Mesures de performances d'un système de multiprogrammation dans un environnement paginé

Roger Cachat

N° d'ordre

THESE

présentée au

CENTRE UNIVERSITAIRE

D'EDUCATION ET DE FORMATION DES ADULTES DE GRENOBLE

pour obtenir le titre

d' INGENIEUR du CONSERVATOIRE NATIONAL des ARTS et METIERS

par

ROGER CACHAT

—o—

MESURES DE PERFORMANCES D'UN SYSTEME DE MULTIPROGRAMMATION DANS UN ENVIRONNEMENT PAGINE

—o—

Thèse soutenue le 6 juin 1973 devant la commission d'examen

Président	Monsieur	L. Bolliet
Président adjoint	Monsieur	P. Namian
Examinateurs	Madame	E. Thibault
	Messieurs	M. Bellot
		J. Du Masle

UNIVERSITE SCIENTIFIQUE
ET MEDICALE DE GRENOBLE

LISTE DES PROFESSEURS

Président : Monsieur Michel SOUTIF
Vice-Président : Monsieur Gabriel CAU

PROFESSEURS TITULAIRES

MM.	ANGLES D'AURIAC Paul	Mécanique des fluides
	ARNAUD Georges	Clinique des maladies infectieuses
	ARNAUD Paul	Chimie
	AUBERT Guy	Physique
	AYANT Yves	Physique approfondie
Mme	BARBIER Marie-Jeanne	Electrochimie
MM.	BARBIER Jean-Claude	Physique expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOUD Fernand	Biosynthèse de la cellulose
	BARRA Jean-René	Statistiques
	BARRIE Joseph	Clinique chirurgicale
	BENOIT Jean	Radioélectricité
	BERNARD Alain	Mathématiques Pures
	BESSON Jean	Electrochimie
	BEZES Henri	Chirurgie générale
	BLAMBERT Maurice	Mathématiques Pures
	BOLLIET Louis	Informatique (IUT B)
	BONNET Georges	Electrotechnique
	BONNET Jean-Louis	Clinique ophtalmologique
	BONNET-EYMARD Joseph	Pathologie médicale
	BONNIER Etienne	Electrochimie Electrométallurgie
	BOUCHERLE André	Chimie et Toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BOUSSARD Jean-Claude	Mathématiques Appliquées
	BRAVARD Yves	Géographie
	BRISSONNEAU Pierre	Physique du solide
	BUYLE-BODIN Maurice	Electronique
	CABANAC Jean	Pathologie chirurgicale
	CABANEL Jean	Clinique rhumatologique et hydrologie
	CALAS François	Anatomie
	CARRAZ Gilbert	Biologie animale et pharmacodynamie
	CAU Gabriel	Médecine légale et Toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques Pures
	CHARACHON Robert	Oto-Rhino-Laryngologie
	CHATEAU Robert	Thérapeutique
	CHENE Marcel	Chimie papetière
	COEUR André	Pharmacie chimique
	CONTAMIN Robert	Clinique gynécologique
	COUDERC Pierre	Anatomie Pathologique
	CRAYA Antoine	Mécanique

Mme	DEBELMAS Anne-Marie	Matière médicale
MM.	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DESRE Pierre	Métallurgie
	DESSAUX Georges	Physiologie animale
	DODU Jacques	Mécanique appliquée
	DOLIQUE Jean-Michel	Physique des plasmas
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	DUGOIS Pierre	Clinique de Dermatologie et Syphiligraphie
	FAU René	Clinique neuro-psychiatrique
	FELICI Noël	Electrostatique
	GAGNAIRE Didier	Chimie physique
	GALLISSOT François	Mathématiques Pures
	GALVANI Octave	Mathématiques Pures
	GASTINEL Noël	Analyse numérique
	GEINDRE Michel	Electroradiologie
	GERBER Robert	Mathématiques Pures
	GIRAUD Pierre	Géologie
	KLEIN Joseph	Mathématiques Pures
Mme	KOFLER Lucie	Botanique et Physilogie végétale
MM.	KOSZUL Jean-Louis	Mathématiques Pures
	KRAVTCHENKO Julien	Mécanique
	KUNTZMANN Jean	Mathématiques appliquées
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
	LAJZEROWICZ Joseph	Physique
	LATREILLE René	Chirurgie générale
	LATURAZE Jean	Biochimie pharmaceutique
	LAURENT Pierre-Jean	Mathématiques appliquées
	LEDRU Jean	Clinique médicale B
	LLIBOUTRY Louis	Géophysique
	LOUP Jean	Géographie
Mle	LUTZ Elisabeth	Mathématiques Pures
MM.	MALGRANGE Bernard	Mathématiques Pures
	MALINAS Yves	Clinique obstétricale
	MARTIN-NOEL Pierre	Seméiologie médicale
	MASSEPORT Jean	Géographie
	MAZARE, Yves	Clinique médicale A
	MICHEL Robert	Minéralogie et Pétrographie
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie nucléaire
	NEEL Louis	Physique du solide
	OZENDA Paul	Botanique
	PAUTHENET René	Electrotechnique
	PAYAN Jean-Jacques	Mathématiques Pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET René	Servomécanismes
	PILLET Emile	Physique industrielle
	RASSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REULOS René	Physique industrielle
	RINALDI Renaud	Physique
	ROGET Jean	Clinique de pédiatrie et de puériculture
	SANTON Lucien	Mécanique
	SEIGNEURIN Raymond	Microbiologie et Hygiène
	SENGEL Philippe	Zoologie
	SILBERT Robert	Mécanique des fluides
	SOUTIF Michel	Physique générale

MM.	TANCHE Maurice	Physiologie
	TRAYNARD Philippe	Chimie générale
	VAILLAND François	Zoologie
	VALENTIN Jacques	Physique nucléaire
	VAUQUOIS Bernard	Calcul électronique
Mme	VERAIN Alice	Pharmacie galénique
M.	VERAIN André	Physique
Mme	VEYRET Germaine	Géographie
MM.	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale
	YOCCOZ Jean	Physique nucléaire théorique

PROFESSEURS ASSOCIES

MM.	BULLEMER Bernhard	Physique
	HANO JUN-ICHI	Mathématiques Pures
	STEPHENS Michaël	Mathématiques appliquées

PROFESSEURS SANS CHAIRE

MM.	BEAUDOING André	Pédiatrie
Mme	BERTRANDIAS Françoise	Mathématiques Pures
MM.	BERTRANDIAS Jean-Paul	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BONNETAIN Lucien	Chimie minérale
Mme	BONNIER Jane	Chimie générale
MM.	CARLIER Georges	Biologie végétale
	COHEN Joseph	Electrotechnique
	COUMES André	Radioélectricité
	DEPASSEL Roger	Mécanique des fluides
	DEPORTES Charles	Chimie minérale
	GAUTHIER Yves	Sciences biologiques
	GAVEND Michel	Pharmacologie
	GERMAIN Jean-Pierre	Mécanique
	GIDON Paul	Géologie et Minéralogie
	GLENAT René	Chimie organique
	HACQUES Gérard	Calcul numérique
	JANIN Bernard	Géographie
Mme	KAHANE Josette	Physique
MM.	MULLER Jean-Michel	Thérapeutique
	PERRIAUX Jean-Jacques	Géologie et Minéralogie
	POULOUJADOFF Michel	Electrotechnique
	REBECQ Jacques	Biologie (CUS)
	REVOL Michel	Urologie
	REYMOND Jean-Charles	Chirurgie générale
	ROBERT André	Chimie papetière
	DE ROUGEMONT Jacques	Neurochirurgie
	SARRAZIN Roger	Anatomie et chirurgie
	SARROT-REYNAULD Jean	Géologie
	SIBILLE Robert	Construction mécanique
	SIROT Louis	Chirurgie générale
Mme	SOUTIF Jeanne	Physique générale

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

Mle	AGNIUS-DELORD Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBLARD Pierre	Dermatologie
	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Yves	Chimie
	BEGUIN Claude	Chimie organique
	BELORIZKY Elie	Physique
	BENZAKEN Claude	Mathématiques appliquées
	BILLET Jean	Géographie
	BLIMAN Samuel	Electronique (EIE)
	BLOCH Daniel	Electrotechnique
Mme	BOUCHE Liane	Mathématiques (CUS)
MM.	BOUCHET Yves	Anatomie
	BOUVARD Maurice	Mécanique des fluides
	BRODEAU François	Mathématiques (IUT B)
	BRUGEL Lucien	Energétique
	BUISSON Roger	Physique
	BUTEL Jean	Orthopédie
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHIAVERINA Jean	Biologie appliquée (EFP)
	CHIBON Pierre	Biologie animale
	COHEN-ADDAD Jean-Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie médicale
	CONTE René	Physique
	COULOMB Max	Radiologie
	CROUZET Guy	Radiologie
	DURAND Francis	Métallurgie
	DUSSAUD René	Mathématiques (CUS)
Mme	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	GENSAC Pierre	Botanique
	GIDON Maurice	Géologie
	GRIFFITHS Michaël	Mathématiques appliquées
	GROULADE Joseph	Biochimie médicale
	HOLLARD Daniel	Hématólogie
	HUGONOT Robert	Hygiène et Médecine préventive
	IDELMAN Simon	Physiologie animale
	IVANES Marcel	Electricité
	JALBERT Pierre	Histologie
	JOLY Jean-René	Mathématiques Pures
	JOUBERT Jean-Claude	Physique du solide
	JULLIEN Pierre	Mathématiques Pures
	KAHANE André	Physique générale
	KUHN Gérard	Physique
	LACOUME Jean-Louis	Physique
Mme	LAJZEROWICZ Jeannine	Physique
MM.	LANCIA Roland	Physique atomique
	LE JUNTER Noël	Electronique
	LEROY Philippe	Mathématiques
	LOISEAUX Jean-Marie	Physique nucléaire
	LONGEQUEUE Jean-Pierre	Physique nucléaire
	LUU DUC Cuong	Chimie organique
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et Médecine préventive
	MARECHAL Jean	Mécanique
	MARTIN-BOUYER Michel	Chimie (CUS)

MM.	MAYNARD Roger	Physique du solide
	MICHOULIER Jean	Physique (IUT A)
	MICOUD Max	Maladies infectieuses
	MOREAU René	Hydraulique (INP)
	NEGRE Robert	Mécanique
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (IUT B)
	PEFFEN René	Métallurgie
	PELMONT Jean	Physiologie animale
	PERRET Jean	Neurologie
	PERRIN Louis	Pathologie expérimentale
	PFISTER Jean-Claude	Physique du solide
	PHELIP Xavier	Rhumatologie
Mlle	RIERY Yvette	Biologie animale
MM.	RACHAIL Michel	Médecine interne
	RACINET Claude	Gynécologie et obstétrique
	RENAUD Maurice	Chimie
	RICHARD Lucien	Botanique
Mme	RINAUDO Marquerite	Chimie macromoléculaire
MM.	ROMIER Guy	Mathématiques (IUT B)
	SHOM Jean-Claude	Chimie générale
	STIEGLITZ Paul	Anesthésiologie
	STOEBNER Pierre	Anatomie pathologique
	VAN CUTSEM Bernard	Mathématiques appliquées
	VEILLON Gérard	Mathématiques appliquées (INP)
	VIALON Pierre	Géologie
	VOOG Robert	Médecine interne
	VROUSSOS Constantin	Radiologie
	ZADWORNY François	Electronique

MAITRES DE CONFERENCES ASSOCIES

MM.	BOUDOURIS Georges	Radioélectricité
	CHEEKE John	Thermodynamique
	GOLDSCHMIDT Hubert	Mathématiques
	SIDNEY STUARD	Mathématiques Pures
	YACOUD Mahmoud	Médecine légale

CHARGES DE FONCTIONS DE MAITRES DE CONFERENCES

Mme	BERIEL Hélène	Physilogie
Mme	RENAUDET Jacqueline	Microbiologie

Fait le 30 mai 1972.

Je tiens à remercier,

Monsieur le Professeur L.BOLLIET qui a bien voulu me faire l'honneur de présider le Jury de cette Thèse et qui a toujours montré la plus grande bienveillance pour mon travail,

Monsieur le Professeur P.NAMIAN qui a accepté d'être Président Adjoint du Jury,

Monsieur J.du MASLE qui a dirigé mon travail et qui m'a permis la réalisation de cette Thèse,

Madame E.THIBAULT, Monsieur M.BELLOT pour leurs suggestions, leurs conseils et leurs précieuses critiques apportées au manuscrit,

Tous mes collègues du Laboratoire qui m'ont aidé dans mon travail.

Je voudrais aussi remercier,

Le service de dactylographie et le service de reproduction de l'IMAG pour la réalisation soignée et rapide de ce document.

C H A P I T R E I

INTRODUCTION

DEFINITIONS PRELIMINAIRES

JOB : Un Job est l'unité de travail pour l'utilisateur; il est constitué d'une suite d'opérations à faire exécuter à une machine.

STEP : Un JOB ou travail est généralement constitué de plusieurs étapes appelées STEPS.

BATCH : Un ensemble de JOBS, ou travaux, présentés comme un flot continu, pour être exécutés par une machine constitue un BATCH de travaux.

ENREGISTREMENT LOGIQUE : L'unité logique de lecture ou d'écriture d'une information est l'enregistrement logique.

BLOC : Plusieurs enregistrements logiques peuvent être regroupés pour former un bloc d'informations.

DATA SET : Un data set est un ensemble d'informations qui a un nom et qui est constitué d'une suite liée d'enregistrements.

BUFFER : Un buffer est une zone tampon de mémoire principale utilisée par certaines fonctions du système pour recevoir ou préparer des enregistrements en cours de transfert.

TACHE : Une tâche est l'unité de travail d'un système de multiprogrammation.

READER : Le reader est une tâche système (dans un système de multiprogrammation) destinée à effectuer la lecture et l'enregistrement des JOBS.

WRITER : Le writer est une tâche système (dans un système de multiprogrammation) destinée à effectuer l'impression des résultats, sur l'imprimante.

INITIATEUR : L'initiateur est une tâche système (dans un système de multiprogrammation) destinée à démarrer l'exécution d'une tâche.

LINK PACK AREA (LPA) : La zone de mémoire principale appelée LPA contient un ensemble de modules chargeables réentrants du système OS/MVT.

SYSTEM QUEUE SPACE (SQS) : La zone de mémoire principale appelée SQS constitue l'espace de travail du système OS/MVT.

L'ordinateur du Laboratoire de Calcul de l'Université de Grenoble assure deux types d'exploitation :

- une exploitation dite "conversationnelle" pendant les heures ouvrables de la journée :

  8 heures à 12 heures
  15 heures à 19 heures

- une exploitation dite "batch" le reste du temps.

1 - Exploitation en mode conversationnel

Elle est réalisée par un système de "partage de temps", générateur de machines virtuelles qui utilise un mécanisme de pagination pour gérer la mémoire virtuelle.

Chaque machine virtuelle est une machine IBM 360 standard dans laquelle peut fonctionner un système d'exploitation quelconque sous le contrôle d'un utilisateur qui est l'opérateur de sa machine virtuelle.

De nombreux utilisateurs peuvent ainsi accéder à la machine réelle.

Ce mode d'exploitation est très apprécié pour la mise au point des programmes.

2 - Exploitation en mode continu (batch)

Elle est réalisée par un système de multiprogrammation qui

permet l'exécution des travaux en flot continu sans intervention de l'utilisateur.

Ce mode d'exploitation est particulièrement recommandé pour les travaux longs et conséquents. L'exploitation en mode conversationnel est actuellement interrompue entre 12 heures et 15 heures pour être remplacée par l'exploitation en mode batch dans le but de satisfaire le maximum d'utilisateurs. Aussi,pendant cette courte période de 3 heures, deux changements de système sont nécessaires .

Un changement de système est une opération coûteuse qui suspend le fonctionnement de la machine pendant 15 minutes environ; en effet, il faut :

- démonter et remonter les disques amovibles pour passer d'un système à l'autre
- formater les volumes fixes (tambours)
- initialiser le nouveau système.

Problème posé

Est-il possible d'assurer aux utilisateurs un service identique ou peu différent en supprimant ces deux changements de système donc en gagnant 30 minutes d'exploitation ?

Solution proposée

Garder l'exploitation en mode conversationnel de 8 heures à 19 heures, et activer entre 12 heures et 15 heures, une machine virtuelle réservée au système de multiprogrammation, qui

assurera l'exploitation en mode batch (en prenant soin de réduire pendant cette période le nombre d'utilisateurs du mode conversationnel).

Intérêt de cette solution

- Les 30 minutes d'arrêt de la machine sont supprimées
- Deux séries de manipulation de piles de disques sont supprimées
- Certains utilisateurs prioritaires dans l'exploitation en mode conversationnel pourraient continuer leurs travaux
- L'enregistrement (lecture) des JOBS pour le système de multiprogrammation pourra être effectué avant l'activation de la machine virtuelle OS/MVT par le système de"partage de temps" (voir utilisation du lecteur virtuel)
- L'impression des résultats, des JOBS exécutés par le système de multiprogrammation, pourra être effectuée après la désactivation de OS/MVT par le système de "partage de temps" (voir utilisation de l'imprimante virtuelle).

Inconvénient de cette solution

Le nombre des travaux exécutés par le système de multiprogrammation sera réduit puisque l'on superpose deux systèmes non conçus l'un pour l'autre, et pouvant apporter des conflits au niveau de l'utilisation des ressources de la machine.

Problème à résoudre

Ajuster certains paramètres du système de multiprogrammation et du système de "partage de temps" en vue de déterminer un point de fonctionnement optimal du système de multiprogrammation dans une machine virtuelle gérée par le système de partage de temps.

Déterminer dans ces conditions les performances du système de multiprogrammation afin de décider si la solution proposée est acceptable.

Organisation du travail

Le nombre de paramètres qui conditionnent le bon fonctionnement d'un système est grand. Nous nous limiterons à l'étude de l'influence de certains paramètres importants.
Pour exprimer les performances du système de multiprogrammation nous ne pourrons que comparer deux modes de fonctionnement de ce système, et établir une différence de performance.
Nous procèderons en deux étapes :

1- Etude du système de multiprogrammation sur la machine réelle

Nous mesurerons le temps d'exécution réel d'un ensemble de JOBS bien définis appelé BATCHTEST qui caractérise l'ensemble des travaux généralement soumis au système de multiprogrammation. En faisant varier certains paramètres du système nous rechercherons le temps d'exécution minimal de l'ensemble des JOBS sur la machine réelle.

2- Etude du système de multiprogrammation fonctionnant dans une machine virtuelle gérée par le système de "partage de temps".

a- Sans aucune autre machine virtuelle active

En faisant varier certains paramètres du système de multiprogrammation nous obtiendrons là aussi, un temps d'exécution minimal pour le même ensemble de JOBS. Nous ferons alors varier certains paramètres du système de "partage de temps" pour améliorer le temps d'exécution de l'ensemble des JOBS. Une comparaison des temps obtenus permettra de matérialiser

les performances du système de multiprogrammation dans un environnement pagine.

b- Avec d'autres machines virtuelles actives en parallèle.

Les performances du système de multiprogrammation vont être modifiées lorsque la charge du système de "partage de temps" augmentera. Cette charge est habituellement constituée d'un nombre important de machines virtuelles standard, dans lesquelles fonctionne un système spécialement conçu pour un environnement de pagination.

La gestion d'une machine virtuelle dans laquelle fonctionne un système de multiprogrammation est une lourde charge peu compatible avec la nature même d'un système de "partage de temps"; cependant lorsque cette machine virtuelle est seule active, la charge du système de partage de temps est relativement faible.

Nous essayerons d'atteindre la charge habituelle du système de partage de temps en augmentant graduellement le nombre de machines virtuelles standard actives en parallèle.

Nous noterons dans ces conditions les nouvelles performances du système de multiprogrammation dans un environnement paginé.

L'étude et la comparaison des différentes mesures de performances permettra de décider si l'utilisation continue du système de "partage de temps" pendant la journée peut être raisonnablement envisagée et dans quelles conditions.

C H A P I T R E II

DESCRIPTION DU MATERIEL

I - DESCRIPTION DE LA MACHINE : CONFIGURATION

L'étude suivante a été réalisée sur l'ordinateur IBM 360 modèle 67 du Laboratoire de Calcul de l'Université de Grenoble dont la configuration est la suivante :

- 1024 K octets de mémoire principale	
- 2 lecteurs/perforateurs de cartes	canal multiple
- 2 imprimantes	canal multiple
- 70 terminaux légers	canal multiple
- 3 terminaux lourds	canal multiple
- 4 dérouleurs de bandes magnétiques	canal multiple
- 2 tambours magnétiques 2301	canal 1
- 12 piles de disques magnétiques 2314	canal 2
- 8 piles de disques magnétiques 2314	canal 3
- 1 écran graphique 2250	

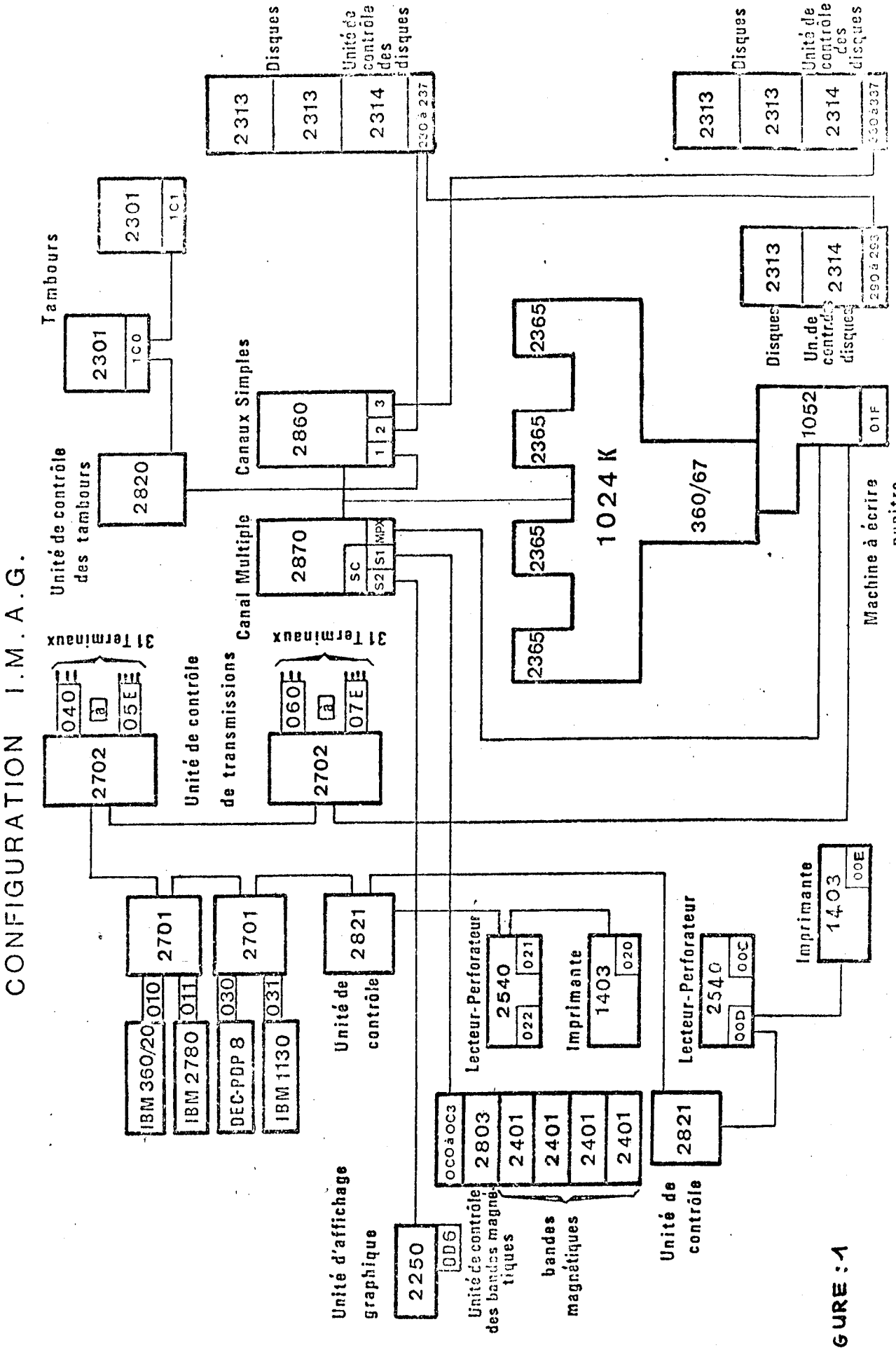

CONFIGURATION I.M.A.G.
Tambours
2301
1C0
1C1
2820
Unité de contrôle des tambours
Canaux Simples
2860
1 2 3
Canal Multiple
2870
SC
S2 S1 MPX
Unité de contrôle de transmissions
31 Terminaux
040
05E
2702
060
07E
2313
Disques
Unité de contrôle des disques
2314
230 à 237
2365
1024 K
360/67
1052
01F
Machine à écrire pupitre
Imprimante
1403
00E
2701
010
011
030
031
IBM 360/20
IBM 2780
DEC-PDP 8
IBM 1130
2821
Unité de contrôle
Lecteur-Perforateur
2540
021
022
Imprimante
1403
020
00C
00D
OC0 à OC3
2803
2401
Unité de contrôle des bandes magnétiques
bandes magnétiques
2250
0D6
Unité d'affichage graphique

FIGURE : 1

II - DESCRIPTION DES SYSTEMES UTILISES

Notre étude porte sur 2 systèmes :

- Le système de multiprogrammation OS/MVT version 20.1
- Le système de partage de temps CP/67 version 3.0 générateur de machines virtuelles qui utilise un mécanisme de pagination avec segmentation pour gérer la mémoire virtuelle.

1- Système de multiprogrammation OS/MVT version 20.1

a- Caractéristiques

Le système OS (Operating System), fonctionnant sur les ordinateurs IBM de la série 360, est à la base un système permettant l'enchaînement automatique de l'exécution des travaux (batch processing).

L'unité de travail pour l'utilisateur du système OS est le JOB.

Un JOB est constitué d'une série de demandes d'exécution de programmes dans un environnement défini.

Un langage de contrôle permet à l'utilisateur :

- de commander l'exécution de ces différents programmes
- de décrire les environnements nécessaires au traitement.

Le système décompose un JOB en unités de travail appelées tâches.

Une tâche comprend généralement l'exécution d'un programme et l'allocation de ressources nécessaires, telles que :

- unité centrale (CPU)
- zone de mémoire principale
- unité d'entrée-sortie
- programme de service du système

Le système OS/MVT (Multiprogramming with a variable number of tasks) est un système OS qui travaille dans un contexte de multiprogrammation.

Ses caractéristiques sont les suivantes :

- Traitement de plusieurs tâches en parallèle :
  Généralement une tâche en exécution n'utilise pas, à un instant donné, toutes les ressources du système (CPU, mémoire principale, etc...)
  L'exécution parallèle de plusieurs tâches (démarrées par plusieurs initiateurs) conduit à une meilleure (voire optimale) utilisation des ressources et, par là même, à une bonne efficacité de l'ensemble du traitement.
  Une tâche en exécution dispose d'une quantité de mémoire principale qui peut être définie par l'utilisateur ou par le système, et qui peut varier d'une tâche à l'autre.

- Sélection judicieuse des JOBS à exécuter dans un ensemble appelé file d'attente d'entrée des travaux :
  Afin d'améliorer les performances du système, la tâche appelée "READER"

construit sur disques une file d'attente des travaux à exécuter à partir d'un lecteur de cartes ou d'un concentrateur de données. De la même façon, la tâche appelée "WRITER" prélève dans les files d'attentes de sortie les résultats de l'exécution des travaux et les achemine vers les imprimantes, perforateurs ou concentrateurs de données concernés. Cette technique rend le système indépendant des opérations d'entrée/sortie sur les unités lentes.

b- Eléments du système

Le système OS/MVT est composé

- d'un programme de contrôle (superviseur)
- d'un ensemble de programmes de traitement (compilateurs, programmes de service...) dont l'exécution est commandée par l'utilisateur, et supervisée par le programme de contrôle.

Programme de contrôle

Une partie de ce programme réside en mémoire principale et l'autre partie se trouve sur volume à accès direct, dit volume système.

Après le chargement initial du système l'organisation de la mémoire principale peut être représentée par la Figure 2.

