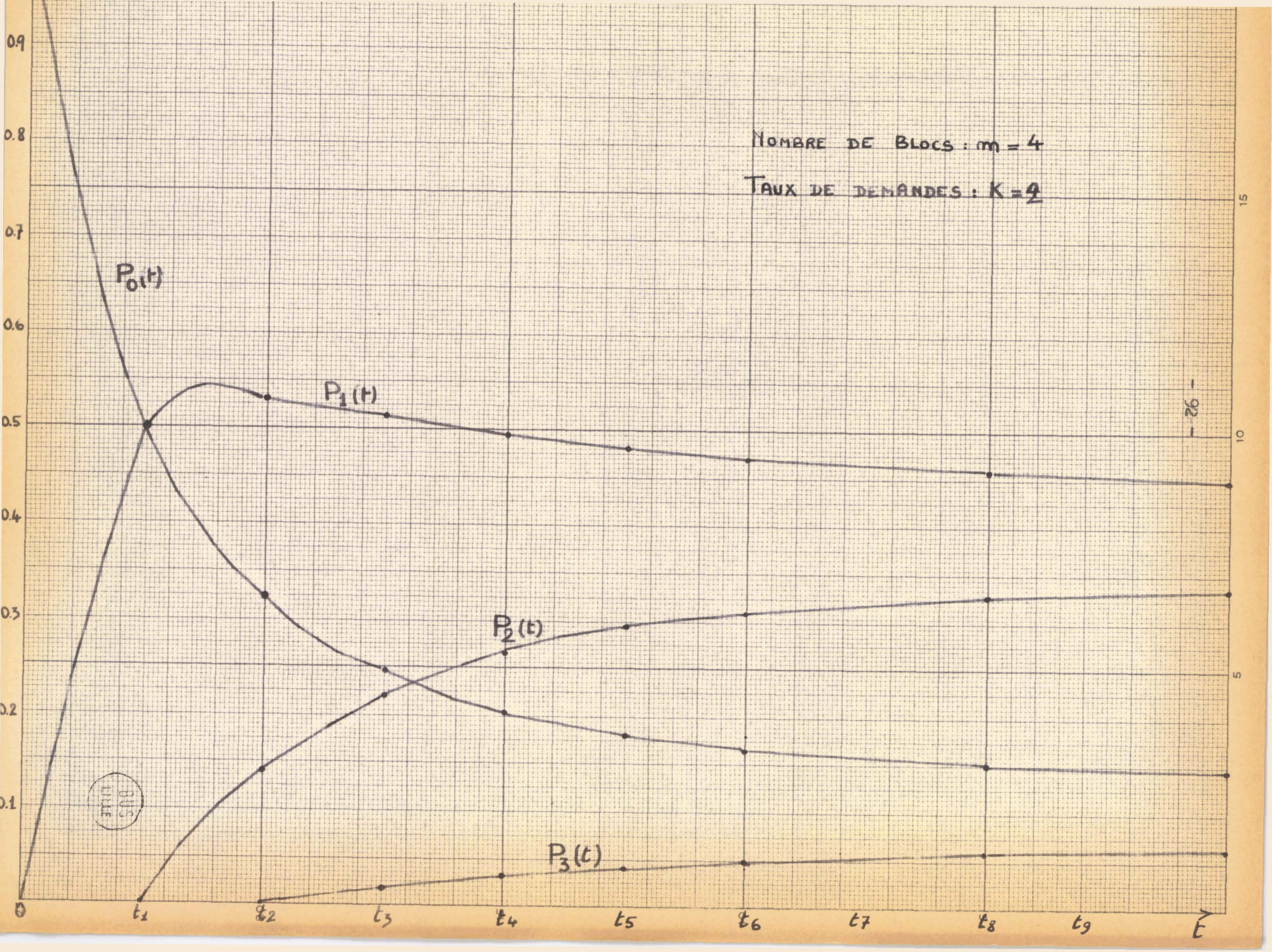
Il résulte de celà que la méthode permet la détermination des distributions, non seulement en régime transitoire mais aussi en régime stationnaire. Afin de contrôler la validité des calculs précédents on a comparé les débits obtenus par les deux méthodes.

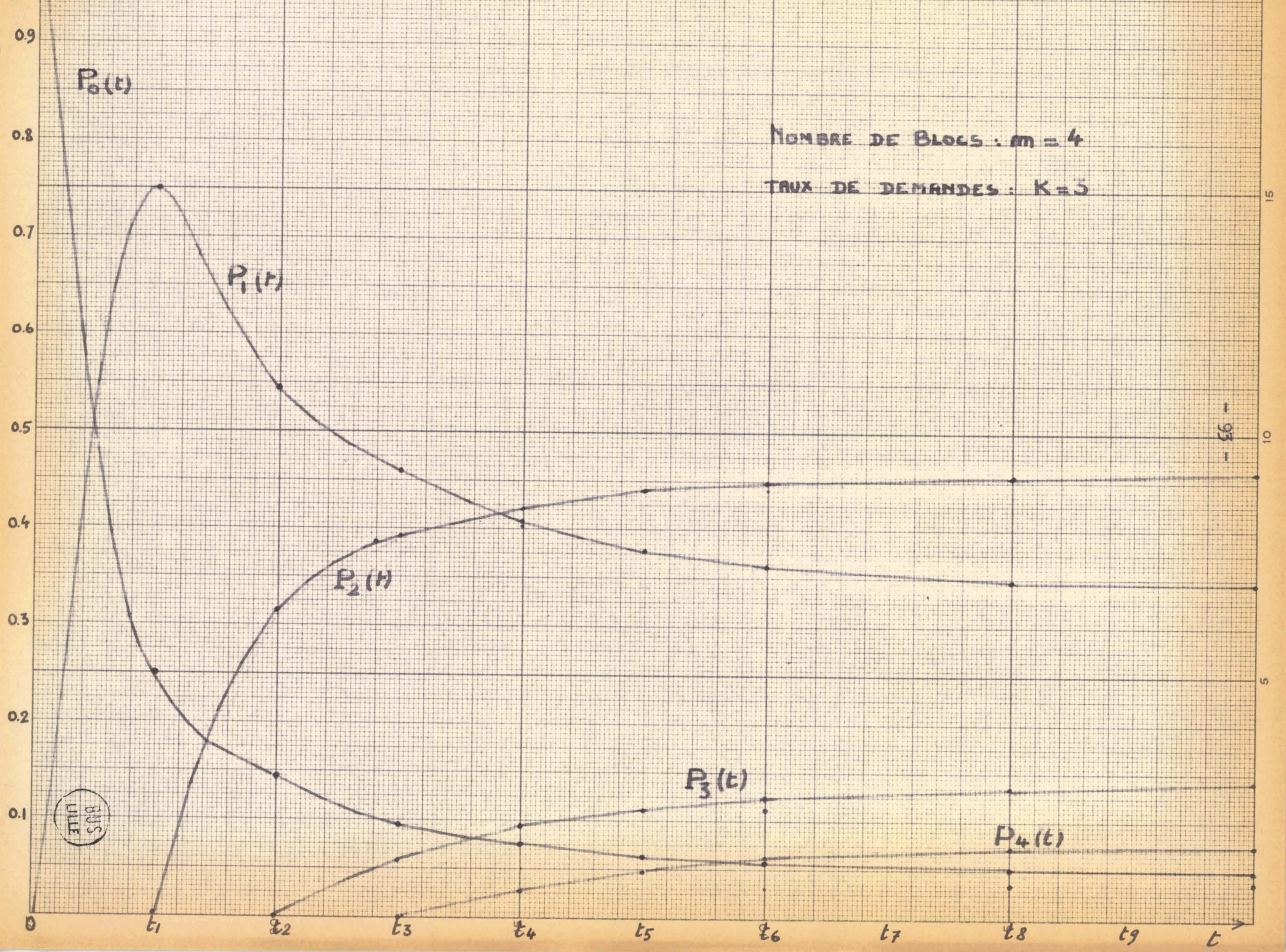
	Régime stationnaire	Chaînes de Markoff
taux k = 1	0,78 mots par cycle	0,81 mots par cycle
taux k = 2	1,27 mots par cycle	1,31 mots par cycle
taux k = 3	1,63 mots par cycle	1,69 mots par cycle.

La différence, qui joue toujours dans le même sens, ne peut être attribuée aux seules erreurs d'intégration graphique et au fait que les probabilités obtenues au bout de 10 itérations ne sont pas rigoureusement fixes. Elle indique la limite de validité des hypothèses simplificatrices faites dans le cas du régime stationnaire. La simplicité relative de cette méthode par rapport à la méthode matricielle fait qu'elle conserve un grand intérêt.

Les courbes des pages suivantes indiquent, pour chacun des cas, les variations en fonction du temps, des probabilités d'occupation. L'unité sur l'axe des abcisses est la durée Δt du cycle de l'unité de contrôle. L'origine des temps est supposée coincider rigoureusement avec l'instant où doit se produire le nouveau départ de la mémoire.


NOMBRE DE BLOCS m = 4
TAUX DE DEMANDES K = 1
$P_0(t)$
$P_1(t)$
$P_2(t)$
$P_3(t)$
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
15
10
5
T_{61}
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t_9
t


Nombre de blocs : m = 4
Taux de demandes : K = 4
$P_0(t)$
$P_1(t)$
$P_2(t)$
$P_3(t)$
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t_9
t
15
10
5
26


NOMBRE DE BLOCS : m = 4
TAUX DE DEMANDES : K = 5
$P_0(t)$
$P_1(t)$
$P_2(t)$
$P_3(t)$
$P_4(t)$
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t_9
t
15
10
5
- 93 -
LILLE BDS

L'évolution des probabilités d'occupation de la mémoire ne permet pas d'évaluer directement la perte de débit provoquée par le branchement ou l'interruption de programme. A cette fin, on doit calculer le débit instantanné pour chaque pas de l'évolution.

$$D(t) = P_1(t).\ 1 + P_2(t)\ .\ 2 + P_3(t)\ .\ 3 + P_4(t)\ .\ 4$$

Les courbes de la page suivante indiquent l'évolution du débit réel et du débit relatif par rapport au débit en régime stationnaire en fonction du temps. On remarque que plus le taux demandes est élevé, plus le régime stationnaire est rapidement atteint.

L'hypothèse selon laquelle la file d'attente n'intervient que peu dans la reprise est partiellement justifiée par ces courbes. La première demande provenant de la file d'attente ne peut se présenter qu'au temps t_5 puisque cette demande ne peut provenir que d'une demande refusée lors des cycles précédents. A cet instant, le débit de la mémoire a atteint une valeur comprise entre 85 et 95 % de sa valeur d'équilibre.

Il est enfin une chose importante à connaître : c'est la perte totale d'accès à la mé moire provoquée par l'interruption ou le branchement. Si on appelle D_m le débit moyen, le nombre de mots délivrés par la mémoire pendant l'état transitoire est :

$$n = D(t_0) + D(t_1) + D(t_2) + \ldots\ldots + D(t_{10})$$

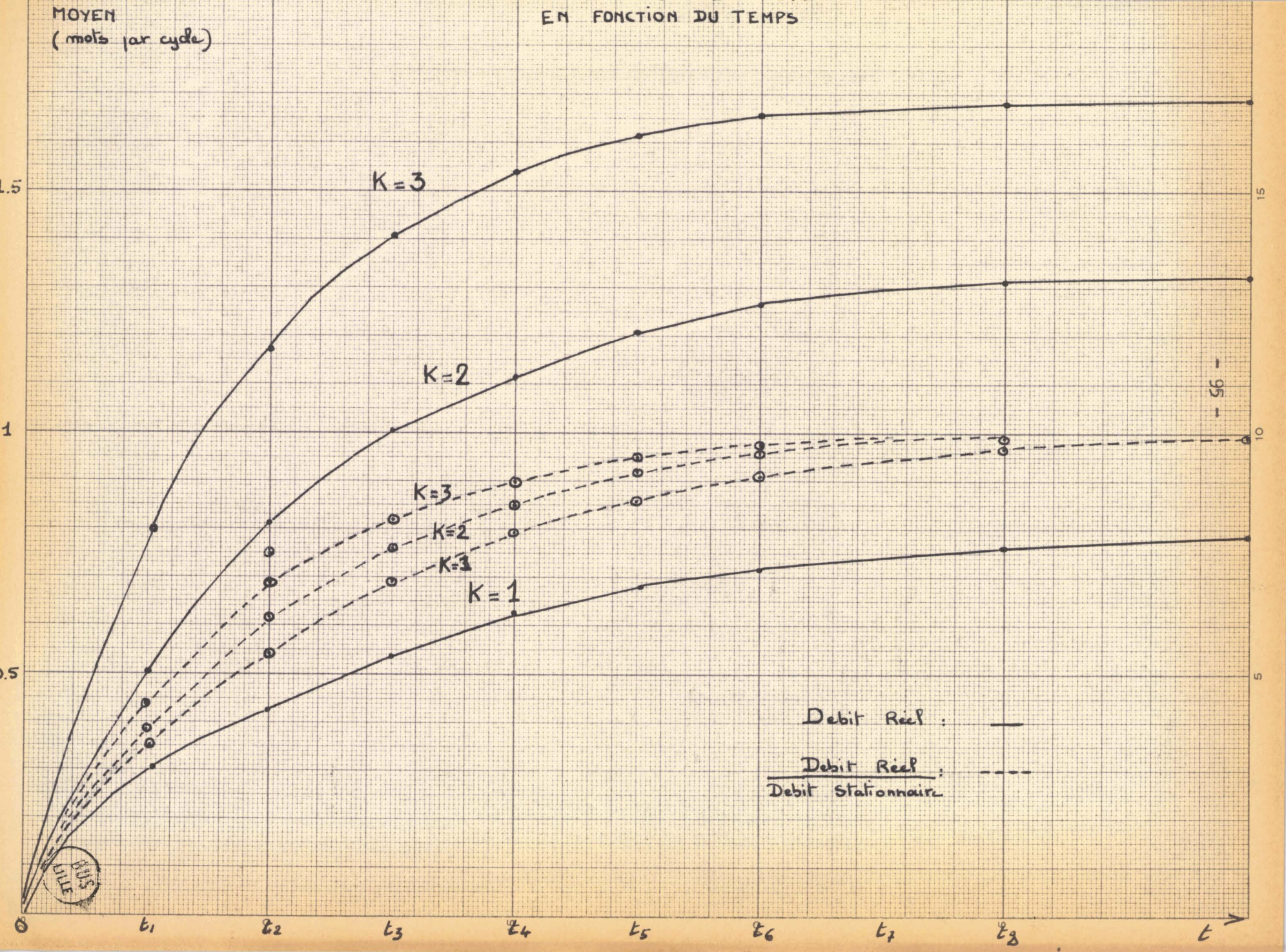
Le nombre de mots délivrés dans le même temps par une mémoire fonctionnant en régime établi serait :

$$n' = \sum D_m$$

Le nombre de mots perdus est : $n' - n$; on peut aussi exprimer cette perte en temps équivalent en se rapportant toujours au débit d'équilibre. Pour les trois valeurs du taux de demandes k, on trouve :

Taux de demandes	mots perdus	temps perdu
k = 1	2,34	3 cycles t soit 0,75 T
k = 2	3,32	2,5 cycles t soit 0,625 T
k = 3	3,73	2,2 cycles t soit 0,55 T

Les mémoires soumises à un taux de demandes élevé ne conduisent qu'à une perte de temps réduite. Il est possible que dans une mémoire où ce taux est faible, la nécessité de réapprovisionner les files d'instructions et de données conduise à accroître ce taux au moment du démarrage d'un nouveau programme de telle sorte que l'optimisation du régime transitoire se produit d'elle-même.


MOYEN
(mots par cycle)
EN FONCTION DU TEMPS
K=3
K=2
K=3
K=2
K=3
K=1
1.5
1
0.5
15
10
5
- 95 -
Debit Réel :
Debit Réel / Debit Stationnaire :
0
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t
BUS LILLE

3) Cas de la mémoire partiellement occupée.

Ordinairement, au moment où se présente la cause qui modifie l'orientation du programme, la mémoire est occupée et un certain nombre de blocs qui se trouvent au travail ne peuvent être remis immédiatement dans leur état initial. La lecture destructive réalisée dans la plus grande partie des mémoires centrales ne permet pas l'interruption d'un cycle de mémoire.

Le nouveau programme ne va donc pas disposer, au départ, de toutes les mémoires. Le vecteur initial ne sera plus le vecteur (1, 0, 0, ...), mais un vecteur (0, 0, 0, ... 1, 0, ... 0). La position de l'élément du vecteur qui est égal à 1 indique le nombre x de blocs occupés au moment de l'interruption.

Pratiquement, il est impossible de définir de manière certaine ce vecteur initial et l'on ne peut que calculer un vecteur probabiliste :

$$(P_1, P_2, P_3, P_4 \ldots\ldots P_m)$$

où chacun des termes représente la probabilité d'occupation en régime stationnaire du nombre de blocs correspondants.

On retrouve alors, en appliquant à ce vecteur la transformation représentée par la matrice d'échanges, le vecteur lui-même puisqu'il représente l'état d'équilibre vers lequel tend le phénomène étudié. Il semble qu'alors, rien ne soit modifié par l'interruption mais, si effectivement le régime de la mémoire ne subit pas de modifications apparentes, une partie des travaux qu'elle poursuit ne présente plus d'intérêt pour le nouveau programme. Le temps que ces travaux prennent à la mémoire est perdu pour le nouveau programme et le débit utile de cette mémoire s'en trouve diminué.

Il est aisé d'évaluer la perte de débit moyenne lors d'une interruption du fonctionnement de la mémoire si l'on connait la valeur moyenne du nombre de blocs occupés immédiatement avant que ne se présente cette interruption. Si on sait que $\bar{x}$ représente la valeur moyenne de l'occupation des blocs, on peut dire, en première analyse, que les $\bar{x}$ mots traités sont devenus inutiles et chiffrer à $\bar{x}$ la perte de débit.

Cette estimation est très approximative puisque, la mémoire étant partiellement occupée au moment de l'interruption, le rôle de la file d'attente que l'on a supposé mineur jusqu'à présent, devient prépondérant : la première demande du nouveau programme peut se voir refuser l'accès immédiat à la mémoire et être déposée en file d'attente. Comme on supposera toujours que le traitement est

séquentiel, il se trouve dans l'impossibilité de démarrer et la perte de temps risque d'être plus importante que dans le cas où la mémoire est initialement et totalement disponible.

Il faut donc nécessairement associer la file d'attente à la matrice de Markoff.

C - ROLE DE LA FILE D'ATTENTE EN REGIME TRANSITOIRE.

Le mécanisme de l'interruption de programme et celui du branchement conditionnel sont sensiblement différents. Dans le second cas, la direction que prendra le programme, bien qu'imprévisible au moment de l'exécution, se maintient dans un domaine d'application unique ; les données traitées restent les mêmes, les registres associés au traitement ne voient pas leur contenu modifié ; il n'y a que le compteur ordinal qui varie et le seul traitement propre à l'opération de branchement consiste à stocker éventuellement l'adresse de retour.

Au contraire, lors de l'apparition d'une interruption de programme, il faut modifier tous les éléments du traitement. Comme le traitement en cours devra être repris ultérieurement, ces éléments doivent être stockés ; étant donné leur nombre et la possibilité d'interruptions à plusieurs niveaux, il serait ruineux de les ranger dans une mémoire spécialisée. C'est donc la mémoire centrale qui sera concernée par ces rangements.

Par ailleurs, au moment où se présente l'interruption, un certain nombre de demandes d'écriture pouvaient se trouver en file d'attente. Ces demandes représentent les résultats d'instructions déjà traitées et qui, au moment de la reprise du programme interrompu seront considérées comme totalement effectuées. Les demandes d'écriture du programme interrompu, même si elles ne sont pas utiles au nouveau programme, doivent impérativement être prises en compte. Seules peuvent être négligées les demandes de lecture.

En conclusion, l'apparition d'une interruption de programme doit :

- Laisser se poursuivre les cycles déjà lancés.
- Laisser subsister dans la file d'attente les demandes d'écriture relatives au programme interrompu.
- Ajouter à la file d'attente une série de demandes d'écriture relatives aux éléments de reprise du programme interrompu.
- Annuler toutes les demandes de lecture non satisfaites.

Ces remarques permettent d'aborder sous un tout autre aspect la question de la longueur de la file d'attente : si on désire que l'exécution du mécanisme

d'interruption soit rapide, il faut que la file d'attente dispose d'assez de place pour pouvoir absorber toutes les demandes de rangement en mémoire sans se saturer, la saturation de la file d'attente provoquant le blocage de traitement. Le dépassement de la longueur de la file d'attente par rapport à sa valeur moyenne doit se résorber assez rapidement pour que des interruptions en chaînes n'aient que peu de risques de saturer cette file.

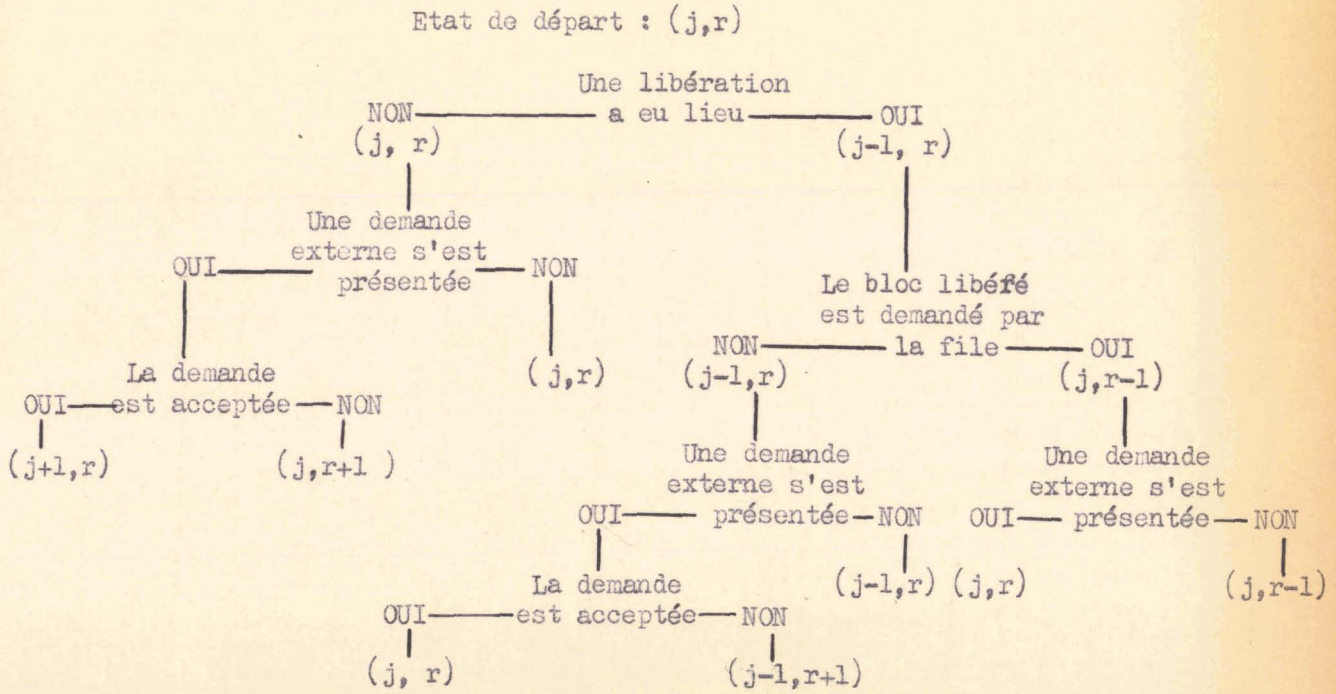
Si la file d'attente intervient dans le calcul, le système composé de m blocs indépendants de mémoire et de s positions de file d'attente peut prendre m.s états différents.

Il faut commencer par évaluer les probabilités de transition à partir d'un état défini. On prendra j comme indice des blocs de mémoire occupé et r comme indice des positions de file d'attente. L'indice i est réservé au temps. Si à l'instant t_i, j blocs de la mémoire et r positions de file étaient occupés, on évaluera les probabilités :

$P_{j,j}$; r,r-1	$P_{j,j+1}$; r,r-1	$P_{j,j-1}$; r,r-1
$P_{j,j}$; r,r	$P_{j,j+1}$; r,r	$P_{j,j-1}$; r,r
$P_{j,j}$; r,r+1	$P_{j,j+1}$; r,r+1	$P_{j,j-1}$; r,r+1

qui représentent toutes les évolutions possibles au cours du cycle Δt qui sépare les instants t_i et t_{i+1}.

Les différents cas qui peuvent se rencontrer sont figurés sur l'organigramme ci-dessous :



Les transitions (j, r) : (j+1, r+1)

(j, r) : (j-1, r-1)

et (j, r) : (j+1, r-1) sont impossibles et les probabilités de transition correspondantes sont donc nulles.

Les expressions des probabilités de transition s'écrivent alors :

$$P_{j,j\,;\,r,r-1} = \frac{j}{m} \cdot H(r)\ \frac{m-k}{m} = \frac{j}{m^2} \cdot H(r).(m-k)$$

$$P_{j,j\,;\,r,r} = \frac{m-j}{m} \cdot \frac{m-k}{m} + \frac{k\,j^2}{m^3}\left(1 - H(r) + \frac{k\,j}{m^2} \cdot H(r)\right.$$

$$P_{j,j\,;\,r,r+1} = \frac{m-j}{m} \cdot \frac{k}{m} \cdot \frac{j}{m} = \frac{k\,j}{m^3}\,(m-j)$$

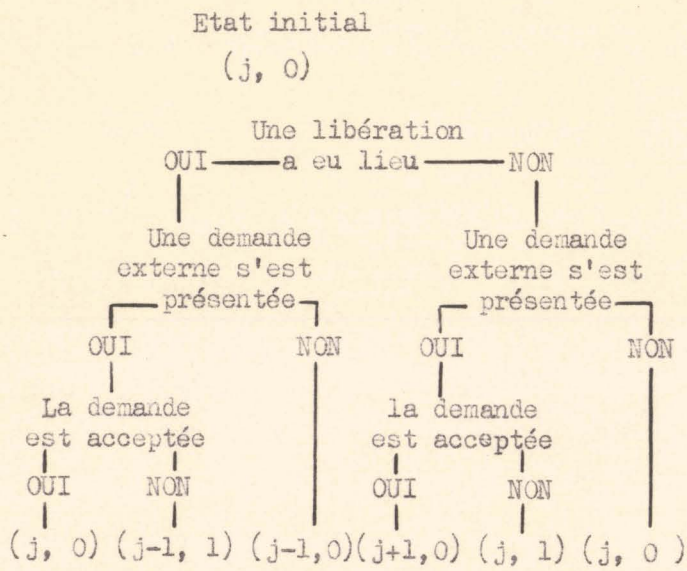
$$P_{j,j-1\,;\,r,r} = \frac{j}{m} \cdot (1 - H(r))\ \frac{m-k}{m}$$

$$P_{j,j-1\,;\,r,r+1} = \frac{j}{m} \cdot (1 - H(r)) \cdot \frac{k}{m} \cdot \frac{m-j}{m}$$

$$P_{j,j+1\,;\,r,r} = \frac{k}{m} \cdot \frac{m-j}{m} \cdot \frac{m-j}{m} = \frac{k}{m} \cdot \left(\frac{m-j}{m}\right)^2$$

Les cas limites, pour la mémoire comme pour la file d'attente, sont à considérer séparément :

a) La file est vide. Il ne peut donc pas exister de demande provenant d'elle.



$$P_{j,j;o,o} = \frac{m-j}{m^2}\left(\frac{jk}{m} + m - k\right)$$

$$P_{j,j-1;0,1} = \frac{kj^2}{m^3}$$

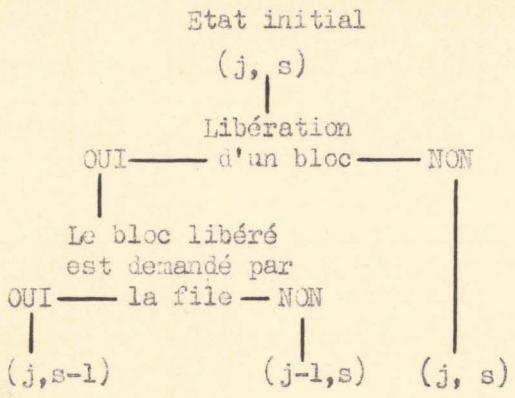
$$P_{j,j-1;0,0} = \frac{j}{m^2} \cdot (m-k)$$

$$P_{j,j+1;0,0} = \frac{k}{m^3} \cdot (m-j)^2$$

$$P_{j,j\,;\,1,1} = \frac{jk}{m^3} \cdot (m-j)$$

b) La file est saturée. Il y a verrouillage du traitement tant que la file reste saturée. Pour qu'une des positions de la file puisse se libérer, il faut

qu'un des blocs occupés de la mémoire se libère, lui-aussi.



$$P_{j,j\,;\,s,s} = \frac{m-j}{m}$$

$$P_{j,j-1\,;\,s,s} = \frac{j}{m} \cdot (1 - H(s))$$

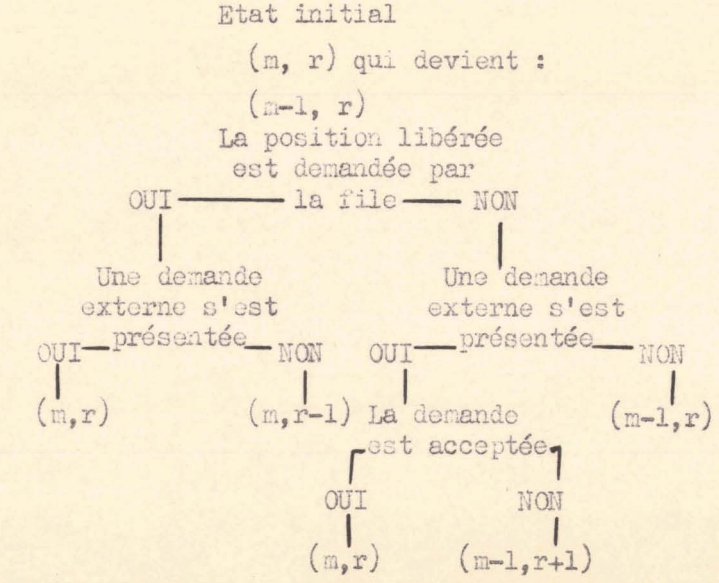
$$P_{j,j\,;\,s,s-1} = \frac{j}{m} \cdot H(s)$$

c) La mémoire est totalement inoccupée. Dans ce cas, aucune demande ne peut provenir de la file d'attente puisque ces demandes ne se présentent que lorsque une libération se produit. Ceci signifie que la file d'attente est donc libre. Le seul changement peut provenir d'une demande extérieure qui sera certainement acceptée.

$$P_{0,1\,;\,0,0} = \frac{k}{m}$$

$$P_{0,0\,;\,0,0} = \frac{m-k}{m}$$

d) La mémoire est saturée. Il existe autant de cycles mineurs de l'unité de contrôle dans un cycle technologique qu'il y a de blocs dans la mémoire. Si tous les blocs sont occupés, la probabilité pour que l'un d'eux soit libéré à l'instant considéré est égale à 1.