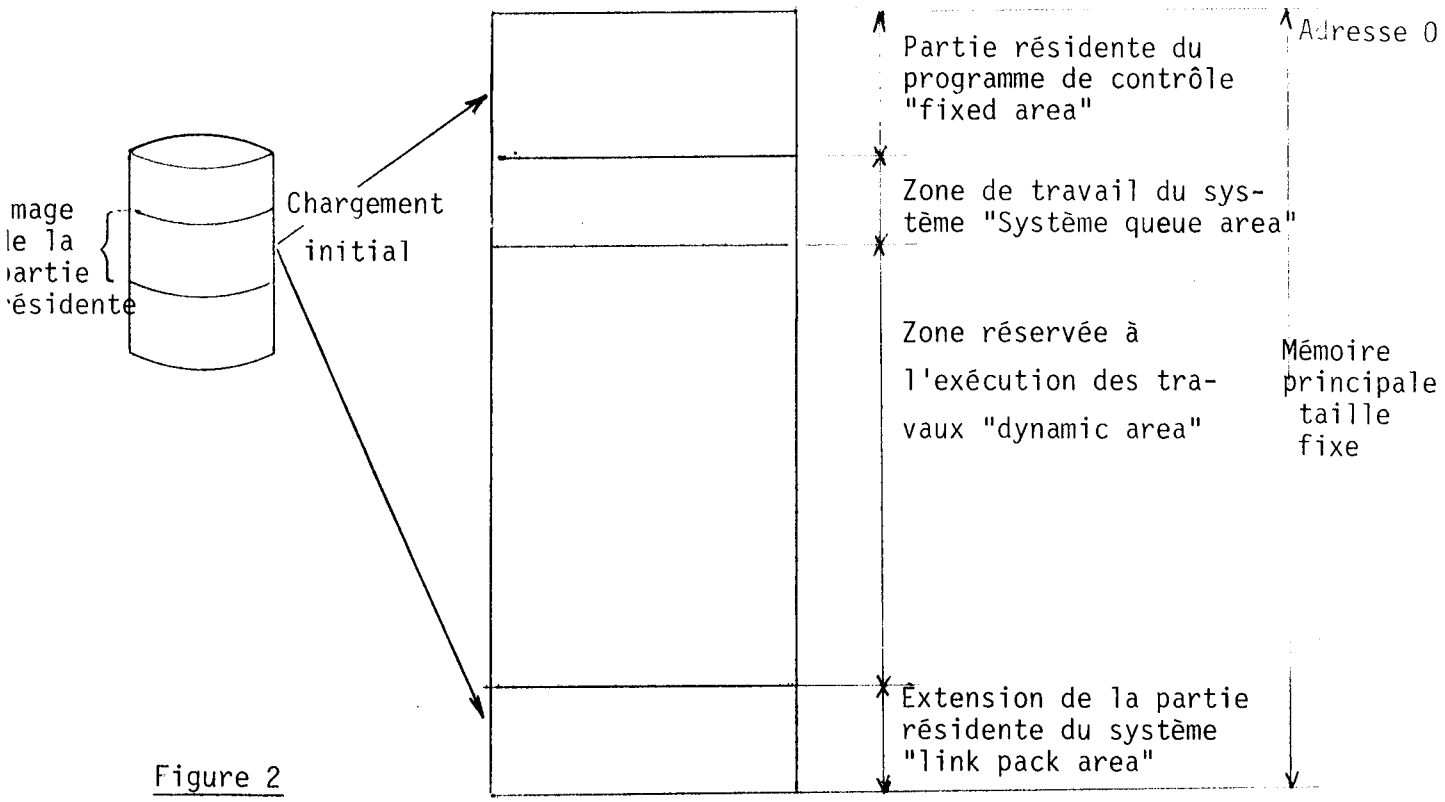


Figure 2

Le programme de contrôle à trois fonctions principales :

- Enchaîner le traitement des JOBS (job management)
- Superviser l'exécution de chaque unité de travail (task management)
- Gérer, organiser et transférer les données (data management)

- LE JOB MANAGEMENT :

* analyse et interprète le langage de contrôle définissant chaque JOB; construit les blocs de contrôle qui décrivent chaque JOB.

* vérifie l'existence des unités d'entrée/sortie demandées et les affecte au JOB intéressé; obtient l'espace nécessaire de mémoire principale; obtient l'espace nécessaire sur les mémoires auxiliaires (volumes à accès direct).

* sélectionne les JOBS à exécuter en fonction de leur type (classe) et de leur priorité.

* transfère les sorties à partir des unités à accès direct (WRITER).

* communique avec l'opérateur par l'intermédiaire d'une console (pupitre).

- LE DATA MANAGEMENT :

* gère la place sur les mémoires auxiliaires
* organise l'information sur les mémoires auxiliaires
* localise l'information
* transfère l'information entre la mémoire principale et les mémoires ou inversement (méthodes d'accès).

- LE TASK MANAGEMENT :

* établit les priorités internes des tâches. Le principe de la priorité est un point important pour la compréhension de la sélection des tâches par le système.
Le système OS/MVT distingue deux priorités :

. la priorité d'entrée des JOBS, spécifiée dans la carte JOB, qui conditionne la sélection d'un JOB dans la file d'attente; cette priorité est exploitée par le JOB MANAGEMENT.

. la priorité d'activation d'une tâche (dispatching priority: DPRTY) qui favorise sa sélection en cours d'exécution.
Un bon choix de la valeur donnée à cette priorité permet d'obtenir une charge équilibrée du système et ainsi d'améliorer les performances.

* crèe les tâches: crèe les blocs de contrôles nécessaires et charge le programme en mémoire principale.

* synchronise les évènements pour permettre le partage des ressources entre plusieurs tâches.

* gère la mémoire principale : la partage entre les tâches actives, en gère l'allocation et le contenu.

* contrôle l'accès de la mémoire principale par l'intermédiaire d'un dispositif de protection mémoire. Une tâche en exécution n'a accès qu'à la zone de mémoire principale qui lui a été attribuée.

* contrôle le temps d'exécution limite des tâches.

* termine les tâches: annule les blocs de contrôle associés à chaque tâche terminée.

La Figure 3 donne un schéma du programme de contrôle d'OS/MVT.

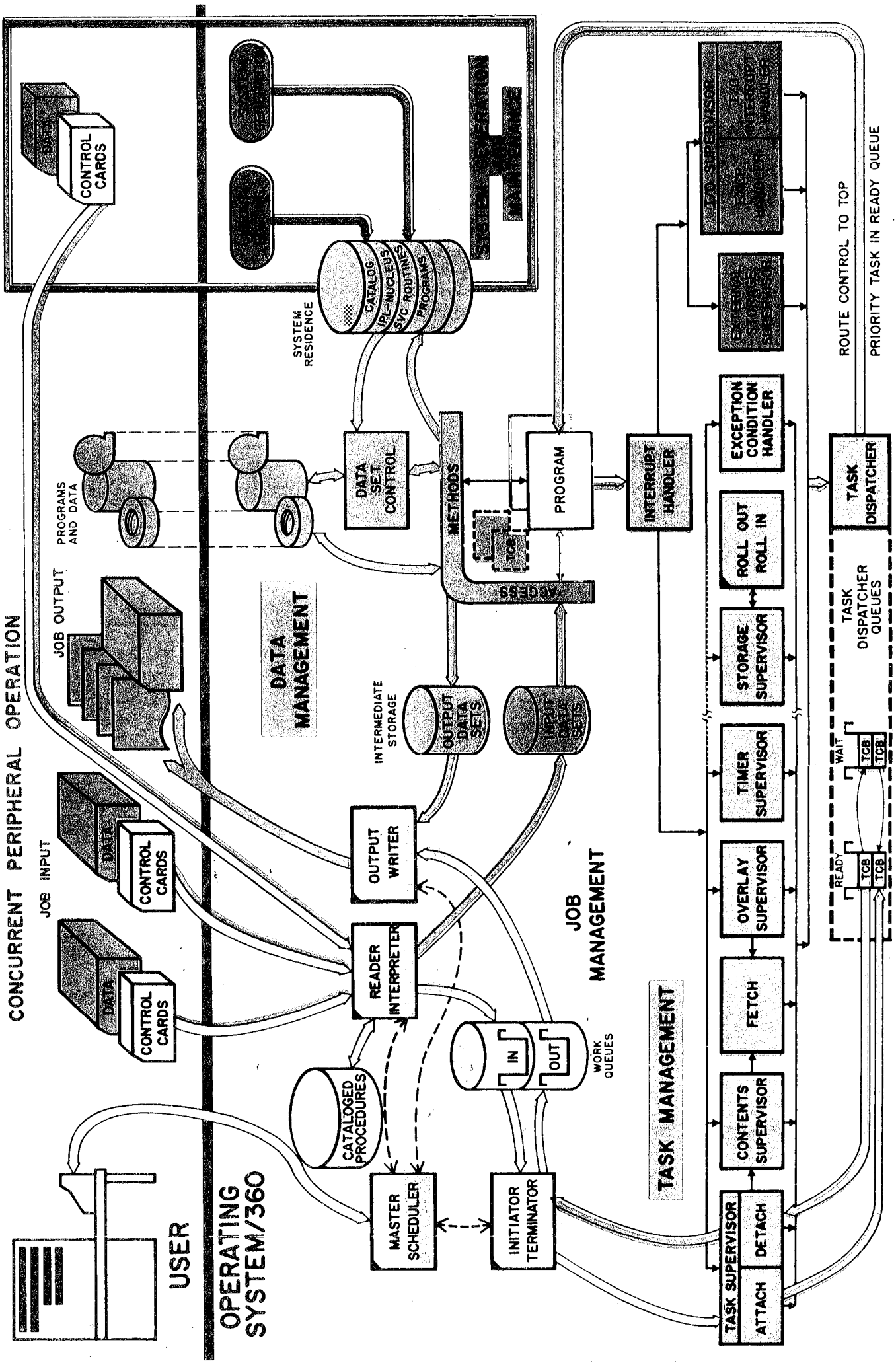
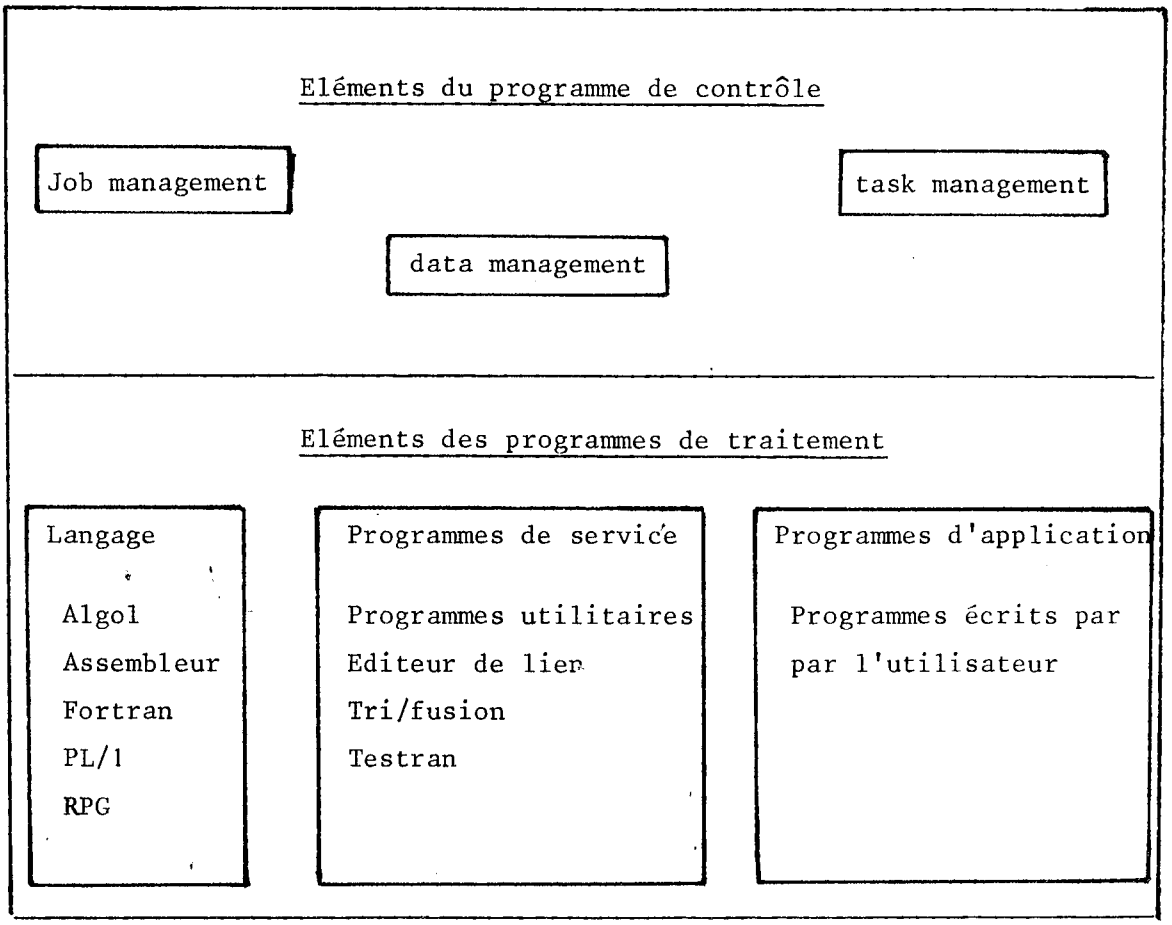

CONCURRENT PERIPHERAL OPERATION
OPERATING SYSTEM/360
USER
JOB INPUT
DATA
CONTROL CARDS
JOB OUTPUT
PROGRAMS AND DATA
SYSTEM RESIDENCE
CATALOG
IPL NUCLEUS
SVC ROUTINES
PROGRAMS
DATA MANAGEMENT
DATA SET CONTROL
METHODS
ACCESS
TCB
PROGRAM
INTERMEDIATE STORAGE
OUTPUT DATA SETS
INPUT DATA SETS
OUTPUT WRITER
READER INTERPRETER
JOB MANAGEMENT
CATALOGED PROCEDURES
IN
OUT
WORK QUEUES
MASTER SCHEDULER
INITIATOR TERMINATOR
TASK MANAGEMENT
TASK SUPERVISOR
ATTACH
DETACH
CONTENTS SUPERVISOR
FETCH
OVERLAY SUPERVISOR
TIMER SUPERVISOR
STORAGE SUPERVISOR
ROLL OUT
ROLL IN
EXCEPTION CONDITION HANDLER
INTERRUPT HANDLER
I/O SUPERVISOR
I/O INTERRUPT HANDLER
EXTERNAL STORAGE SUPERVISOR
TASK DISPATCHER
TASK DISPATCHER QUEUES
READY
WAIT
TCB
ROUTE CONTROL TO TOP
PRIORITY TASK IN READY QUEUE
OPERATING SYSTEM/360 CHART V25-6156

FIGURE: 3

Programmes de traitement

Les programmes de traitement sont des programmes standards organisés en bibliothèques, sur des volumes à accès direct. Ils peuvent être appelés par l'utilisateur pour assurer différents services.

Le tableau ci-dessous résume les éléments de l'operating system.



2- Système de partage de temps CP-67 version 3.0

a- Caractéristiques

CP-67 est un système de "partage de temps" générateur de machines virtuelles, fonctionnant sur les ordinateurs IBM 360 modèle 67.

Une machine virtuelle est la simulation d'un ordinateur 360 standard et de ses unités d'entrée-sortie associées.

CP-67 permet à plusieurs utilisateurs d'accéder simultanément à l'ordinateur, et gère pour chacun d'eux, une machine virtuelle de configuration déterminée.

Le 360 virtuel créé par CP est, pour l'utilisateur et ses programmes, analogue à un 360 réel.

CP alloue les ressources de la machine réelle à chaque machine virtuelle, par tranche de temps.

Les machines virtuelles étant simulées par CP, leur configuration peut varier et diffèrer de la configuration réelle.

Chaque utilisateur contrôle sa machine virtuelle à partir d'un terminal à distance qui représente donc sa "console opérateur".

Comme dans une machine réelle, un "operating system" quelconque (OS/MVT par exemple) peut fonctionner dans une machine virtuelle.

Pour utiliser le système CP, l'utilisateur doit avoir une identification (USERID) et un mot de passe, reconnus par CP, quand il "démarre" sa machine virtuelle.

Dans le système CP, sont associés à chaque USERID :

- une information de facturation
- une classe
- des options
- une table décrivant la machine virtuelle

Chaque machine virtuelle doit disposer au moins de :

- 256 K de mémoire virtuelle
- deux unités de disque
- une console (le terminal)
- un lecteur perforateur
- une imprimante

b- Technique d'adressage

CP assure la gestion de la mémoire virtuelle à l'aide du mécanisme de pagination. La mémoire virtuelle est divisée en blocs de 4096 octets appelés pages. Les pages sont rangées à la demande par le système sur une mémoire auxiliaire à accès direct. Quand une page passe de l'état inactif à l'état actif (ou inversement) elle est transférée de la mémoire secondaire dans la mémoire principale (ou inversement). Quand le mécanisme de pagination fonctionne pour une machine virtuelle, une autre machine virtuelle peut être activée. Ce mécanisme de pagination est transparent pour l'utilisateur; il est mis en oeuvre par un dispositif spécial au 360 modèle 67 qui traduit dynamiquement les adresses virtuelles en adresses réelles.

c- Traitement des opérations d'entrée-sortie

Les opérations d'entrée-sortie d'une machine virtuelle A sont traitées par CP. Celui-ci doit les convertir en opérations d'entrée-sortie sur la machine réelle.

CP intercepte le démarrage d'une opération d'entrée-sortie utilisateur (SIO) et assure les fonctions suivantes :

- translation de l'adresse de l'unité virtuelle en adresse de l'unité réelle.
- translation de l'adresse mémoire virtuelle en adresse mémoire réelle.
- construction des programmes canaux réels (CCW) pour l'utilisateur
- démarrage de l'opération d'entrée-sortie si le canal est libre.
- vérification de la présence des pages nécessaires, en mémoire réelle.

Pendant ce temps, d'autres machines virtuelles peuvent être activées. Quand CP reçoit une interruption indiquant la fin de l'opération d'entrée-sortie, il rend le contrôle à la machine virtuelle A et lui reflète les conditions de fin.

d- Traitement des enregistrements que doit lire ou écrire la machine virtuelle.

Généralement ces enregistrements sont regroupés dans le spooling de CP (sur disque) et identifiés par le nom de la machine virtuelle à laquelle ils doivent être transférés. Ce transfert se fait par le lecteur virtuel, commandé par la machine virtuelle.

L'opérateur peut attacher réellement un lecteur de cartes

à la machine virtuelle. Les cartes lues ne passent alors plus par le spooling de CP. Il en est de même pour l'impression des enregistrements.

e- Fonctions console

Les fonctions console de CP permettent à l'utilisateur de contrôler sa machine virtuelle à partir du terminal (il en est l'opérateur).

- Il commande l'IPL
- Il peut à tout instant arrêter le fonctionnement de sa machine virtuelle. (touche attention).
- Il peut afficher (faire imprimer) le contenu de toute zone de sa mémoire virtuelle, le contenu des registres.
- Il peut en plus communiquer avec les autres utilisateurs.

III - DESCRIPTION DU BATCH TEST

1- Caractéristiques d'un batch test

Utilisé pour optimiser le fonctionnement d'un système, le batch test doit avoir des caractéristiques qui reflètent la physionomie de l'ensemble des JOBS généralement traités par le système.

Ces caractéristiques conditionnent le degré d'amélioration des performances du système.

Les paramètres fondamentaux d'un batch test sont les suivants :

- le temps global d'exécution
- le degré d'utilisation du CPU
- l'activité d'entrée-sortie
- la taille des fichiers traités par le "reader" et le "writer"
- la taille des programmes

La valeur de ces paramètres peut être déterminée à l'aide de programmes de statistiques des systèmes (facturation).

2- Description du batch test utilisé

Notre batch test est constitué de 7 JOBS utilisant des composants variés du système OS/MVT. Le tableau ci-dessous dresse les particularités de chaque JOB :

Nom programme	Classe	Composants du système utilisé	Taille mémoire utilisée en K octets	Temps d'exécution moyen	Temps CPU moyen 1/100 sec.	Nbre d'enregist. disque traités	Nbre d'enregist. SYSIN traités	Nbre d'enregist. SYSOUT traités	Nbre total d'enregist. traités	Nbre moyen d'enregist. traités par sec. de CPU
BENCH11	B	Compilateur Algol	180	13mn 20s	513	1252	334	486	2072	400
		chargeur(E)	100		11 992	1198	0	1640	2838	23
BENCH08	A	Compilateur Cobol	100	7mn 50s	240	78	76	81	235	100
		Editeur de liens	96		53	64	0	18	82	154
		(E)	100		559	802	0	0	802	144
BENCH03	A	Compilateur Cobol	100	12mn 13s	1040	284	660	700	1644	160
		Editeur de liens	96		92	292	0	18	310	340
		(E)	70		178	42	0	4	46	26
BENCH01	A	IEBGENER	70	11mn 46s	72	301	0	3	304	425
		COBO	120		1430	0	1	0	1	
BENCH12	B	Assembleur F	70	6mn 16s	977	245	104	214	563	58
		Editeur de liens	96		41	19	0	17	36	88
		(E)	70			14	0	4	4	28
BENCH16	C	Compilateur Fortran G	100	8mn 51s	225	105	56	191	352	158
		Editeur de liens	96		118	252	0	81	333	280
		(E)	70		118	0	3	249	252	215
BENCH10	C	FORT	70	7mn 38s	31277	0	36	712	748	3

(E) = Exécution du module chargeable

Dans cette étude, les 7 JOBS pourront être exécutés en parallèle dans un temps moyen de 12 minutes, pour une utilisation de 8 minutes de CPU.

3- Utilisation de la priorité d'activation pour le batch test

La priorité d'activation d'une tâche (dispatching priorité) peut être définie par l'utilisateur, au niveau de la carte de contrôle :

```
//   EXEC    PGM=XXX,DPRTY=(n,p)
          0 ≤ n ≤  15
          0 ≤ p ≤  15
```

Alors la priorité d'activation aura pour valeur :

$$dp = 16n + p \qquad \text{avec} \qquad 0 \leq dp \leq 255$$

Les composants du système (compilateurs, assembleur, éditeur de liens, programmes utilitaires, etc...) sont appelés par le batch par l'intermédiaire de procédures cataloguées.

Une priorité d'activation a donc été associée à chaque appel d'un composant du système, et ceci dans chaque procédure cataloguée. Pour calculer la priorité d'activation il faut connaître pour chaque composant :

- le temps CPU moyen qu'il utilise
- le nombre moyen d'opérations d'entrée-sortie réalisées pendant ce temps.

Ces deux paramètres permettent de définir la classe à laquelle appartient le JOB et par là-même de lui associer sa priorité d'activation.

4- Conditions d'exécution du batch par le système OS/MVT

- Démarrage du système OS/MVT :
  Les paramètres liés à chaque expérience seront transmis au système OS/MVT pendant la procédure d'initialisation du système. Pour chaque type d'expérience un chargement du système sera donc nécessaire.

- Démarrage du batch test :
  Les caractéristiques du batch test resteront constantes pour l'ensemble des expériences. Les initiateurs seront démarrés (actifs) avant l'activation de la tâche READER; l'exécution des JOBS débutera immédiatement après leur lecture.

Le READER restera actif jusqu'à la fin de la lecture des JOBS.

La tâche WRITER restera active pendant toute la durée d'exécution du batch.

C H A P I T R E III

ETUDE DU SYSTEME DE MULTIPROGRAMMATION OS/MVT SUR LA MACHINE REELLE

Le système OS/MVT est un système modulaire et paramétré (soit au moment de la génération du système, soit à l'initialisation). Ces paramètres permettent :

1/ D'adapter le système à la configuration de la machine :
- taille de mémoire principale
- unités d'entrée/sortie disponibles
- etc...

Ces paramètres sont fixés au moment de la génération du système.

2/ D'adapter le système aux besoins de l'exploitation :
- sélection des fonctions
- sélection des compilateurs
- etc...

Ces paramètres sont fixés au moment de la génération du système.

3/ D'adapter le système au type de l'exploitation: ces paramètres sont définis au moment de la génération du système (valeur par défaut). Ils peuvent être modifiés au démarrage du système pour la durée de la session :
- liste des modules résidents
- taille de la zone de travail du système
- etc...

4/ D'organiser les composants du système sur les unités à accès direct.

5/ De sélectionner des tâches paramétrées (READER WRITER INITIATEUR) au cours de la session.

D'une manière générale, le système OS/MVT est ralenti, au niveau de la quantité globale de travail fourni par les entrées/sorties.

Pour en diminuer le nombre, plusieurs possibilités sont offertes :

- augmenter le nombre de fonctions très utilisées en mémoire principale
- augmenter la taille des blocs traités par le READER et le WRITER avec augmentation en parallèle de la quantité de mémoire utilisée par certains composants
- augmenter la taille de mémoire principale utilisée par certains composants pour réduire l'utilisation des fichiers de travail.

Dans chacun de ces cas, la quantité de mémoire principale disponible pour l'exécution des travaux (dynamic area) diminue de même que le nombre de JOBS actifs en parallèle. Un compromis est donc nécessaire.

Nous étudierons l'influence sur le temps d'exécution du batch test, des paramètres suivants :

- modules résidents en mémoire principale
- type de l'unité à accès direct supportant certains composants du système
- paramètres du READER
- nombre d'INITIATEURS

Pour chaque expérience, nous noterons trois temps :

T_e = Temps global d'exécution du batch test
C'est le temps écoulé entre le premier message "JOB STARTED" et le dernier message "JOB ENDED".

T_s = Temps de la session
C'est le temps écoulé entre le premier message "JOB STARTED" et l'arrêt de l'activité des périphériques.

T_r = Temps de présence du reader
C'est le temps écoulé entre les messages : "RDR STARTED" et "RDR ENDED".

L'organisation standard des volumes à accès direct montés en permanence est la suivante :

Type	Nom	Adresse	Type de volume (1)	Contenu (2)
Tambour	20MVT9	1C0	S	BS
Tambour	20MVT8	1C1	S	BS
Disque	20MVT1	230	S	BS
Disque	20MVT2	231	S	BS + UT
Disque	20MVT3	330	S	BS + UT
Disque	20MVT4	331	S	BS + UT
Disque	IMAG79	232	P	BU + UT
Disque	IMAG83	233	P	BU + UT
Disque	ILL003	234	P	BU + UT
Disque	ISN003	235	P	BU + UT
Disque	CNRS01	332	P	BU + UT
Disque	ISN004	333	P	BU + UT
Disque	IMAG84	334	P	BU + UT

Disque	IMAG85	290	P	BU + UT
Disque	IMAG80	292	P	BU + UT
Disque	IMAG81	293	P	BU + UT

(1) { S = volume système
P = volume résident permanent

(2) { BS = bibliothèques système
BU = bibliothèque utilisateurs
UT = data sets de travail du système

I - INFLUENCE DES MODULES RESIDENT EN MEMOIRE PRINCIPALE (LINK PACK AREA)

Comme dans tous les systèmes de programmation, OS/MVT a un noyau résident en mémoire principale. Ce noyau est complété par des modules de gestion des JOBS (JOB MANAGEMENT) implantés dans une zone de mémoire nommée "LINK PACK AREA".

Cette zone peut être étendue dynamiquement par une option au démarrage du système. Elle contient alors :

- des modules de traitement d'appel au superviseur (instruction SVC) provenant de SYS1.SVCLIB (RSVC)

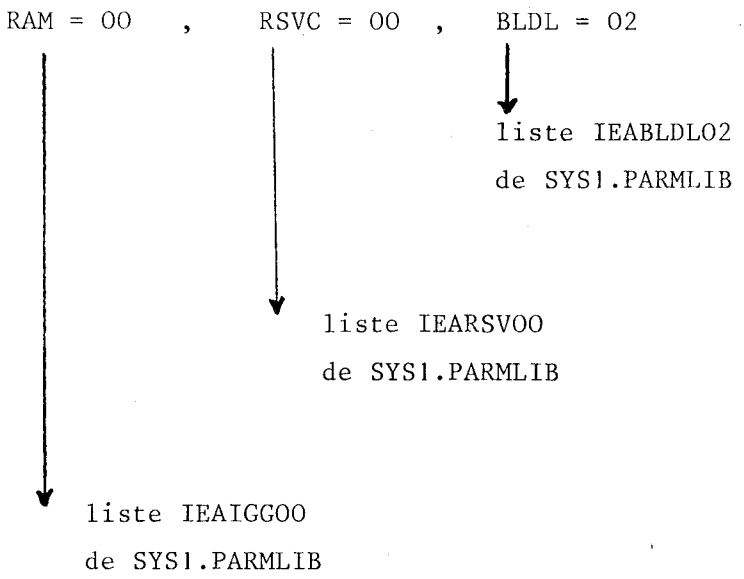
- des modules de rattrapage d'erreurs d'entrée sortie provenant de SYS1.SVCLIB.

- d'autres modules réentrants provenant de SYS1.LINKLIB (reader, initiateur) ou de SYS1.SVCLIB (méthodes d'accès) (RAM).

- une table contenant les adresses absolues sur supports à accès direct de modules appartenant à SYS1.LINKLIB. Ces modules ont une fréquence d'emploi assez importante mais leur implantation en mémoire principale n'est pas justifiée (BLDL).

Pour réaliser cette opération, il faut définir des listes standard de noms de modules de chaque catégorie. Chaque liste à un nom symbolique définissant sa catégorie et un numéro d'ordre. L'ensemble de de ces listes est contenu dans une bibliothèque appelée SYS1.PARMLIB.

Au démarrage du système, on transmettra les noms et les numéros d'ordre de chaque liste désirée. Par exemple :



Dans le cas d'un démarrage "standard" le système a pour convention de consulter les listes dont le nom se termine par 00.

1- Détermination de ces modules : expérience AMAP

Une procédure d'analyse de système "AMAP" : (advanced multiprogramming Analysis Procedure) permet d'obtenir un état très précis de la charge du hardware d'une installation et du système d'exploitation.

Le programme AMAP est un programme de mesure actif pendant la durée d'un batch test. Il permet d'effectuer un certain nombre de relevés destinés à être analysés dans une étape ultérieure.

REMARQUE | Pour les besoins de l'exploitation, cette expérience a été réalisée en ASP/MVT où ASP est un sous-système de gestion des entrées-sorties du système OS/MVT.
On pourra se reporter à la brochure référence IV.

a- Description du BATCH TEST de l'expérience AMAP

Le batch test est composé d'un ensemble de 20 JOBS utilisant un échantillon varié des composants du système :

Nom		Composant utilisé		Procédure
TEST01	B	Assembleur F	Loader	ASMFCG
TEST02	B	Fortran H	Edit liens	FORTHCLG
TEST03	A	PL1 F	Loader	PL1LFCG
TEST04	C	Algol F	Edit liens	ALGOFCLG
TEST05	B	Algol F	Edit liens	ALGOFCLG
TEST06	A	Cobol F	Edit liens	COBFCLG
TEST07	C	Fortran G	Edit liens	FORTGCLG
TEST08	B	FORMAC	PL1F Edit liens	FORMAC
TEST09	B	PL1F	Loader GO	PL1LFCG
TEST10	B	Algol W		AWBATCH
TEST11	C	Algol F	Edit liens	ALGOFCLG
TEST12	B	Fortran G	Edit liens	FORTGCLG
TEST13	B	PL1F	Loader	PL1LFCG
TEST14	B	Assembleur G	Loader	ASMGCG
TEST15	B	WATFOR		TPWATFOR
TEST16	A	IEHLIST		
TEST17	B	PL360	Edit liens	PL360CLG

TEST18	B	Cobol U	Edit liens		COBUCLG
TEST19	B	IEFBR14			
TEST20	A	IEBGENER	2xTRI	2xIEBPTPCH	

Ces JOBS n'utilisent aucun fichier, disque ou bande, privé. Les accès disques sont donc dûs à l'utilisation :

- des bibliothèques du système
- des data sets de travail

b- Exécution du batch test

Elle doit être effectuée avec un démarrage spécial d'ASP/MVT; le programme AMAP est implanté dans le système et contrôle ensuite l'exécution du batch test. Les relevés concernant les mesures du batch test sont gardés dans un fichier et seront analysés par la suite.