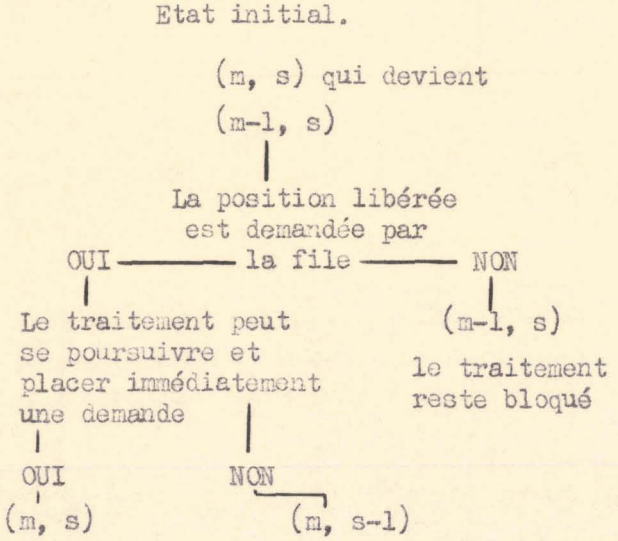
$$P_{m,m;r,r} = H(r)\,\frac{k}{m} + (1-H(r)) \cdot \frac{k}{m^2}$$

$$P_{m,m;r,r-1} = H(r) \cdot \frac{m-k}{m}$$

$$P_{m,m-1;r,r} = (1 - H(r)) \cdot \frac{m-k}{m}$$

$$P_{m,m-1;r,r+1} = (1-H(r)) \cdot \frac{k}{m} \cdot \frac{m-1}{m}$$

e) La mémoire et la file sont toutes deux saturées. La saturation de la mémoire conduit aux mêmes conclusions que dans le cas précédent, mais, de plus, la saturation de la file entraîne le blocage du traitement si la demande ne peut trouver place dans la mémoire ou dans la file.



$$P_{m,m\ ;\ s,s} = H(r) \cdot \frac{k}{m}$$

$$P_{m,m\ ;\ s,s-1} = H(r) \cdot \frac{m-k}{m}$$

$$P_{m,m-1\ ;\ s,s} = 1 - H(r)$$

Toutes les situations possibles étant prévues et les termes des probabilités de transition étant définis, on peut établir la matrice complète de Markoff et, à partir du choix d'un vecteur initial, calculer l'évolution du système formé de la file d'attente et de la mémoire partagée. Pour des mémoires formées d'un faible nombre de blocs, associées à des files d'attente de longueur réduite, l'application de la méthode utilisée précédemment dans un cas plus simple, est possible. On aboutira néanmoins à des matrices d'ordre relativement élevé. Dans le cas où la mémoire est partagée en de nombreux blocs et où la longueur de la file d'attente devient importante, le calcul direct n'est plus possible. Si, par exemple, on cherche à définir par ce moyen le comportement en régime transitoire d'une mémoire divisée en 32 blocs, associée à une file d'attente de 8 positions, il faut écrire et traiter des matrices d'ordre 256, ce qui est pratiquement impossible.

A l'analyse rigoureuse que représente l'étude par les chaînes de Markoff, il faut ajouter quelques hypothèses simplificatrices tenant compte des caractéristiques particulières du fonctionnement de la mémoire en régime transitoire.

Si, pour éviter la lourdeur des calculs correspondants, on désire ne pas envisager les possibilités présentées simultanément par la file d'attente et

par la mémoire, il faut nécessairement introduire certaines corrélations entre les états respectifs de l'une et de l'autre. Ces corrélations résultent de ce que l'on peut prévoir sur l'état initial de la file, de la distribution des blocs au travail et du taux de demandes. Il faut alors différencier les divers cas que l'on peut rencontrer et, en particulier, l'interruption de programme et le branchement conditionnel.

1) Cas de l'interruption de programme.

L'interruption de programme impose le rangement immédiat d'un nombre assez important de données relatives à l'ancien programme. Pour réaliser rapidement cette opération, il faut transférer toutes ces données dans la file d'attente. Avant que ne se présente l'interruption, la longueur occupée de celle-ci était la valeur moyenne calculée dans le cas du régime stationnaire. Si on admet que sur deux demandes de lecture provenant de l'unité centrale, une concerne les instructions et l'autre les données et que, à chaque donnée lue, correspond une donnée réinscrite dans la mémoire, le quart de ce que contient alors la file d'attente doit être conservé à fin d'écriture ; le restant qui représente des demandes de lecture peut être détruit. On choisira la longueur de la file d'attente de telle sorte que le quart des demandes restantes augmenté de toutes les demandes propres à l'interruption conduise la file au environs de la saturation.

Pour que le système revienne à un régime normal, la file d'attente doit se vider progressivement, la probabilité pour qu'elle passe par de faibles taux d'occupation avant de parvenir à son état d'équilibre est donc très faible. Les probabilités de transition correspondantes peuvent donc être négligées.

Au moment où se produit l'interruption, la mémoire a atteint un régime d'équilibre et les blocs qui sont au travail vont se trouver libérés, les uns à la suite des autres sur une période correspondant au temps de cycle d'un bloc. L'afflux de demandes, lié à l'interruption, va provoquer un accroissement de l'activité de la mémoire. Il en résultera que le retour au régime stationnaire se fera à partir d'un taux d'occupation plus élevé que le taux normal. Les probabilités de transition correspondant à des taux d'occupation plus faibles que le taux normal sont faibles et peuvent être, elles aussi, négligées.

2) Cas du branchement conditionnel.

Le branchement conditionnel est le second cas susceptible de provoquer une rupture de régime dans le débit de la mémoire. Il est impossible de prévoir quelles seront les instructions et les données à traiter à la suite de l'instruction de branchement. Comme cependant cette rupture se produit à l'intérieur d'un même programme, les informations stockées dans les registres des circuits de traitement ne sont pas modifiées. Le chargement de la file d'attente tel qu'il se produit pour l'interruption n'aura pas lieu, Le taux d'occupation de la file d'attente restera voisin de ce qu'il est en régime continu ; son comportement est lié à la manière dont la logique d'anticipation réagit devant le retard qu'elle a pris. Dans tous les cas, il faudra que le débit de la mémoire et la charge de la file soient supérieurs pendant la période transitoire à ce qu'ils sont en régime normal. Selon la vitesse de réaction de la logique d'anticipation, on peut centrer les termes probabilistes les plus importants plus ou moins loin de la valeur moyenne dans le sens d'une augmentation du débit et de la longueur de la file.

Il est important de remarquer à ce propos qu'il peut être utile de freiner la vitesse de réaction de la logique d'anticipation. Imaginons que la logique d'anticipation du programme pose immédiatement toutes les demandes qui lui sont nécessaires pour retrouver son avance. La première de ces demandes ne peut être satisfaite qu'après un cycle de mémoire de telle sorte que l'information désignée dans l'instruction ne sera transmise aux circuits de traitement qu'au bout de deux cycles de mémoire. Les instructions qui suivent n'ont aucune chance d'être prises en compte pendant ce temps et, si elles sont lancées immédiatement, elles risquent de prendre la place d'autres demandes plus urgentes alors qu'elles ne seront certainement pas exploitées.

Cette remarque reste vraie dans le cas de l'interruption.

Exemple d'application de ces approximations.

Une mémoire est partagée en quatre blocs indépendants et le taux moyen de demande est k = 2. L'examen des distributions de longueurs de la file d'attente montre que :

- Pour une longueur s = 3, la probabilité de saturation est de 0,27
- Pour une longueur s = 4, la probabilité de saturation est de 0,16
- Pour une longueur s = 5, la probabilité de saturation est de 0,09
- Pour une longueur s = 6, la probabilité de saturation est de 0,04.

On prendra une longueur de file de 5.

Les calculs en régime permanent indiquent d'autre part que la longueur moyenne de la file est de 2,409.

Lors d'une interruption de programme, il faut ranger dans la mémoire trois mots complets dans des adresses successives. Cette valeur de trois représente un minimum : le compteur ordinal, le contenu d'un registre d'index, le contenu de l'accumulateur et un registre qualitatif indiquant l'état des bascules annexes, le niveau de priorité et un masque lié au programme interrompu.

Si 2,4 demandes occupent la file au moment de l'interruption, une, en moyenne sera une demande de lecture et deviendra inutile. Il devient donc possible de loger dans la file d'attente les trois demandes de rangement provenant de l'interruption. Il en résulte que la file contiendra au moins trois demandes ; la mémoire sera elle aussi occupée immédiatement puisque les demandes provenant du nouveau programme lancé porteront sur des instructions disposées dans des adresses successives.

Il est donc logique de considérer que :

1) La file se trouvera dans les états 3, 4 ou 5 (saturée)
2) La mémoire se trouvera dans les états 2, 3 ou 4 (totalement occupée)

La matrice de Markoff réduite à ces états limités est d'ordre 9 alors que la matrice complète est d'ordre 25.

Il faut ajouter aux 9 états admis comme transitoires un dixième qui regroupe tous les autres états en supposant que, dès que le système atteint l'un d'eux, la mémoire a retrouvé son régime stationnaire.

En conservant la notation à quatre indices représentant dans l'ordre : l'état initial de la mémoire, l'état final de la mémoire, l'état initial de la file et l'état final de la file, la matrice réduite peut s'écrire :

		indices finaux								
		23	24	25	33	34	35	43	44	45
indices initiaux	23	$p_{22/33}$	$p_{22/34}$	$p_{22/35}$	$p_{23/33}$	$p_{23/34}$	$p_{23/35}$	$p_{24/33}$	$p_{24/34}$	$p_{24/35}$
	24	$p_{22/43}$	$p_{22/44}$	$p_{22/45}$	$p_{23/43}$	$p_{23/44}$	$p_{23/45}$	$p_{24/43}$	$p_{24/44}$	$p_{24/45}$
	25	$p_{22/53}$	$p_{22/54}$	$p_{22/55}$	$p_{23/53}$	$p_{23/54}$	$p_{23/55}$	$p_{24/53}$	$p_{24/54}$	$p_{24/55}$
	33	$p_{32/33}$	$p_{32/34}$	$p_{32/35}$	$p_{33/33}$	$p_{34/34}$	$p_{33/35}$	$p_{34/33}$	$p_{34/34}$	$p_{34/35}$
	34	$p_{32/43}$	$p_{32/44}$	$p_{32/45}$	$p_{33/43}$	$p_{33/44}$	$p_{33/45}$	$p_{34/43}$	$p_{34/44}$	$p_{34/45}$
	35	$p_{32/53}$	$p_{32/54}$	$p_{32/55}$	$p_{33/53}$	$p_{33/54}$	$p_{33/55}$	$p_{34/53}$	$p_{34/54}$	$p_{34/55}$
	43	$p_{42/33}$	$p_{42/34}$	$p_{42/35}$	$p_{43/33}$	$p_{43/34}$	$p_{43/35}$	$p_{44/33}$	$p_{44/34}$	$p_{44/35}$
	44	$p_{42/43}$	$p_{42/44}$	$p_{42/45}$	$p_{43/43}$	$p_{43/44}$	$p_{43/45}$	$p_{44/43}$	$p_{44/44}$	$p_{44/45}$
	45	$p_{42/53}$	$p_{42/54}$	$p_{42/55}$	$p_{43/53}$	$p_{43/54}$	$p_{43/55}$	$p_{44/53}$	$p_{44/54}$	$p_{44/55}$

En faisant l'approximation sur H(r) : H (r) = r/m, on trouve les termes suivants :

$$\begin{pmatrix}
.46875 & .12500 & .00000 & .12500 & .00000 & .00000 & .00000 & .00000 & .00000 & .28125 \\
.25000 & .50000 & .12500 & .00000 & .12500 & .00000 & .00000 & .00000 & .00000 & .00000 \\
.00000 & .50000 & .50000 & .00000 & .00000 & .00000 & .00000 & .00000 & .00000 & .00000 \\
.09365 & .02347 & .00000 & .47656 & .09365 & .00000 & .03025 & .00000 & .00000 & 28125 \\
.00000 & .00000 & .00000 & .37500 & .50000 & .09375 & .00000 & .03125 & .00000 & .00000 \\
.00000 & .00000 & .00000 & .00000 & .25000 & .75000 & .00000 & .00000 & .00000 & .00000 \\
.00000 & .00000 & .00000 & .12500 & .09375 & .00000 & .40625 & .00000 & .00000 & .37500 \\
.00000 & .00000 & .00000 & .00000 & .00000 & .00000 & .50000 & .50000 & .00000 & .00000 \\
.00000 & .00000 & .00000 & .00000 & .00000 & .00000 & .00000 & .50000 & .50000 & .00000 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

La dernière ligne de la matrice représente le régime stationnaire. On suppose que si ce régime est atteint, la mémoire le conserve indéfiniment.

Le vecteur initial représentant la distribution des occupations de la file d'attente et de la mémoire n'est pas défini et ne peut s'obtenir que par l'examen des distributions avant l'interruption. On suppose que le régime stationnaire est atteint et que la distribution est celle qui a été calculée pour ce cas.

Pour la mémoire, les probabilités d'occupation sont :

Nombre de positions occupées	0	1	2	3	4
Probabilités	0,064	0,250	0,372	0,250	0,064

Pour la file d'attente, les probabilités d'occupation sont :

Nombre de positions occupées	0	1	2	3	4	5
Probabilités	0,091	0,182	0,218	0,180	0,131	0,086

Les distributions instantannées des occupations de la mémoire vont se trouver accrues au cours du traitement de l'interruption et la file d'attente va recevoir pendant ce temps de deux à trois demandes supplémentaires de telle sorte que l'on peut considérer que les distributions indiquées ci-dessus se trouvent translatées dans le sens d'une plus grande charge. On prendra une position supplémentaire pour la mémoire et trois positions pour la file.

Pour les vecteurs initiaux définis par les deux chiffres désignant les positions occupées de la mémoire et de la file, on trouve alors la distribution suivante des probabilités :

1,3	1,4	1,5	1,6	1,7	1,8	1,9
0,0058	0,0116	0,0141	0,0115	0,0084	0,0055	0,0058
2,3	2,4	2,5	2,6	2,7	2,8	2,9
0,023	0,046	0,0545	0,045	0,033	0,021	0,023
3,3	3,4	3,5	3,6	3,7	3,8	3,9
0,034	0,068	0,081	0,067	0,049	0,032	0,034
4,3	4,4	4,5	4,6	4,7	4,8	4,9
0,023	0,051	0,065	0,059	0,044	0,029	0,033

Les probabilités portées sur la dernière ligne représentant la saturation de la mémoire sont la somme des probabilités pour des occupations de trois ou quatre blocs occupés, avant que ne se présente l'interruption. Si quatre blocs étaient occupés, la file d'attente reçoit une demande supplémentaire. Les cas qui sont portés à droite de la ligne pointillée représentent la saturation de la file d'attente.

La probabilité pour que la capacité de la file d'attente soit dépassée est de 0,50. Dans ce cas, le temps pendant lequel le traitement sera bloqué est, en moyenne, de :

$$T.0,364 + 2.T.0,268 + 3.T.\ 0,174 + 4.T.0,192 = 2,18\ T$$

T étant la durée du cycle mineur.

Dans le cas où il y a dépassement de la capacité de la file d'attente, le vecteur initial présentera un terme r, nombre de positions occupées dans la file, égal à 5, dès que le blocage aura disparu. On peut constater, dans le cas qui est étudié, l'intérêt que présente une file d'attente longue pour l'amélioration des performances de la mémoire en régime transitoire.

L'évolution des occupations dans la mémoire et dans la file sera connue lorsqu'on appliquera à chacun des vecteurs initiaux possibles, la transformation définie par la matrice de Markoff. Pour connaître la probabilité de chaque cas au bout de n.T, il faut faire le produit de ce vecteur initial par la puissance n de la matrice.

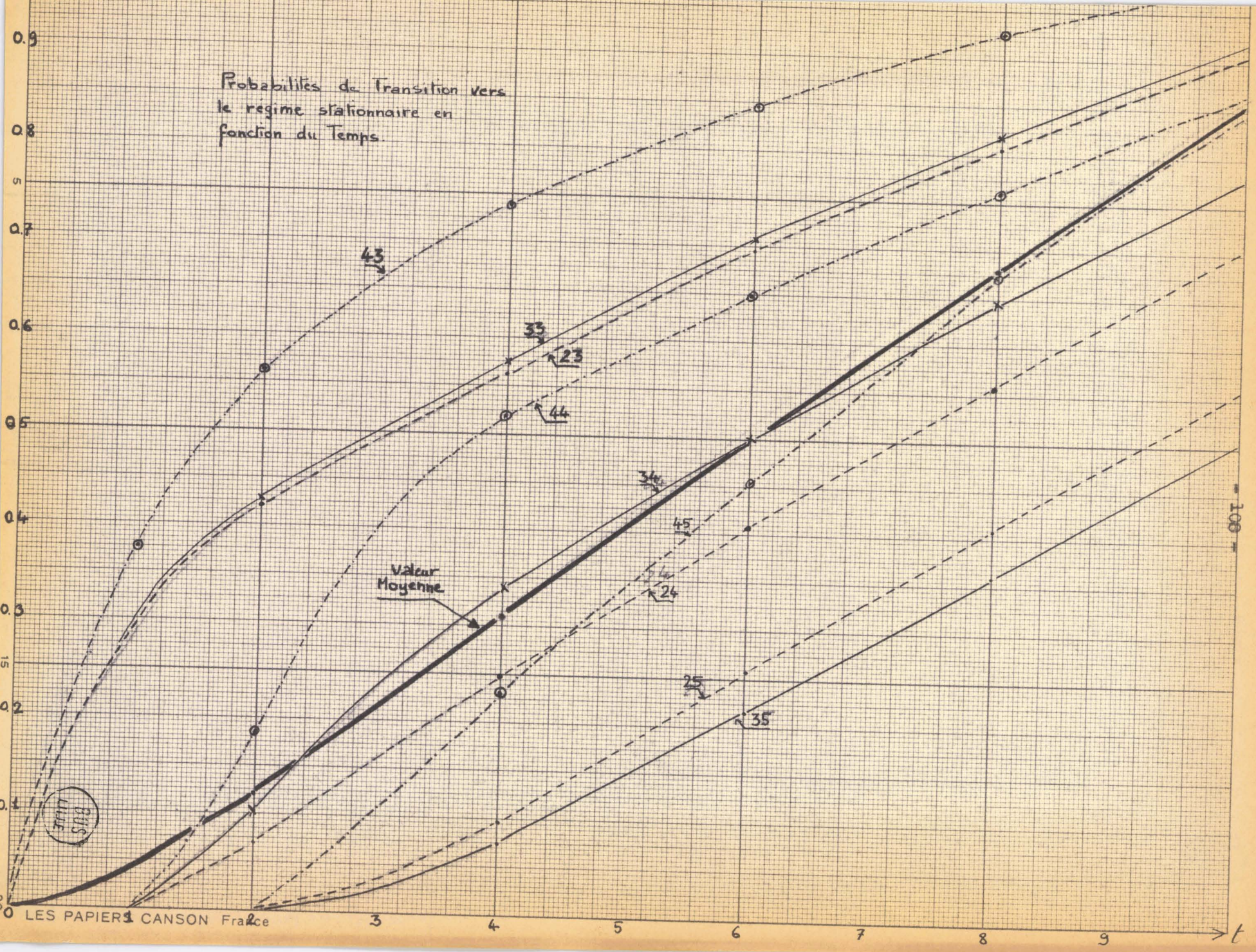
Les résultats de ce calcul, pour les différents vecteurs initiaux possibles sont représentés à la page suivante sous forme de courbes. La présence d'un état stationnaire équivalent au régime continu est concrètisée par la présence d'une probabilité égale à 1 dans la dernière ligne de la matrice. Il est donc certain que si n tend vers l'infini, la puissance n de la matrice de Markoff va converger vers une matrice :

$$\begin{pmatrix} 0\ 0\ 0\ 0\ 0 \ldots . 1 \\ 0\ 0\ 0\ 0\ 0 \ldots . 1 \\ \ldots\ldots\ldots \\ 0\ 0\ 0\ 0\ 0 \ldots . 1 \end{pmatrix}$$

On peut chiffrer les états intermédiaires en indiquant la probabilité de transition entre le vecteur initial et l'état stationnaire. Les courbes représentées à la page suivante représentent :

- En abcisse, le temps, compté en cycles mineurs de l'unité de contrôle.
- En ordonnée, la probabilité de transition entre l'instant initial et l'instant repéré.
- En paramètre, figurent les divers vecteurs initiaux possibles.

La somme pondérée par la probabilité propre de chacun des vecteurs initiaux et représentée en traits forts, indique l'évolution moyenne de la mémoire vers le régime stationnaire.


Probabilités de Transition vers le régime stationnaire en fonction du Temps.
43
33
23
44
34
45
24
25
35
Valeur Moyenne
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
5
6
7
8
9
t
LES PAPIERS CANSON France

On peut, à partir de ces résultats numériques, chiffrer la perte de rendement de la mémoire lors d'une interruption. Cette perte de rendement se déduit de la variation du débit de la mémoire donc de l'estimation de son taux d'occupation.

Cette mesure n'est pourtant pas exacte puisqu'une partie des travaux que poursuit la mémoire au moment de l'interruption devient inutile. Son débit utile est donc inférieur à son débit réel.

La courbe de la page suivante indique la variation du taux moyen d'occupation, déduite de l'analyse des matrices de Markoff. Avant l'interruption, le taux est de 2. Au moment où l'interruption se présente, le taux réel ne change pas, mais le taux utile est ramené au tiers de sa valeur précédente en admettant qu'il y a deux opérations de lecture pour une d'écriture.

La première courbe correspond au cas optimum où toutes les demandes présentes dans la file d'attente sont immédiatement satisfaites.

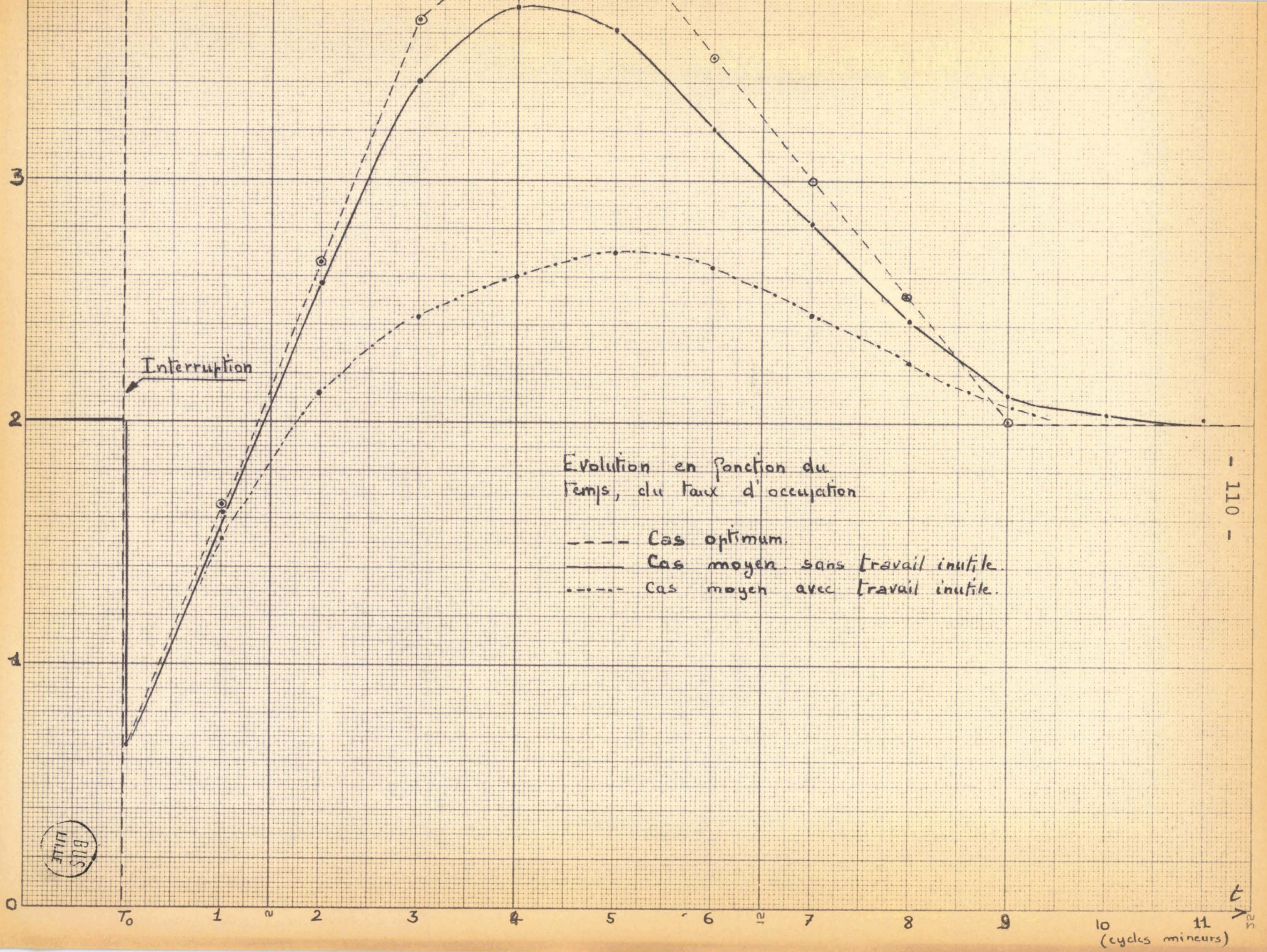
La seconde courbe fait intervenir des conflits entre ces demandes mais suppose que les mémoires qui faisaient auparavant des opérations de lecture peuvent être rendues immédiatement disponibles.

La dernière, enfin, tient compte de ces mémoires occupées inutilement qui s'opposent à la prise en charge des demandes présentes dans la file d'attente.

Les trois courbes présentent la caractéristique commune suivante : le taux d'occupation passe par un maximum supérieur à sa valeur moyenne. Ceci résulte de la saturation de la file d'attente qui provoque un regain d'activité provisoire de la mémoire afin de retrouver la profondeur d'anticipation du régime établi.

En fait, la description du processus d'interruption choisi est la plus simple qu'on puisse imaginer car on a supposé que le taux de demandes restait constant, alors qu'il peut être utile d'en autoriser l'accroissement, afin de retourner plus vite à l'état stationnaire. Dans cette hypothèse, les probabilités de transition de la matrice de Markoff deviennent variables et l'étude probabiliste se complique. Il faut préciser alors les priorités relatives des différentes demandes, en particulier pour les écritures qui restent à faire pour l'ancien programme, par rapport aux lectures du nouveau.

On se contentera d'une évaluation approximative basée sur des considérations probabilistes simples et sur des hypothèses assez vraisemblables.


Interruption
Evolution en fonction du Temps, du taux d'occupation
Cas optimum.
Cas moyen sans travail inutile.
Cas moyen avec travail inutile.
0
1
2
3
T_0
1
2
3
4
5
6
7
8
9
10
11
(cycles mineurs)
t

BIS LILLE

D - TEMPS PERDU PAR LE TRAITEMENT.

Cette notion est plus réaliste que celle de variation du débit de la mémoire envisagée jusqu'à présent ; il ne suffit pas que la mémoire travaille vite, encore faut-il qu'elle délivre aussi vite que possible les informations nécessaires au traitement.

Lors d'une interruption, l'information la plus urgente est la première instruction du nouveau programme ; elle doit être suivie de l'information que désigne cette instruction. Il est donc logique de donner à ces deux demandes le plus haut niveau de priorité. Malheureusement, la seconde ne peut être posée que lorsque la première a été satisfaite. Si on désire donner à cette seconde demande le plus grand nombre de chances d'être satisfaite, il ne faut pas lancer d'autres travaux dans la mémoire, mais alors on retrouve une organisation purement séquentielle et l'anticipation ne pourra pas se former. L'optimisation immédiate ne peut donc se faire qu'au détriment de l'optimisation à long terme. Si, par contre, on préfère retrouver rapidement la profondeur normale d'anticipation, cela se fera au détriment des premières instructions du programme.

Il faut enfin préciser à quel moment il importe de ranger en mémoire les données du programme interrompu. Il est, en effet, possible de les ranger immédiatement pour respecter l'ordre chronologique mais il semble préférable d'attendre que le nouveau programme ait atteint son équilibre. Il est probable que le programme interrompu ne sera pas rappelé avant un certain temps et les demandes d'écriture qui le concernent doivent se trouver au niveau de priorité le plus bas. Elles ne peuvent y être maintenues pendant un certain temps qu'à l'aide d'un dispositif spécial de blocage mais alors elles immobilisent des positions de la file d'attente, accroissant les risques de saturation de celle-ci et donc de blocage du traitement.

1) Optimisation immédiate.

Le nombre de positions occupées de la mémoire, au moment de l'interruption étant $\bar{x}$, le temps nécessaire pour obtenir la première instruction est :

$$T_1 = t_{lec} \frac{m - \bar{x}}{m} + \left(t_{lec} + \frac{T}{2}\right) \cdot \frac{\bar{x}}{m}$$

t_{lec} représente la fraction du cycle de mémoire au bout duquel le résultat de la lecture est disponible. On l'appelle, dans une mémoire classique, le temps d'accès. Cette fraction est comprise entre 0,3 et 0,5 fois le temps de cycle.

Si on réserve, à la recherche de la donnée définie par cette première instruction, la plus grande priorité, on trouve, pour obtenir cette donnée, un temps d'attente compté à partir du moment où l'instruction est disponible et sans tenir compte d'un éventuel calcul d'adresse :

$$T_2 = t_{lec} \frac{m - \bar{x}}{m} \left(\frac{m - \bar{x}.tlec/T}{m}\right) + \frac{\bar{x}}{m} . t_{lec/T} \left(\frac{T}{2} + t_{lec}\right))$$

La quantité $\bar{x}. t_{lec/T}$ représente le nombre de blocs restant occupés au moment où commence la recherche de la donnée.

Il faudra ensuite chercher l'instruction suivante :

$$T_3 = t_{lec}$$

et de même pour les temps de recherche des mots suivants.

Cette solution qui assure la reprise la plus rapide du nouveau programme revient à considérer la mémoire comme séquentielle, le bénéfice du partage des blocs n'apparaît que par la possibilité de chevauchement partiel des cycles successifs. Si, par exemple $t_{lec} = 0,5\ T$ et $\bar{x} = m/2$, on trouve $T_1 = 0,75\ T$, $T_2 = 0,6125\ T$ et $T_3 = 0,5\ T$; par la suite le débit de la mémoire ne peut dépasser deux mots par cycle technologique.

2) Optimisation à long terme.

La solution opposée consiste à règler au plus vite les travaux propres au régime transitoire pour préparer le retour au régime établi. Il y a donc lieu, tout d'abord, de ranger dans la mémoire les données du programme interrompu, puis de donner la priorité aux demandes des files d'anticipation, afin de leur permettre de retrouver leur avance.

Les mots à enregistrer, suivis des demandes de la file d'anticipation d'instruction vont occuper la mémoire pendant un temps relativement court, car les zones d'écriture et de lecture sont le plus souvent séquentielles et les demandes relatives à chacun de ces deux groupes porteront sur des blocs voisins. Il est certain néanmoins que le traitement ne pourra reprendre qu'après une attente importante.

S'il y a p mots du programme interrompu à écrire dans la mémoire et si la profondeur de la file d'anticipation des instructions est q, il faut lancer p + q demandes préalables vers la mémoire avant la première demande de donnée. En admettant que le débit de la mémoire reste égal ou un peu supérieur à ce qu'il est en régime continu, on peut évaluer le temps d'attente. Avec un débit D compté en mots par cycle de mémoire, ce temps vaut :

$$T_{attente} = (p + q) / D$$

En supposant que le caractère séquentiel des demandes permet d'optimiser la distribution des demandes entre les différents blocs, on trouve :

$$T_{attente} = (p + q) \text{ cycles mineurs}$$
$$= (p + q) \cdot T / m$$

A la valeur moyenne comprise entre ces deux valeurs extrêmes, il faut ajouter le temps d'attente et le temps de lecture pour la première donnée du nouveau programme.