Le temps global de l'exécution du batch test est de 745 secondes.

c- Analyse des résultats

De l'ensemble des résultats fournis par AMAP, nous ne présenterons que les trois points les plus importants.

i- Charge des unités à accès direct

3 classes de Data sets seront utilisées sur ces unités pour cette expérience :

- les bibliothèques du système
- les data sets de travail
- les files d'attente d'ASP

Bibliothèques du système : elles sont réparties sur 4 unités de disques et 2 tambours de la façon suivante :

Volume	bibliothèque système	fréquence d'utilisation (1)
2OMVT8	SYS1.PROCLIB	M
2OMVT9	SYSCTLG	G
	SYS1.NUCLEUS	F
	SYS1.SVCLIB	G
	SYS1.LOGREC	F
	SYS1.LINKLIB	G
2OMVT1	SYS2.LINKLIB	G
2OMVT2	SYS1.UGRLINK	F
	SYS1.LIBFMAC	F
	SYS1.AWLKLIB	F
	SYS1.WATFOR	F
	SYS1.PL36OLIB	F
	SYS1.PLOSLIB	F
2OMVT3	SYS1.MACLIB	G
	SYS1.SORTLIB	F

20MVT4	SYS1.FORTLIB	M
	SYS1.PL1LIB	M
	SYS1.ALGLIB	M
	SYS1.COBLIB	F

(1) { G = Grande
M = Moyenne
F = Faible

REMARQUE : Le tambour 20MVT9 est le volume système résident.

Data sets de travail : Ils sont pré-alloués au démarrage du système ASP/MVT par des initiateurs I_j suivant un algorithme défini dans le système.
Ils sont nommés symboliquement Ut_i (i=1,2,3).
Chaque initateur I_j alloue 3 data sets Ut_i sur des unités disques séparées.
L'algorithme de répartition fait apparaître l'ordre d'allocation résumé dans le tableau ci-dessous.

	Initiateur 1	I_2	I_3	I_4	I_5	I_6	I_7
UT1	IMAG84	IMAG85	IMAG84	IMAG84	IMAG85	IMAG84	IMAG83
UT2	IMAG79	IMAG84	ISNO03	ISNO03	IMAG84	IMAG79	IMAG84
UT3	ISNO04	20MVT2	CNRS01	20MVT3	20MVT2	20MVT4	ILLO03

Files d'attente d'ASP : Par définition du système, les files d'attente d'ASP prennent place sur des unités disque et dans notre exploitation sur des volumes complets; ceci sous-entend qu'il ne peut y avoir des data sets des 2 premières classes sur ces mêmes volumes. Pour notre expérience 2 unités appelées Q1 et Q2 retiennent ces files d'attente.

Résultats obtenus par AMAP : Les différents relevés fournis par AMAP nous permettent de dresser le taux d'utilisation de chaque unité à accès direct. Dans ce tableau nous regroupons le nombre d'accès à chaque unité et son pourcentage de charge sachant que le temps total de l'expérience a été de 745 sec.

Type	Nom	Adresse	Type de volume (1)	Nb UT_i	Nb bibliothèque système	Nb d'enregistrements traités	Capacité par seconde	% charge
Tambour	20MVT9	1C0	S	0	5	27665	57	65,1
Tambour	20MVT8	1C1	S	0	1	7251	57	17,0
Disque	20MVT1	230	S	0	1	11426	40	38,3
	Q1	291	S	0	0	2186	40	7,3
	Q2	335	S	0	0	2138	40	7,1
	20MVT4	331	S	1	4	1428	40	4,7
	20MVT2	231	S	2	6	1081	40	3,6
	IMAG84	334	P	7	0	962	40	3,2
	IMAG79	232	P	2	0	787	40	2,6
	IMAG83	233	P	1	0	757	40	2,5
	ILL003	234	P	1	0	596	40	2,0
	20MVT3	330	S	1	2	531	40	1,7
	IMAG85	290	P	2	0	335	40	1,1
	ISN003	235	P	2	0	235	40	0,7
	CNRS01	332	P	1	0	207	40	0,6
	ISN004	333	P	1	0	176	40	0,5
	IMAG80	292	P	0	0	57	40	0,1
Disque	IMAG81	293	P	0	0	15	40	0,0

(1) S = volume système
P = volume permanent

CONCLUSIONS :

. Bibliothèques du système :

Le disque 20MVT3 qui contient la bibliothèque du système SYS1.MACLIB et SYS1.SORTLIB est deux fois moins chargée que les disques 20MVT2 et 20MVT4.

Il est donc plus apte à recevoir d'éventuelles bibliothèques système.

. Data sets de travail du système :

Le disque IMAG84 contient sept data sets pré-alloués.

En échangeant les adresses de Q1(291) et de IMAG81(293), le disque IMAG81(PDISK) est alors susceptible de recevoir des data sets pré-alloués.

Cet échange a permis d'obtenir une meilleure répartition des data sets pré-alloués.

ii- Analyse de l'utilisation de SYS1-SVCLIB

Cette analyse permet d'obtenir la liste des membres les plus utilisés de SYS1-SVCLIB dans l'ordre décroissant.

Résultat du test :

Nom du membre	Taille(octet)	Fait partie de la liste standard	Nb d'accès
IGC0007B	292		620
IGC0006[$^{12}_{0}$]	728		451

Catégorie RSVC	IGC0107B	976		448
	IGC0003E	1024		423
	IGC0303D	736		314
	IGC0003D	440		258
	IGC5403D	640		241
	IGC0403D	912		241
	IGG0190M	1024	oui	225
	IGG0190L			224
	IGG0199M			224
	IGG0190N			222
	IGG0190S			221
	IGG0191A			216
	IGG0196A		oui	216
	IGG0190Y			215
	IGG0200G		oui	210
	IGC0002[$^{12}_{0}$]		oui	209
	IGG02002		oui	208
	IGG0209Z		oui	208
	IGG0200H		oui	208
	IGC0001I		oui	207
	IGG0199X	1024	oui	201

Catégorie RAM	IGG019BC	240		66
	IGG019AC	280		16
	IGG019AV	128		10
	IGG019BD	296		6

CONCLUSIONS

La taille de la LINK PACK AREA est fixée, par expérience en fonction de la taille de la mémoire principale disponible.

Ce paramètre permet de déterminer le nombre de modules à inclure dans chacune des listes standards IEARSVOO et IEAIGGOO de SYS1.PARMLIB.

Ces listes standards seront construites à partir des résultats de l'expérience AMAP.

On peut remarquer le nombre important d'accès à certains membres de SYS1.SVCLIB (plus de 600). Ce data set étant en exploitation normale, le plus utilisé du système OS/MVT.

iii- Analyse de l'utilisation de SYS1.LINKLIB

Cette analyse permet d'obtenir la liste des membres les plus utilisés de SYS1.LINKLIB dans l'ordre décroissant :

Résultat du test :

Nom du module	taille(octet)	nb d'accès
IEFSDO61	45888	110
IEFSDO62	9088	71
IEFW21SD	20888	71
IEFWAOOO	26840	70
DEVMASKT	1192	70
DEVNAMET	440	69
IEFWDOOO	25328	68
IEFVH1	4848	42
IEFQMSSS	3336	42
IEFVHN	1464	41
IEEVRC	3400	41

IEFSDO86	3368	38
IEFSDO8O	7200	38
IEEVSTAR	3952	22
IEEVRCTL	1728	22
IEEVRJCL	416	22
IEFSDO6O	3360	22
IEFIRC	1464	20

CONCLUSIONS

Le nombre d'accès aux membres les plus utilisés de SYS1.LINKLIB est en moyenne six fois inférieur au nombre d'accès aux membres les plus utilisés de SYS1.SVCL1B.
De plus, la taille des modules sélectionnés est importante; il est donc utopique de les rendre résidents.

Par définition du système, SYS1.LINKLIB peut être scindé en plusieurs bibliothèques. Nous utiliserons donc la hiérarchie de mémoire disponible tambour disque, pour mettre sur tambour (dans SYS1.LINKLIB), les membres sélectionnés par l'expérience AMAP.

La totalité des modules restant sur disque dans SYS2.LINKLIB.

Parmi les membres sélectionnés, nous distinguerons encore les plus utilisés à savoir :

IEFSDO61 IEFSDO65 IEFSD104 IEFV4221 IEFW425D
IEFSDO62
IEFW21SD IEFVMCVL IEFVM1 IEFXA
IEFWAOOO IEFWCOOO
DEVMASKT
DEVNAMET
IEFWDOOO IEFW41SD

et leurs noms seront regroupés par ordre alphanumérique dans le membre IEABLDOO de SYS1.PARMLIB.

Ce membre permettra de construire en mémoire la table BLDL qui contient le nom et l'amplacement sur tambour de ces modules.

2- Influence des modules résident en mémoire principale

Deux expériences ont été réalisées pour montrer l'intérêt de rendre résidents en "LINK PACK AREA" les modules déterminés par l'expérience AMAP précédente.

Pour chaque expérience, l'exécution du batch sera faite :

1/ sans LINK PACK AREA étendue
2/ avec LINK PACK AREA étendue par des modules occupant 22K de mémoire.

a- Expérience 1

Conditions

- Les data sets importants du système résident sur tambour
- La file d'attente sur tambour
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- sans LINK PACK AREA (LPA) étendue

$T_{session}$ = 16 mn 49 s

$T_{exécution}$ = 11 mn 12 s

T_{reader} = 54 s

- avec LPA étendue

$T_{session}$ = 16 mn 32 s

$T_{exécution}$ = 10 mn 53 s

T_{reader} = 58 s

b- Expérience 2

Conditions

- data sets importants du système sur disque
- file d'attente sur disque
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- Sans LPA étendue

$T_{session}$ = 19 mn

$T_{exécution}$ = 13 mn 13 s

T_{reader} = 2 mn 42 s

- avec LPA étendue

$T_{session}$ = 18 mn 56 s

$T_{exécution}$ = 13 mn 5 s

T_{reader} = 2 mn 7 s

On peut mesurer le gain de temps d'exécution obtenu avec LPA étendue :

Pour l'expérience 1 : 19 s soit 2,8 %
Pour l'expérience 2 : 8 s soit 1 %

Le fait de rendre résidents :

- les modules du membre IEAIGGOO de SYS1.PARMLIB
- les modules du membre IEARSVOO de SYS1.PARMLIB
- la table BLDL correspondant au membre IEABLDOO

déterminés par l'expérience AMAP à donc permis d'obtenir un gain de temps d'exécution moyen de : 2 %

3- Conclusion

Ce résultat se justifie par le fait que la plupart des modules ajoutés en mémoire sont relativement peu utilisés par ce batch test; en effet, si l'on examine la composition des différents JOBS de ce batch, on peut remarquer qu'ils sont constitués par des STEPS dont le temps d'exécution est relativement long. Toutefois, on peut raisonnablement penser que l'effet de la présence d'une LPA étendue serait nettement plus important dans le cas d'un batch-test composé de nombreux STEPS ayant un temps d'exécution plus modeste; ce qui conduirait à une utilisation plus intensive des modules de la LPA.

II - INFLUENCE DU TYPE DE L'UNITE A ACCES DIRECT SUPPORTANT CERTAINS COMPOSANTS DU SYSTEME

Nous étudions deux types de supports à accès direct :

- des unités de tambour magnétique 2301
- des unités de disque 2314

Rappels des caractéristiques du disque 2314 Model A (Référence II)

Volume d'information à accès direct de capacité maximale supérieure à 29 millions d'octets.
Est constitué de 203 cylindres (200 utilisables) de 20 pistes chacun.
Chaque piste peut contenir au maximum 7294 octets.
Le temps de rotation d'un 2314 est de 25 ms
Le transfert d'information entre mémoire centrale et disque se fait à la vitesse maximum de 312 000 octets par seconde.
Le temps moyen de positionnement du bras est d'environ 75 ms.

Rappels des caractéristiques du tambour 2301 (Référence II)

Volume d'information à accès direct de capacité maximum supérieure à 4 millions d'octets.
Constitué de 200 pistes pouvant contenir au maximum 20483 octets chacune.
Le temps de rotation d'un tambour est de 8,6 millisecondes.
Le transfert d'information entre mémoire centrale et tambour se fait à la vitesse maximum de 1,2 millions d'octets par seconde.
Le tambour 2301 a donc une capacité 7 fois inférieure à celle du disque 2314.
Le transfert d'information se fait pratiquement 3 fois plus rapidement entre mémoire centrale et tambour qu'entre mémoire centrale et disque et ceci sans positionnement de bras.

Nous connaissons mal à priori l'influence des supports d'information sur le comportement du système. Toutefois à l'aide de notre expérience passée, nous avons dégagé les composants les plus importants du système :

- data sets du volume système résident : (volume à partir duquel on effectue le chargement initial du système : IPL) peu avantageux mais nécessaire en standard
- file d'attente des travaux
- data sets de travail

Notre but est de mesurer la répartition optimale, pour notre installation, de ces composants à travers les différents niveaux de la hiérarchie mémoire: mémoire centrale, tambour, disques.

1- Etude de la position du volume système résident

Dans cette étude, le volume système résident contient les data sets suivants :

SYS1.LOGREC
SYSCTLG
SYS1.NUCLEUS
SYS1.SVCLIB
SYS1.LINKLIB

Le data set SYS1.LINKLIB contient les modules sélectionnés par l'expérience AMAP.

Exécution du batch
- avec volume système résident sur disque
- avec volume système résident sur tambour

Conditions de l'expérience

- file d'attente des jobs sur disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- avec volume système résident sur disque

$T_{session}$ = 18 mn 56 s

$T_{exécution}$ = 13 mn 05 s

T_{reader} = 2 mn 07 s

- avec volume système résident sur tambour

$T_{session}$ =

$T_{exécution}$ = 10 mn 55 s'

T_{reader} = 56 s

Le fait d'utiliser un tambour 2301 comme volume système résident a permis d'obtenir un gain de temps d'exécution de : 2 mm 10 s soit un pourcentage de gain de : 16,6 %.

2- Etude de la position de la file d'attente des travaux (SYS1.SYSJOBQUE)

Exécution du batch -avec SYS1.SYSJOBQE disque
-avec SYS1.SYSJOBQE sur tambour 20MVT8

Conditions de l'expérience

- volume système résident sur tambour
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- avec SYS1.SYSJOBQE sur disque

$T_{session}$ =

$T_{exécution}$ = 10 mn 55 s

T_{reader} = 56 s

- avec SYS1.SYSJOBQE sur tambour

$T_{session}$ = 16 mn 23

$T_{exécution}$ = 10 mn 53 s

T_{reader} = 58 s

Le fait de mettre la file d'attente des travaux sur tambour a permis d'obtenir un gain de temps d'exécution de 2 secondes soit 0,3 % de gain.

3- Etude de la position des data sets de travail

Certains composants du système (compilateur, éditeur de liens, etc...) utilisés par le batch, ont besoin d'un ou de plusieurs data sets de travail séquentiels de nom SYSUTn, dont le plus utilisé est SYSUT1.

Ces data sets peuvent être définis au coup par coup par les steps d'un job ou globalement, au démarrage d'un initiateur soit sur disque, soit sur tambour.

La taille de ces data sets est fonction des besoins du composant .

Dans le cas où SYSUT1 est implanté sur disque, la taille de ce data set n'est pratiquement pas limitée étant donné la capacité et le nombre de ces unités.

Dans le cas où SYSUT1 est implanté sur un tambour nous devons calculer sa taille pour allouer 6 data sets SYSUT1 sur ce tambour.

a- Expérience

Exécution du batch
- avec data sets de travail sur disques 2314
- avec data sets de travail sur tambours 2301

Conditions de l'expérience

- volume système résident sur disque
- SYS1.SYSJOBQE sur disque

- sans LPA étendue
- 7 initiateurs
- Reader avec 1 buffer de 3200 octets

Résultat

- avec data sets de travail sur disque

T_s = 19 mn

T_e = 13 mn 13 s

T_r = 2 mn 42 s

- avec data sets de travail sur tambour

T_s =

$T_{exécution}$ = 13 mn 38 s

T_{reader} = 3 mn 47 s

Le fait de mettre les data sets de travail SYSUT1, sur tambour n'apporte pour le batch aucune amélioration.

b- Cas particulier

Il est cependant possible d'obtenir un gain de temps d'exécution important en utilisant le tambour comme support du data set SYSUT1. Prenons comme exemple la compilation d'un important programme écrit en langage PL/1. Le compilateur PL/1 utilise une zone de travail en mémoire principale appelée région, dont la taille est fixée par l'utilisateur (cette taille s'exprime en K octets ou K = 1024 octets).

Cette zone de travail est étendue automatiquement par un data set (SYSUT1) sur une unité à accès direct lorsqu'elle n'est pas de taille suffisante pour le programme à compiler.

L'utilisation de ce data set est donc d'autant plus importante, que la taille du programme à compiler est grande, par rapport à la taille de la région allouée au compilateur.

Conditions de l'expérience

- volume système résident sur disque
- SYS1.SYSJOBQE disque

Résultats

- avec utilitaire sur disque

temps de compilation REGION=100K : 2 mn 47 s
REGION=82K : 2 mn 52 s

- avec utilitaire sur tambour

temps de compilation REGION=100K : 2 mn 15 s
REGION=82K : 2 mn 14 s

REGION=100K Le gain de temps d'exécution est de 32 s soit de 19 %
REGION=82K Le gain de temps d'exécution est de 38 s soit de 22 %

donc en moyenne : 20,5 % de gain

4- Conclusions

En résumé, pour l'ensemble de cette étude nous avons pu montrer que :

- Lorsque le volume système résident est sur tambour, le gain de temps d'exécution est important (16 %)

  Ce résultat est dû au nombre d'accès important dans le data set SYS1.SVCLIB principalement, et dans le data set SYS1.LINKLIB au cours de l'exécution du batch test.

- Lorsque la file d'attente des travaux réside sur tambour, aucune amélioration n'est apportée au temps d'exécution.

  Ce résultat est principalement lié aux caractéristiques du batch test: 7 jobs constitués de steps relativement longs en temps CPU. La file d'attente des travaux est dans ce cas pratiquement inexistante. Si la file d'attente des travaux était plus importante l'influence du type de support serait nécessairement plus sensible.

- Lorsque les data sets de travail du système résident sur tambour aucune amélioration n'est apportée au temps d'exécution.

  Ce résultat se justifie par le fait que dans le batch, les data sets de travail ne sont pratiquement pas utilisés. Il n'est donc pas justifié d'implanter les data sets de travail sur tambour sauf dans quelques cas très particuliers (compilations importantes par exemple).

En résumé, nous utiliserons de préférence le tambour magnétique comme volume système résident c'est-à-dire comme support des data sets système: SYS1.SVCLIB principalement ainsi que SYS1.LINKLIB.

III - INFLUENCE DES PARAMETRES DU READER

Le flot d'entrée traité par le reader est constitué par un ensemble de JOBS comprenant environ 1400 cartes auxquelles il faut ajouter les cartes contrôles provenant des procédures cataloguées utilisées.

Globalement nous obtenons 1722 images de cartes dont 224 cartes contrôle, soit :

13 % de cartes contrôle
87 % de cartes données

La tâche reader est lancée par une commande qui appelle une procédure du type suivant : figure 4.

```
/RDR      JOB MSGLEVEL=1                                                    00001000
/R        EXEC RDR                                                          00001100
**   POUR MVT STAND ALONE UNTOUEMENT                                        00001200
** SOURCE : IBM , REF : SYSTEM PROGRAMMER'S GUIDE                          00001300
** FILE # S360-20 , FORM C28-6550-6 , CHAPITRE   , P. 227-261             00001400
XIEFPROC   EXEC     PGM=IEFIRC,                  READER FIRST LOAD        00001500
X     REGION=56K,                                                         00001600
X  PARM='30100105C01C2491C011PDISK   FCCCC00'  DEFAULT OPTIONS           00001700
**        BPPTTTCCMMMIIICCCRLSSSSSSSSAAAADD  PARM FIELDS                  00001800
**        B    PROGRAMMER NAME AND ACCOUNT NUMBER NEEDED                  00001900
**           PROGRAM CANNOT BE ROLLED OUT                                 00002000
**        PP   PRIORITY=01                                                00002100
**        TTT  JOB STEP INTERVAL=01 MINUTES                               00002200
**        OOO  PRIMARY SYSOUT SPACE=50 TRACKS                             00002300
**        MMM  SECONDARY SYSOUT SPACE=10 TRACKS                           00002400
**        III  READER/INTERPRETER DISPATCHING PRIORITY=249                00002500
**        CCC  JOB STEP DEFAULT REGION=100K                               00002600
**        R    DISPLAY AND EXECUTE COMMANDS=1                             00002700
**        L    BYPASS LABEL OPTION=1                                      00002800
**        SSSSSSSS  SYSOUT UNIT NAME= SYSDA                               00002900
**        AAAA COMMAND AUTHORITY FOR MCS=E000-ALL COMMANDS          MCS   00003000
**           MUST BE AUTHORIZED                                     MCS   00003100
**        DD   JCL AND ALLOCATION MESSAGE LEVEL DEFAULTS=00      OCI968  00003200
/IEFPROC.IEFRDER DD UNIT=CC2,VOL=SER=000300,LABEL=(1,SL),DSNAME=BENCH
/IEFRDER DD UNIT=2540,                  CARD READER                       00003300
X             LABEL=(,NL),                    UNLABELED                 00003400
X             VOLUME=SER=SYSIN,               VOLUME NAME               00003500
X  DCB=(BLKSIZE=80,RECFM=F,BUFL=80,BUFNO=1)                              00003600
XIEFPDSI DD DSN=SYS1.PROCLIB,DISP=SHR                                   00003700
XIEFDATA DD UNIT=PDISK,                                                 00003800
X   SPACE=(80,(500,500),RLSE,CONTIG),                                   00003900
X DCB=(BUFNO=1,LRECL=80,BLKSIZE=3200,RECFM=FB,BUFL=3200,DSORG=PS)       00004000
```

FIGURE 4

Dans cette procédure 2 cartes contrôle DD définissent les fichiers d'entrée et de sortie du reader :

- IEFRDER définit le fichier d'entrée constitué d'enregistrements de longueur 80 non bloqués. Il est possible,en utilisant le sous-paramètre OPTCD=C du paramètre DCB,de chaîner la lecture des enregistrements. Dans ce cas, plusieurs opérations de lecture seront traitées par le système comme une seule opération continue.
  Cette technique nécessite l'utilisation d'au moins deux mémoires tampons

- IEFDATA définit le fichier de sortie séquentiel où le reader range les data sets(cartes)inclus dans le flot d'entrée.
  Les caractéristiques de ce fichier sont paramétrées (sous-paramètres du paramètre DCB).

Les fonctions de transfert de ces data sets (images de cartes) sur le fichier défini par la carte DD IEFDATA(fichier sur unité à accès direct) utilisent une fraction de temps CPU non négligeable.

Nous nous proposons d'étudier l'influence des paramètres du reader sur le temps d'exécution du batch à savoir :

- type de l'unité d'entrée des JOBS (lecteur de cartes-bande)
- taille des mémoires tampons du reader (IEFDATA)
- nombre de mémoires tampons du reader (IEFDATA)
- chaînage de la lecture (IEFRDER)

1- Influence du type de l'unité de lecture des JOBS

La comparaison est établie entre une entrée des JOBS à partir d'une bande magnétique et à partir d'un lecteur de cartes.

Le reader dispose d'un buffer de 3200 octets, l'unité de bandes magnétiques transfère 750 cartes/seconde et l'unité de lecture de cartes 1000 cartes/minute.

En valeur absolue, la lecture d'une bande magnétique est donc 45 fois plus rapide que la lecture des cartes.

Conditions de l'expérience

- volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- sans LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- avec entrée sur bande magnétique

$T_{session}$ = 19 mn

$T_{exécution}$ = 13 mn 13 s

T_{reader} = 2 mn 42 s

- avec entrée sur cartes

$T_{session}$ = 19 mn 25 s

$T_{exécution}$ = 14 mn 25 s

T_{reader} = 3 mn 6 s

Le gain de temps d'exécution du batch avec lecture des JOBS sur bande magnétique est de 1 mn 12 s soit de 8,3 % par rapport à une lecture sur carte.

Nous utiliserons donc de façon standard, pour la suite des expériences, la lecture des JOBS sur bande magnétique.

2- Influence de la taille des mémoires tampons du reader (IEFDATA)

Deux expériences ont été réalisées avec lecture des JOBS sur bande magnétique.

L'expérience 1 attribue une seule mémoire tampon au reader, nous ferons varier la taille de cette mémoire tampon.

L'expérience 2 attribue deux mémoires tampons au reader, nous ferons également varier la taille de ces deux mémoires tampons.

a- Expérience 1

Conditions

- volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- sans LPA étendue
- 7 initiateurs

Résultats obtenus avec une seule mémoire tampon

taille de la mémoire tampon	$T_{session}$	$T_{exécution}$	T_{reader}
80	19 mn 26 s	14 mn 51 s	5 mn 15 s
400	20 mn 13 s	14 mn 23 s	3 mn 15 s
1600	19 mn 14 s	13 mn 38 s	3 mn
3200	19 mn	13 mn 13 s	2 mn 42 s

Le temps d'exécution décroît de façon continue lorsque la taille de la mémoire tampon croît de 80 à 3200 octets. Le gain de temps d'exécution est alors 1 mn 38 sec. soit de 11 %.

b- Expérience 2

Conditions

- volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- sans LPA étendue
- 7 initiateurs

Résultats obtenus avec 2 mémoires tampons

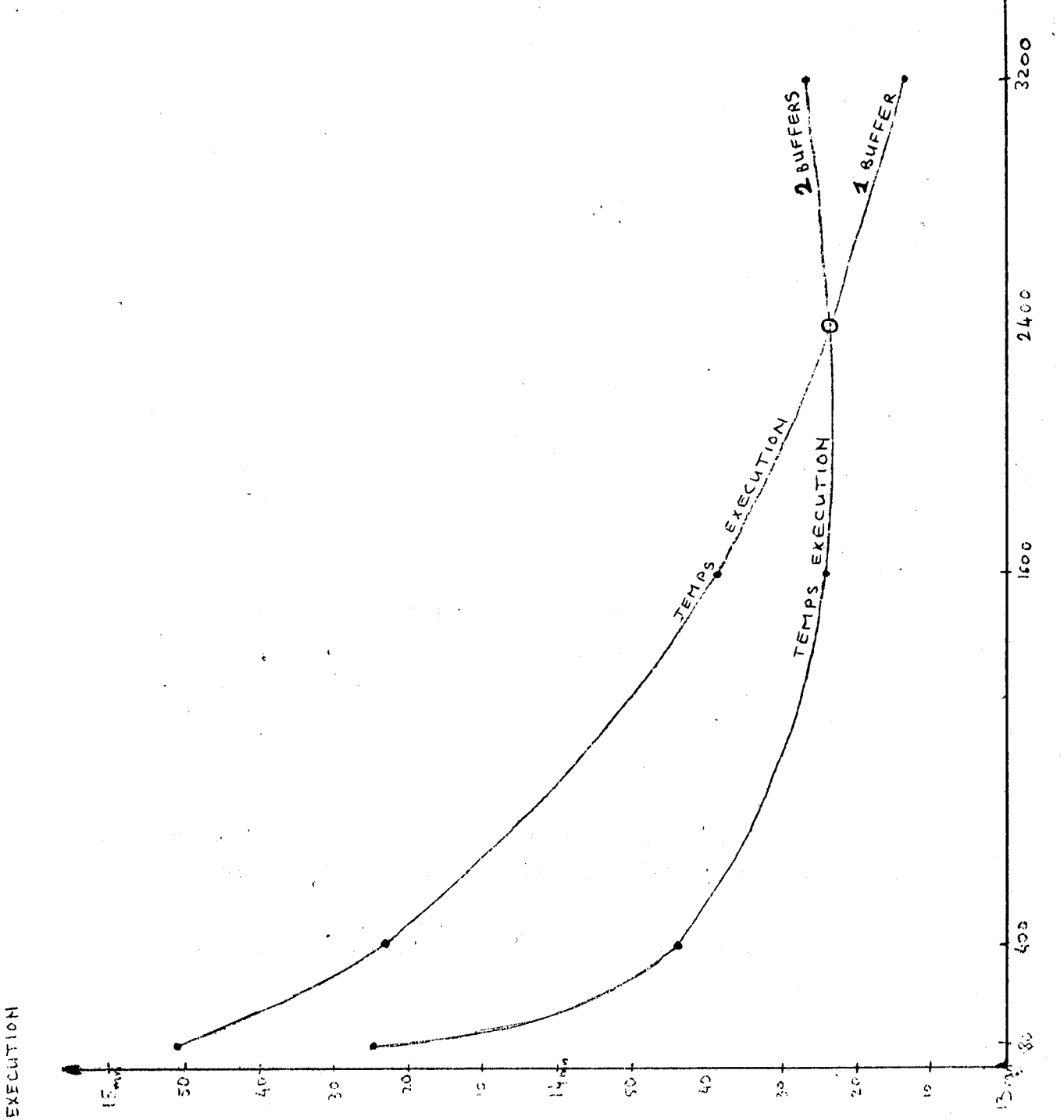
Taille de chacune des 2 mémoires tampons	$T_{session}$	$T_{exécution}$	T_{reader}
80	19 mn 48 s	14 mn 24 s	3 mn 44 s
400	19 mn 8 s	13 mn 44 s	3 mn 5 s
1600	19 mn	13 mn 24 s	3 mn
3200	18 mn 44 s	13 mn 27 s	2 mn 37 s

Le temps d'exécution décroît de façon continue lorsque la taille des 2 mémoires tampons croît de 80 à 1600 octets. Il croît ensuite légèrement lorsque la taille passe de 1600 à 3200 octets. Le gain de temps obtenu est de 57 sec. soit de:

6,6 %

L'ensemble des résultats des deux expériences est montré par la Figure 5. Le temps d'exécution minimum ayant été obtenu avec une mémoire tampon de 3200 octets.