3) Solution intermédiaire.

Le traitement peut commencer avant que les files d'attente d'écriture et d'anticipation soient totalement servies. Il le fera à un rythme réduit.

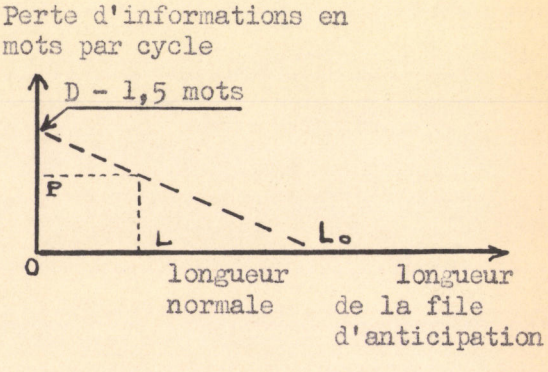
On peut tenter d'évaluer :

a) la perte de temps pour le traitement en fonction de la réduction de longueur des files d'anticipation.

b) le taux de croissance qui conduit à la perte de temps globale minimum.

a) Perte de temps.

Lorsque les files d'anticipation sont normalement garnies, le débit se situe aux environs de D, valeur calculée en régime stationnaire. Lorsque ces files sont vides, le traitement ne dispose que d'accès séquentiels à la mémoire de telle sorte que, en se basant sur le rapport de deux lectures pour une écriture, le taux d'activité de la mémoire est de 1,5 mots par cycle technologique. On va supposer que la loi de variation entre ces deux valeurs extrêmes est linéaire. La loi est représentée sur le graphique ci-contre.



En appelant P, la perte instantannée ainsi exprimée, on peut écrire que la perte globale entre l'instant zéro de l'interruption et l'instant t_{final} où la file a retrouvé sa longueur est :

$$\text{Perte totale} = \int_{t_0}^{t_{final}} P(t).dt$$

La perte instantannée d'informations est : $P(t) = P_{max} - \frac{P_{max}}{L_0} \cdot L(t)$

P_{max} désigne la valeur maximum : D - 1,5 mots.

b) Taux de croissance optimum.

En supposant que le débit de la mémoire reste pratiquement constant, on peut écrire que l'information perdue par le traitement sert à accroître la longueur des files d'anticipation :

$$\frac{dL}{dt} = P(t) = P_{max} - \frac{P\ max}{L_0} \cdot L(t)$$

d'où on déduit la variation de la longueur de la file d'anticipation :

$$L(t) = L_0 \left(1 - e^{\frac{-P\ max}{L_0}\ t} \right)$$

et $$P(t) = P_{max}\ e^{\frac{-Pmax}{L_0}\ t}$$

La perte totale est donc : $P_{totale} = L_0$, ce qui est logique.

Cette perte d'informations est indépendante de la vitesse d'évolution du régime transitoire.

Si on limite la vitesse d'accroissement de la file d'anticipation, cela revient à introduire un coefficient inférieur à 1 dans l'expression de dL/dt ou à ajouter une constante négative au second terme. Dans les deux cas, cela conduit à augmenter les pertes totales d'information. Il semble donc qu'il soit préférable de favoriser au maximum les files d'anticipation, même si cela se fait, provisoirement, au détriment du traitement proprement dit.

Pratiquement, il faut tenir compte d'abord du fait que l'équation différentielle ne représente pas exactement l'évolution du phénomène et que elle ne peut tenir compte des particularités telles que le caractère séquentiel des zones concernées, l'existence d'au moins deux files distinctes d'anticipation et les aléas sur le débit instantanné réclamé par le traitement. En fait, un dernier élément doit intervenir, c'est le caractère prioritaire de certains traitements immédiats, en particulier dans les systèmes fonctionnant en temps réel pour lesquels il peut être urgent de procéder à certains traitements, même si ceux qui suivent doivent s'en trouver un peu retardés. Dans ces cas, on obtiendra une solution simple en n'autorisant une file d'anticipation à présenter une demande que lorsqu'un certain intervalle de temps se sera écoulé depuis qu'elle a obtenu la précédente.

CHAPITRE V

MODELE DE GESTION D'UNE MEMOIRE PARTAGEE.

L'accroissement du temps de service, chiffré dans le chapitre précédent, va à l'encontre du but recherché qui est d'améliorer les performances de la machine. Si on ne considère que l'unité centrale, on peut distinguer deux grandes familles d'informations : les instructions et les opérandes. La recherche alternée des unes puis des autres ne peut se faire qu'en séquence dans une organisation classique ; il en résulte une perte de temps supplémentaire. Seuls les échanges avec les organes périphériques sont alors assurés d'un meilleur rendement.

Si on suppose que le traitement est, par nature, séquentiel, on peut imaginer d'en augmenter la vitesse en procédant à une recherche anticipée des instructions à exécuter. Cette supposition ne tient pas compte des ruptures de séquence programmées ou imprévues dues respectivement aux branchements et aux interruptions de programme d'origine externe au programme. On verra plus loin comment il est possible de traiter ces cas particuliers.

A - PRINCIPES D'UNE ORGANISATION DE RECHERCHE ANTICIPEE.

Une telle organisation fait l'objet d'une description très détaillée dans la publication citée en référence (6) qui décrit la machine IBM 360/91. On ne reprendra les principaux termes de cette description que pour en extraire ce qui est spécifique de la recherche anticipée.

Le dispositif impose deux groupes d'opérations :

a) Lire les instructions avant de devoir les exécuter et les disposer dans une file d'attente où elles seront prises dans l'ordre où elles ont été introduites (file d'instructions).

b) Extraire de l'instruction la partie adresse et les bits de modification d'adresse ; chercher les opérandes avant la prise en charge de l'instruction puis les placer dans une autre file d'attente (file d'opérandes).

L'unité de contrôle ne peut alors trouver les informations qu'elle recherche que dans la position la plus basse de chacune de ces deux files d'attente.

Il est nécessaire de décrire l'organisation adoptée pour les différentes mémoires temporaires destinées à contenir des demandes ou des résultats avant de tenter d'en chiffrer les performances. De nombreuses contraintes apparaissent du fait de la simultanéité, pour lesquelles le choix d'une solution est difficile du fait de l'impossibilité de chiffrer ses performances.

Une mémoire partagée organisée en vue de la recherche anticipée comporte nécessairement :

- m blocs de mémoire complets, c'est-à-dire disposant chacun d'un registre d'adresses, d'un registre d'entrée et d'un registre de sortie de l'information.
- Une file d'attente pour stocker les demandes non immédiatement satisfaites. Cette file, dont la longueur peut être déterminée à partir des calculs précédents, doit permettre de stocker dans chacune de ses positions, l'adresse demandée, la nature de l'opération - Lecture ou écriture - et éventuellement une indication de priorité.
- Deux files de rangement des informations lues par anticipation, l'une pour les instructions, l'autre pour les données.
- Une file d'attente pour les informations à écrire dans la mémoire. Il est à noter que, si les deux précédentes files sont en avance sur le traitement, celle-ci, par contre, se trouve toujours en retard, ce qui pose un délicat problème lorsqu'il est nécessaire de procéder à des échanges entre ces deux groupes de files.

L'organigramme de la page suivante indique le principe de fonctionnement d'une telle mémoire. Les files de rangement des instructions et des données n'apparaissent pas explicitement dans l'organigramme. On considère que ces files d'attente font plutôt partie de l'unité centrale que de la mémoire. Il est d'ailleurs indiqué, dans toutes les publications relatives à la recherche anticipée, que la gestion de ces deux files d'attente est confiée à l'unité de contrôle et non à la mémoire. Les demandes présentées par ces files sont donc considérées comme des demandes externes, au même titre que celles qui proviennent des canaux d'entrées-sorties.

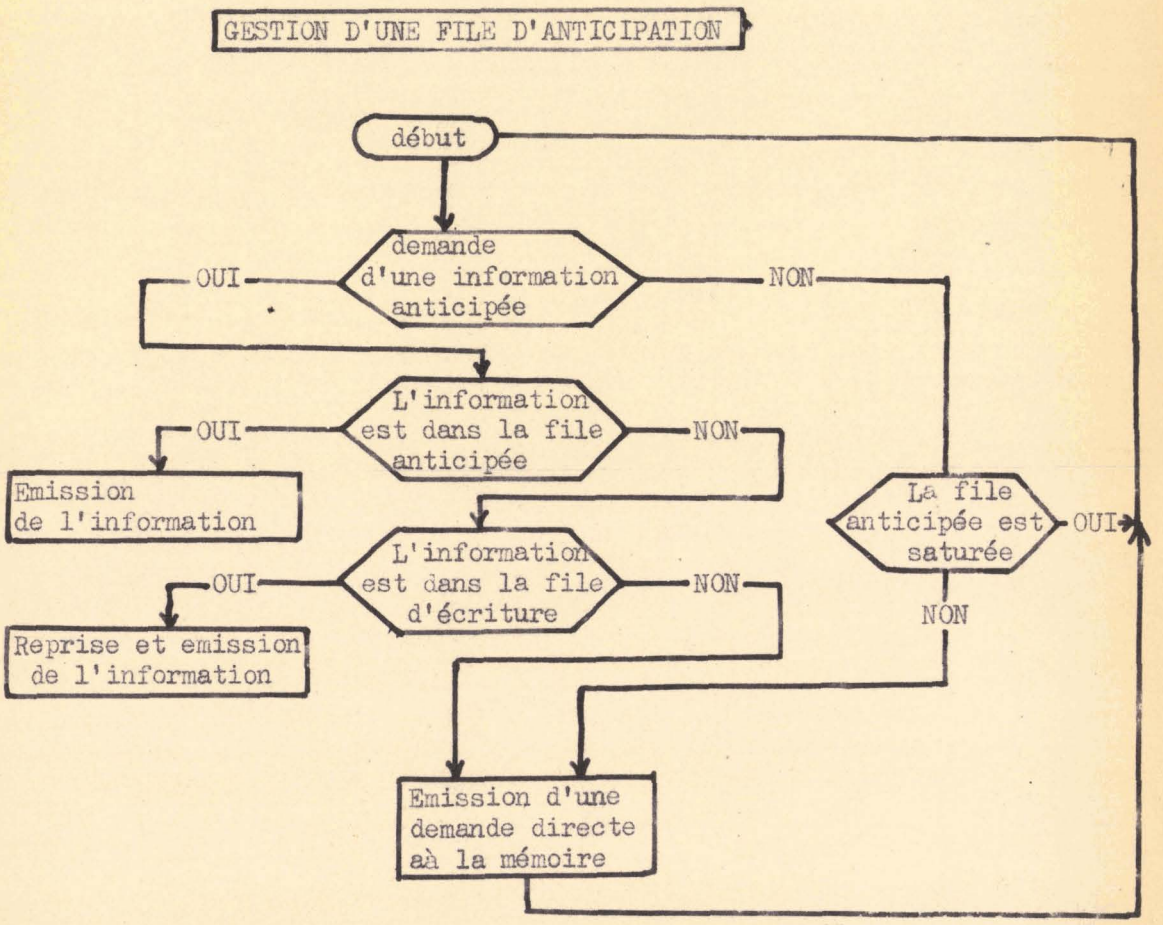
On a supposé que un cycle de l'unité de gestion de la mémoire ne pouvait prendre en charge qu'une seule demande. Si on admet que les demandes contenues dans la file d'attente sont prioritaires sur les demandes externes, d'une part et que les demandes qui se trouvent dans la file d'attente ne peuvent concerner

que des blocs occupés, on constate que l'élément déterminant est la libération d'un bloc au travail. Comme les demandes sont prises en charge, une par une, les libérations ne peuvent être simultanées.

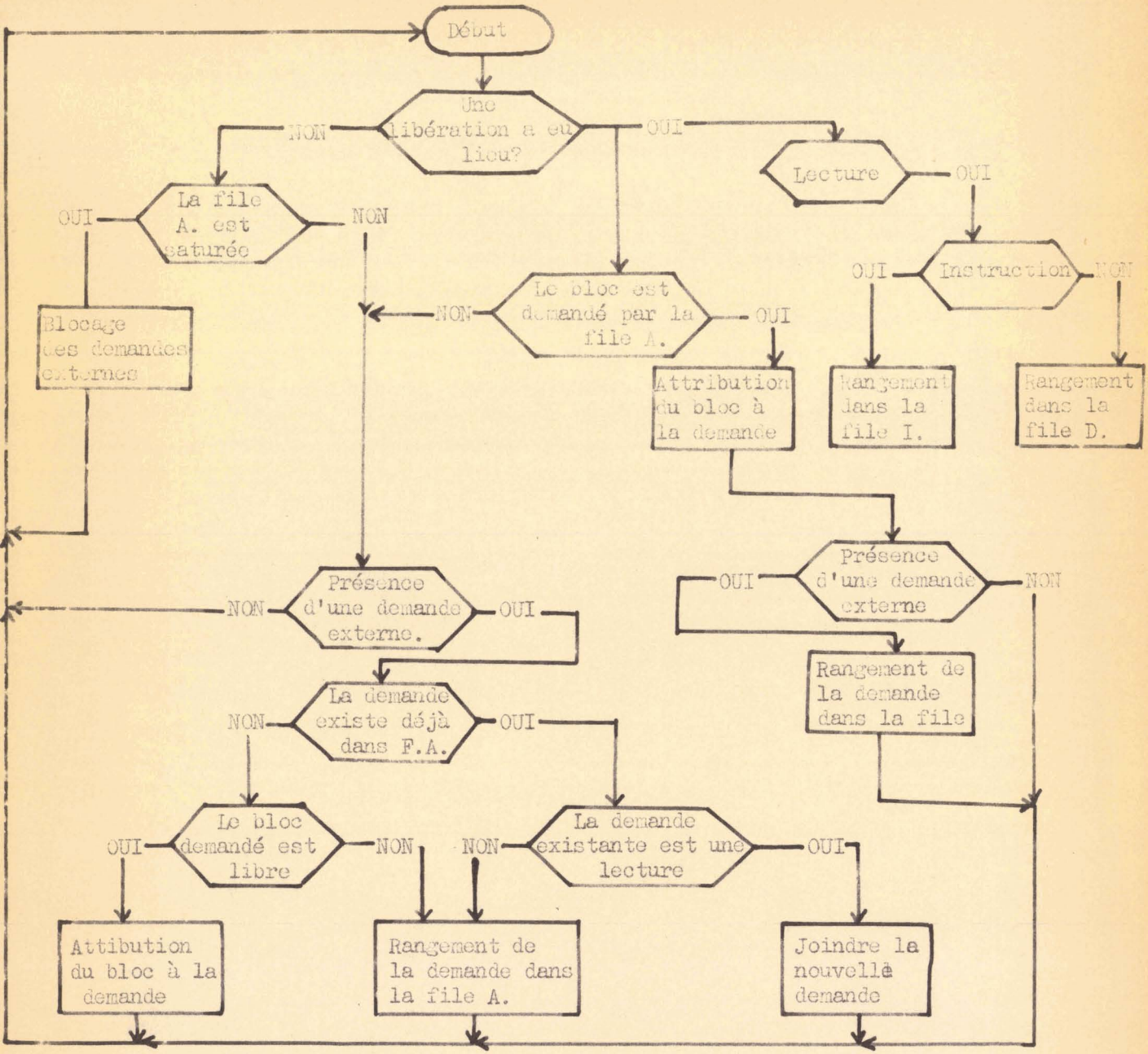
Par la suite, on adoptera les dénominations suivantes pour les files :

- File d'attente des demandes F. A.
- File d'attente des instructions F. I.
- File d'attente des données F. D.
- File d'attente d'écriture F. E.

Il ne semble pas utile de prévoir, pour les entrées-sorties, l'existence d'une file d'anticipation puisque la gestion des périphériques n'est pas synchronisée sur celle du traitement interne. Dans les échanges entre la mémoire et les organes périphériques, l'information traitée est, presque toujours, déposée dans des adresses successives: Il est alors inutile de connaitre le contenu d'une adresse dont le rang serait n dans la zone concernée par l'échange pour demander le contenu de l'adresse de rang n + 1.