FIGURE : 5


EXECUTION
50
40
30
20
50
40
30
20
400
1600
2400
3200
TAILLE
EN OCTET
2 BUFFERS
1 BUFFER
TEMPS EXECUTION
TEMPS EXECUTION

3 - Influence du nombre de mémoires tampons du reader (IEFDATA)

Pour cette expérience, nous fixerons la taille de mémoire tampon à 80 octets; nous noterons le temps d'exécution du batch en attribuant successivement 1, 2,..., 10 mémoires tampons au reader.

L'entrée des travaux étant effectuée à partir de bande magnétique.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- sans LPA étendue
- 7 initiateurs

Résultats

Nombre de mémoires tampons	$T_{session}$	$T_{exécution}$	T_{reader}
1	19 mn 26 s	14 mn 51 s	5 mn 15 s
2	19 mn 48 s	14 mn 24 s	3 mn 44 s
4	20 mn 37 s	14 mn 41 s	3 mn 44 s
6	20 mn 23 s	14 mn 44 s	3 mn 29 s
10	20 mn 6 s	14 mn 35 s	3 mn 13 s

Le temps d'exécution du batch diminue de 27 secondes soit de 3 % lorsque l'on passe de 1 à 2 mémoires tampons de 80 octets.

Il reste sensiblement le même lorsque le nombre de mémoires tampons est compris entre 2 et 10.

4- Influence du chaînage de la lecture (IEFRDER)

Deux expériences ont été réalisées; la première utilise le lecteur de cartes pour l'entrée des JOBS, la seconde utilise le dérouleur de bandes magnétiques; le reader ayant 1 buffer de 3200 octets.

a- Expérience 1

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- 7 initiateurs
- reader avec 1 buffer de 3200 octets
- sans LPA étendue

Résultats avec entrée des JOBS au lecteur de cartes

- sans chaînage de la lecture

$T_{session}$ = 19 mn 25 s

$T_{exécution}$ = 14 mn 25 s

T_{reader} = 3 mn 6 s

- avec chaînage de la lecture

$T_{session}$ = 19 mn 15 s

$T_{exécution}$ = 13 mn 51 s

T_{reader} = 3 mn 4 s

b- Expérience 2

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- sans LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats avec entrée des JOBS sur bande magnétique

- sans chaînage de la lecture

$T_{session}$ = 19 mn

$T_{exécution}$ = 13 mn 13 s

T_{reader} = 2 mn 42 s

- avec chaînage de la lecture

$T_{session}$ = 19 mn 27 s

$T_{exécution}$ = 13 mn 27 s

T_{reader} = 2 mn 24 s

Le gain de temps d'exécution obtenu avec chaînage de la lecture est de :

Expérience 1 : 34 secondes soit de 4 %

Expérience 2 : Le temps d'exécution n'est pas amélioré

5- Conclusions

Les expériences précédentes nous permettent de tirer les conclusions suivantes :

- Le type de l'unité d'entrée des JOBS joue un rôle important (gain de temps d'exécution de 8,3 % avec entrée de JOBS à partir de bande magnétique au lieu de lecteur de cartes).
  En utilisant un lecteur plus rapide pour l'enregistrement des travaux, le reader constitue une charge moins lourde pour le système; il reste actif moins longtemps (2 mn 42 au lieu de 3 mn 6 s dans l'expérience réalisée).

- Augmenter la taille de la mémoire tampon (IEFDATA) du reader améliore nettement le temps d'exécution du batch.
  Ceci s'explique par une réduction importante du nombre d'opérations d'entrée/sortie pour la mise sur unité à accès directe et la relecture des data sets inclus dans le flot d'entrée des JOBS (blocage des enregistrements). De plus, le reader reste actif 2 mn 42 s au lieu de 5 mn 15 s lorsque cette taille passe de 80 à 3200 octets. Il constitue donc une charge moins importante pour le système.

  Mais cette solution a un inconvénient: en effet cette mémoire tampon fait partie de la zone de mémoire principale utilisée :

  . par le reader
  . par le composant du système qui relira le data set créé par le reader

  Pour obtenir ce gain de temps il est donc nécessaire d'augmenter parallèlement à la taille de la mémoire tampon, la taille de la mémoire principale utilisée :

  . par le reader
  . par le composant du système (exemple: compilateur)

Ce gain de temps entraine une perte de place, un juste compromis est donc nécessaire.

- L'influence du nombre de mémoires tampons du reader est nettement moins sensible sur le temps d'exécution du batch (3 % de gain passant de 1 à 3 mémoires tampon).
  Il est donc préférable d'augmenter la taille de la mémoire tampon plutôt que le nombre de mémoires tampons.

- L'influence du chaînage de la lecture apporte une amélioration sensible lorsque l'entrée des travaux est effectuée à partir du lecteur de cartes.(4 % pour le batch).
  La charge apportée par le reader est sensiblement plus faible puisque plusieurs opérations d'entrée/sortie sont traitées par le système comme une seule.
  Le reader reste actif moins longtemps : 2 mn 24 s au lieu de 2 mn 42 s.

En résumé, pour les expériences suivantes, l'entrée des JOBS sera effectuée de préférence à partir de bande magnétique; et n'étant pas limités au point de vue place, nous utiliserons pour le reader une mémoire tampon de 3200 octets.

IV - INFLUENCE DU NOMBRE D'INITIATEURS

A un instant donné, l'exécution d'une tâche n'immobilise pas généralement toutes les ressources du système :

- CPU
- Mémoire principale
- unités d'entrée-sortie

En exécutant plusieurs tâches en parallèle, démarrées par plusieurs initiateurs, les ressources du système peuvent être mieux utilisées . C'est le principe même de la multiprogrammation.

Nous nous proposons de déterminer le nombre d'initiateurs qu'il faut rendre actifs pour obtenir un temps d'exécution minimum du batch. Les caractéristiques du batch joueront, là encore, un grand rôle sur les résultats obtenus.

Notre batch test est constitué de JOBS variés que l'on regroupe en 3 sous-ensembles :

- JOBS n'utilisant que le CPU Ex: BENCH10
- JOBS faisant beaucoup d'entrées-sorties
- JOBS mixtes Ex: BENCH

Les fichiers traités par ces JOBS sont répartis sur différentes unités de disques.

1- Expérience

Le batch est exécuté successivement avec 1,2,3... puis 7 initiateurs.

Conditions

- Volume système résident sur tambour
- SYS1.SYSJOBQE sur tambour
- avec LPA étendue
- Reader avec 1 buffer de 3200 octets

Résultat : Figure 7

Nbre d'initiateurs	$t_{session}$	$t_{exécution}$	t_{reader}
1	18 mn 35 s	17 mn 57 s	49 s
2	14 mn 3 s	14 mn 2 s	53 s
3	16 mn 28 s	11 mn 19 s	55 s
4	16 mn 49 s	11 mn 14 s	56 s
5	16 mn 52 s	11 mn 14 s	57 s
6	17 mn 10 s	11 mn 30 s	1 mn 4 s
7	16 mn 32 s	10 mn 53 s	58 s

Le tableau ci-dessous donne le pourcentage de gain de temps d'exécution obtenu par rapport au temps d'exécution du batch avec un seul initiateur :

Nombre d'initiateurs	% de gain
2	22 %
3	37 %
4	37,5 %
5	37,5 %
6	36 %
7	39,5 %

On peut constater que :

TEMPS
SESSION

NOMBRE D'INITIATEURS

SYSTEME OS / MVT

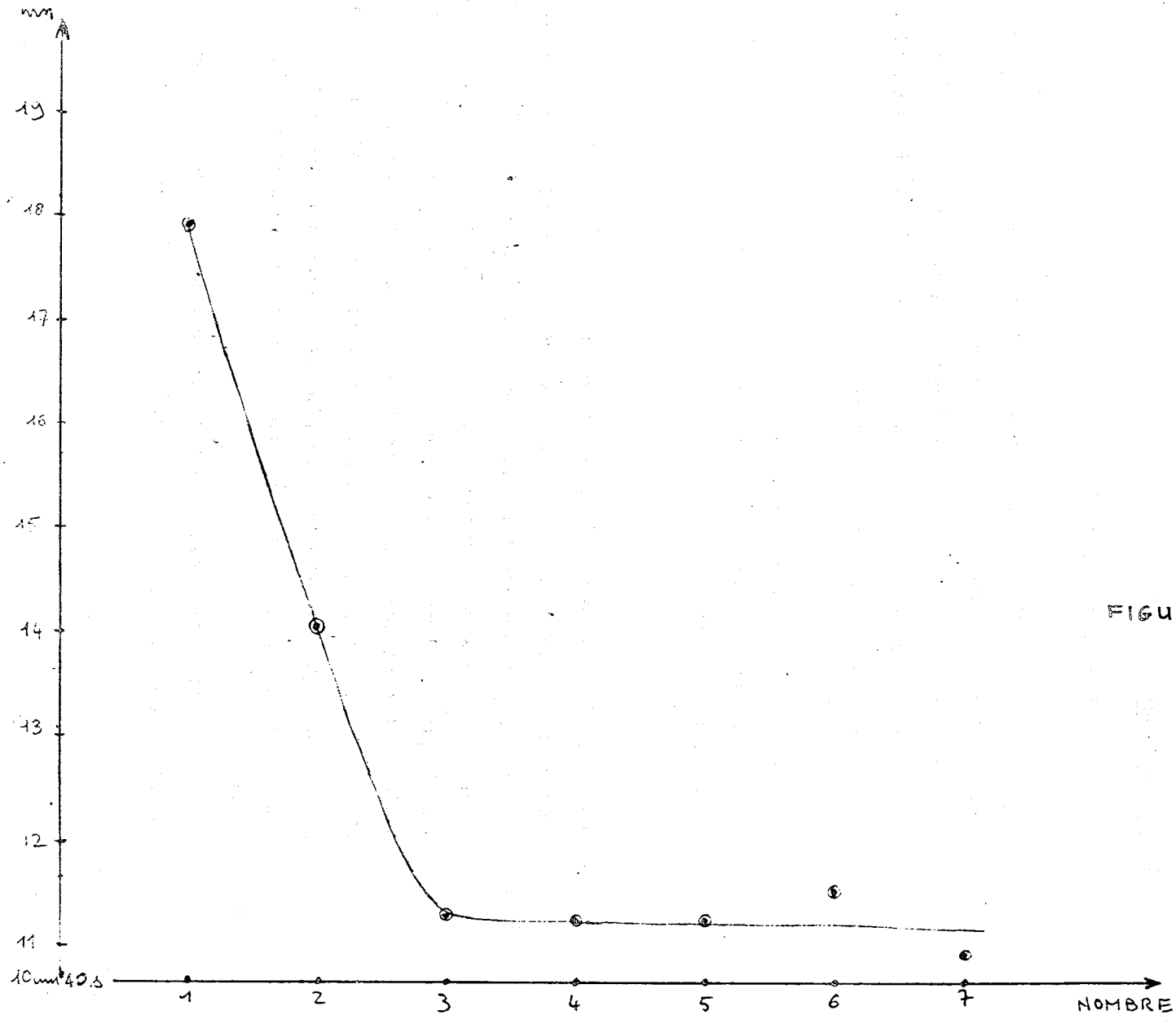


FIGURE : 7

- Le passage de 1 à 3 initiateurs améliore nettement le temps d'exécution du batch.

- Le passage de 3 à 7 initiateurs n'apporte qu'un gain de temps d'exécution faible.

2- Conclusions

L'expérience réalisée montre :

- Un gain de temps d'exécution important dû à la multiprogrammation (37 % lorsque l'on passe de 1 à 3 initiateurs)
  Ce gain de temps est cependant lié aux caractéristiques du batch test. Nous pouvons considérer que ce gain de temps sera maximum, quand les JOBS, qui s'exécutent en parallèle, utilisent :

  . des ressources variées (CPU...)
  . des fichiers bien répartis sur les unités d'entrée-sortie afin d'éviter les conflits.

- Une stabilité du temps d'exécution au delà de 3 initiateurs.
  Pour notre batch test, cette stabilité peut s'expliquer par une prépondérance des entrées-sorties par rapport à l'utilisation du CPU. Il est à noter que lorsque le nombre d'initiateurs augmente, la région utilisée par chaque JOB diminue. Cependant, augmenter le nombre d'initiateurs minimise les risques de sous-emploi de la machine en cas d'arrêt de l'un des initiateurs. Ce cas est important dans la pratique, et permet de conclure que la situation la plus favorable à la multiprogrammation, a le plus de chance d'être obtenue avec un nombre maximum d'initiateurs actifs.

  Il sera aussi plus probable que des travaux de type différent soient en concurrence, d'où une meilleure utilisation des ressources.

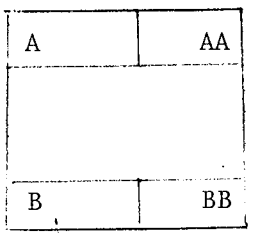
Pour montrer l'importance du type de travail à exécuter sur l'efficacité de la multiprogrammation, nous nous proposons de donner un contre-exemple.

3- Contre exemple de multiprogrammation

La position des fichiers, sur lesquels travaillent les programmes, joue un rôle capital dans l'efficacité de la multiprogrammation.

Etudions l'exemple :

- soient 2 programmes fortran FORT1 FORT2
- FORT1 lit les enregistrements du fichier séquentiel A et les recopie dans le fichier séquentiel AA.
- FORT2 lit les enregistrements du fichier séquentiel B et les recopie dans le fichier séquentiel BB.
- Position des fichiers : ils sont tous sur le même volume



A et AA sont au début du disque
B et BB sont à la fin du disque
Chaque fichier a une taille de
1 cylindre et contient des enregistrements non bloqués.

Le temps d'exécution global des 2 programmes est de :

Avec 1 initiateur : 1 mn 1 s
Avec 2 initiateurs : 1 mn 47 s

Donc, dans ce cas particulier, nous observons une perte de temps importante; en effet, les 2 programmes qui s'exécutent en parallèle sont antagonistes.

V - CONCLUSIONS

1- Tableau récapitulatif des résultats obtenus séparément

Modifications apportées	Gain de temps d'exécution obtenu
- 22 K de modules résidents supplémentaires.	2 %
- Utilisation de tambours magnétiques pour les data sets du système au lieu de disques.	16 %
- entrée de JOBS sur le lecteur de bande magnétique au lieu du lecteur de cartes.	8,3 %
- Mémoire tampon de 3200 octets pour le reader au lieu de 80 octets	9 %
- Chaînage de la lecture pour le reader	4 %
- Passage de 1 à 7 initiateurs.	39 %

2- Choix des valeurs optimales des paramètres de MVT étudiés

Le tableau ci-dessus nous permet de définir les meilleures conditions de fonctionnement du système OS/MVT :

- 22 K de modules résidents supplémentaires en "link pack area"
- Data sets système sur tambour 2301
  - . SYS1.LOGREC
  - . SYSCTLG
  - . SYS1.NUCLEUS
  - . SYS1.SVCLIB
  - . SYS1.LINKLIB (modules très utilisés)
- Data set SYS1.SYSJOBQE sur tambour
- 7 initiateurs actifs
- reader avec 1 mémoire tampon de 3200 octets
- chaînage de la lecture (OPTCD=C)
- entrée des JOBS (input stream) à partir d'une bande magnétique.

3- Gain de temps d'exécution obtenu

Le temps d'exécution du batch OS était de 14 mn 24 s dans les conditions suivantes :

- data sets système sur disque 2314
- data sets SYS1.SYSJOBQE sur disque 2314
- sans LPA étendue
- 7 initiateurs actifs
- reader avec 2 mémoires tampons de 80 octets
- sans chaînage du traitement
- entrée des JOBS à partir d'une bande magnétique

Avec le système amélioré ce temps d'exécution a été de :

10 mn 53 s

Donc un gain de temps de :

3 mn 31 s soit de 25 %

C H A P I T R E IV

ETUDE DU SYSTEME DE MULTIPROGRAMMATION OS/MVT FONCTIONNANT DANS UNE MACHINE VIRTUELLE GEREE PAR LE SYSTEME DE PARTAGE DE TEMPS CP

Lorsque le système de multiprogrammation OS/MVT fonctionne dans une machine virtuelle gérée par le système CP, celui-ci réalise un certain nombre d'opérations pour la machine virtuelle, qui dégradent les performances du système OS/MVT.

Ces opérations sont les suivantes :

- pagination et traitement des fautes de page
- répartition et utilisation du CPU
- traitement des interruptions d'entrée-sortie avec réflexion à la machine virtuelle
- simulation des instructions d'entrée/sortie (recompilation des programmes canaux virtuels de OS/MVT en programmes canaux réels)
- traitement des entrées-sorties
- simulation des instructions privilégiées
- réflexion des interruptions programme
- " " SVC
- translation des adresses

Après avoir déterminé dans différentes conditions le temps d'exécution du batch avec le système OS/MVT sur la machine réelle, nous allons réaliser les mêmes expériences en faisant fonctionner le système OS/MVT dans un contexte de machines virtuelles.

Pour déterminer les performances du système OS/MVT sous CP, nous comparerons le temps réel minimum d'exécution du batch sous CP et le temps d'exécution obtenu dans les mêmes conditions sur la machine réelle.

Nous établirons un premier état des performances lorsque la machine virtuelle OS/MVT est seule utilisatrice de CP :

- après optimisation de certains paramètres de MVT
- " " " " de CP

Nous établirons un deuxième état de performances lorsque la machine virtuelle OS/MVT fonctionne en parallèle avec N autres machines virtuelles. Nous établirons enfin un état comparatif de performances avec le système de monoprogrammation OS/PCP.

Pour toutes les expériences, le système CP dispose de :

- 98 K de mémoire réelle
- une pile de disques 2314 (sur le canal 2) contenant les modules du système
- deux piles de disques 2314 (une sur le canal 2 l'autre sur le canal 3) pour les opérations de spooling.
- un tambour 2301 pour la pagination

Les autres unités à accès direct sont utilisées par le système OS/MVT.

La configuration de la machine virtuelle OS est la suivante :

- 1024 K de mémoire virtuelle
- 2 lecteurs de cartes dédiés, c'est-à-dire supportés par la machine virtuelle et non par CP.
- 2 imprimantes dédiées
- 1 tambour magnétique 2301 dédié
- 14 piles de disques 2314 dédiées

Les temps mesurés au cours des expériences sous CP sont les temps réels relevés sur un chronomètre :

T_s : temps réel de la session

T_e : temps réel d'exécution

T_r : temps réel de présence du reader

P A R T I E I

MACHINE VIRTUELLE OS/MVT SEULE

UTILISATRICE DE CP

Procédons en deux étapes :

. Tout d'abord, sans intervenir sur CP, déterminons les performances optimales de OS/MVT. Pour celà, CP ne s'occupe que de la seule machine virtuelle OS/MVT ce qui minimise les conflits au niveau de la mémoire et des canaux et intervenons sur la valeur des paramètres d'OS/MVT.

. Après avoir obtenu une meilleure performance, faisons varier les paramètres de CP, sans intervenir sur OS/MVT, et définissons ainsi la meilleure structuration des ressources de CP, pour atteindre le temps minimal d'exécution de notre batch test, et par déduction le meilleur environnement pour OS/MVT.

A - MODIFICATION DES PARAMETRES DE MVT

I - INFLUENCE DES MODULES RESIDENT EN MEMOIRE VIRTUELLE

1- Expérience

L'expérience AMAP (cf: CHAPITRE III) nous a permis de dresser la liste des modules très utilisés du système OS/MVT. Nous mesurons sous CP l'influence de leur implantation permanente en mémoire virtuelle (LPA étendue). Nous rappelons que la taille de l'ensemble de ces modules est de 22K.

Exécutons le batch - sans LPA étendue
- avec LPA étendue

Conditions de l'expérience

- Volume système résident sur disque
- 7 initiateurs
- SYS1.SYSJOBQE disque
- reader avec 1 buffer de 3200 octets

Résultats

- temps réels de l'expérience sans LPA étendue

$t_{s\ réel}$: 35 mn 55 s

$t_{e\ réel}$: 28 mn 45 s

$t_{r\ réel}$: 4 mn 50 s

- Temps réels de l'expérience avec LPA étendue occupant 22K de mémoire virtuelle

$t_{s\ réel}$: 33 mn 40 s

$t_{e\ réel}$: 26 mn 15 s

$t_{r\ réel}$:

Le fait de rendre résidents en mémoire virtuelle (dans la LPA) les modules déterminés par les expériences AMAP a permis d'obtenir un gain de temps d'exécution de :

2 mn 30 s soit un pourcentage de 8,7 %

2- Conclusions

Le pourcentage de gain de temps obtenu est nettement plus important que sur la machine réelle - 8,7 % au lieu de 2 %.
Le résultat s'explique par le fait qu'en OS/MVT sous CP, le chargement en mémoire d'un module par une opération d'E/S, à partir d'un disque est une opération couteûse puisque CP doit traduire les programmes canaux "virtuels" de OS/MVT en programmes canaux réels.

Il est donc évidemment intéressant de rendre résidents en mémoire de la machine virtuelle, les modules très utilisés du système OS/MVT puisqu'alors ils seront transférés par le mécanisme de pagination. De plus, le fait de rendre résidents des modules d'OS pour échanger des E/S sur disque contre de la pagination sur tambour est gravement défavorisé par le type de système de la mémoire en OS ("Content Supervisor" Ref.VIII)

basé sur une structure de listes telles que,pour retrouver un module en mémoire, il faut balayer une liste distribuée sur toute la mémoire, d'où une activité de pagination non négligeable.

Toutefois, en augmentant le nombre de modules résidents, on accroît la taille de la "Link Pack Area", et,de ce fait,l'on diminue la taille de la mémoire destinée aux utilisateurs.

II - INFLUENCE DU TYPE DE L'UNITE A ACCES DIRECT SUPPORTANT CERTAINS COMPOSANTS DU SYSTEME OS/MVT.

Il aurait été intéressant, pour étudier OS/MVT sous CP de faire varier l'implantation des data sets du système OS/MVT en les plaçant sur des unités à accès direct de type différent (disque ou tambour). Mais la version actuelle du système CP ne permet pas l'utilisation d'un tambour comme support de certains composants du système OS/MVT.

En particulier pour des problèmes de synchronisation et de vitesse :

- le chargement initial (IPL) est impossible à partir du tambour; il n'est donc pas possible d'utiliser le tambour comme volume système résident
- la file d'attente des travaux (SYS1.SYSJOBQE) ne peut non plus résider sur tambour.

Nous nous limitons donc à faire varier l'implantation du data set SYS1.LINKLIB de OS/MVT (disque ou tambour).

Pour cette expérience,un tambour est réservé à CP pour la pagination et peut contenir 900 pages de mémoire virtuelle (maximum de 900 x 4 = 3600K) ce qui est suffisant pour notre seule machine virtuelle. De ce fait, le tambour utilisé par OS ne peut ni détériorer les performances ni affecter le mécanisme de pagination de CP.

1- Expérience

Exécutons le batch avec SYS1.LINKLIB sur disque 2314 puis sur tambour 2301.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- avec SYS1.LINKLIB sur disque

$T_{session}$ = 33 mn 40 s

$T_{exécution}$ = 26 mn 15 s

T_{reader} =

- avec SYS1.LINKLIB sur tambour

$T_{session}$ =

$T_{exécution}$ = 23 mn 2 s

T_{reader} = 3 mn 57 s

2- Conclusions

En installant SYS1.LINKLIB sur tambour, le gain sur le temps d'exécution était, à priori, évident, mais il était nécessaire de le chiffrer (3 mn 13 s soit 12,3 %); en effet le contexte particulier de cette expérience:-1 seule machine virtuelle, 1 seul tambour pour la pagination et 1 tambour attribué à OS/MVT- peut nous servir de référence pour la

suite de nos différents travaux.

De plus, ce gain de temps est important pour le déplacement sur tambour, du seul data set SYS1.LINKLIB.

Sur la machine réelle, le déplacement sur tambour du volume système résident (SYS1.SVCLIB en particulier) a permis d'obtenir un gain de temps de 16 %.

On peut donc conclure que l'influence du type de l'unité à accès direct supportant les composants essentiels du système OS/MVT est plus importante dans un contexte de pagination que sur la machine réelle.

III - INFLUENCE DES PARAMETRES DU READER

1- Influence du type de l'unité d'entrée des JOBS

La comparaison sera faite entre une entrée des JOBS sur lecteur de cartes et sur lecteur de bande magnétique dédiés à la machine virtuelle.

Exécution du batch
- avec entrée sur carte
- avec entrée sur bande magnétique

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- avec entrée sur cartes

$T_{session}$ = 33 mn 13 s

$T_{exécution}$ = 26 mn 3 s

T_{reader} = 5 mn 3 s

- avec entrée sur bande magnétique

$T_{session}$ = 33 mn 40 s

$T_{exécution}$ = 26 mn 15 s

T_{reader} =

Sous CP le temps d'exécution du batch n'est pas amélioré par une lecture des JOBS sur une unité plus rapide (bande magnétique)

2- Influence de la taille des mémoires tampons du reader

Comme sur la machine réelle deux expériences ont été réalisées:

La première attribue une mémoire tampon sur reader; nous ferons varier sa taille .

La seconde attribue deux mémoires tampons, nous ferons également varier la taille de ces deux mémoires tampons.

a- Expérience 1

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- sans LPA étendue
- 7 initiateurs

Résultats avec une mémoire tampon

Taille de la mémoire tampon	$T_{session}$	$T_{exécution}$	T_{reader}
80	34mn 4s	31mn 8s	6mn 44s
400	33mn 10s	30mn	5mn 33s
1600	34mn	28mn 4s	5mn 6s
3200	35mn 55s	28mn 45s	4mn50s

Le temps d'exécution diminue régulièrement lorsque la taille de la mémoire tampon augmente de 80 à 3200 octets.

Le gain de temps obtenu est de :

2 mn 23 secondes soit 7,6 %

b- Expérience 2

Exécution du batch avec pour le reader deux mémoires tampons de successivement 80, 400, 1600, 3200 octets.

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- sans LPA étendue
- 7 initiateurs

Résultats avec deux mémoires tampon

Taille de chacune des 2 mémoires tampons	$T_{session}$	$T_{exécution}$	T_{reader}
80	35mn 50s	31mn 30s	
400	34mn 35s	30mn 45s	6mn 15s
1600		31mn 3s	5mn 20s
3200		28mn 46s	5mn 7s

Le temps d'exécution diminue régulièrement lorsque la taille de chaque mémoire tampon augmente de 80 à 3200 octets.

Le gain de temps obtenu est de :

2 mn 44 s soit de 8,7 %

Pour l'ensemble des deux expériences, le gain de temps obtenu est donc en moyenne de 8,2 % lorsque la taille de la mémoire tampon passe de 80 à 3200 octets.

3- Influence du nombre de mémoires tampons du reader

Réalisons deux expériences :

Expérience 1 : la taille de la mémoire tampon étant fixée à 80 octets, attribuons successivement : 1, 2, 4, 6, 10 mémoires tampons au reader.

Expérience 2 : la taille de la mémoire tampon étant successivement de 400, 1600, 3200 octets, attribuons une puis deux mémoires tampons au reader.

a- Expérience 1

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- sans LPA étendue
- 7 initiateurs

Résultats avec mémoire tampon de 80 octets

nombre de mémoires tampons	$T_{session}$	T Exécution	T reader
1	34mn 4s	31mn 8s	6mn 44s
2	35mn 50s	31mn 30s	
4	34mn 27s	30mn 34s	5mn 52s
6	34mn 45s	29mn 20s	5mn 42s
10	33mn 18s	29mn 22s	5mn 49s

Le temps d'exécution diminue régulièrement lorsque l'on passe de 1 à 10 mémoires tampons de 80 octets :

Gain de temps : 1mn 46s soit 5,7 %

b- Expérience 2

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- sans LPA étendue
- 7 initiateurs

Résultats

Mémoire tampon	$T_{session}$	$T_{exécution}$	T_{reader}
1 x 400	33mn 10s	30 mn	5mn 33s
2 x 400	34mn 35s	30mn 45s	6mn 15s
1 x 1600	34mn	28mn 4s	5mn 6s
2 x 1600		31mn 3s	5mn 20s
1 x 3200		28mn 45s	4mn 50s
2 x 3200		28mn 46s	5mn 7s

Le temps d'exécution augmente faiblement lorsque le reader dispose de deux mémoires tampons au lieu d'une.

4- Influence du chaînage de la lecture

Deux expériences ont été réalisées comme sur la machine réelle.

La première utilise le lecteur de cartes (dédié) pour l'entrée des JOBS.