ORGANIGRAMME D'UN PROCESSUS DE RECHERCHE ANTICIPEE



Remarque - Le fait qu'il existe deja une demande concernant la même adresse dans la file A, indique que le bloc concerné n'est pas libre. Si la demande existante est une écriture, la nouvelle demande prend place après celle-ci: Il n'y a donc aucun risque de détruire l'ordre séquentiel du programme. S'il s'agit d'une lecture, il est possible de n'effectuer qu'un seul cycle de mémoire pour les deux demandes.

Ordinairement, le fonctionnement des quatre files fondamentales est régi par les règles suivantes :
La file d'attente d'instructions qui est en fait une file d'anticipation, fournit son contenu, dans un ordre rigoureusement séquentiel, au bloc de commande. Cette file place une demande dès que l'une de ses positions se trouve libérée, de manière à se trouver toujours saturée.

La file d'attente de données qui fonctionne, elle aussi, en anticipation, suit la file d'attente instruction. Dès que la partie adresse d'une instruction éventuellement modifiée par l'adjonction d'une base, d'un index, contrôlée par des circuits de protection en multiprogrammation, a été obtenue, cette adresse fait l'objet d'une demande d'écriture s'il s'agit d'extraire le contenu de la mémoire. Si, au contraire, il s'agit de ranger une information dans la mémoire, il est impossible de demander le bloc correspondant tant que l'instruction n'aura pas été exécutée.

La file d'attente des demandes reçoit des adresses lorsque les demandes qui se sont présentées n'ont pu être satisfaites et procède à l'examen de toutes les demandes qu'elle contient, chaque fois qu'une libération se produit.

La file d'écriture reçoit les données à inscrire dès qu'elles sont fournies par les organes de traitement ou les canaux d'accès. Pour chacunes d'elles, la file transmet une demande à la mémoire, c'est-à-dire à la file d'attente des demandes.

Ordinairement, ces quatre files sont indépendantes et les différences qui existent dans leur contenu -adresses ou données- ainsi que dans leur gestion, justifient, à première vue, ce choix.

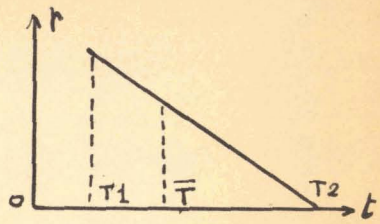
1) La file d'attente de données.

Elle doit être capable de fournir aux circuits de traitement les données demandées, sans délai. Pour cela, il faut qu'elle soit capable de régulariser convenablement les variations de débits entre son entrée et sa sortie.

Sa longueur ne peut être déterminée que par la connaissance de la loi de distribution des intervalles de temps entre deux demandes successives. Cette loi est discontinue car les temps de travaux sont souvent des multiples d'un même temps de cycle de base de l'unité de traitement. Elle est très liée à la nature des instructions cablées de la machine.

Afin de chiffrer, même très approximativement, les grandeurs qui interviennent, on fera l'hypothèse suivante :

La loi de distribution des intervalles de temps est linéairement décroissante et comprise entre deux valeurs extrêmes T_1 et T_2, les instructions les plus longues étant les plus rares.



La première condition à respecter est l'égalité des débits moyens à l'entrée et à la sortie.

L'intervalle de temps moyen est : $\overline{T} = \frac{T_2 - T_1}{3} + T_1$

Si, sur k demandes présentées à la mémoire, une fraction : a concerne les données, il faut respecter la condition :

$$a.\ k = \overline{T} / T_{cycle}$$

La fragilité de l'hypothèse faite sur la distribution des intervalles de temps entre deux demandes successives à la file de données et le fait que la loi de distribution des temps de services soit issue d'un calcul numérique ne permettent pas de calculer la longueur de la file par une méthode rigoureuse, inspirée des méthodes précédentes.

On peut obtenir cependant une précision supplémentaire par la définition des écarts quadratiques moyens sur chacune des deux lois statistiques :

$$\text{Pour les sorties : } \sigma = \int_{T_1}^{T_2} (t - \frac{T_2 - T_1}{T_2} + T_1)^2 \cdot \frac{2(T_2 - t)}{(T_2 - T_1)^2} \cdot dt = \frac{T_2 - T_1}{3.\ 2}$$

Pour les entrées, il faut procéder à une intégration graphique. Dans le cas de quatre blocs de mémoire, on trouve, dans les quatre cas étudiés, les valeurs suivantes de σ :

Taux de demandes	Temps de service moyen	écart quadratique
1	1,17 T_{cycle}	0,36 T_{cycle}
2	1,82 T_{cycle}	0,93 T_{cycle}
3	3,18 T_{cycle}	1,72 T_{cycle}
3,6	4,91 T_{cycle}	2,44 T_{cycle}

On peut alors exprimer la longueur de la file donnée en posant la condition suivante : Pendant le temps de service moyen, la longueur doit être suffisante pour satisfaire un taux moyen de demandes de la part de l'unité de traitement. Si on estime qu'une régularisation des débits sur les valeurs moyennes est par trop insuffisante, on peut ajouter à chacun des termes, les écarts quadratiques

moyens dans le sens le plus défavorable, c'est-à-dire en les ajoutant au temps de service et en les retranchant de l'intervalle entre deux demandes.

On prendra, par exemple, toujours dans le cas d'une mémoire partagée en quatre blocs, les valeurs suivantes pour les temps extrêmes de traitement :

Temps minimum : $T_1 = 0,2\ T$; Temps maximum : $T_2 = 5\ T$

Ceci conduit à une valeur moyenne : $\overline{T}_d = 1,6\ T$

et une valeur quadratique moyenne : $\sigma_d = 1,1\ T$

On trouve alors pour la longueur de la file donnée :

	Longueur
- Taux global de demandes, k = 1	
régularisation en valeurs moyennes :	2
régularisation en ecarts types :	3
- Taux global de demandes, k = 2	
Régularisation en valeurs moyennes :	4
Régularisation en écarts types :	6
- Taux global de demandes, k = 3	
Régularisation en valeurs moyennes :	6
Régularisation en écarts types :	10
- Taux global de demandes, k = 3,6	
Régularisation en valeurs moyennes :	10
Régularisation en écarts types :	15

2) La file d'attente d'instructions.

L'alimentation de la file de données ne peut se faire que lorsque les adresses de celles-ci sont connues. Comme, dans presque tous les cas, l'adresse sur laquelle porte le traitement figure dans l'instruction, il faut que la file d'instructions soit en avance sur la file de données. Cette avance doit être telle que les demandes de données puissent se faire dès qu'une position est libérée dans la file. Une position supplémentaire dans la file d'instructions, par rapport à la file de données semble pouvoir assurer cette condition. Si on analyse le processus complet qui, à partir du contenu du compteur ordinal anticipé, conduit à la fourniture à l'unité de traitement de tous les éléments nécessaires à l'exécution de l'instruction, on constate qu'il faut nécessairement disposer, l'un à la suite de l'autre, deux cycles de demandes et que le temps correspondant est donc deux fois le temps de service. La file d'attente d'instructions doit donc être notablement plus longue que la file d'attente de données ; en principe, le double, mais si l'on admet une certaine compensation statistique, il est possible de ne pas atteindre ce rapport.

Un autre risque se présente alors ; c'est que les demandes de données issues de la file d'instructions soient prématurées et que la mémoire ne puisse délivrer le résultat de la lecture dans une file de données qui est encore saturée. On peut alors imaginer de ne déclancher la recherche de la donnée que lorsque l'instruction a atteint une certaine profondeur dans la file.

3) La file d'attente d'écriture.

Lorsque l'une des unités ayant accès à la mémoire doit y déposer une information, celle-ci est transmise avec son adresse de rangement vers l'unité de contrôle qui examine si le bloc demandé est disponible. Si c'est le cas, les deux mots sont transmis au bloc et traités par celui-ci. Sinon, ils sont stockés dans une file d'attente. La file d'attente d'écriture est différente de celle de lecture car elle doit, non seulement contenir l'adresse, mais aussi l'information à y déposer.

Le problème majeur posé par l'existence de cette file est la nécessité de prévoir un rappel possible d'une information pour laquelle une demande d'écriture a été posée mais non encore réalisée. Il importe d'abord d'interdire la lecture tant que l'écriture n'a pas été faite. On l'obtient, dans le cas où le traitement reste uniquement linéique, en desservant les demandes présentes dans les files d'attente dans l'ordre où elles se sont présentées et en assurant toujours la priorité aux demandes d'écriture sur les demandes de lecture. On peut cependant imaginer de prendre directement l'information qui est présente dans les mémoires rapides de la file d'attente sans attendre qu'elle ait été, au préalable, rangée dans la mémoire. Il faut alors comparer chacune des demandes de lecture aux demandes d'écriture en attente dans la file et créer une liaison directe entre cette file et les liaisons de sortie.

Cette situation, qui peut sembler exceptionnelle, sera pratiquement très fréquente du fait que la mémoire sert de stockage temporaire dès que le nombre de registres de traitement est insuffisant. Pour l'éviter, il faudrait imposer aux programmeurs et aux programmes de traduction de ne redemander une information déposée dans la mémoire que lorsque la probabilité pour qu'elle y ait été rangée soit devenue très grande. Cette contrainte supplémentaire est inutile s'il existe un moyen relativement simple d'établir un "pont" entre l'écriture et la lecture. Il semble, en effet, fondamental de tenter d'éviter que l'art du programmeur puisse influer de manière notable sur les performances d'un système.

B - OPTIMISATION DE LA RECHERCHE ANTICIPEE.

L'ensemble des registres figurant dans les files d'attente nécessaires à la recherche anticipée possèdent un certain nombre de caractéristiques communes bien que leur rôle soit différent. Jusqu'à présent, ce sont surtout les différences qui ont été les facteurs déterminants de leur organisation logique ; les files sont séparées les unes des autres et les relations qui peuvent exister entre elles sont le fait de structures logiques supplémentaires et particularisées.

Si tous les registres de stockage provisoire associés à la mémoire pouvaient être banalisés, c'est-à-dire recevoir indifféremment les informations propres à chacune des files, il en résulterait les avantages suivants :

- L'unicité de la logique doit permettre de diminuer le prix des circuits de contrôle pour un nombre donné de positions ou d'accroître ce nombre à prix équivalent.
- La probabilité de saturation de l'une des files est diminuée puisque chacune d'elles peut emprunter à ses voisines le nombre de positions excédentaire dont elle a provisoirement besoin ; ceci dans la mesure où ces voisines comportent des positions libres. Pour des files indépendantes, il y a saturation chaque fois que l'une d'elles est saturée ; il en résulte immédiatement le blocage du traitement. Dans le cas de registres banalisés, ce blocage n'intervient que lorsque toutes les positions sont occupées.
- Les échanges entre les positions de stockage peuvent être normalisés et la logique qui les contrôle n'existera qu'à un seul exemplaire.
- En cas de panne, pour des machines dans lesquelles on désire accroitre le taux de sécurité, il suffit d'exclure la position déficiente ; la machine peut continuer à fonctionner avec, il est vrai, des performances un peu réduites.

Afin de banaliser toutes les positions des files d'attente, on peut partir du principe suivant : tout ce qui est commun à toutes les files sera réalisé par circuits ; tout ce qui particularise l'une des files par rapport aux autres sera mémorisé dans les positions correspondantes sous forme d'un mot identificateur ; ce mot qui n'aura de sens que pour les circuits de contrôle de la mémoire sera créé et exploité par eux.

Comme l'accès au groupe des registres doit pouvoir se faire en plusieurs endroits, les structures telles que les piles et les registres à décalage doivent être rejetées.

1) Caractéristiques communes aux files d'attente.

a) Toutes les files concernent la mémoire et chacune de leurs positions désigne une adresse de la mémoire centrale. L'adresse qui doit être stockée dans une position de file comporte deux parties : la première, formée des poids faibles désigne le bloc, la seconde formée des poids forts désigne la position dans le bloc. Seule la seconde partie est transmise au bloc au moment où il entreprend le travail commandé. La première partie, qui est de longueur moindre, est traitée par la logique de contrôle ; elle doit donc être en permanence à la disposition de celle-ci dès que la demande a été introduite dans la file.

b) Toutes les positions de file d'attente doivent contenir une information. Pour les opérations de lecture, cette information n'apparaîtra que lorsque le bloc sélectionné aura terminé son cycle tandis que dans les opérations d'écriture, l'information à traiter est chargée dans la file en même temps que la demande.

c) A l'intérieur d'une même file, particulièrement si les positions de cette file sont distribuées de manière aléatoire, dans les registres, doit exister un ordre de rangement qui puisse être modifié, consulté et prévu de telle manière que ces opérations puissent se faire simultanément sur toutes les positions allouées à la file. Pour la file d'attente instructions, cet ordre correspond à la séquence de programme, pour les autres files, il représente la priorité relative de chacune des demandes.

2) Principe de l'organisation proposée.

Une organisation de recherche anticipée peut se schématiser par l'adjonction d'une mémoire tampon entre la mémoire centrale et les circuits d'exécution. Comme les opérations effectuées, les liaisons qui s'établissent et les données traitées sont de nature diverses, il a semblé naturel de spécialiser les fonctions. Les remarques faites plus haut ne peuvent alors prendre effet. Si, au contraire, on part de ces points, on peut considérer qu'une seule mémoire tampon doit être capable d'assurer la totalité des travaux.

On appelle mémoire de recherche, l'ensemble des registres destinés à stocker les demandes. Cette mémoire de recherche servira systématiquement d'intermédiaire entre la mémoire centrale et les circuits utilisateurs, c'est-à-dire que tout échange entre eux se fera par son intermédiaire. Si le temps de cycle de la mémoire de recherche est assez court, cette solution ne présente

pas d'inconvénient car elle ne diminue qu'à peine le débit. Par contre, elle simplifie beaucoup la logique de contrôle, la file d'attente n'étant plus un demandeur prioritaire comme il était supposé jusqu'à présent, mais bien le seul demandeur ayant accès à la mémoire centrale.

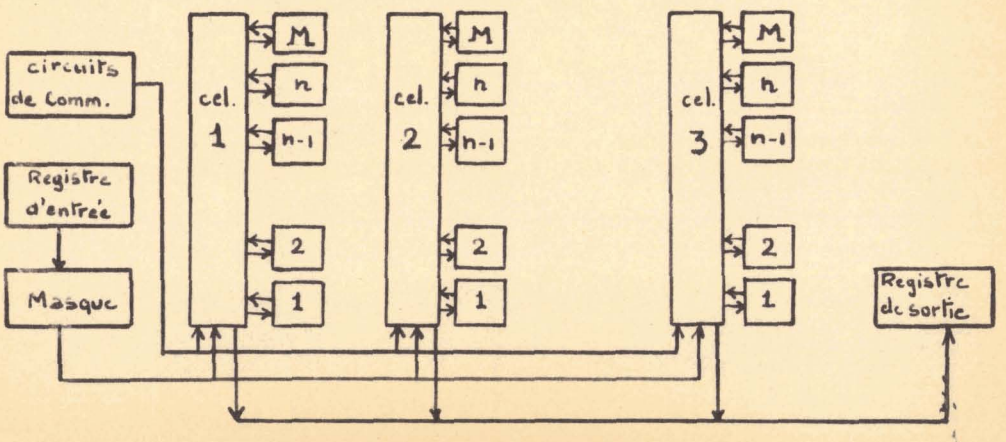
Lorsque l'un des circuits ayant accès à la mémoire formule une demande, celle-ci est caractérisée par une adresse de la mémoire centrale. On peut, d'autre part, considérer que la mémoire de recherche contient une image réduite de la mémoire centrale. Cette image ne reproduit de la mémoire centrale que ce qui est, instantanément, utile au traitement en cours. Comme cette image est réduite, elle doit contenir par chaque recopie du contenu de la mémoire, ce contenu lui-même et l'adresse correspondante. L'adresse émise par un circuit utilisateur à l'accasion d'une demande est donc une partie de l'information stockée dans une position de la mémoire de recherche.