La seconde utilise le dérouleur de bande magnétique.

a- Expérience 1

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE sur disque
- avec LPA étendue
- 7 initiateurs
- reader avec un buffer de 3200 octets

Résultats avec entrée des JOBS au lecteur de cartes

- sans chaînage de la lecture

$T_{session}$ = 33 mn 13 s

$T_{exécution}$ = 26 mn 3 s

T_{reader} = 5 mn 3 s

- avec chaînage de la lecture

$T_{session}$ =

$T_{exécution}$ = 26 mn

T_{reader} = 4 mn 50 s

Le gain de temps d'exécution est négligeable avec chaînage de la lecture.

b- Expérience 2

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initateurs
- reader avec 1 buffer de 3200 octets

Résultats avec entrée des JOBS au lecteur de cartes

- sans chaînage de la lecture

$T_{session}$ = 33 mn 40 s

$T_{exécution}$ = 26 mn 15 s

T_{reader} =

- avec chaînage de la lecture

$T_{session}$ =

$T_{exécution}$ = 25 mn 40 s

T_{reader} = 3 mn 43 s

Avec chaînage de la lecture le gain de temps d'exécution est de 35 s soit de 2 %.

5- Conclusions

Comparons ces résultats à ceux obtenus sur la machine réelle :

- Le type de l'unité d'entrée des JOBS, qui a une influence importante sur le temps d'exécution, sur la machine réelle (8,3 % de gain dans l'expérience réalisée), n'en a pas sous CP. Ceci peut s'expliquer par le fait que sous CP, le temps CPU passé pour la recompilation des programmes canaux permettant la lecture des JOBS est important par rapport au temps CPU utilisé par le reader.
  Nous serons donc conduits à déterminer le temps moyen passé, au cours de la session, pour la recompilation des programmes canaux.

- Augmenter la taille de la mémoire tampon du reader a les mêmes avantages sous CP que sur la machine réelle. (gain de temps d'exécution de l'ordre de 8 % en passant de 80 à 3200 octets).
  Ceci s'explique donc par la réduction du nombre d'entrées-sorties pour la mise sur unité à accès direct et la relecture des data sets inclus dans le flot d'entrée des JOBS (blocage des enregistrements).
  Mais ce nombre d'entrées-sorties est cependant faible par rapport au nombre d'entrées-sorties total réalisées au cours de la session, ce qui explique que sous CP une amélioration supplémentaire n'est pas obtenue.
  Nous avons vu qu'augmenter la taille de la mémoire tampon, utilisée par le reader, entraine une perte de place en mémoire principale.
  Cette perte de place n'est plus un inconvénient, puisque sous CP, la mémoire virtuelle est de taille moins limitée. Cette solution est donc très avantageuse.

- Augmenter le nombre de mémoires tampons du reader n'a pas d'intérêt sous CP, le ralentissement étant principalement dû à la réalisation

des opérations d'entrée-sortie donc à leur nombre.
Il est donc préférable d'augmenter la taille de la mémoire tampon du reader plutôt que le nombre de ces mémoires tampons.

- Le chaînage de la lecture n'apporte sous CP qu'une amélioration très faible (2%). Si la charge apportée par le reader au système OS/MVT est, avec chaînage de la lecture, légèrement plus faible, le nombre d'opérations d'entrée/sortie réel à effectuer pour la lecture des travaux reste le même. Ce qui peut expliquer une influence plus faible, sous CP, du chaînage de la lecture.

IV - INFLUENCE DU NOMBRE D'INITIATEURS

La machine virtuelle ayant une taille de 1024K, les 7 JOBS du test peuvent être exécutés en parallèle.

1- Expérience

Exécution du batch avec successivement 1, 2, 3...7 initiateurs.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- reader avec 1 buffer de 3200 octets

Résultats : temps réels

Nbre d'initiateurs	$T_{session}$	$T_{exécution}$	T_{reader}
1	32mn 45s	31mn 25s	2mn 5s
2	27mn 59s	27mn 39s	3mn 9s
3	27mn 15s	26mn 15s	2mn 55s
4	27mn 33s	26mn 25s	3mn 53s
5	27mn 35s	26mn 15s	3mn 50s
6	34mn 35s	26mn 35s	4mn 10s
7	33mn 40s	26mn 15s	

Le tableau ci-dessous donne le pourcentage de gain de temps d'exécution obtenu par rapport au temps d'exécution avec un seul initiateur.

Nombre d'initiateurs	Gain de $T_{exécution\ réel}$	% de gain
2	3mn 46s	12 %
3	5mn 10s	16,4 %
4	5mn	16 %
5	5mn 10s	16,4 %
6	4mn 50s	15,4 %
7	5mn 10s	16,4 %

Le passage de 1 à 3 initiateurs améliore nettement le temps d'exécution réel du batch.

Le passage de 3 à 7 initiateurs n'améliore pas le temps d'exécution réel du batch.

2- Conclusions

En comparant ces résultats aux résultats obtenus sur la machine réelle nous remarquons :

- qu'en faisant passer le nombre d'initiateurs de 1 à 3, le pourcentage de gain de temps d'exécution obtenu est de 16,4 % sous CP au lieu de 37 % sur la machine réelle (donc 2 fois moins important).
  Ce résultat s'explique par le fait qu'une grande partie du temps est passé dans CP pour le compte de la machine virtuelle. CP ralentit donc l'effet de la multiprogrammation.

- Une même stabilité du temps d'exécution au delà de 3 initiateurs. Là encore la prépondérance des entrées/sorties qui se traduisent sous CP par la recompilation des programmes canaux, peut expliquer cette stabilité.

Les avantages de la multiprogrammation sont donc nettement réduits sous CP.

Cette conclusion nous conduit à l'étude d'un système OS de monoprogrammation PCP dans ce même contexte de pagination : (cf CHAPITRE 5)

V - CONCLUSIONS

1- Tableau récapitulatif des résultats obtenus séparément

Modification apportée	Gain de temps d'exécution obtenu
- 22K de modules résidents supplémentaires	8,7 %
- SYS1.LINKLIB sur tambour magnétique 2301	12,3 %
- entrée des JOBS sur lecteur de bande magnétique au lieu du lecteur de cartes	0 %
- mémoire tampon de 3200 octets pour le reader au lieu de 80 octets	8,2 %
- chaînage de la lecture pour le reader	1 %
- passage de 2 à 7 initiateurs	12 %

2- Choix des valeurs optimales des paramètres de MVT étudiés

Ces valeurs sont les suivantes :

- 22K de modules résidents supplémentaires en LPA
- data set SYS1.LINKLIB sur tambour 2301
- 7 initiateurs actifs
- reader, avec 1 mémoire tampon de 3200 octets
- avec chaînage de la lecture (OPTCD=C)
- entrée des JOBS (input stream) sur lecteur de cartes ou lecteur de bande magnétique dédié à la machine virtuelle OS/MVT.

3- Gain de temps d'exécution réel obtenu

Le temps d'exécution du batch OS sous CP était de 31 mn 30 s dans les conditions suivantes :

- data set SYS1.LINKLIB sur disque 2314
- pas de module résident
- 7 initiateurs actifs
- reader avec 2 mémoires tampons de 80 octets
- sans chaînage de la lecture
- avec entrée des JOBS sur bande magnétique

Avec le système amélioré, ce temps d'exécution a été de :

23 mn 2 s

Soit un gain de temps de: 8 mn 28 s soit de 27 %

4- Remarque: influence de CP sur l'ordre d'activation des JOBS

Dans un système de multiprogrammation, les JOBS sont activés en fonction :

- de leur "dispatching priorité" : chaînage des TCB
- des opérations d'entrée-sortie qu'ils réalisent (tâches en attente, tâche prête, tâche active)
- de leur classe

Le problème est de savoir si CP a une influence sur l'activation des JOBS en multiprogrammation.

Trois expériences ont été réalisées dans les mêmes conditions, en OS/MVT stand alone, et en OS/MVT sous CP.

- Volume système résident sur disque
- sans LPA étendue
- 7 initiateurs
- SYS1.SYSJOBQE sur disque

L'ordre de démarrage des JOBS est le suivant :

BENCH11

	Classe
BENCH11	B
BENCHO8	A
BENCHO3	A
BENCHO1	A
BENCH12	B
BENCH16	C
BENCH10	C

Ordre de fin d'exécution en OS/MVT stand alone et en OS/MVT sous CP

Expérience 1 : Exécution du batch avec reader de 2 buffers de 80 octets :

Ordre de fin OS	Ordre de fin OS/CP
BENCHO8	BENCH10
BENCH12	BENCHO8
BENCH10	BENCHO1
BENCH16	BENCH16
BENCHO1	BENCH11
BENCH11	BENCH12
BENCHO3	BENCHO3

Expérience 2 : Exécution du batch avec reader de 2 buffers de 400 octets

Ordre de fin OS	Ordre de fin OS/CP
BENCHO8	BENCH10
BENCH12	BENCHO8
BENCH10	BENCHO1
BENCH16	BENCH16
BENCHO1	BENCH11
BENCH11	BENCH12
BENCHO3	BENCHO3

Expérience 3 : Exécution du batch avec reader de 1 buffer de 3200 octets

Ordre de fin OS	Ordre de fin OS/CP
BENCH12	BENCH10
BENCHO8	BENCHO8
BENCH10	BENCHO1
BENCH16	BENCH11
BENCHO3	BENCH12
BENCH11	BENCH16
BENCHO1	BENCHO3

Portons sur un graphique pour chaque expérience, le temps relatif de démarrage (STARTED) de chaque JOB et le temps relatif de fin (ENDED).

Ceci en - OS/MVT stand alone
- OS/MVT sous CP (temps réel)

Relevons pour chaque JOB, le temps CPU

- en OS/MVT seul
- en OS/MVT sous CP (temps CPU + OVERHEAD)

Relevons pour chaque JOB le temps d'exécution :

- en OS/MVT seul
- en OS/MVT sous CP (temps réel)

RDR = 2 BUFFERS DE 80 - 7 INITIATEURS

ORDRE DE FIN D'EXECUTION	OS	08	12	10	16	01	11	03
	OS/CP	10	08	01	16	11	12	03

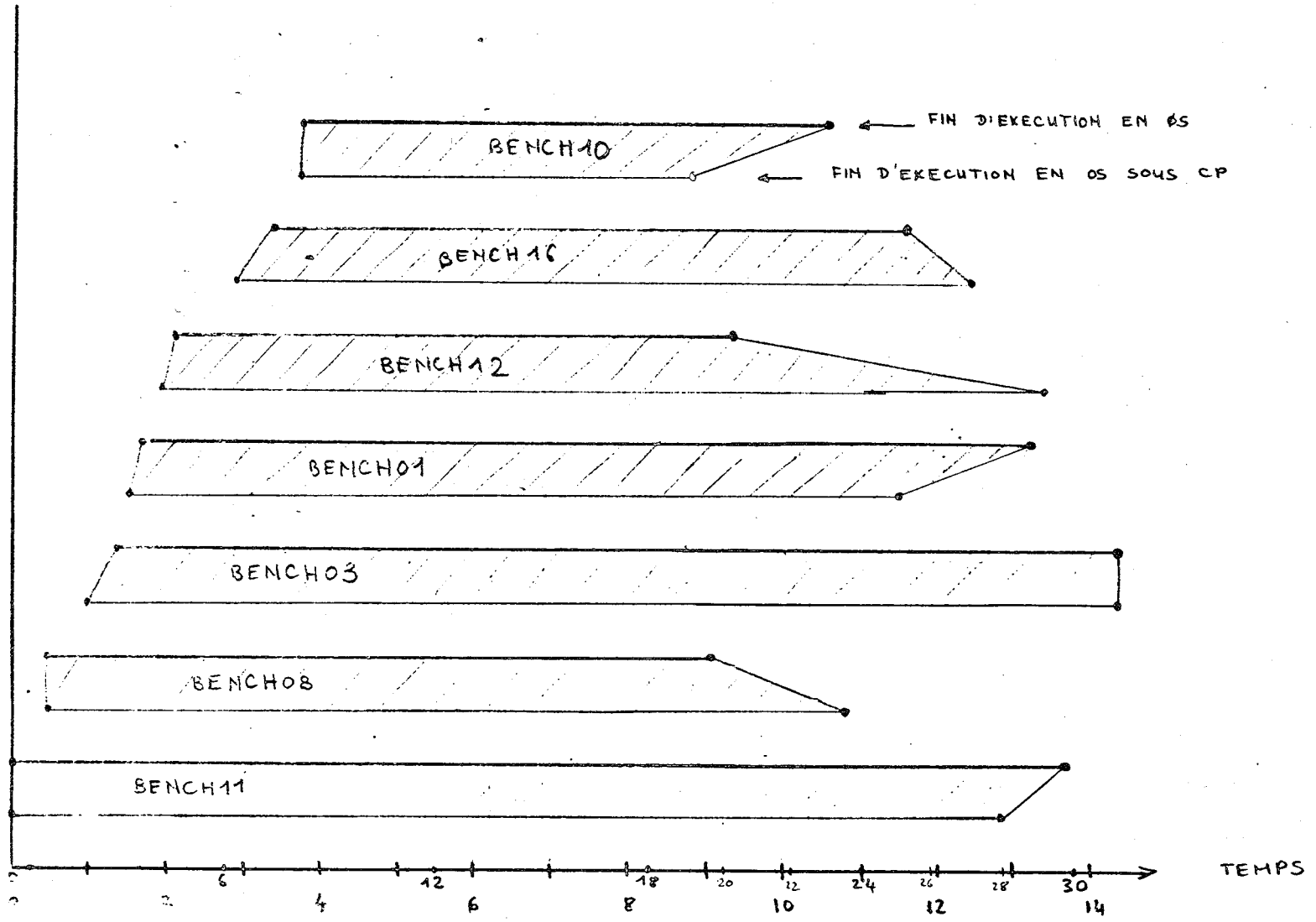


FIGURE : 8

RDR = 2 BUFFERS DE 400 7 INITIATEURS

ORDRE DE FIN D'EXECUTION

OS	12	08	10	16	01	11	03
OS / CP	10	08	01	12	11	16	03

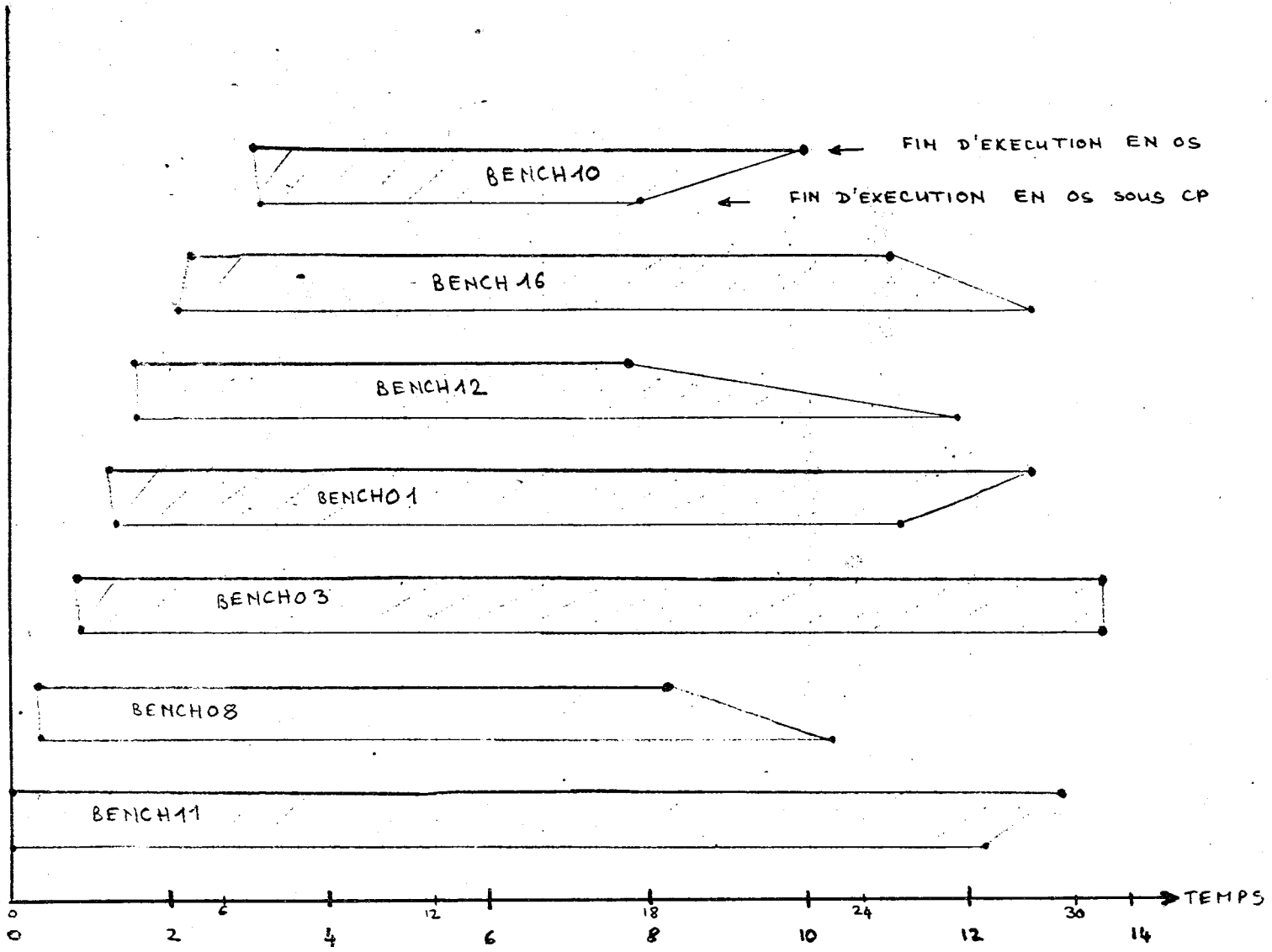


FIGURE : 9

RDR = 1 BUFFER DE 3200 — 7 INITIATEURS

ORDRE DE FIN D'EXECUTION								
	OS	12	08	10	16	03	11	01
	OS/CP	10	08	01	11	12	16	03

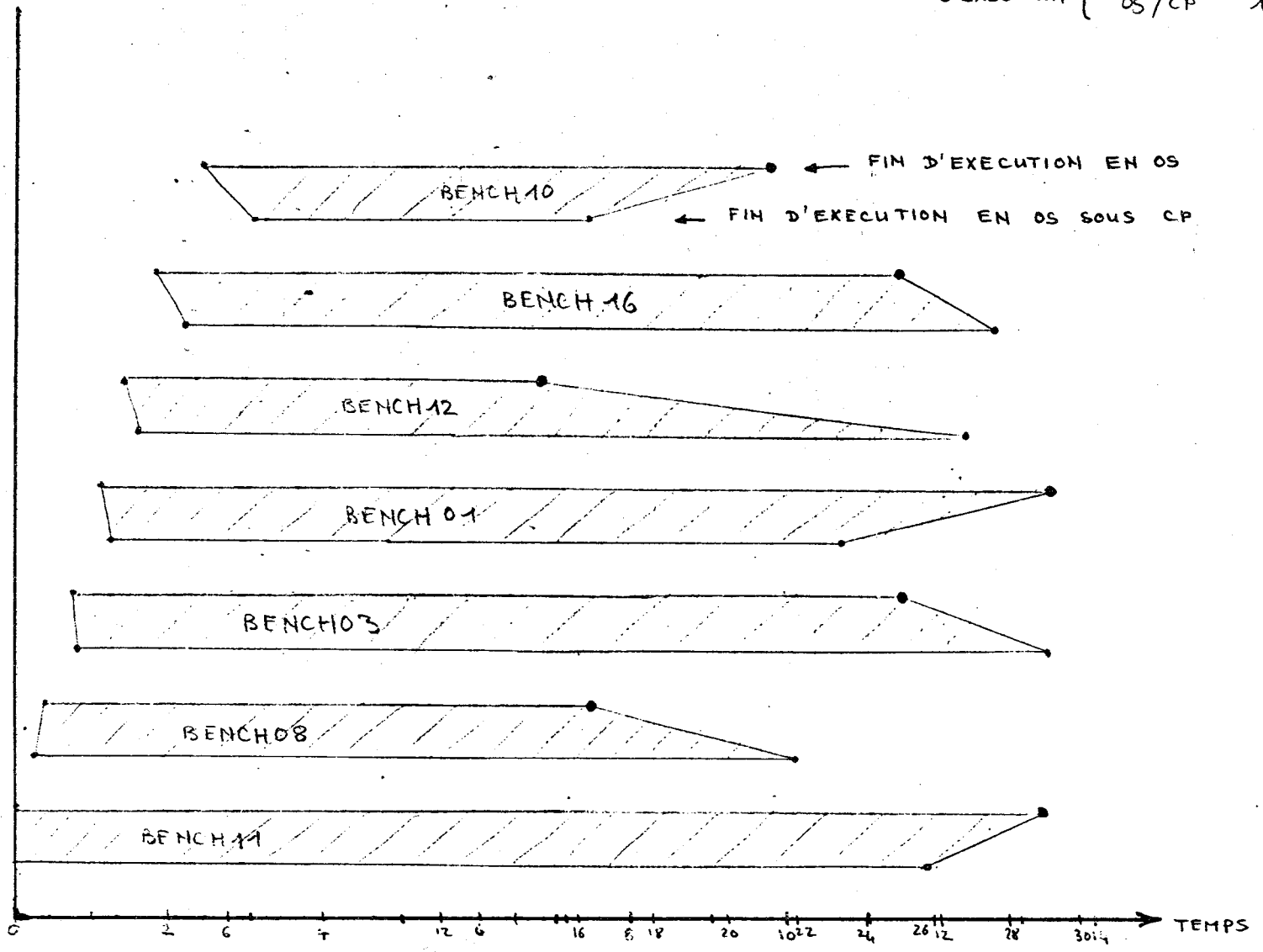


FIGURE : 10

BENCH OS/MVT SEUL

TEMPS D'EXECUTION (ENDED-STARTED)

Nom programme step Classe		Taille demandée	Taille utile	RDR 2 x 80	RDR 2 x 400	RDR 1 x 3200	Temps moye
BENCH11	ALGOL	180	178	13mn 44	13mn 10s	13mn 10s	13mn 21s
B	GO	d	100				
BENCH08	COB	100	100	8mn 36s	7mn 52s	7mn 2s	7mn 50s
A	LKED	96	96				
	GOTRI	d	94				
	IEBPTPCH	170	62				
BENCH03	COB	100	100	13mn 6s	12mn 53s	10mn 40s	12mn 13
A	LKED	96	96				
	GO	70	62				
BENCH01	IEBGENER	70	62	11mn 39s	11mn 30s	12mn 10s	11mn 46
A	COBO	120	112				
BENCH12	ASM	50	70	7mn 10s	6mn 15s	5mn 24s	6mn 16
B	LKED	96	96				
	GO	70	62				
BENCH16	FORT	100	86	8mn 14s	8mn 48s	9mn 33s	8mn 51
C	LKED	96	96				
	GO	70	64				
BENCH10	FORT	70	64	6mn 49s	8mn 49s	7mn 16s	7mn 38
C							
SESSION				14mn 24s	13mn 44s	13mn 13s	13mn 47

d = 100K région par défaut mise par le READER

BENCH OS/MVT sous CP

TEMPS D'EXECUTION (ENDED-STARTED)

Nom programme step Classe		Taille demandée	Taille utile	RDR 2 x 80	RDR 2 x 400	RDR 1 x 3200	Temps moyen
BENCH11	ALGOL	180	178	10mn 56s	10mn 50s	10mn 47s	10mn 51s
B	GO	d	100				
BENCH08	COB	100	100	9mn 9s	9mn 3s	8mn 50s	9mn
A	LKED	96	96				
	GOTRI	d	94				
	IEBPTPCH	170	62				
BENCH03	COB	100	100	11mn	11mn 2s	10mn 55s	10mn
A	LKED	96	96				
	GO	70	62				
BENCH01	IEBGENER	70	62	9mn 35s	9mn 30s	9mn 8s	9mn 23s
A	COBO	120	112				
BENCH12	ASM	50	70	10mn 32s	10mn 12s	10mn 25s	10mn 23s
B	LKED	96	96				
	GO	70	62				
BENCH16	FORT	100	86	10mn 08s	10mn 22s	10mn 20s	10mn 17s
C	LKED	96	96				
	GO	70	64				
BENCH10	FORT	70	64	6mn 33s	6mn 39s	6mn 28s	6mn 33s
C							
SESSION				11mn 17s	11mn 17s	11mn 8s	11mn 14s

d = 100K région par défaut mise par le READER

BENCH OS/MVT sous CP

TEMPS D'EXECUTION REEL (ENDED réel - STARTED réel)

Nom programme step Classe		Taille demandée	Taille utile	RDR 2 x 80	RDR 2 x 400	RDR 1 x 3200	Temps moyen
BENCH11	ALGOL	180	178	28mn 10s	27mn 20s	25mn 50s	27mn 7s
B	GO	d	100				
BENCH08	COB	100	100	22mn	22mn 30s	21mn 10s	21mn 53s
A	LKED	96	96				
	GOTRI	d	94				
	IEBPTPCH	170	62				
BENCH03	COB	100	100	28mn 30s	28mn 45s	27mn 05s	28mn 7s
A	LKED	96	96				
	GO	70	62				
BENCH01	IEBGENER	70	62	22mn	22mn 5s	20mn 35s	21mn 33s
A	COBO	120	112				
BENCH12	ASM	50	70	25mn	23mn 10s	23mn 25s	23mn 51s
B	LKED	96	96				
	GO	70	62				
BENCH16	FORT	100	86	21mn 40s	23mn 20s	22mn 55s	22mn 38s
C	LKED	96	96				
	GO	70	64				
BENCH10	FORT	70	64	10mn 25s	10mn 45s	9mn 35s	10mn 11s
C							
SESSION				31mn 30s	30mn 35s	28mn 45s	30mn 20s

d = 100K région par défaut mise par le READER

BENCH OS/MVT SEUL

TEMPS CPU

Nom programme step Classe		Taille demandée	Taille utile	RDR 2 x 80	RDR 2 x 400	RDR 1 x 3200	Temps moyen
BENCH11	ALGOL	180	178	4s 87	4s 69	4s 69	2mn
B	GO	d	100	1mn 56s 82	1mn 57s 10	1mn 57s 47	
BENCH08	COB	100	100	2s 01	1s 95	2s 02	8s
A	LKED	96	96	0s 43	0s 52	0s 51	
	GOTRI	d	94	5s 51	5s 54	5s 54	
	IEBPTPCH	170	62				
BENCH03	COB	100	100	9s 78	9s 26	9s 24	12s
A	LKED	96	96	0s 81	0s 79	0s 79	
	GO	70	62	1s 99	1s 93	2s 08	
BENCH01	IEBGENER	70	62	0s 65	0s 66	0s 71	15s
A	COBO	120	112	14s 21	14s 34	14s 53	
BENCH12	ASM	50	70	9s 31	9s 44	9s 44	10s
B	LKED	96	96	0s 37	0s 34	0s 34	
	GO	70	62	0s 14	0s 14	0s 13	
BENCH16	FORT	100	86	2s 18	2s 24	2s 25	4s
C	LKED	96	96	1s 32	1s 23	0s 26	
	GO	70	64	1s 15	1s 20	1s 15	
BENCH10	FORT	70	64	5mn 1s 30	5mn 5s 63	5mn 7s 58	5mn 5s
C							

d = 100K région par défaut mise par le READER

BENCH OS/MVT sous CP

TEMPS CPU

Nom programme step Classe		Taille demandée	Taille utile	RDR 2 x 80	RDR 2 x 400	RDR 1 x 3200	Temps moye
BENCH11	ALGOL	180	178	6s 85	6s 42	5s 81	2mn 45s
B	GO	d	100	2mn 40s 59	2mn 40s 51	2mn 35s 41	
BENCHO8	COB	100	100	3s 75	3s 50	3s 22	
A	LKED	96	96	1s 44	1s 60	1s 66	10s
	GOTRI	d	94	6s 97	6s 80	6s 69	
	IEBPTPCH	170	62				
BENCHO3	COB	100	100	13s 84	13s 73	13s 14	
A	LKED	96	96	1s 84	1s 78	1s 50	17s
	GO	70	62	1s 87	1s 74	0s 88	
BENCHO1	IEBGENER	70	62	0s 99	1s 08	0s 97	16s
A	COBO	120	112	14s 71	13s 88	16s 58	
BENCH12	ASM	50	70	13s 79	13s 77	13s 90	
B	LKED	96	96	0s 59	0s 69	0s 58	14,5s
	GO	70	62	0s 13	0s 07	0s 08	
BENCH16	FORT	100	86	3s 84	2s 78	3s 94	
C	LKED	96	96	2s 86	2s 64	2s 10	7s
	GO	70	64	0s 61	1s 43	1s 26	
BENCH10	FORT	70	64	5mn 47s 5	5mn 46s 65	5mn 45s 72	5mn 46s
C							

d = 100K région par défaut mise par le READER

Reportons dans un tableau le pourcentage du temps d'exécution de chaque JOB par rapport au temps d'exécution du batch :

Nom du programme	Activité I/O	% en OS MVT seul	% en OS MVT sous CP	Différence
BENCH11	grande	97 %	90 %	-7 %
BENCH08	moyenne	57 %	72 %	+15 %
BENCH03	moyenne	89 %	93 %	+4 %
BENCH01	très grande	85 %	71 %	-14 %
BENCH12	faible	45 %	79 %	+34 %
BENCH16	assez faible	64 %	75 %	+11 %
BENCH10	très faible	55 %	33,5 %	-21 %

(1) Les JOBS ayant une grande activité d'entrée-sortie (BENCH11, BENCH01) se terminent plus rapidement en OS/MVT sous CP.

(2) Les JOBS ayant une faible activité d'entrée-sortie (BENCH12, BENCH16) se terminent moins rapidement en OS/MVT sous CP. Ceci est dû au verrouillage des pages au profit des jobs limités par I/O qui de ce fait défavorisent les jobs limités par CPU.

B - MODIFICATION DES PARAMETRES DE CP

Les paramètres du système OS/MVT ayant été choisis pour un fonctionnement sous CP, nous allons étudier l'influence de certains paramètres de CP sur le comportement du système OS/MVT :

- variation de la configuration de la machine réelle et de la machine virtuelle
  - . taille mémoire réelle
  - . taille mémoire virtuelle
  - . utilisation du lecteur et imprimante virtuels

- gestion privilégiée de la machine virtuelle OS/MVT
  - . verrouillage (c'est-à-dire installation permanente) de pages en mémoire réelle

I - INFLUENCE DE LA TAILLE DE LA MEMOIRE REELLE ET DE LA MEMOIRE VIRTUELLE

1- Expérience

Fonctionnant sous CP, le système OS/MVT peut disposer d'une mémoire virtuelle de taille variable, et qui peut être supérieure à la taille de la mémoire réelle de l'ordinateur.

Une étude a été réalisée à l'IMAG pour déterminer le meilleur rapport entre la taille de la mémoire réelle de l'ordinateur et la taille

de la mémoire virtuelle de la machine OS.

Nous ne reporterons donc ici qu'un tableau récapitulatif des résultats obtenus.

Conditions des expériences

Les expériences ont été réalisées alors que CP ne supportait qu'un seul utilisateur: la machine OS fonctionnant avec le système OS/MVT version 15/16.

- Variation de la taille de la mémoire réelle
  Cette variation a été obtenue par une modification apportée à la séquence d'IPL de CP, qui permet à l'opérateur d'indiquer la taille désirée de la mémoire réelle.

- Variation de la taille de la mémoire virtuelle
  La machine virtuelle OS disposait d'une mémoire virtuelle de successivement :
  - 512K
  - 1024K
  - 2048K
  - 3072K

Résultats des expériences avec notre batch test

Taille mémoire virtuelle	Taille mémoire réelle	Conditions	Temps Exécution	% wait
512K	512K	reader: 2 buffers de 400 octets 3 initiateurs(100K)	32 mn	28,59
	384K		40 mn	34,90
	256K		52 mn	
	200K		85 mn	19,17
1024K	512K	reader: 4 buffers de 400 octets 4 initiateurs(180K)	30 mn	40,90
	320K		35 mn	29,18
	200K		82 mn	
2048K	512K	reader: 8 buffers de 400 octets 5 initiateurs(290K)	29 mn 30 s	27,84
	328K		37 mn	
	256K			
	216K		68 mn	
3072K	512K	reader: 16 buffers de 400 octets 7 initiateurs(290K)	31 mn	19,27
	384K		36 mn	20
	256K		57 mn	

2- Conclusions

Portons sur un graphique, pour chaque taille de mémoire virtuelle, le temps d'exécution du batch par rapport à la taille de la mémoire réelle de la machine :

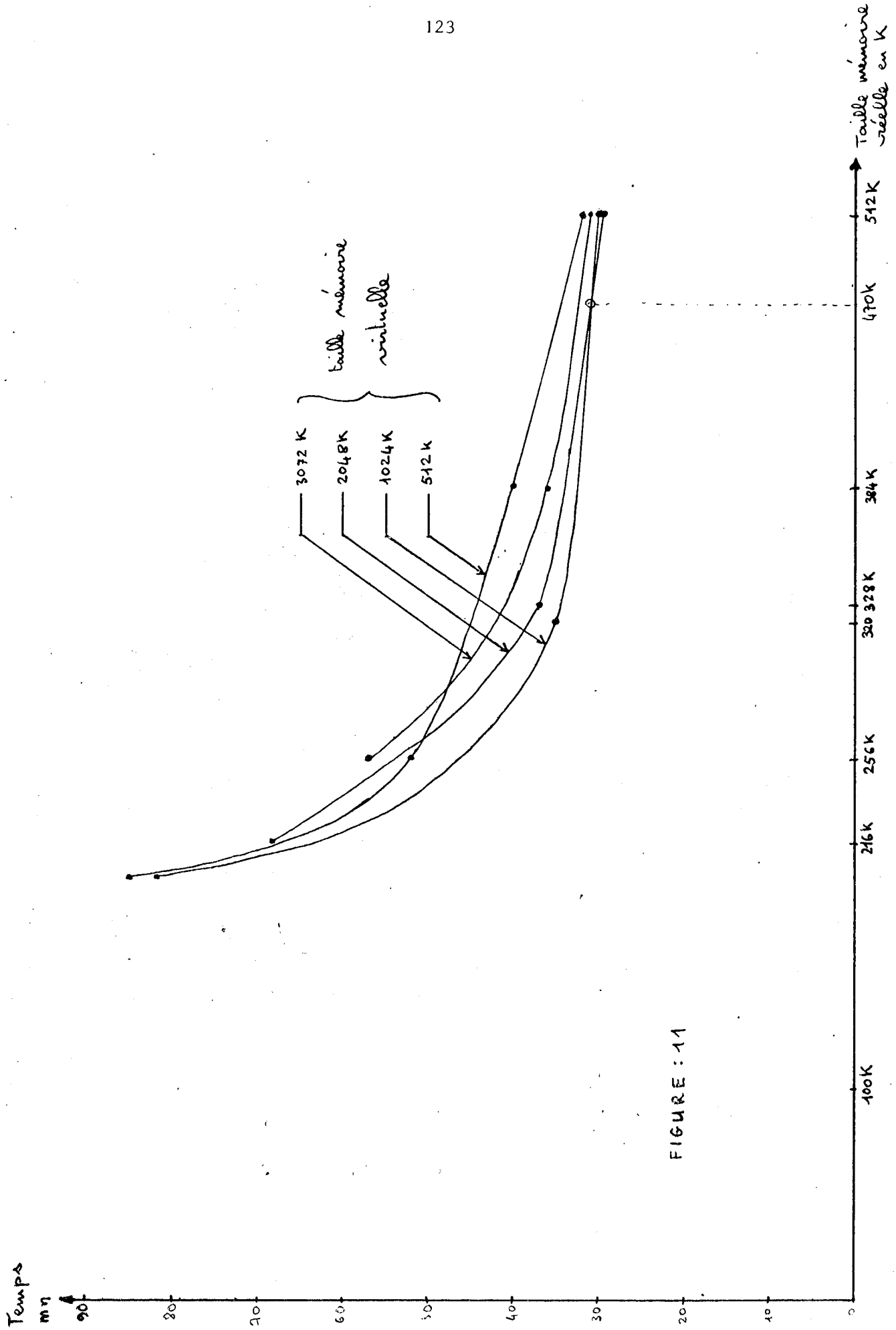

Temps
mn
90
80
70
60
50
40
30
20
10
0
100K
216K
256K
320 328K
384K
470K
512K
Taille mémoire réelle en K
3072 K
2048K
1024K
512 k
taille mémoire initielle

FIGURE : 11

Pour une taille de mémoire réelle inférieure à 470K, le meilleur temps d'exécution du batch a été obtenu avec une mémoire virtuelle de 1024K.

Pour une taille de mémoire réelle de 512K, le meilleur temps d'exécution du batch a été obtenu avec une mémoire virtuelle de 2048K, le moins bon avec une mémoire virtuelle de 512K :

Mémoire réelle 512K
Mémoire virtuelle 512K
→ $T_{exécution}$ = 32 mn

Mémoire réelle 512K
Mémoire virtuelle 2048K
→ $T_{exécution}$ = 29 mn 30 s

Soit 2 mn 30 s de gain de temps d'exécution, donc un pourcentage de gain de 7,8 %.

Les détails de ces expériences sont décrits dans l'article référencé en bibliographie I.

Nous pouvons cependant noter que l'utilisation d'une mémoire virtuelle de taille plus importante a permis de réduire le nombre d'opérations d'entrées-sorties réalisées par le système OS/MVT :(par une augmentation de la taille des blocs traités par le reader, et une augmentation de la région utilisée par les JOBS) donc d'améliorer le temps d'exécution.

II - UTILISATION DU LECTEUR DE CARTES ET DE L'IMPRIMANTE VIRTUELLE

La configuration de la machine virtuelle OS comprend, entre autres unités virtuelles : un lecteur de cartes et une imprimante virtuels. Ces 2 unités sont en fait matérialisées par une zone de disque dans un mécanisme de "spooling". De cette façon,CP lui-même effectue la lecture des JOBS sur un lecteur de cartes réel et crèe le fichier de spooling "lecteur virtuel" pour la machine OS; de même,les résultats destinés à l'impression sont regroupés dans le fichier de spooling "imprimante virtuelle" puis envoyés par CP sur l'imprimante réelle.

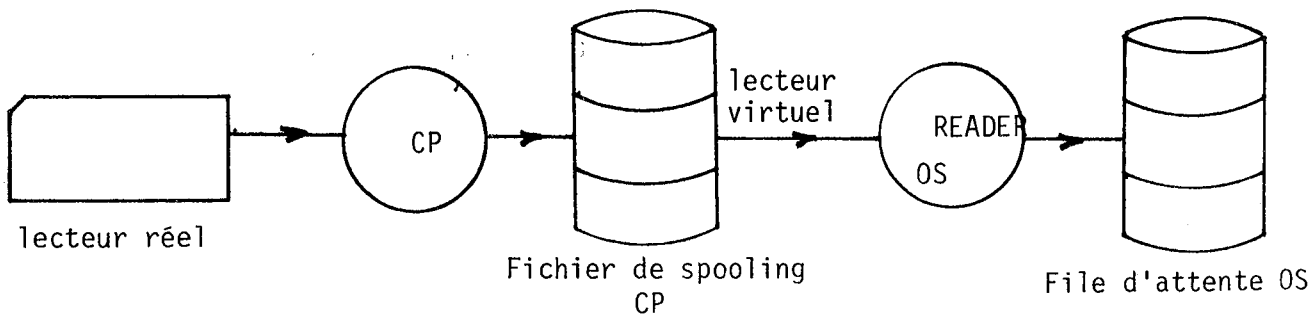
Intérêt de cette solution

Les JOBS peuvent être lus par CP sans que la machine virtuelle OS soit active. Il en est de même pour les sorties sur imprimante qui peuvent être stockées par CP et imprimées réellement sans la présence de la machine OS.

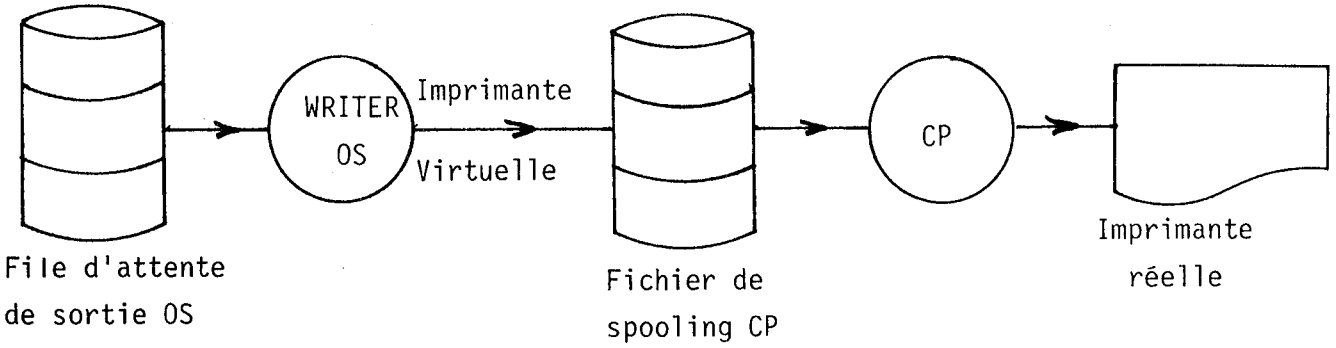
Inconvénient de cette solution

Elle implique un double transfert des fichiers d'entrée et de sortie :

En entrée : - envoi des JOBS dans le spooling de CP
- lecture du fichier de spooling CP par le reader de OS/MVT et création de la file d'attente OS :



En sortie : - lecture des files d'attente de sortie et transfert, par le writer OS/MVT, des résultats dans le spooling de CP.
- envoi des résultats sur imprimante réelle.



Il est aussi possible d'attacher à une machine virtuelle des organes périphériques réels donc de remplacer un "lecteur virtuel" par un lecteur réel et une "imprimante virtuelle" par une imprimante réelle. Dans ce cas, la machine OS s'occupe elle-même des entrées et des sorties sans intervention de CP (lecteurs et imprimantes dédiées à la machine OS utilisés dans la partie A) donc supprime le spooling de CP.

L'expérience suivante détermine l'influence des lecteurs et imprimantes sur le temps d'exécution du batch.

1- Expérience

Exécution du batch avec :

- lecteur + imprimante réelle
- lecteur virtuel, imprimante réelle
- lecteur réel, imprimante virtuelle
- lecteur et imprimante virtuels

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

	$T_{session}$	$T_{exécution}$	T_{reader}
lecteur réel impr.réelle	33mn 40s	26mn 15s	4mn
lecteur virtuel Impr.réelle	32mn 35s	26mn 5s	3mn 42s
lecteur réel impr.virtuelle	28mn 40s	27mn 10s	4mn 15s
lecteur virtuel impr.virtuelle	28mn 20s	26mn 37s	3mn 30s

2- Conclusions

L'utilisation du lecteur virtuel améliore le temps d'exécution du batch de :

10s soit ≃ 0,6 %

et

33s soit ≃ 2 %

Cette amélioration est dûe au fait que le type de l'unité d'entrée des JOBS n'est plus un lecteur de cartes mais un disque 2314.

L'utilisation de l'imprimante virtuelle augmente le temps d'exécution du batch de :

55s soit 3 %

et 32s soit 2 %

Cette perte de performance est dûe au fait que le "writer" OS/MVT doit véhiculer les résultats du spooling OS dans le spooling CP. Cette recopie augmente les conflits au niveau des canaux, donc le temps réel d'exécution augmente. Cependant, le temps réel de la session diminue considérablement (13 %) puisque les résultats restent dans le spooling de CP. Cette solution pourra être adoptée si l'on diffère la sortie des résultats de l'exécution des JOBS sur l'imprimante réelle.

3- Remarque: suppression du spooling de OS/MVT en sortie

Il est aussi possible de supprimer le spooling du système OS en dirigeant directement les sorties de résultats sur les imprimantes (virtuelles sous CP) :

(//ddname DD UNIT=1403)

Cette solution est cependant difficilement utilisable en exploitation car elle nécessite la maintenance d'un système spécialisé avec :

- Définition d'un grand nombre d'unités d'impressions pour absorber le flot continu des résultats (occasionnné par la multiprogrammation)

- mise en place de procédures spécialisées pour orienter et ventiler correctement ces résultats sur les différentes imprimantes.

Une expérience réduite a été réalisée pour montrer l'intérêt de cette solution :

Exécutons le programme BENCH11 (imprimant une grande quantité de résultats: 2240 lignes) dans différentes conditions :

		Temps d'exécution	Temps d'exécution + Temps d'impression
Sortie des résultats par le WTR	sur imprimante réelle	5mn 40s	11mn 32s
	sur imprimante virtuelle sans impression réelle	5mn 42s	7mn 12s
Sortie directe des résultats	sur imprimante réelle	7mn 47s	9mn 14s
	sur imprimante virtuelle sans impression réelle	5mn 33s	7mn 5s

Nous notons une amélioration du temps total (exécution + impression) lorsque les résultats sont envoyés directement sur imprimante virtuelle :

1,6 %

Le temps d'exécution étant amélioré de :

2,7 %

Cette solution a cependant des inconvénients et ne permet pas le changement:

- de papier
- de chaîne
- de bande pilote

NOTE : Bien que la solution ne soit pas réalisable pour une exploitation (affecter une imprimante réelle à un fichier de sortie) il est aussi possible de sortir les résultats directement sur imprimante réelle : dans ce cas particulier nous notons en supprimant le spooling d'OS :

gain temps total (exécution + impression) : 20 %
perte temps d'exécution : ,27 %

Les opérations de sortie, réalisées directement sur un dispositif lent (imprimante) ralentissent l'exécution.

III - VERROUILLAGE DES PAGES TRES UTILISEES DE LA MACHINE VIRTUELLE OS/MVT, EN MEMOIRE REELLE.

Le système CP offre la possibilité de fixer en mémoire réelle du 360/67, une ou plusieurs pages de la mémoire d'une machine virtuelle.

Déterminons dans une première étape les zones de mémoire virtuelle à fixer en mémoire réelle, pour améliorer le fonctionnement du système OS/MVT sous CP sans détériorer les performances du mécanisme de pagination.

Le système OS/MVT partage la mémoire principale en 4 zones :

"FIXED AREA"

Zone contenant la partie résidente du système OS/MVT chargée au moment de l'opération IPL.

"SYSTEM QUEUE SPACE" (SQS) ou "SYSTEM QUEUE AREA" (SQA)

Zone contenant les tables et blocs de contrôle construits par OS/MVT.

"DYNAMIC AREA"

Zone subdivisée en régions, réservée aux programmes des utilisateurs et aux tâches système.

"LINK PACK AREA" (LPA)

Extension de la partie résidente du système OS/MVT chargée au moment de l'opération IPL, contenant un ensemble de programmes réentrants du système.

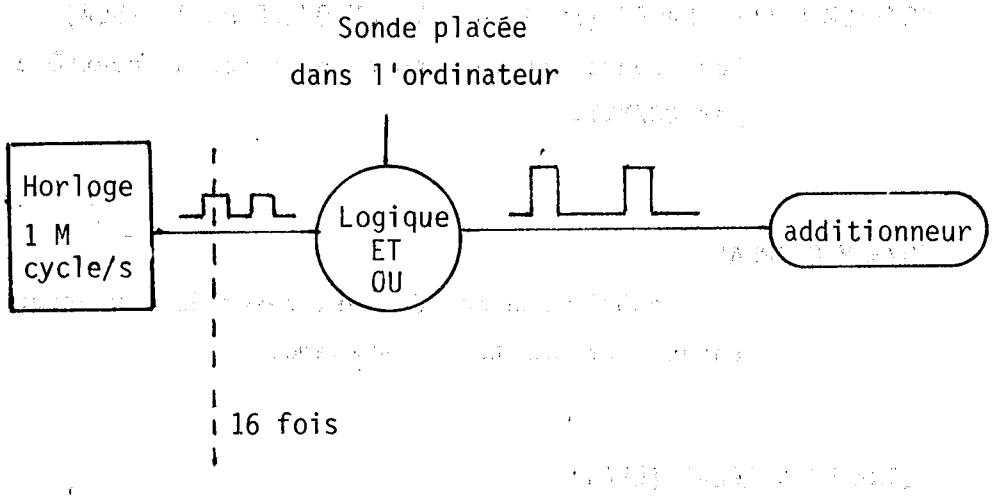
Les tailles de ces différentes zones sont :

	N° 1ère page	N°dernière page	Taille
FIXED AREA	0	18	73K
SQS	18	34	65K
DYNAMIC AREA	34	244	842K
LPA	245	255	44K

1- Détermination de ces pages

Une liste des pages (blocs de 4K octets) très utilisées de OS/MVT a été obtenue à l'aide d'expériences réalisées grâce à deux "BCU" (BASIC COUNTER UNIT).

Un BCU est un additionneur d'impulsions qui peut être représenté par le schéma suivant :



Pour l'expérience les sondes ont été placées sur les positions binaires (bits) du PSW de la machine afin de compter :

- le temps passé par page de 4K octets
  (sélection d'un bit dans la partie adresse)

- le temps passé en mode superviseur (sélection du bit "WAIT" et du bit "PROBLEM") qui est égal (en termes logiques) au temps :

$$\neg \text{ WAIT} \wedge \neg \text{ PROBLEM}$$

  Ce temps représente le temps passé dans le système CP.

Chaque "BCU" dispose de 16 sondes liées à 16 compteurs. Les 2 BCU ont été branchés pour que les 30 premières sondes mesurent le temps passé dans les pages 0 à 29 de la mémoire réelle.. La 31e sonde mesure le temps en mode superviseur. La 32e sonde mesure le temps total de l'expérience.

Pour chaque expérience, on calcule le pourcentage du temps passé dans chaque page par rapport au temps passé en mode superviseur.

Expérience I

Cette expérience porte sur 5 mesures.

Chaque mesure a été effectuée pour 1000 secondes de fonctionnement du système OS/MVT au cours d'une exploitation normale.

Résultats

Page	Temps	% Temps passé dans chaque page / Temps mode superviseur
0	226s 814ms	18,3 %
1	24s	1,9 %
2	4s 781ms	0,4 %
3	7s 715ms	0,6 %
4	100s 160ms	8,1 %
5	7s 688ms	0,6 %
6	20s 259ms	1,6 %
7	274s 403ms	22,2 %
8	75s 966ms	6,2 %
9	246s 521ms	20 %
10	161s 756ms	13,1 %
11	38s 571ms	3,1 %
12	48s 337ms	3,9 %
13	65s 975ms	5,4 %
14	27s 235ms	2,2 %
15	13s 025ms	1,1 %
16	7s 126ms	0,6 %
17	4s 469ms	0,4 %
18	7s 259ms	0,6 %
19	2s 033ms	0,2 %
20	6s 891ms	0,6 %
21	4s 249ms	0,3 %
22	24s 355ms	2 %
23	53s 100ms	4,3 %
24	11s 522ms	0,9 %
25	5s 630ms	0,5 %
26	4s 241ms	0,3 %
27	2s 675ms	0,2 %
28	5s 025ms	0,4 %
29	16s 263ms	1,3 %

Temps en mode superviseur = 1231s 508ms

Temps total = 5000s

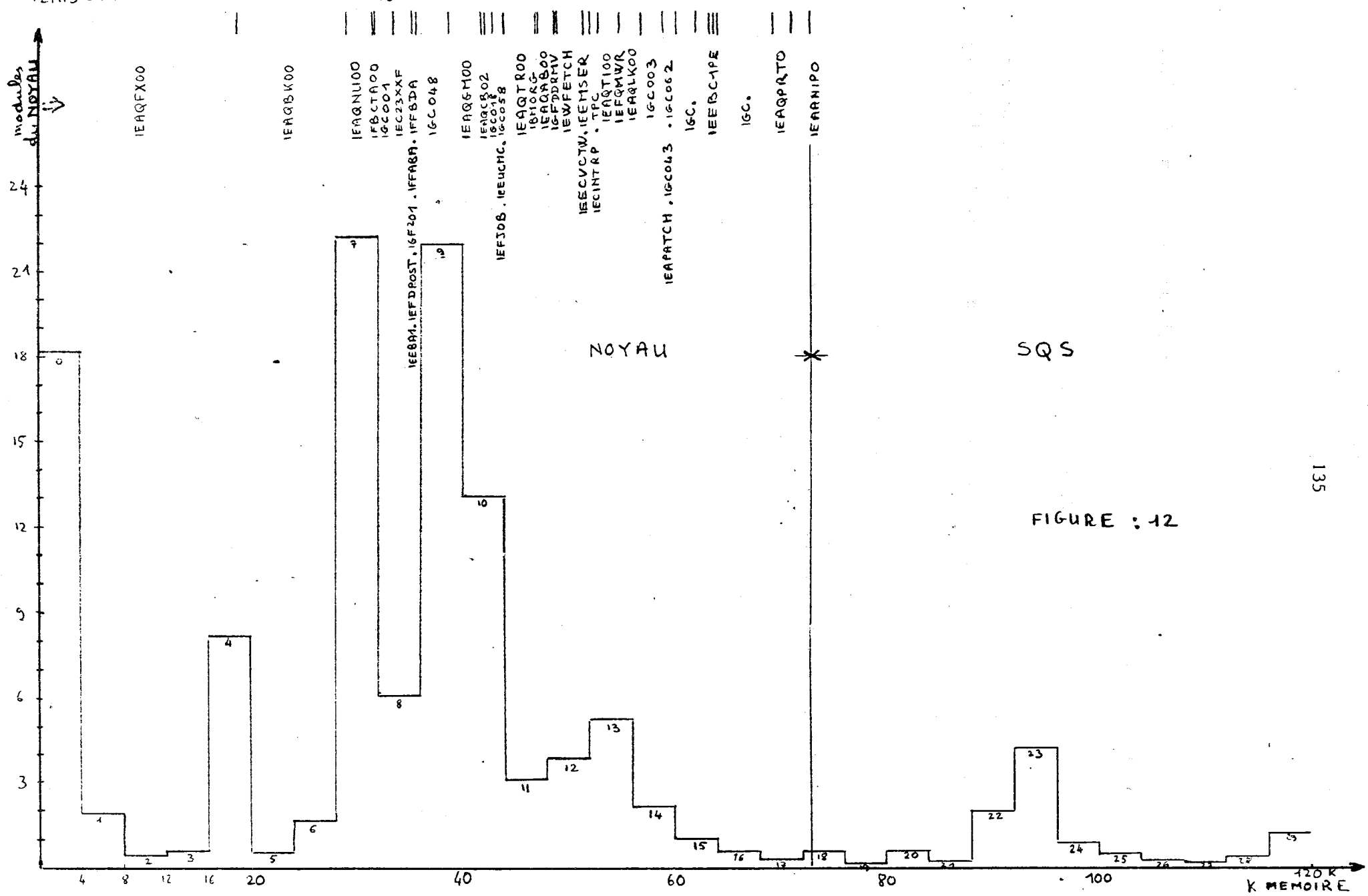

% TEMPS/PAGE 4 K
TEMPS SUPERVISEUR
% DU TEMPS PASSE DANS OS/MVT
modules du NOYAU
IEAQFX00
IEAQBK00
IEAQNU00
IFBCTA00
IGC001
IEC23XXF
IEEBM1.IEFDPOST.IGF201.IFFARA.IFFBDA
IGC048
IEAQGM00
IEAQCB02
IGC018
IGC058
IEFJOB.IEEUCHC.
IEAQTR00
IGF10RG
IEAQAB00
IGFDDRMV
IEWFETCH
IEECVCTW.IEEMSER
IECINTRP.TPC.
IEAQT100
IEFQMWR
IEAQLK00
IGC003
IEAPATCH.IGC043.IGC062
IGC.
IEEBC1PE
IGC.
IEAQPRT0
IEARNIP0
NOYAU
SQS
24
21
18
15
12
9
6
3
4
8
12
16
20
40
60
80
100
120 K
K MEMOIRE

FIGURE : 12

La liste des pages du noyau dans l'ordre décroissant d'utilisation est la suivante :

7,9,0,10,4,8,13,12,11,14,1,6,15,3,5,16,2,17

Le contenu de ces pages est noté dans la figure 12. La liste des pages de l'espace de travail du système (SQS) dans l'ordre décroissant d'utilisation est la suivante :

23,22,29,24,18,20,25,28,21,26,27,19

Les 10 pages les plus utilisées sont dans l'ordre décroissant :

7,9,0,10,4,8,13,23,12,11

Expérience II

Cette expérience porte sur une mesure effectuée pour 7889 sec. de fonctionnement du système OS/MVT au cours d'une exploitation normale.

Résultats

Page	Temps	% Temps passé dans chaque page / Temps mode superviseur
0	351s 796ms	15,1 %
1	42s 445ms	1,8 %
2	6s 449ms	0,3 %
3	12s 621ms	0,6 %
4	176s 245ms	7,6 %
5	10s 569ms	0,5 %
6	34s 838ms	1,5 %
7	433s 339ms	18,8 %
8	134s 547ms	5,8 %
9	295s 482ms	12,8 %
10	195s 948ms	8,4 %
11	56s 158ms	2,4 %
12	69s 557ms	3 %
13	115s 565ms	5 %
14	30s 584ms	1,3 %
15	25s 795ms	1,2 %
16	5s 679ms	0,3 %
17	17s 053ms	0,7 %
18	5s 669ms	0,3 %
19	14s 701ms	0,6 %
20	12s 385ms	0,5 %
21	3s 408ms	0,1 %
22	43s 309ms	1,9 %
23	92s 626ms	4 %
24	29s 964ms	1,5 %
25	12s 162ms	0,5 %
26	6s 190ms	0,3 %
27	2s 917ms	0,1 %
28	8s 089ms	0,3 %
29	33s 526ms	1,4 %

Temps en mode superviseur = 2316s 553ms

Temps total = 7889s 196ms

% TEMPS/PAGE 4K
% TEMPS SUPERVISEUR

% DU TEMPS PASSE DANS OS/MVT

EXPERIENCE II

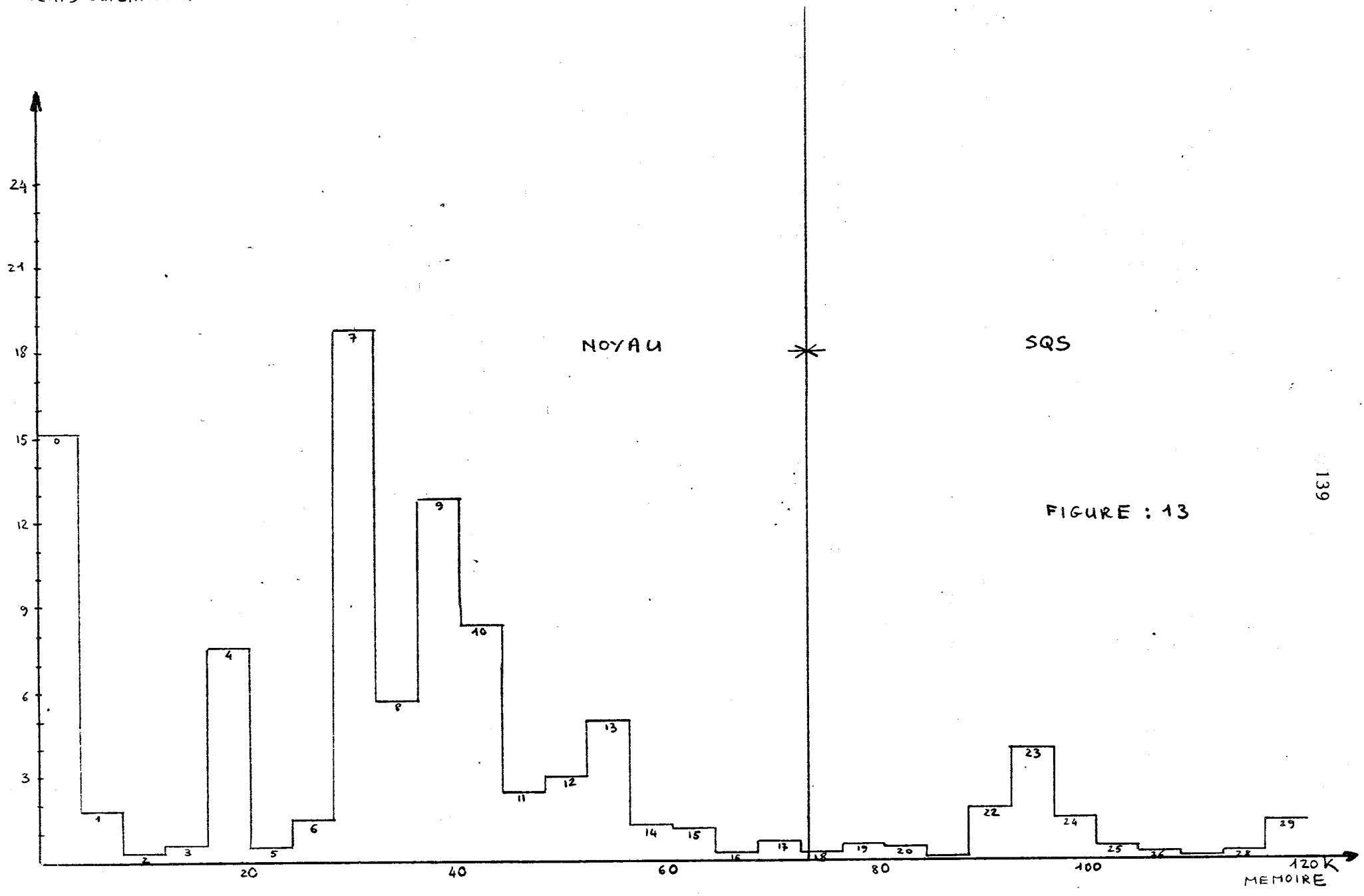


FIGURE : 13

La liste des pages du noyau dans l'ordre décroissant d'utilisation est la suivante :

7,0,9,10,4,8,13,12,11,1,6,14,15,17,3,5,2,16

La liste des pages de la SQS dans l'ordre décroissant d'utilisation est la suivante :

23,22,29,24,19,20,25,28,26,18,21,27

Les 10 pages les plus utilisées sont dans l'ordre décroissant :

7,0,9,10,4,8,13,23,12,11

Expérience III

Cette expérience porte sur quatre mesures de 500 secondes de fonctionnement du système OS/MVT.