La seconde caractéristique fondamentale de la mémoire de recherche sera donc qu'elle doit être organisée comme une mémoire associative. La partie du contenu qui sert à la sélection est l'adresse de la mémoire centrale ou, pour l'écriture, une partie de cette adresse.

3) La mémoire à logique distribuée.

La mémoire à logique distribuée est devenue l'un des modèles les plus employés de mémoires associatives. Sa désignation est la traduction de la phrase américaine "Distributed logic memory" ou D.L.M.

La mémoire à logique distribuée est un groupe de cellules travaillant en parallèle et disposant en commun d'une unité de contrôle, d'une voie d'entrée et d'une voie de sortie. Chaque cellule contient autant de positions binaires qu'il faut stocker de bits plus une position dite d'activité que l'on appellera la bascule M. Une cellule pour laquelle M = 1 est dite active, dans le cas contraire, elle est au repos.



La logique de chaque cellule est capable d'effectuer la comparaison entre son propre contenu et le mot présenté à l'entrée. Elle peut aussi réaliser les opérations habituelles de lecture et d'écriture depuis ou vers les registres d'entrée ou de sortie. La communication entre les cellules est faite par les bascules M, chaque bascule pouvant être activée par sa propre logique ou par les logiques des cellules voisines.

Lors d'une opération faisant intervenir le contenu du registre d'entrée, un masque peut être appliqué à ce registre de manière à ne faire porter la comparaison ou l'écriture que sur une partie des mots traités ou des mots enregistrés.

Les opérations qui peuvent être commandées à la mémoire sont :

A - ACTIVER : Activer (faire M = 1) toutes les cellules dont le contenu correspond au mot présenté à l'entrée.

AE - ACTIVER EXCLUSIVEMENT : Activer toutes les cellules dont le contenu correspond au mot présenté à l'entrée et mettre au repos toutes les autres.

AD, AG - ACTIVER A DROITE (A GAUCHE). Activer les cellules situées à droite (ou à gauche) des cellules dont le contenu correspond au mot présenté à l'entrée.

AEG, AED - ACTIVER EXCLUSIVEMENT A DROITE (A GAUCHE). Activer comme dans le cas précédent et mettre au repos les cellules qui n'ont pas été activées.

E - ECRIRE : Stocker le mot présenté à l'entrée dans toutes les cellules.

EC - ECRIRE CONDITIONNELLEMENT : Ecrire le mot présenté à l'entrée dans les seules cellules actives.

L - LIRE : Lire le contenu de toutes les cellules actives. Le mot qui apparaît dans le registre de sortie est formé de l'union des contenus des cellules actives.

Pour toutes ces opérations, le traitement se fait en parallèle. La partie la plus importante de la logique associée à chaque cellule est un comparateur logique dont la sortie débouche soit sur la bascule M de la cellule, soit sur celles de ses voisines, soit sur des portes de communication entre les bascules de la cellule et les registres d'entrée et de sortie.

A la mémoire de recherche ainsi définie, il faut associer quelques circuits externes :

- Une unité de contrôle capable de délivrer des séquences d'ordres à la mémoire de recherche et à la mémoire centrale et d'aiguiller les informations entre les divers registres. Cette unité, en même temps que les ordres, délivrera les masques nécessaires à l'identification des données présentées aux cellules.

Le seul élément conditionnel sur lequel l'unité de contrôle peut orienter le traitement des demandes est lié à l'ordre de lecture. Cet ordre provoque l'union logique des mots contenus dans les cellules actives, de telle sorte qu'il est impossible de savoir si le contenu du registre de sortie est un mot unique ou la superposition de plusieurs mots. Pour lever cette indétermination, la logique de lecture doit être capable de déterminer si une, plusieurs ou aucune des cellules de la mémoire de recherche étaient actives au moment de la lecture.

- Pour les files anticipées de lecture, il faut un circuit capable de formuler les demandes à l'instant voulu et en définissant l'adresse convenable. Pour le programme et les échanges avec les canaux d'entrée - sortie, il suffit d'un compteur puisque les demandes portent sur des adresses disposées en séquence. Pour les données dont les adresses figurent dans les instructions, il faut une logique capable d'isoler la partie adresse de l'instruction, d'apporter les modifications d'adressage et de translation et de déceler la présence d'indicateurs d'adressage indirect.

- Enfin, pour accélérer le rythme de la mémoire de recherche, il est utile de disposer certaines informations à l'extérieur de la mémoire. On pourrait imaginer une organisation où la mémoire de recherche existerait seule, mais le nombre de cycles qu'elle aurait à fournir pour traiter chaque demande serait bien plus important, or, au niveau de sa réalisation pratique, ce sont les limitations de vitesse qui semblent les plus restrictives ; c'est donc elles qu'il faut considérer en premier lieu.

On pourrait imaginer de nombreuses variantes à l'organisation qui est décrite, les unes plus rapides, les autres plus économiques. Les deux principes retenus sont d'une part la sélection des cellules par leur contenu et, d'autre part, un rangement interne organisé de telle manière qu'il permette l'accès à la cellule cherchée soit directement, soit par une opération indirecte à un seul niveau.

Dans les échanges avec les circuits utilisateurs, la mémoire de recherche fonctionne comme une mémoire associative.

Dans les échanges avec la mémoire, il faut tenir, dans les cellules, deux comptabilités :

1) Le recensement des cellules libres et des cellules occupées.
2) Les priorités des demandes enregistrées dans les cellules occupées et leur enchaînement.

Il faut donc créer deux chaînes distinctes. Or, pour définir une cellule par une autre, cette dernière doit contenir une information qui définisse, sans ambiguité, la cellule qui la suit. Il est impossible d'utiliser à cette fin, l'adresse de la demande car une même adresse peut faire l'objet de plusieurs demandes. Les cellules seront donc numérotées et chacune contiendra les positions nécessaires pour contenir le numéro servant à désigner une autre cellule. Ceci revient à dire que la mémoire de recherche doit être aussi adressable.

Elle peut l'être directement à l'aide d'une logique supplémentaire.

Elle peut aussi l'être indirectement par son contenu à condition de déposer dans chaque cellule le numéro qui la définit. La seconde solution semble la plus homogène puisqu'elle ne modifie pas la logique de sélection. De plus, les bascules destinées à contenir le numéro de la cellule contiennent des informations fixes et n'ont pas besoin d'être cablées.

4) Chaîne des cellules disponibles.

Au départ, chaque cellule contient le numéro d'une autre cellule disponible à prendre à sa suite. La position correspondante de la cellule s'appelle le chaînon de disponibilité (CD). La chaînon de la dernière position n'existe pas puisque si on charge cette cellule, la mémoire de recherche est saturée.

Lorsqu'une cellule est chargée, son chaînon est disposé, à l'extérieur de la mémoire dans un registre dit : registre de chaînon de disponibilité (RCD) qui contient donc le numéro de la première cellule disponible. Les cellules sont donc rangées dans l'ordre de leur entrée en service au rythme des demandes qui se présenteront.

Lorsqu'une cellule est libérée, on charge dans la cellule définie par RCD, le numéro de cette cellule libérée. Il y a donc reprise dans l'ordre des cellules qui ont terminé leur travail.

L'absence de chaînon de disponibilité dans une cellule indique que cette cellule est la dernière disponible; la mémoire de recherche est donc saturée et il faut bloquer les demandes.

Exemple de fonctionnement

Numéro de cellule	1	2	3	4	5	6	7	8	9
Occupation	X		X	X		X		X	
Chainon de disponibilité		9			2		5		

Supposons qu'à l'instant considéré, le registre chaînon de disponibilité contienne le numéro 7.

a) Si une demande se présente, le chaînon de 7, c'est-à-dire 5 dans le cas présent, prend place dans RCD. La cellule 5, s'il ne se produit pas de libération entre temps, sera mise à la disposition de la prochaine demande.

b) Si une libération se produit, soit, par exemple la cellule 4, on charge dans la cellule désignée par RCD soit 7, un chaînon désignant la cellule libérée, c'est-à-dire 4. L'ancienne chaîne qui était 7-5-2-9 est devenue 7-4-5-2-9. Pour refermer la chaîne, il faut aussi déposer dans la cellule libérée l'ancien chaînon contenu dans 7.

L'avantage de cette organisation provient de ce que les cellules concernées par les opérations de modification sont toutes immédiatement disponibles, l'une parce qu'elle vient d'être libérée, la seconde parce qu'elle est désignée par RCD et la dernière, dont le numéro apparait dans le chaînon de la seconde.

Il en résulte cependant que les cellules qui ont terminé un travail ont de fortes chances de ne pas rester disponibles longtemps. Sur de courtes périodes, l'activité des différentes cellules risque donc d'être très variable. Cela n'a pas d'importance puisque le vieillissement des circuits électroniques n'est pas lié à leur activité. Si l'on désirait remettre la cellule à la fin de la chaîne, il faudrait procéder en pas à pas et balayer toute la chaîne à partir de la cellule désignée par RCD à l'aide, par exemple, des ordres ACTIVER DROITE ou ACTIVER GAUCHE.

5) Chaîne des priorités.

Dans les cellules occupées, un rangement doit intervenir de manière à respecter les priorités des demandes les unes par rapport aux autres. Une demande ne peut pas recevoir, au moment de son enregistrement un degré de priorité fixe car la moins favorisée risquerait d'attendre indéfiniment un accès au bloc qu'elle demande.

Si l'on désire conserver une organisation simple, on peut admettre deux règles de priorités : les demandes sont prises en charge dans l'ordre rigoureux où elles se présentent ; il faut alors établir pour les cellules occupées un chaînage identique à celui des cellules disponibles.

Une seconde solution, plus souple et d'un rendement plus élevé consiste à n'établir de priorités relatives qu'entre les demandes portant sur un même bloc de la mémoire centrale. L'ordre dans lequel les organes demandeurs exploiteront ces demandes sera rétabli par elles et la mémoire de recherche se contente d'y répondre en exploitant au mieux les temps disponibles.

Les chaînes qui s'établissent ne portent plus que sur les demandes qui portent sur le même bloc et il doit donc exister autant de chaînes qu'il y a de blocs. Il n'est plus aussi facile de traiter ces chaînes car on ne peut envisager l'existence d'autant de registres de tête de chaîne qu'il y a de blocs, à l'extérieur de la mémoire de recherche. De plus, l'ordre établi dans chaque chaîne est imposé par l'ordre dans lequel les demandes se sont présentées et ne peut être modifié.

Chaque chaîne doit comporter un début qui est la première demande à s'être présentée donc celle qui jouit de la plus grande priorité, une fin de chaîne qui est la dernière demande présentée et qui doit être connue afin de permettre l'enregistrement d'une nouvelle demande à sa suite. Puisque la comptabilité de ces chaînes ne peut être assurée à l'extérieur de la mémoire de recherche, il faut stocker dans chaque cellule les informations nécessaires au chaînage.

Les chaînons reliant les demandes pourront être pris dans les mêmes positions que les chaînons de disponibilité puisque la cellule ne peut appartenir simultanément aux deux chaînes.

Il faudra de plus deux positions, l'une indiquant le début d'une chaîne de priorité et l'autre sa fin. On appellera D et F les positions binaires ainsi définies. Il faut deux positions indépendantes puisqu'une même cellule peut être à la fois le début et la fin d'une chaîne si elle contient la seule demande relative à un bloc donné.

La gestion des chaînes de priorité doit être capable de traiter les demandes qui se présentent comme celles qui sont totalement satisfaites et qui disparaissent de la mémoire de recherche. En fait, certaines demandes peuvent rester dans la mémoire alors qu'elles n'ont plus besoin d'accès à l'un des blocs de la mémoire centrale. Il s'agit des cellules qui participent aux files

d'attente d'instructions et de données et qui, ayant été chargées par anticipation, ne seront libérées que lorsque la demande effective se sera présentée.

Ces demandes ne font partie ni des chaînes de priorité, ni des chaînes de disponibilité et il faut prévoir, à leur sujet, un traitement particulier.

Introduction d'une demande dans la chaîne. Lorsqu'une demande se présente, il faut la ranger par rapport aux demandes arrivées avant elles et portant sur le même bloc de la mémoire centrale. La sélection dans la mémoire de recherche se fera donc sur la partie de l'adresse complète qui désigne le bloc. La logique de contrôle de la lecture permet de savoir si, une, plusieurs ou aucune des cellules activées a répondu à la demande.

Si aucune des cellules n'a répondu à la demande, c'est que le bloc est libre et que la demande peut être transférée immédiatement vers ce bloc. Elle est néanmoins enregistrée. Elle comporte les bits D et F.

Si une demande unique existait déjà dans la chaîne, elle contient à la fois les bits D et F. On lui conservera le bit D et on lui ajoutera le contenu de RCD dans les positions réservées au chaînon. La demande qui se présente sera enregistrée dans la cellule désignée par RCD avec le bit F.

Si plusieurs demandes précédaient la demande traitée, il suffit de lire celle qui contient le bit F, d'y déposer le contenu de RCD et de charger la demande traitée dans la cellule désignée par RCD avec le bit F.

Exploration des demandes de la chaîne. Chaque fois qu'un bloc de la mémoire centrale a terminé un cycle, une demande est faite à la mémoire de recherche sur la base de la partie de l'adresse désignant ce bloc et du bit D. S'il n'y avait aucune cellule demandant ce bloc, ce dernier est laissé au repos. S'il y avait une ou plusieurs demandes en attente, l'une d'elles contient ce bit D. On transfère alors cette demande vers la mémoire et on analyse les positions qui définissent son emplacement dans la chaîne des priorités.

Si la demande contenait le bit F, c'est qu'elle est la seule à demander ce bloc de la mémoire centrale ; il n'y a alors rien à faire puisque le fait de la servir élimine la chaîne correspondante.

Si la demande ne contenait pas le bit F, elle n'est pas la seule demande de la chaîne. Le chaînon qu'elle contient définit la cellule qu'il faudra servir à sa suite ; on place le bit D dans cette dernière cellule.

Libération d'une cellule. Lorsqu'une demande a été desservie, la cellule est introduite dans la chaîne de disponibilité. Dans le cas d'une demande de lecture, cela se produit lorsque le circuit qui réclame l'information à lire en a fait la demande et que cette demande a été satisfaite. Pour une demande d'écriture, on peut considérer que la demande peut être éliminée de la mémoire de recherche dès qu'elle a été transmise vers la mémoire centrale. Afin d'éviter cependant qu'une demande de lecture sur la même adresse ne soit différée trop longtemps, on peut conserver la demande tant que dure le cycle du bloc de la mémoire centrale.

C - FORMAT DES DONNEES TRAITEES.

Comme les cellules sont banalisées, chacune doit être capable de contenir les informations nécessaires à toutes les opérations possibles. Il n'y a pratiquement pas beaucoup de place mal utilisée car les positions les plus nombreuses contiennent des informations nécessaires à toutes les demandes.

A l'intérieur d'une cellule on trouve :

- Les données lues ou à écrire. Ces données qui représentent la partie la plus importante de la cellule peuvent être contenues dans des positions non connectées à la logique de sélection. Elles ne participent pas aux opérations d'activation des cellules et ne servent qu'à des stockages provisoires.

- Les adresses complètes dans lesquelles les données doivent prendre place ou dans lesquelles il faut les lire. Une adresse complète se subdivise en deux parties : le numéro du bloc défini sur les poids faibles de l'adresse et la position dans le bloc définie sur les poids forts ; cette dernière partie, la plus volumineuse, sera appelée l'adresse réduite.

- Un indicatif du canal demandeur qui se trouve à l'origine de l'information enregistrée dans la cellule.

- Un chaînon qui servira soit à indiquer la position de la cellule dans la chaîne de disponibilité soit à définir sa priorité par rapport aux demandes portant sur le même bloc

- Deux positions binaires contenant les indications de début et de fin (D et F) de la chaîne de priorité à laquelle appartient la cellule.