Résultats

Page	Temps	% Temps passé dans chaque page / Temps mode superviseur
0	93s 487ms	14,2 %
1	11s 568ms	1,8 %
2	2s 061ms	0,3 %
3	3s 006ms	0,5 %
4	47s 603ms	7,3 %
5	9s 098ms	1,4 %
6	9s 478ms	1,4 %
7	121s 090ms	18,3 %
8	36s 903ms	5,6 %
9	86s 177ms	13,1 %
10	56s 826ms	8,6 %
11	20s 880ms	3,1 %
12	25s 148ms	3,8 %
13	29s 797ms	4,5 %
14	8s 768ms	1,3 %
15	11s 433ms	1,7 %
18	1s 218ms	0,2 %
19	317ms	0,0 %
20	2s 690ms	0,4 %
21	759ms	0,1 %
22	12s 899ms	2 %
23	26s 613ms	4 %
24	7s 171ms	1,1 %
25	1s 875ms	0,3 %
26	1s 576ms	0,2 %
27	976ms	0,1 %
28	2s 035ms	0,3 %
29	2s 518ms	0,4 %
30	8s 348ms	1,3 %
31	8s 371ms	1,3 %

Temps en mode superviseur = 655s 715ms

Temps total = 2000s 189ms

% TEMPS/PAGE 4K
TEMPS SUPERVISEUR

% DU TEMPS PASSE DANS OS/MVT

EXPERIENCE III

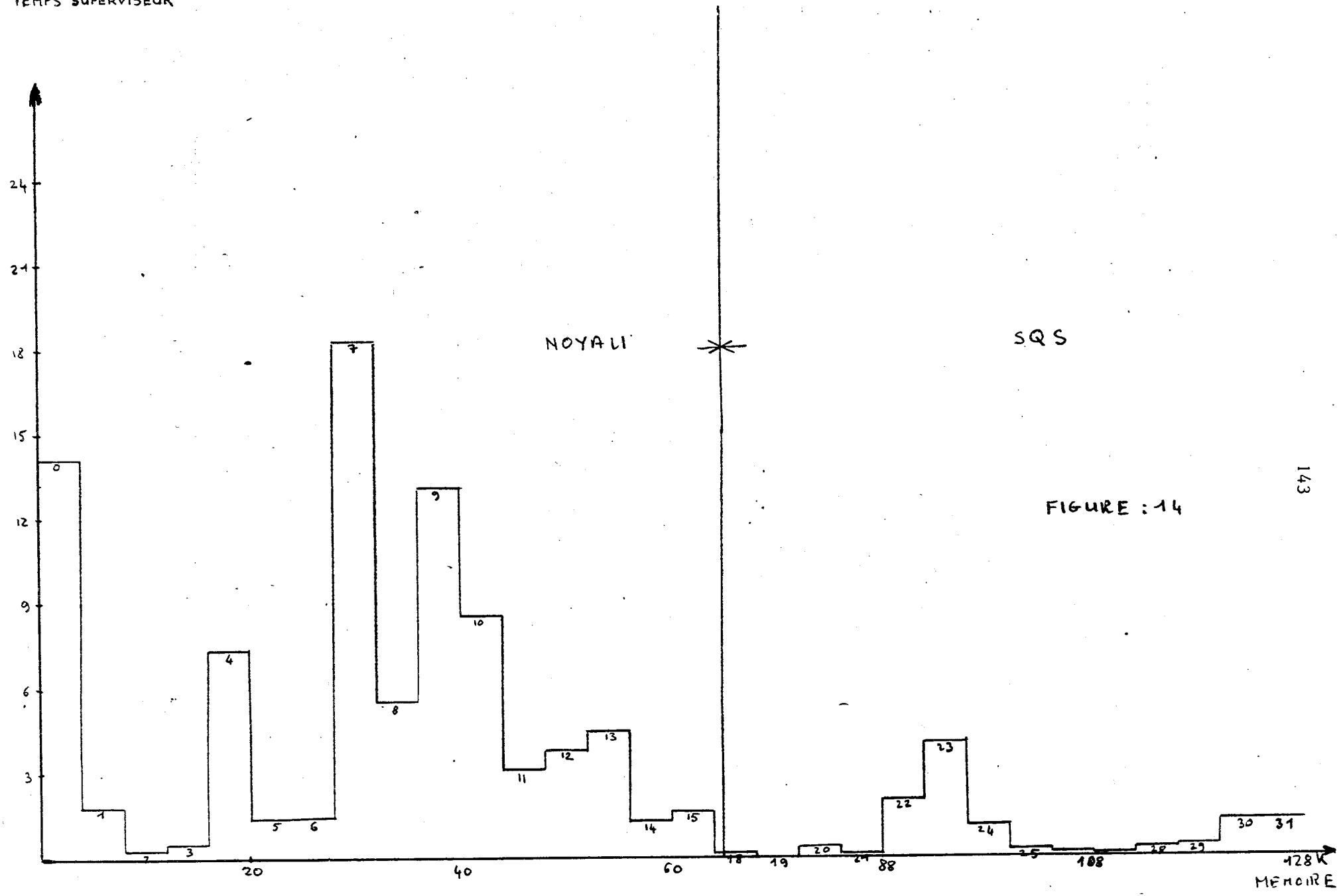


FIGURE : 14

La liste des pages du noyau dans l'ordre décroissant d'utilisation est la suivante :

7,0,9,10,4,8,13,12,11,1,15,6,5,14,3,2

La liste des pages de la SQS dans l'ordre décroissant d'utilisation est la suivante :

23,22,31,30,24,20,29,28,25,26,18,27,21,19

Les 10 pages les plus utilisées dans l'ordre décroissant sont :

7,0,9,10,4,8,13,23,12,11

Expérience IV

Cette expérience porte sur une mesure effectuée pendant l'opération d'IPL jusqu'au démarrage d'un premier travail.

Résultats

Page	Temps	% Temps passé dans chaque page / Temps mode superviseur
0	1391s 263ms	
1	131s 141ms	
2	12s 059ms	
3	16s 273ms	
4	256s 608ms	
5	61s 313ms	
6	49s 867ms	
7	1392s 057ms	
8	550s 724ms	
9	815s 333ms	
10	635s 333ms	
11	370s 686ms	
12	337s 796ms	
13	69s 075ms	
14	148s 429ms	
15	18s 206ms	
26	9s 098ms	
27	11s 019ms	
28	9s 346ms	
29	46s 732ms	

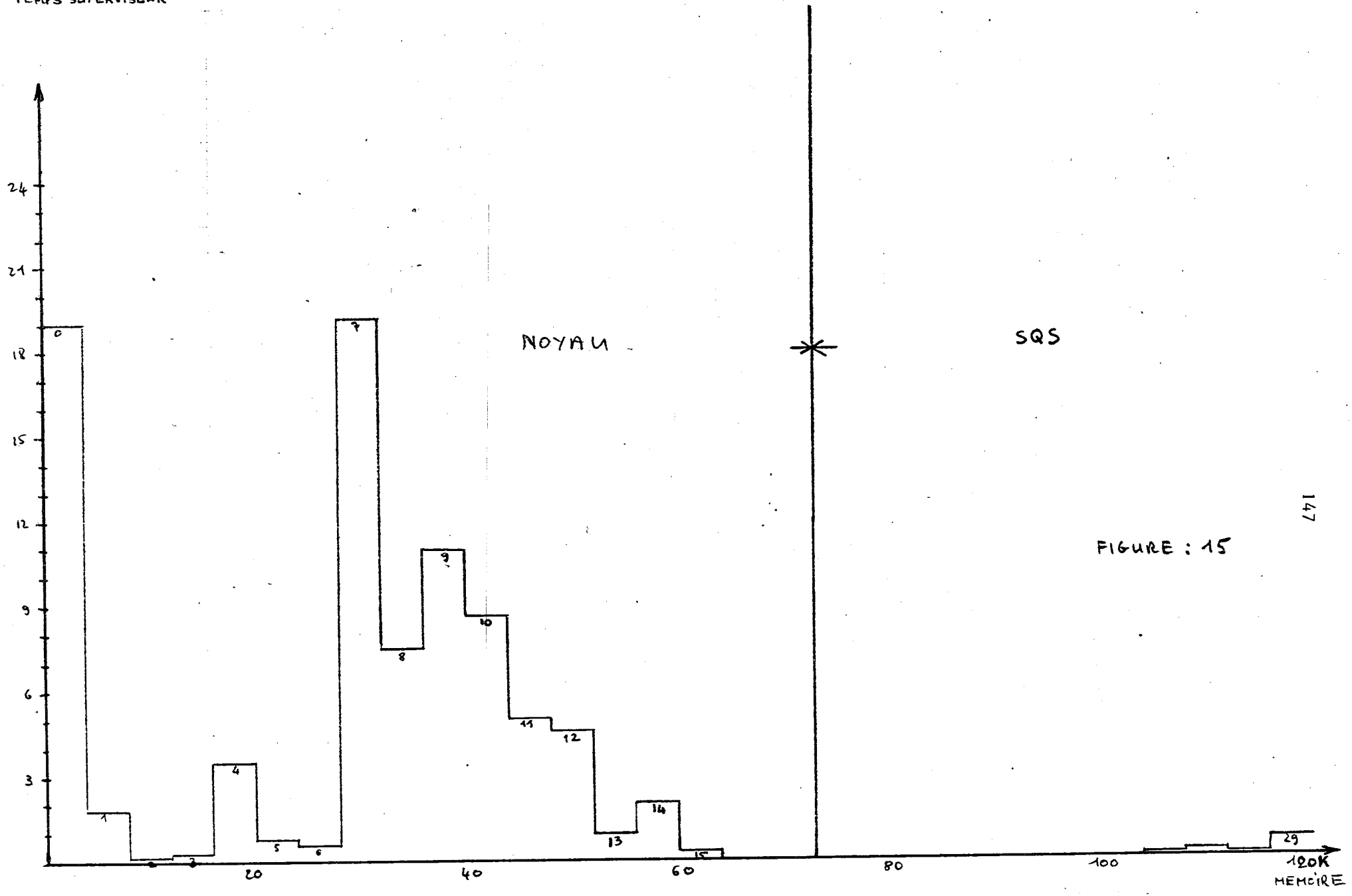
Temps en mode superviseur = 7301s 972ms

Temps total = 142735s 100ms

% TEMPS/PAGE 4K
TEMPS SUPERVISEUR

% DU TEMPS PASSE DANS OS/MVT A L'IPL

EXPERIENCE IV



147

FIGURE : 15

La liste des pages du noyau dans l'ordre décroissant d'utilisation est la suivante :

7,0,9,10,8,11,12,4,14,1,13,5,6,15,3,2

La liste des pages de la SQS dans l'ordre décroissant d'utilisation est :

29,27,28,26

Les 10 pages les plus utilisées sont dans l'ordre décroissant:

7,0,9,10,8,11,12,4,14,1

a- Liste des pages les plus utilisées du noyau de OS/MVT

Les expériences "BCU" nous permettent de dresser la liste des 10 pages les plus utilisées du noyau de OS/MVT dans l'ordre décroissant :

7,0,9,10,4,8,13,12,11,1

Portons dans un tableau , le pourcentage moyen d'utilisation de ces 10 pages au cours de la session ainsi que la liste des modules du noyau figurant dans chacune de ces 10 pages :

N° Page	% d'utilisation	Modules du noyau concernés	Fonctions
7	19,8	Fin de IEAQBK00	Partie fixe. Adresses de blocs de contrôle
		IEAQNU00	Partie fixe de mémoire basse
		IFBCTA00	Table fixe
		début de IGC001	Routine de "WAIT"
0	15,8	IEAQFX00	Partie fixe . Adresses de blocs de contrôle.
9	15,3	Fin de IGC048	Routine de "DEQUEUE"
		début de IEAQGM00	Routine de "GETMAIN, FREEMAIN"
10	10	Fin de IEAQGM00	
		IEAQCB02	
		IGC018	Recherche du directory d'un data set partitionné- conversion.
		IEFJOB	
		IEEUCMC	
		IGC058	
		début de IEAQTR00	Traitement des interruptions SVC de niveau 2.
4	7,8	Fin de IEAQFX00	Partie fixe. Adresses de blocs de contrôle
		début de IEAQBK00	Partie fixe. Adresses de blocs de contrôle
8	5,9	Fin de IGC001	Routine de "WAIT"
		IEC23XXF	
		IEEBA1	Routine d'attention
		IEFDPOST	Routine de "POST"
		IGF201	
		IFFABA	Table fixe
		IFFBDA	Table fixe
		début de IGC048	Routine "DEQUEUE"
13	5	IECINTRP	Traitement des interruptions d'entrée/sortie
		TPC	
		IEAQTI00	Traitement des interruptions timer de niveau 2
		IEFQMWR	
		début de IEAQLK00	Recherche de modules en mémoire(contents supervisi
12	3,6	Fin de IEAQAB00	Routine d'"ABEND"
		IGFDDRMV	
		IEWFETCH	Routine de recherche et de changement de modules
		IEECVCTW	Routine de "WAIT" (ECB et UCM)
		IEEMSER	Table résidente du "MASTER SCHEDULER"
11	2,9	Fin de IEAQTR00	Traitement des interruptions SVC de niveau 2
		IBMORG	Table de SVC
		début de IEAQAB00	Routine d'"ABEND"

b- Liste des pages les plus utilisées de l'espace de travail du système OS/MVT (SQS)

Les expériences "BCU" ont montré que les pages de l'espace de travail du système OS/MVT étaient moins fréquemment utilisées que les pages du noyau; les 7 pages les plus utilisées de la SQS sont :

23,22,30,31,29,24,20

Pour déterminer à un instant donné la position des différents blocs de contrôle actifs pour le système OS/MVT, utilisons un programme qui donne la liste des blocs de contrôle attachés à chaque tâche. Cette liste est construite en partant de la tâche de plus haute priorité et en suivant le schéma de la figure 16 :

OS/MVT SCHEMA DES BLOCS DE CONTROLE

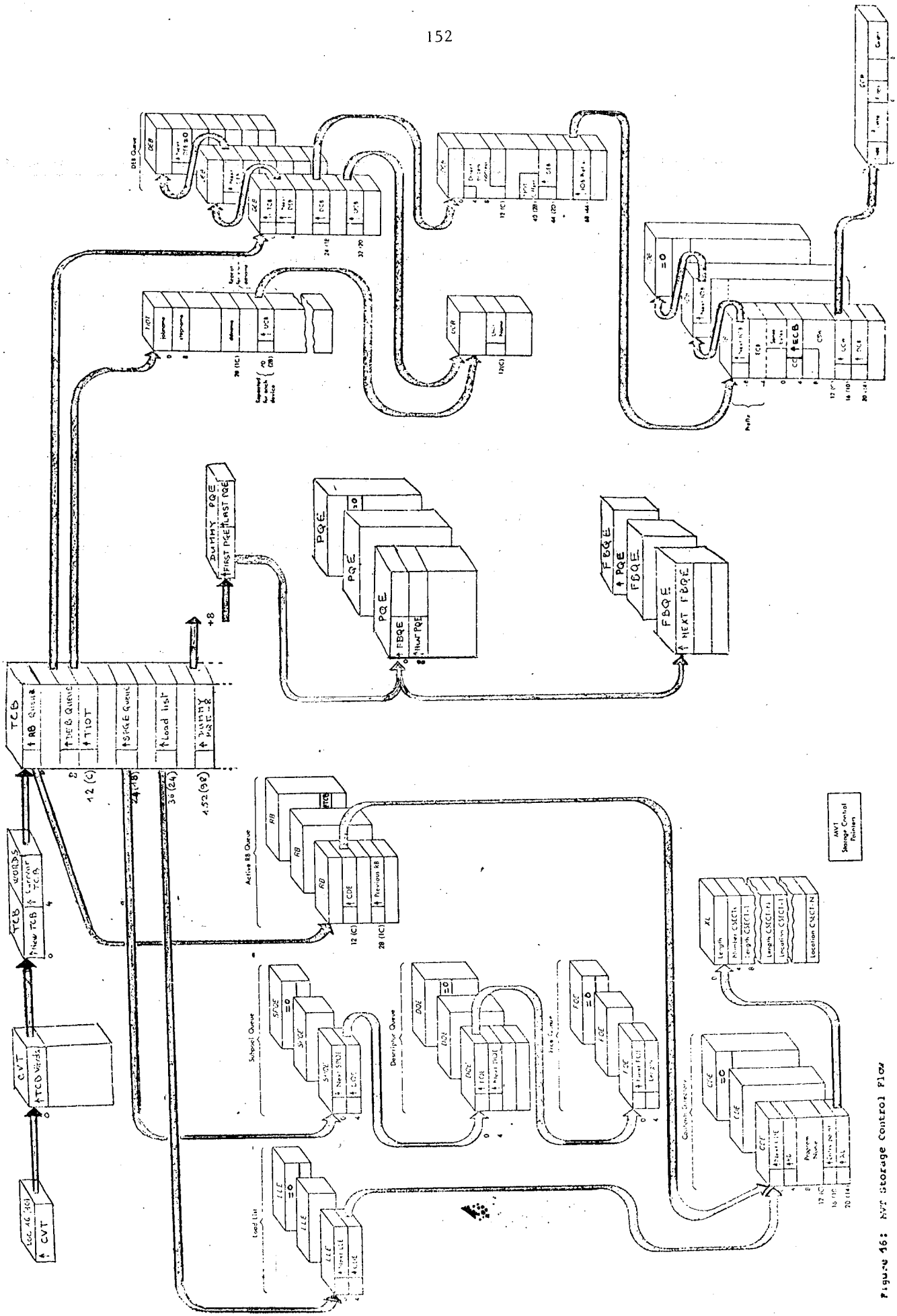


Figure 16: MVT Storage Control Flow

Il existe deux versions de ce programme nommé DPFORMAT.

Version (1)

Avec sortie sur l'imprimante de l'état des blocs de contrôle: un état des blocs de contrôle est pris en mode programme après avoir masqué les interruptions externes . Cette version fournit en fin d'analyse

- le nombre par page de chacun des blocs de contrôle
- la somme par page de tous les blocs de contrôle

Version (2)

Contrôle de l'analyse et sortie des résultats sur écran cathodique 2250. Dans cette version intéractive, un état des blocs de contrôle est pris interruptions externes masquées.

On peut afficher en fin d'analyse

- le nombre par page d'un bloc de contrôle quelconque
  - . par l'envoi d'un message de 4 caractères à partir du clavier du 2250 qui indique les types de blocs à visualiser.
    EX: "TIOT" état des TIOT
    :
    "DEB " état des DEB
  - . en utilisant les touches du bloc de fonctions du 2250: à chaque touche correspond un type de blocs de contrôle

- la somme par page de tous les blocs de contrôle :

  . par l'envoi du message "TOTA" à partir du clavier
  . en utilisant la touche TOTA du bloc de fonctions

De nouveaux états peuvent être obtenus à la demande

  . par l'envoi du message "DEBU"
  . en utilisant la touche DEBU du bloc de fonctions

Ce programme devient inactif :

  . par l'envoi du message "FIN"
  . en utilisant la touche FIN du bloc de fonctions

Résultats donnés par ce programme

Les pages de la SQS contenant le plus de blocs de contrôle sont dans l'ordre décroissant :

34,33,32,29,30,28

NOMBRE DE BLOC PAR PAGE

PAGE	CVT	TCB	PQE	FBQE	TIOT	SPQE	DQE	FQE	RB	LLE	CDE	XL	DEB	UCB	DCB	IOB	ECB	CCW	TOTAL
** NOYAU **																			
0 00																	1		1
1 01														10					10
2 02														13			1		14
4 04		3							7										15
5 05		2							3										5
7 07	1																		1
8 08													1		1	1			3
10 0A																1			1
** SQS **																			
19 13			1																1
23 17		2											4						6
24 18		3	1		3				3				6						16
25 19		3			3				2		2	1	1		1	1		1	15
26 1A		3			3		1		1		4	2	1		1				16
27 1B		4	2		2		2		3		4	5							22
28 1C		6			5	4	18		6	2	5	3	1			1			51
29 1D		3			5	11	14		6	20	8	12	2						81
30 1E		1	1		3	8	16		3	4	4	5	2						47
32 20			2			15	25		3	40	13	11	5		2	1			117
33 21		2			4	10	25		2	28	27	30	1						129
34 22						6	14	3	1	34	91	89							238

FIGURE : 17

2- Influence du verrouillage de certaines pages

a- Verrouillage des pages du noyau de OS/MVT déterminées par les expériences BCU

Exécution du batch en fixant en mémoire réelle successivement 0,3,6,12,18 pages du noyau de OS/MVT.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader de 1 buffer de 3200 octets

Résultats

Nbre de pages fixées	N°de pages fixées	$T_{exécution}$	T_{reader}
0 page		26mn 15s	4mn 25s
3 pages	7,0,9	25mn 37s	4mn 27s
6 pages	7,0,9,10,4,8	25mn 50s	4mn 20s
12 pages	7,0,9,10,4,8, 13,12,11,14,15,6	25mn 28s	4mn 25s
Tout le noyau 18 pages	0 → 17	26mn 9s	4mn 18s
35 pages	0 → 34	26mn 20s	

Le temps d'exécution minimum a été obtenu en fixant les 12 pages les

plus utilisées du noyau de OS/MVT :

Gain de temps : 47s soit 3 %

b- Verrouillage des pages de l'espace de travail de OS/MVT déterminées par le programme DPFORMAT et les expériences BCU.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- sans aucune page fixée
  - $T_{exécution}$ = 26mn 15s
  - T_{reader} = 4mn 25s
- avec 8 pages fixées (22,23,29,30,31,32,33,34)
  - $T_{exécution}$ = 26mn 34s
  - T_{reader} = 4mn 42s

En fixant en mémoire réelle les 8 pages les plus utilisées de la SQS le temps d'exécution est sensiblement le même.

c- Verrouillage des pages contenant les modules résidents de OS/MVT (LPA)

La taille totale de la LPA est pour cette expérience de 44K; nous fixons donc 11 pages en mémoire réelle.

Conditions de l'expérience

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue
- 7 initiateurs
- reader avec 1 buffer de 3200 octets

Résultats

- sans page fixée

$T_{exécution}$ = 26mn 15s

T_{reader} = 4mn 25s

- avec 11 pages fixées

$T_{exécution}$ = 27mn

T_{reader} = 4mn

Le fait de fixer en mémoire réelle les pages de la "link pack area" n'a pas amélioré le temps d'exécution du batch.

3- Conclusions

L'influence du verrouillage de certaines pages de la machine virtuelle OS/MVT en mémoire réelle, ne peut être que très peu sensible dans les conditions de l'expérience.

En effet, la machine virtuelle OS/MVT qui est seule active, a en permanence un grand nombre de ces pages installées en mémoire réelle. La pagination ne se fait donc qu'à un très faible niveau.

On peut penser que les améliorations notées seraient nettement plus importantes si plusieurs machines virtuelles étaient actives en parallèle.

Le verrouillage des 12 pages les plus utilisées du noyau de OS/MVT améliore de façon sensible le temps réel d'exécution du batch (3 %). Ces 12 pages contiennent des programmes qui s'étendent généralement sur plusieurs pages et dont la structure ne tient pas compte de la taille de ces pages.

Le mécanisme de pagination est affecté par la structure des programmes à exécuter; les fautes de pages sont d'autant plus fréquentes:

- que la page est souvent référencée
- que le programme auquel elle appartient s'étend sur plusieurs pages.

On peut donc penser que le verrouillage des pages du noyau apporte une amélioration du fonctionnement du système OS/MVT sous CP. Cette amélioration se faisant au détriment des autres machines virtuelles puisque les pages fixées restent en mémoire réelle et diminuent donc l'espace libre.

Le verrouillage des pages de la zone de travail du système

OS/MVT n'entraîne pas d'amélioration. Cette zone contient des blocs de contrôle très souvent utilisés et de très petite taille. Dans le contexte de l'expérience, (une seule machine virtuelle active) ils sont pratiquement en permanence en mémoire réelle donc très peu paginés.

Le verrouillage des pages de la LPA n'apporte pas non plus d'amélioration au temps d'exécution. Cette zone contient des modules de petite taille (généralement inférieure à 1024 octets) qui sont très souvent utilisés donc pratiquement en permanence en mémoire réelle.

Remarquons encore que le verrouillage d'un trop grand nombre de pages (EX: 18 pour le noyau) augmente le temps d'exécution du batch donc affecte le mécanisme de pagination. L'influence relativement faible du verrouillage des pages de la machine virtuelle en mémoire réelle permet de conclure que le système CP "pagine" bien.

IV - CONCLUSIONS

1- Tableau récapitulatif des résultats obtenus séparément

Modifications apportées	Gain de temps d'exécution du batch
Utilisation d'une mémoire virtuelle de 2048K au lieu de 512K	7,8 %
Utilisation du lecteur virtuel au lieu du lecteur réel	2,5 %
Blocage en mémoire réelle de 12 pages très utilisées du noyau de MVT : (7,0,9,10,4,8,13,12,11,14,15,6)	3 %

2- Choix des valeurs optimales des paramètres de CP étudiés

Pour améliorer le fonctionnement du système OS/MVT sous CP on pourra fixer dans CP les paramètres suivants :

- Machine virtuelle OS/MVT disposant d'une mémoire virtuelle 2048K octets.
- utilisation du lecteur virtuel pour la machine OS.
- 12 pages les plus utilisées de la machine virtuelle OS/MVT fixées en mémoire réelle.

3- Gain de temps d'exécution obtenu

En fixant ces paramètres dans CP, le gain de temps d'exécution est d'environ 13,3 %.

Le temps d'exécution du batch, avec le seul système OS/MVT optimisé sous CP était de 23mn 2s. En fixant dans CP les paramètres indiqués, il est diminué d'environ 3mn c'est-à-dire qu'il est de 20mn environ.

C - PERFORMANCES DE MVT SEUL UTILISATEUR DE CP

I - EVALUATION

Pour déterminer les performances de MVT fonctionnant seul sous CP, nous prendrons comme temps de référence le meilleur temps d'exécution du batch sur la machine réelle, obtenu en utilisant les deux tambours magnétiques pour le système OS/MVT :

10 mn 53 secondes

Etablissons deux états de performance :

Après optimisation de certains paramètres du système OS/MVT en utilisant les deux tambours magnétiques pour le système CP :

Le temps d'exécution du batch par le système OS/MVT optimisé sous CP est de :

23 mn 2 s

Le temps d'exécution a augmenté de 12 mn 9 s.

Les performances du système OS/MVT ont diminué de :

52,7 %

par rapport à son fonctionnement sur la machine réelle.

Après optimisation de certains paramètres du système OS/MVT et de certains paramètres du système CP en utilisant un tambour magnétique pour le système MVT et pour le système CP :

Le temps d'exécution du batch par le système OS/MVT optimisé fonctionnant sous CP optimisé est de :

20 mn

Le temps d'exécution a augmenté de 9 mn 7 s.

Dans les meilleures conditions de fonctionnement des deux systèmes, les performances du système OS/MVT fonctionnant dans une machine virtuelle gérée par le système CP ont diminué de 45 % par rapport à son fonctionnement sur la machine réelle.

Cette diminution est importante et montre à quel point sont antagonistes les systèmes MVT et CP.