- Un certain nombre de positions binaires de sélection servant à indiquer soit la nature du travail demandé à la mémoire soit la progression de celui-ci.

- Un numéro de cellule composé de positions fixes connectées à la logique de sélection et servant à définir la position d'une cellule à partir de son adresse.

En dehors des données enregistrées, toutes ces positions de la cellule peuvent servir, seules ou associées à d'autres, à activer la cellule à partir d'un mot présenté dans le registre d'entrée.

Les échanges entre la mémoire de recherche et l'extérieur peuvent se classer en deux catégories :

Les entrées :

- Demandes d'écriture depuis un circuit utilisateur.
  Information / Adresse / Bit d'écriture.
- Demande de lecture par anticipation
  / Adresse / bit de lecture / Numéro du demandeu
- Demande de lecture immédiate.
  / Adresse /bit de lecture /numéro du demandeur
- Résultat d'une opération de lecture.
  Information/ numéro du bloc.

Les sorties :

- Emission vers un demandeur du résultat d'une lecture.
  Information / Numéro du demandeur /
- Ordre de lecture à la mémoire
  / Adresse complète / Bit de lecture
- Ordre d'écriture à la mémoire.
  Information / Adresse complète / Bit d'écriture.

Les signalisations :

Bien qu'il n'y ait pas d'échange d'informations, il faut prévoir qu'un bloc qui a terminé une opération d'écriture puisse le signaler à la logique de contrôle.

Il est nécessaire de prévoir que toutes les opérations d'échange qui peuvent s'établir entre la logique de la mémoire de recherche et les demandeurs ou la mémoire centrale soient signalés aux circuits de contrôle.

On supposera donc que tous les transferts indiqués plus haut sont initialisés par un appel à ces circuits de contrôle sous une forme permettant de reconnaître leur origine ou leur nature.

Les ordres émis par cette logique de contrôle concernent soit la mémoire de recherche soit des transferts entre les registres d'entrée et de sortie de celle-ci et l'extérieur. Les ordres internes appartiennent à la liste dressée plus haut :

Exemple : AE (numéro de bloc) si 0 : signifie ACTIVER EXCLUSIVEMENT les cellules à travers un masque qui ne laisse passer que le numéro de bloc, à la condition que la précédente lecture n'ait concerné aucune des cellules.

Les transferts avec l'extérieur seront décrits sous forme littérale.

D - SEQUENCES DE TRAITEMENT DES DEMANDES.

Parmi les positions binaires de sélection indiquant la nature et l'état d'avancement des demandes stockées dans les cellules, on trouvera :

- Une position I indiquant une demande de lecture immédiate.
- Une position A indiquant une demande de lecture anticipée.
- Une position E indiquant une demande d'écriture.
- Une position R indiquant, pour les demandes de lecture, qu'elles attendent la disponibilité du bloc qu'elles désignent.
- Une position S indiquant, pour ces mêmes demandes, que la mémoire a terminé la lecture et que le résultat se trouve à la disposition de la demande de lecture immédiate qui doit se présenter.
- Une position T indiquant qu'un bloc de la mémoire centrale se trouve actuellement au travail pour la cellule. Cette position T permet de savoir à chaque instant quels sont les blocs de la mémoire centrale qui sont au travail.
- Deux positions binaires D et F indiquant le début et la fin de la chaîne de disponibilité de chaque bloc de mémoire centrale.

Séquence de traitement des demandes anticipées.

La logique d'anticipation présente à l'entrée de la mémoire de recherche un mot composé de:

ADRESSE	BIT A	N° DU CANAL DEMANDEUR

		CODE	MASQUE	CONDITION
	Le mot est-il dans la file d'attente d'écriture?	AE	ADR., E	
		LIRE		
	Si non allera (1)	BRA.		SI 0
	Transférer la partie information qui vient d'être lue depuis le registre de sortie vers le registre d'entrée.	Registre sortie vers registre entrée		
		Registre C.D. vers registre entrée		
	Ranger le mot: INFORMATION | ADRESSE | BIT S | N° CANAL dans l'adresse définie par RCD.	AE	CHAINON	
		LIRE		
		Registre sortie vers registre C.D.		
	Mettre dans RCD le nouveau chainon	EC		
	FIN			
(1)	Le bloc demandé est-il libre?	AE	N°BLOC,T	
		LIRE		
	Si non aller à (2)	BRA		SI 1
	Ranger le mot: ADRESSE | BIT T | BIT A | N° CANAL DANS l'adresse définie par RCD.	Suite de la séquence identique à la séquence précédente.		
	Mettre dans RCD le nouveau chainon.			
	Lire la partie adresse de la cellule et transférer cette adresse vers la mémoire centrale.	LIRE		
		Registre sortie vers la mémoire		(partie adresse)
	FIN			
(2)	Chercher la fin de la chaine. Cette fin existe-t-elle?	AE	N°BLOC,F	
		LIRE		
	Si oui aller à (3)	BRA		SI 1
	Ranger le mot: ADRESSE | BIT R | BIT A | BITS D ET F | N° CANAL dans la cellule définie par RCD.	Suite identique à la séquence précédente		
	Mettre dans RCD le nouveau chainon.			
	FIN			
(3)	Lire la fin de chaine. Placer le contenu de RCD dans le chainon de cette cellule. Supprimer le bit F qui s'y trouve.	Registre C.D. vers registre d'entrée		
		EC	CHAINON,F	
	Ranger le mot: ADRESSE | BIT R | BIT A | BIT F | N° CANAL dans la cellule désignée par RCD	Suite identique à la séquence précédente		
	Mettre dans RCD le nouveau chainon			
	FIN			

Séquence de traitement des demandes provenant de la mémoire.

Un bloc de mémoire qui a reçu les informations nécessaires à la poursuite de son cycle fonctionne de manière indépendante. Lorsque le cycle est terminé, il le signale en transférant vers la logique de contrôle son numéro et, s'il s'agit d'une lecture, l'information lue.

	Code	MASQUE	CONDITION
L'opération demandée était une lecture?	AE LIRE	N°BLOC,T,E	
Si oui aller à (1)	BRA		SI 0
Retirer la cellule de la chaine d'attente	AE LIRE	N°BLOC?T CHAINON	
Mettre la cellule dans la chaine de disponibilité.	E	T	T=0
	Registre sortie		vers registre entrée
FIN	AE	N°CELLULE	
	EC	D	
	Registre C.D.		vers registre d'entrée
(1) La demande de lecture était anticipée?	AE	N°CELLULE	
	EC	CHAINON	
Si oui aller à (2)	AE	N°BLOC,T,I	
	BRA LIRE		SI 0
Transférer l'information vers le canal demandeur.	Registre sortie		vers canal demandeur
Retirer la cellule de la chaine d'attente		Suite identique à la séquence précédente.	
Mettre la cellule dans la chaine de disponibilité.			
(2) La demande étant anticipée, il faut ranger l'information accompagnée du bit S dans la cellule. Cette demande est disponible pour une prochaine interrogation.	AE	N°BLOC,T,A	
	EC	Inform.,S	

Remarque. Si l'on désire augmenter la vitesse de traitement, on peut imposer au bloc de signaler la nature du travail qu'il vient de terminer.

Il est par ailleurs évident que la signalisation, pour une demande de lecture, sera émise par le bloc lorsque l'information sera disponible et non lorsque le cycle sera terminé.

Séquence de traitement des demandes immédiates.

Lorsqu'une demande de lecture ne peut être transmise à la mémoire de recherche avant de prendre effet par la logique d'anticipation, le circuit qui la présente est supposé attendre afin de respecter l'ordre séquentiel du traitment. Habituellement, néammoins, la demande doit trouver dans la mémoire de recherche, l'information réclamée.

	CODE	MASQUE	CONDITION
La demande a-t-elle fait l'objet d'une anticipation	AE	ADRESSE, N°CANAL BIT A	
Si non reprendre la séquence des demandes anticipées.	LIRE BRA		SI 0
La demande a-t-elle été servie	AE	ADRESSE N°CANAL BIT S	
si non FIN. (Le canal maintient sa demande tant qu'il n'aura pas obtenu satisfaction)	LIRE BRA		SI 0
Transférer le contenu de la cellule sélectionnée vers le canal demandeur	REgistre sortie vers canal demandeur		
Mettre la cellule dans la chaine de disponibilité.	Registre C.D. vers registre entrée AE EC	 CHAINON CHAINON	

Remarques. Comme l'adresse peut faire simultanément l'objet de demandes issues de canaux différents, la sélection initiale porte aussi sur le numéro du canal demandeur.

On pourrait distinguer deux cas de demandes non desservies:

- Ou bien le bloc est au travail et il ne peut être question alors d'obtenir plus vite satisfaction.
- Ou bien le bloc n'est pas encore au travail et on peut satisfaire l'urgence de cette demande en modifiant sa position dans la chaine des demandes relatives au bloc.

On reprendra alors la séquence de traitement des demandes anticipées en ajoutant à la cellule le bit D. On rétablira la chaine en déposant dans cette cellule un chainon désignant celle qui, auparavant, contenait de bit D et dans laquelle il faut le détruire.

La demande a été ainsi introduite au début de la chaine et non à la fin.

Séquence de traitement des demandes d'écriture.

La mémoire de recherche enregistre les demandes d'écriture et les conserve jusqu'à ce que l'opération soit terminée dans la mémoire. Cette disposition entraine un taux d'occupation un peu plus élevé dans la mémoire de recherche mais permet la relecture immédiate d'une information présente dans la chaine d'écriture.

Cette remarque est fondamentale car, dans ces conditions, la mémoire de recherche peut jouer exactement le rôle, que l'on confie ordinairement à une mémoire ultra-rapide supplémentaire, de stockage provisoire des résultats intermédiaires d'un calcul.

	CODE	MASQUE	CONDITION
Le bloc est-il disponible?	AE	N°BLOC,T	
	LIRE		
Si non aller à (1)	BRA		SI 1
	Registre C.D. vers registre entrée		
Ranger le mot: INFORMATION | ADRESSE | BIT T | BIT E | NUMERO DE CANAL dans l'adresse définie par RCD	AE	CHAINON	
	LIRE		
	Registre sortie vers registre C.D.		
Mettre dans RCD le nouveau chainon	EC	Le mot correspondant à la demande	
Lire la partie adresse de la cellule			
transférer cette adresse vers la mémoire accompagnée d'une demande d'écriture.			
FIN			

(1) La suite de la séquence est identique à la séquence correspondant à une demande de lecture anticipée puisque la demande doit être introduite dans la chaine des demandes relatives au bloc.

Seul le bit A sera remplacé par un bit E.

E - POSSIBILITES ACTUELLES DE REALISATION.

On a longtemps dit que la réalisation de mémoires associatives ne deviendrait pratiquement possible qu'avec l'apparition de nouvelles technologies se prêtant mieux à la nature complexes des décisions qui doivent s'y prendre. On met, à l'heure actuelle beaucoup d'espoirs dans les logiques supra-conductives mais elles restent encore au stade du laboratoire; de plus il semble difficile d'obtenir les vitesses souhaitables dans l'application présente. Lorsqu'on parle de mémoire associatice, on envisage souvent la réalisation d'une mémoire centrale totalement organisée de cette manière. Pour une mémoire telle que celle qui est proposée, et qui ne contiendra que quelques dizaines de cellules, l'échelle est totalement différente.

Etant donnée la part active que prennent les éléments mémorisés dans les décisions qui les concernent, il semble naturel de faire appel à des circuits logiques.

Les principaux avantages qui en résultent sont:

- Vitesse de fonctionnement élevée ce qui est ici une condition indispensable.
- Compatibilité de niveau avec les circuits de commande et d'accés d'où il résulte une simplification des inter-connexions et un gain de temps, les circuits d'adaptation d'impédances apportant toujours dans une chaine logique des délais supplémentaires.
- Modularité aisée permettant d'associer des éléments de base identiques dans des cofigurations parfaitement adaptées à chaque cas particulier.

Une technologie à semi-conducteurs sous forme de circuits intégrés semble donc préférable à toute autre. Il est dès à présent possible d'affirmer qu'on obtiendrait une telle structure à un prix abordable et ceci d'autant plus son organisation étant totalement répétitive, la standardisation peut étre poussée à l'extrême.

CONCLUSION.

On s'accorde habituellement à reconnaitre qu'au dessus d'un certain seuil, les performances s'obtiennent au prix d'un effort technologique dont le prix croît comme l'exponentielle de l'accroissement de vitesse obtenu. Les circuits de stockage n'échappent pas à cette loi. Comme le besoin de disposer de mémoires rapides, de capacité moyenne ou grande se fait de plus en plus pressant, le partage de la mémoire devient une nécéssité. Le coût d'une mémoire partagée, s'il n'est pas tout à fait proportionnel au nombre de blocs indépendants, suit une loi assez voisine. Les performances que l'on peut en attendre autorisent la conception d'une classe de machines hors de portée des possibilités d'une mémoire unique.

Bien qu'alors les performances ne puissent se chiffrer que de manière statistique, il est possible d'en connaitre exactement la valeur à condition de définir tout aussi exactement les conditions dans lesquelles elle se trouve exploitée et l'organisation logique des circuits chargés de la contrôler. Bien que complexe dans sa définition, une telle organisation peut être unifiée et donc être confiée à des éléments très répétitifs, relativement simples et donc d'un prix et d'une mise en oeuvre abordables.

Un autre élément caractèristique des techniques du traitement de l'information est la pauvreté des méthodes sytèmatiques permettant de définir la structure des machines. L'algébre logique et la théorie des circuits combinatoires constituent des outils sûrs pour l'étude des fonctions de base. A l'autre extrémité, les théories des algorithmes, des langages et des méthodes de programmation se développent et fournissent des moyens prècis d'aborder les questions relatives à l'utilisation des systèmes de traitement de l'information.

Dans le domaine intermédiaire de la conception de ces systèmes, il n'existe actuellement que peu de méthodes qui permettent de définir au mieux les assemblages de fonctions capables de concilier au mieux les exigences des utilisateurs et les contraintes que rencontrent les technologiciens. Il ne fait aucun doute que, si ces méthodes se développent, ce sera sur des bases statistiques. Nous serions heureux si ce travail pouvait fournir quelques éléments sur la méthode à suivre dans une telle entreprise.

BIBLIOGRAPHIE

1 - Processing data in bits and pieces. F.P.BROOKS
Procedings of the international conference on information processing. UNESCO - PARIS 1959

2 - B 5500 reference manual. BURROUGHS.

3 - Sabrac - A new génération sérial computer.M.LEHMAN, R.ESHED, Z.NETTER. IEEE Transactions on electronic computers. Decembre 1963.

4 - System design of a small, fast digital computer. H.SHORR, N.E.WISEMAN. IEEE Transactions on electronic computers. Decembre 1963.

5 et 6 - Papers on system 360 model 91.
I.B.M. Journal of research and development. Janvier 1967
Some remarks on system development. M.J.FLYNN. P.R.LOW.
Storage system. L.J.BOLAND, G.D.GRANITO, A.U.MARCOTTE, R.E.GOLDSCHMIDT, D.M.POWERS.

7 - Analysis of a basic queuing problem in computer system.P.E. BOUDREAU. M.KAC. I.B.M. Journal of research and development. Avril 1961

8 - An improved cell memory. R.S.GAINES, C.Y.LEE.
IEEE Transactions on electronic computers. Fevrier 1965.

9 - Bulk processing in distributed logic memory. B.A.CRANE, J.A.GITHENS.
IEEE Transactions on electronic computers. Avril 1965.

10 - A cantent adressable distributed logic memory with application to information retrivial. C.Y.LEE, A. PAUL.
Proceedings of the IRE. Janvier 1963.

11 - Special report on electronic memories.
Electronics. Janvier et Fevrier 1968.

12 - An introduction to probability theory and its applications. Tome 1. W.FELLER. Publié chez John WILLEY.

BIBLIOTHEQUE UNIVERSITAIRE
Section de SCIENCES
LILLE