II - JUSTIFICATION : temps passé dans CP

Le coût de la gestion de la machine virtuelle OS/MVT se traduit par une utilisation du CPU au niveau du système CP.

1- Temps global passé dans CP:(overhead)

La commande CP : Q T fournit :

- le temps CPU total utilisé par la machine virtuelle
- le temps CPU virtuel (temps CPU utilisé par le système OS/MVT)

Temps CPU passé dans le système CP = TOTCPU - VIRCPU

L'expérience montre que le temps CPU total est en moyenne deux fois supérieur au temps CPU virtuel.

Le pourcentage de temps CPU passé dans le système CP pour la gestion de la machine virtuelle OS/MVT est donc en moyenne égal à 50 % du temps CPU total.

2- Temps passé pour la recompilation de programmes canaux.

Une partie du temps CPU passé dans CP est dûe à la recompilation des programmes canaux virtuels de OS/MVT en programmes canaux réels pour la réalisation des opérations d'entrée-sortie.

Pour mesurer le temps passé en recompilation de programmes canaux et le nombre de ces recompilations, effectuons l'expérience suivante :

Dans le module CCWTRANS du système CP plaçons deux compteurs :

- le premier incrémenté de 1 à chaque passage
- le second cumulant à chaque passage le temps passé dans ce module.

Nous obtenons les résultats suivants :

	Nbre de recompilation	Temps passé en recompilation en unité de temps CP	Temps total passé dans CP en unité de temps CP	% du temps passé en recompilation par rapport au temps passé dans CP
Pour l'initialisation du système OS	9607	1787566	4852577	36
Pour l'exécution du batch test	84510	8209526	28745247	28

28 % du temps passé dans CP est donc dû à la recompilation des programmes canaux, pour la machine OS ; ce pourcentage justifie en partie la dégradation des performances du système de multiprogrammation sous CP.
Ce pourcentage est très faible (inférieure à 10 %) pour une exploitation CP sans machine virtuelle OS.

P A R T I E II

MACHINE VIRTUELLE OS/MVT EN PARALLELE AVEC

N MACHINES VIRTUELLES ACTIVES

Les mesures effectuées jusqu'ici ont été faites alors que le système CP ne supportait qu'un seul utilisateur : la machine virtuelle OS.

L'étude suivante détermine :

- l'influence sur le temps d'exécution du batch test d'une charge croissante du système CP.

- un nouvel état des performances du système OS/MVT lorsque CP supporte une charge raisonnable.

Pour les expériences réalisées dans cette partie, les deux tambours magnétiques seront utilisés par le système CP.

Nous pourrons ainsi augmenter la charge de CP sans diminuer les performances de son mécanisme de pagination.

De plus la machine virtuelle OS ne sera pas favorisée (pas de fixation de pages en mémoire réelle).

A - ETUDE DE LA CHARGE CROISSANTE DE CP

I - DETERMINATION DE N POUR OBTENIR LA CHARGE HABITUELLE DE CP.

Pour augmenter la charge du système CP, on active en parallèle la machine OS et N machines virtuelles standards utilisant le système CMS.

Chaque machine virtuelle effectue un travail standard constitué par :

- l'édition d'un petit fichier
- la compilation d'un programme Fortran, PL1 ou Algol
- le chargement et l'exécution du programme compilé

Ces quatre opérations sont effectuées d'une manière cyclique et sont mises en jeu par une procédure d'enchaînement automatique. Les machines virtuelles fournissent donc un travail continu à CP.

L'augmentation de charge est contrôlée par une machine virtuelle spécialisée MESURE, qui, à intervalles de temps réguliers, mesure l'activité du système CP et relève des renseignements sur :

- le taux de multiprogrammation
- le temps problème
- le temps de WAIT
- le temps pris par CP lui-même
- le nombre de fautes de page par seconde
- le nombre de pages lues par CP

- le nombre de pages écrites par CP
- l'activité des canaux

REMARQUE

Chaque machine virtuelle autre que la machine OS, effectue ses entrées-sorties sur le mini-disque qui lui correspond; Tous ces mini-disques appartiennent à la même unité réelle et, en conséquence, un conflit apparaitra rapidement au niveau du canal auquel est reliée cette unité.
Ces machines virtuelles crèent donc des conditions relativement défavorables.

Expérience

La charge de base du système CP sera :

- la machine virtuelle OS
- la machine mesure qui apporte une charge négligeable

Exécutons dans la machine OS, le batch habituel en activant successivement en parallèle : 0,3,5,8 machines virtuelles standards.

Conditions

- Volume système résident sur disque
- SYS1.SYSJOBQE disque
- avec LPA étendue

- 7 initiateurs
- reader avec un buffer de 3200 octets

Résultats

Nbre de machines virtuelles supplémentaires		$T_{session}$	$T_{exécution}$	T_{reader}
0		33mn 40s	26mn 15s	
1 Fortran 1 PL1 1 Algol	3		32mn 56s	5mn 3s
2 Fortran 2 PL1 1 Algol	5	47mn 48s	40mn 48s	8mn
3 Fortran 2 PL1 1 Algol	8		1h 43mn 30s	15mn 10s

Regroupons dans la Figure 18 les résultats fournis par la machine MESURE et portons sur un graphique : Figure 19 les résultats de l'expérience.

MOYENNE DES RESULTATS DE LA MACHINE MESURE

Nbre machines virtuelles standards	Taux Multiprogrammation de CP	Temps Problème %	Temps Wait %	Temps CP %	Nombre de fautes de page(s)	Nbre de pages sur tambour Max=900	Nbre de calculs de la taille mémoire/s	Commentaires
3	4, 12 un peu faible	43, 64 très bon	7, 51 très bon	48,85 un peu élevé	55	354	0	Le temps problème est très bon. Occupation mémoire faible. Bon fonctionnement. Pas de saturation.
5	5, 93 moyen	34, 51 un peu faible	28,37 correct	37,12 correct	39	427	1	Correspond à un fonctionnement de CP au cours d'une exploitation habituelle (sans machine virtuelle OS).
8	7, 9 Trop élevé	18, 34 très faible	51, 22 très élevé	30,44 presque normal	20	420	32	Conflit mémoire : très grand nbre de calcul de taille mémoire/seconde Conflit canal dû aux nombreuses entrées/sorties provenant des machines virtuelles.

Figure 18

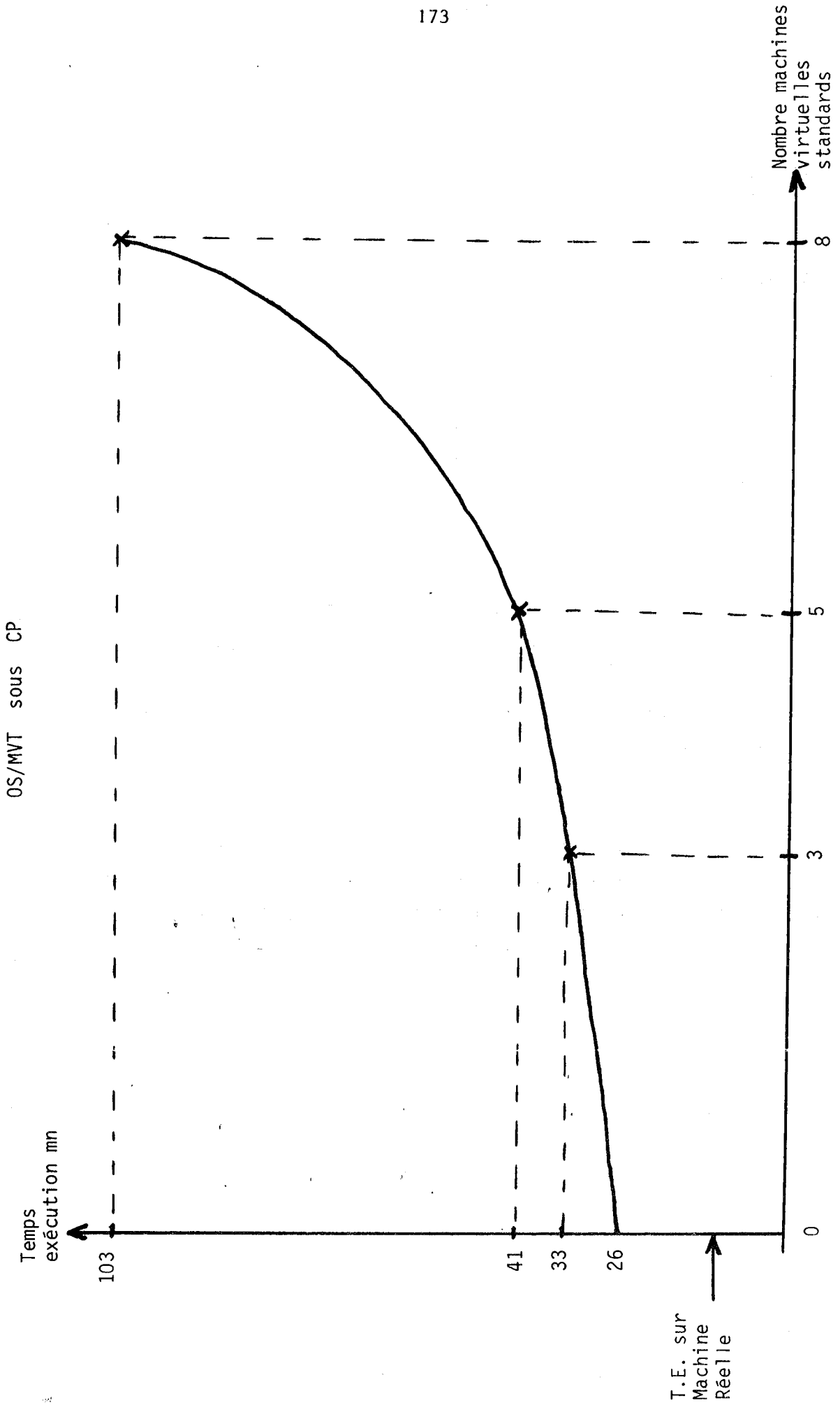

OS/MVT sous CP
Temps exécution mn
103
41
33
26
T.E. sur Machine Réelle
0
3
5
8
Charge habituelle de CP
Nombre machines virtuelles standards

Figure 19

II - CONCLUSIONS

Le temps d'exécution a donc augmenté très rapidement avec le nombre de machines virtuelles standards supplémentaires :(Figure 19).

Il est de 26 mn 15 s lorsque MVT est seul utilisateur de CP; en ajoutant :

- 3 machines virtuelles, il est multiplié par 1,25
- 5 machines virtuelles, il est multiplié par 1,55
- 8 machines virtuelles, il est multiplié par 3,95

La machine virtuelle OS/MVT impose donc à CP une lourde charge, peu compatible avec le fonctionnement en parallèle, de machines virtuelles standards. Une charge voisine de la charge habituelle du système CP a été obtenue ici, par l'activation de la machine OS/MVT et de 5 machines virtuelles standards (Figure 18).

Compte tenu de la particularité des machines virtuelles utilisées pour l'expérience (fournissant un travail continu au système CP) nous pouvons penser que la machine virtuelle OS/MVT apporte une charge équivalente à 30 utilisateurs standards).

B - PERFORMANCE DE MVT DANS CET ENVIRONNEMENT

En activant en parallèle :

- la machine virtuelle OS/MVT
- la machine mesure
- 5 machines virtuelles du type que nous avons défini

afin d'obtenir la charge habituelle du système CP, le temps réel d'exécution du batch test, par le système OS/MVT sous CP est de :

40 mn 48 s

Ce temps d'exécution est presque 4 fois supérieur au temps d'exécution du batch test, par le système OS/MVT sur la machine réelle dans les meilleures conditions (10 mn 53 s). Les performances du système OS/MVT sous CP diminuent considérablement lorsque la charge de CP devient plus importante. Il n'est donc pas pensable d'utiliser en exploitation dans une machine virtuelle un système aussi complexe que OS/MVT.

(Rappelons que dans cette partie, les deux tambours magnétiques étaient utilisés par CP).

La multiprogrammation réalisée par OS/MVT est incompatible avec celle réalisée par CP; nous sommes donc amenés à étudier sous CP le comportement de systèmes OS plus simples :

- PCP système de monoprogrammation
- MFT avec une seule partition

et nous laisserons au système CP le soin de réaliser la multiprogrammation en activant en parallèle plusieurs machines virtuelles OS.

C H A P I T R E V

ESSAIS COMPARATIFS AVEC LES SYSTEMES

OS PCP et MFT sous CP

La dégradation des performances du système OS/MVT, fonctionnant dans une machine virtuelle gérée par le système CP, est importante.

Une étude complémentaire a été réalisée en utilisant les systèmes PCP et MFT pour exécuter dans une ou plusieurs machines virtuelles OS le même ensemble de travaux.

Dans ce chapitre, laissons au système CP le soin d'effectuer la multiprogrammation, et activons plusieurs machines virtuelles OS, fonctionnant en monoprogrammation avec les systèmes PCP et MFT une partition.

Le nombre de machines virtuelles OS actives en parallèle, limité par le "hardware" à quatre pour nos expériences permet d'obtenir un taux de multiprogrammation satisfaisant. Chaque machine virtuelle OS a, à sa disposition, au moins quatre piles de disques 2314, les deux tambours magnétiques étant utilisés par CP pour la pagination.

Ne favorisons pas les machines virtuelles OS (pas de fixation de pages en mémoire réelle).

Les temps d'exécution totaux notés dans ce chapitre comprennent :

- les temps d'exécution des JOBS
- les temps d'impression dans le spooling de CP (utilisation du lecteur et de l'imprimante virtuels).

Lorsque l'ensemble des travaux est partagé entre plusieurs machines virtuelles OS, le temps d'exécution sera pris comme étant la moyenne des temps d'exécution dans chaque machine virtuelle :

$$T_{EM} = \frac{\sum_{i=1}^{N} T_{E_i}}{N}$$

T_{E_i} = temps d'exécution relevé dans chaque machine virtuelle OS

N = nombre de machines virtuelles OS

Ce choix est justifié par le fait que, sur une expérience de longue durée il est possible de répartir l'ensemble des travaux à exécuter entre les différentes machines, de façon à obtenir un même temps d'activité de chacune d'elles.

I - ETUDE DU SYSTEME OS PCP sous CP

Le système PCP utilisé pour ces expériences a tous ses data sets regroupés sur une seule pile de disques 2314.

A l'initialisation du système, chargeons en mémoire virtuelle la liste standard des modules préconisés par IBM.

1- Etude de la variation du nombre de machines virtuelles OS/PCP et du nombre de machines virtuelles standards.

Exécutons l'ensemble des JOBS du batch test, répartis dans successivement 1,2,3 puis 4 machines virtuelles OS/PCP.

Pour un nombre de machines virtuelles données,augmentons la charge de CP en activant en parallèle des machines virtuelles standards.

Regroupons dans la Figure 20 ci-dessous l'ensemble des résultats obtenus.

Portons sur un graphique : Figure 21, les temps d'exécution de l'ensemble des JOBS du batch test répartis entre 1,2,3 puis 4 machines virtuelles OS/PCP.

Portons sur un graphique Figure 22, les temps d'exécution de ce même ensemble de travaux répartis entre 3 machines virtuelles OS/PCP fonctionnant en parallèle avec des machines virtuelles standards.

Machines virtuelles	Temps problème %	Temps WAIT %	Temps dans CP %	PCP1		PCP2		PCP3		PCP4		Temps d'exécution moyen total avec impression (SPOOL CP)
				JOBS	TE	JOBS	TE	JOBS	TE	JOBS	TE	
1 PCP 1 mesure	49 -	41	10	BENCH11 " 10 " 03 " 12 " 16 " 08 " 01	31'25"							31' 25"
2 PCP 1 mesure	57	24	19	BENCH11 " 03 " 08	14'55"	BENCH10 " 12 " 16 " 01	19'5"					17'
3 PCP 1 mesure	60	19	21	BENCH11	6'40"	BENCH01 " 12 " 10	16'35"	BENCH08 " 03 " 16	11'10"			11' 28"
4 PCP 1 mesure	70	1	29	BENCH11	10'30"	BENCH12 BENCH16 BENCH01	12'45"	BENCH03 BENCH08	9'50"	BENCH10	13'20"	11' 36"
2 PCP 5 standard 1 mesure	37	48	15	BENCH11 BENCH08 BENCH03	33'50"	BENCH01 BENCH12 BENCH16 BENCH10	22'35"					28' 10"
3 PCP 5 standard 1 Mesure	47	33	20	BENCH11 BENCH16	23'20"	BENCH01 BENCH12 BENCH10	22'45"	BENCH08 " 03	12'45"			19' 37"
3 PCP 8 standard 1 Mesure	74	6	20	BENCH11 " 16 " 01	48'	BENCH10	18'10"	BENCH08 BENCH12 BENCH03	17'50"			29'

TE = temps d'exécution avec impression dans le spooling CP

Figure 20

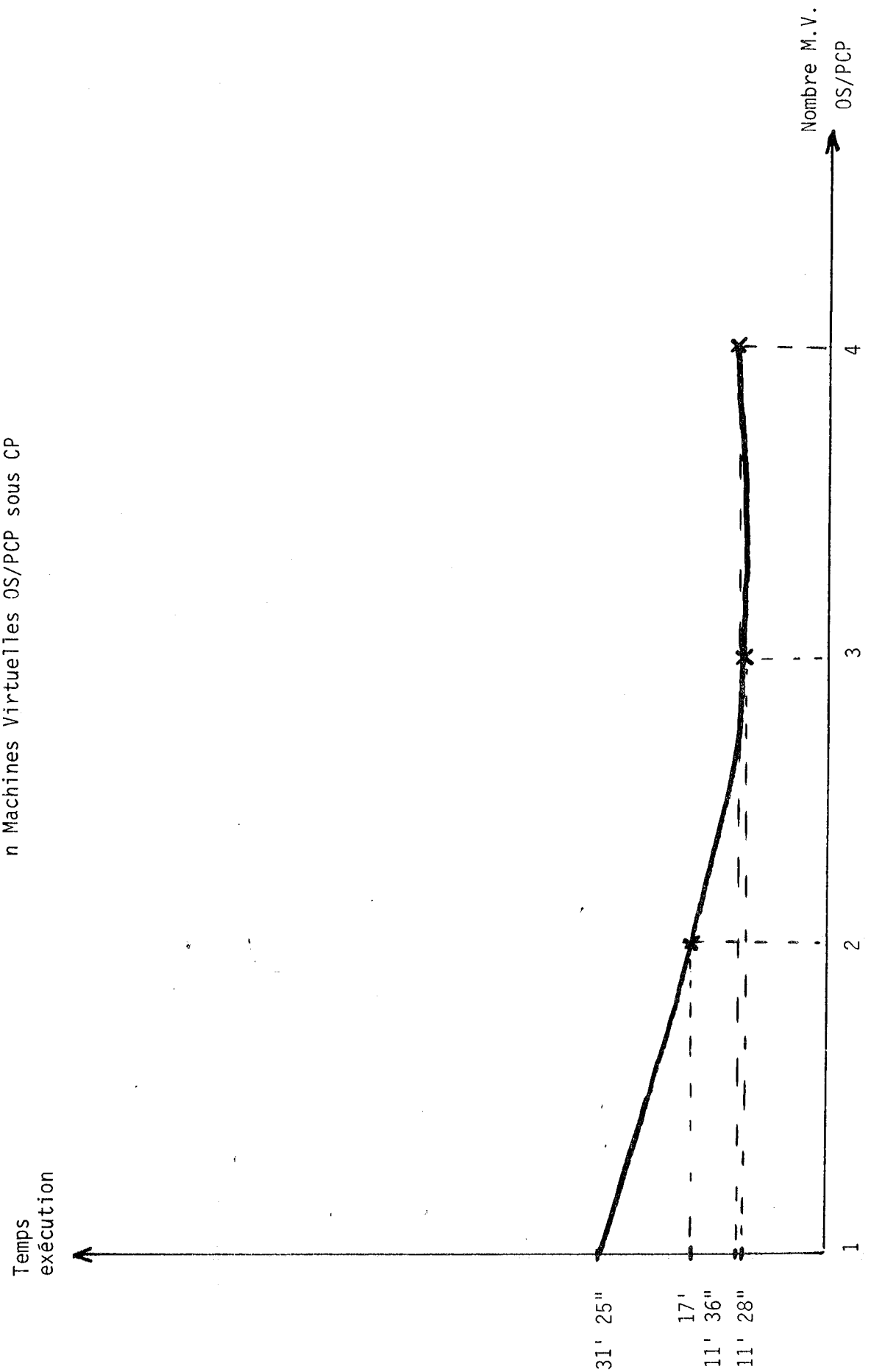

n Machines Virtuelles OS/PCP sous CP
Temps exécution
31' 25"
17' 36"
11' 28"
1
2
3
4
Nombre M.V. OS/PCP

Figure 21

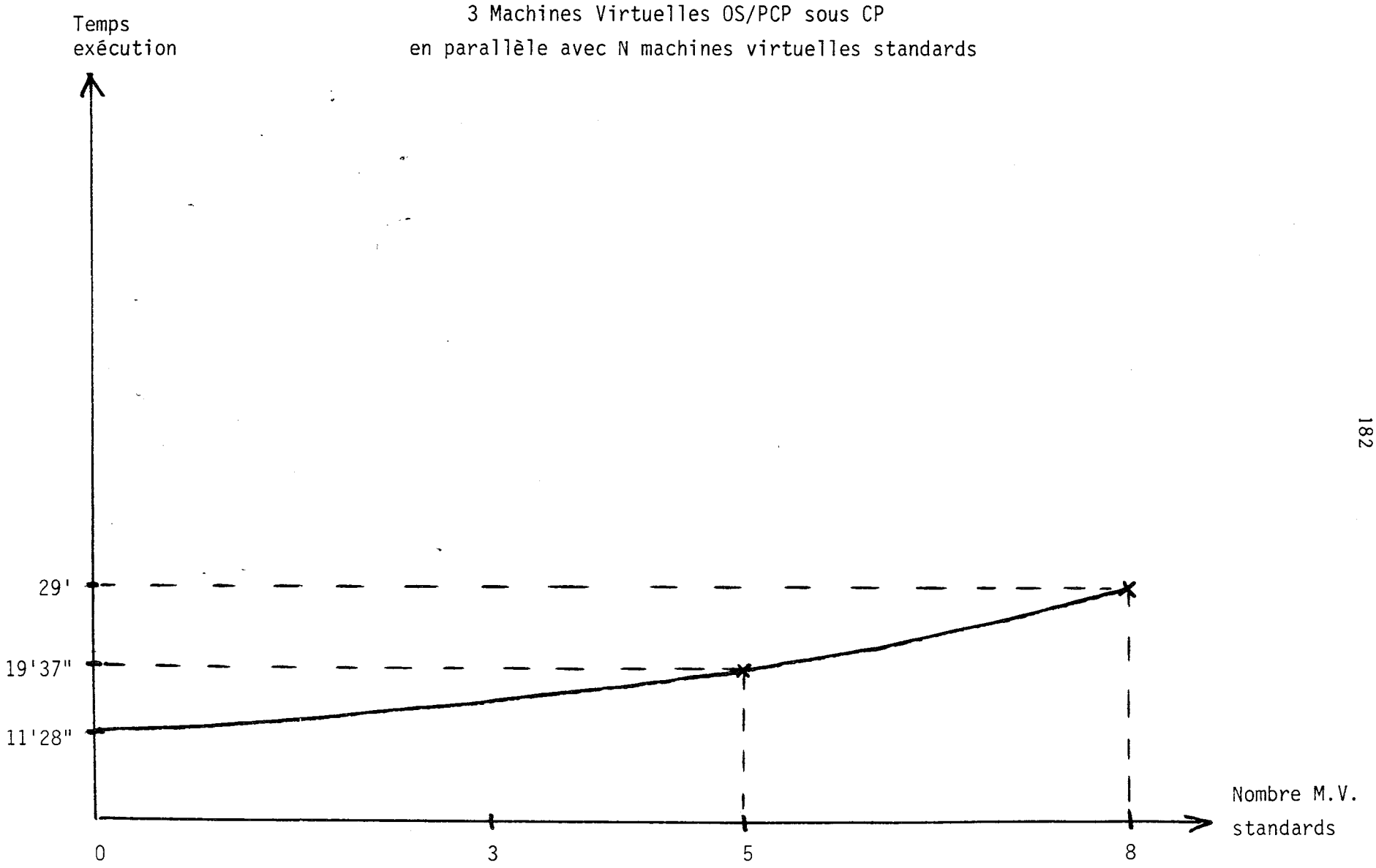

3 Machines Virtuelles OS/PCP sous CP
en parallèle avec N machines virtuelles standards
Temps
exécution
29'
19'37"
11'28"
0
3
5
8
Nombre M.V.
standards

Figure 22

2- Performances du système OS/PCP sous CP

a- Machines virtuelles OS/PCP seules utilisatrices de CP

La consultation du tableau des résultats (Figure 20) de l'expérience précédente permet de noter un très bon temps moyen d'exécution (avec impression des résultats dans le spooling de CP) des JOBS en activant en parallèle 3 ou 4 machines virtuelles OS PCP :

11 mn 28 s et 11 mn 36 s

Rappelons le temps d'exécution du batch test, dans les meilleures conditions, sur la machine réelle, par le système OS/MVT :

10 mn 53 s

(Notons que ce temps ne tient pas compte de l'impression des résultats sur l'imprimante, et qu'il représente le temps total d'exécution de tous les JOBS du batch test). Si les temps d'exécution du batch test sont très voisins (4,3 % de différence) il est délicat de chiffrer les performances des systèmes OS/PCP sous CP, par rapport à celles du système OS/MVT sur la machine réelle. Trop de différences existent entre ces deux systèmes et entre les conditions dans lesquelles ont été réalisées les expériences.

b- Machines virtuelles OS/PCP en parallèle avec des machines virtuelles standards.

Lorsque la charge du système CP augmente, les performances des machines virtuelles OS/PCP diminuent de façon sensible.

En activant en parallèle :

- 3 machines virtuelles OS/PCP
- 5 machines virtuelles standards
- la machine mesure

Le temps total d'exécution du batch avec impression dans le spooling de CP est de :

19 mn 37 s

Prenons là encore, comme temps de référence, le meilleur temps d'exécution du batch par le système de multiprogrammation OS/MVT sur la machine réelle:

10 mn 53 s

Nous notons, dans ces conditions, une augmentation de 43 % du temps d'exécution sans qu'il soit possible d'établir une comparaison précise, entre les performances des systèmes OS/PCP sous CP par rapport à celles du système OS/MVT sur la machine réelle.

3- Charge apportée à CP

Pour la gestion d'une machine virtuelle OS/PCP, le temps CPU passé dans CP est égal à 20 % du temps CPU total utilisé. Donc très nettement inférieur au pourcentage de temps CPU passé dans CP pour la gestion d'une machine virtuelle OS/MVT (50 %).

Notons de plus que la charge apportée à CP, par l'activation de 3 machines virtuelles OS/**PCP** et de 5 machines virtuelles standards, est inférieure à la charge habituelle du système CP. La machine virtuelle mesure nous donne pour ce fonctionnement les résultats suivants :

- Temps "problème" : 46,98 %
- Temps "WAIT" : 33,95 %
- Temps passé dans CP : 19,07 %

II - ETUDE DU SYSTEME OS MFT sous CP

La gestion par le système CP de plusieurs machines virtuelles OS fonctionnant avec un système de monoprogrammation ayant donné des résultats satisfaisants, utilisons, dans cette étude, un système OS MFT avec une seule partition (exécution de JOBS de façon séquentielle).

Pour supprimer la multiprogrammation au niveau du système OS, démarrons la tâche "writer" à la fin de l'exécution des travaux.

A l'initialisation du système OS , chargeons en mémoire virtuelle la liste standard des modules préconisés par IBM.

1- Etude de la variation du nombre de machines virtuelles OS/MFT et du nombre de machines virtuelles standards.

Exécutons l'ensemble des JOBS du batch test répartis dans successivement 1,2,3 puis 4 machines virtuelles OS/MFT.

Augmentons ensuite la charge de CP en activant en parallèle des machines virtuelles standards.

Regroupons dans la Figure 23 ci-dessous l'ensemble des résultats obtenus.

Portons sur un graphique : Figure 24, les temps d'exécution de l'ensemble des JOBS du batch test répartis entre 1,2,3 puis 4 machines virtuelles OS/MFT une partition.

Portons sur un graphique : Figure 25, les temps d'exécution de ce même ensemble de travaux répartis entre 3 machines virtuelles OS/MFT fonctionnant en parallèle avec des machines virtuelles standards.

M A C H I N E S M F T

Machines virtuelles	Temps problème %	Temps WAIT %	Temps dans CP %	MFT1		MFT2		MFT3		MFT4		TE Moyen	TI Moyen	Temps total
				TE	TI	TE	TI	TE	TI	TE	TI			
1 MFT 1 mesure	37	49	13	36'	6'							36'	6'	42'
2 MFT 1 Mesure	68	15	17	18'40"	3'40"	24'20"	2'30"					21'30"	3'10	24'40"
3 MFT 1 Mesure	68	13	19	14'50"	3'10"	17'50"	1'20"	15'10"	3'10			16'	2'40"	18'40"
4 MFT 1 mesure	64	9	27	17'	2'30"	22'10"	1'	8'	1'40"	9'30"	1'40"	14'20"	2'20"	16'40"
3 MFT 3 standard 1 Mesure	63	8	28	16'40"	4'	21'30"	2'	15'40"	4'40"			18'	3'30"	21'30"
3 MFT 5 standard 1 Mesure	65	9	26	17'30"	5'10"	21'15"	1'40"	17'40"	4'40"			18'50"	3'55"	22'45"
3 MFT 8 standard 1 Mesure	63	9	27	17'	9'	20'35"	3'10"	18'50"	6'20"			18'50"	6'10"	25'

TE = Temps d'exécution

TI = Temps d'impression dans le spooling CP

Figure 23

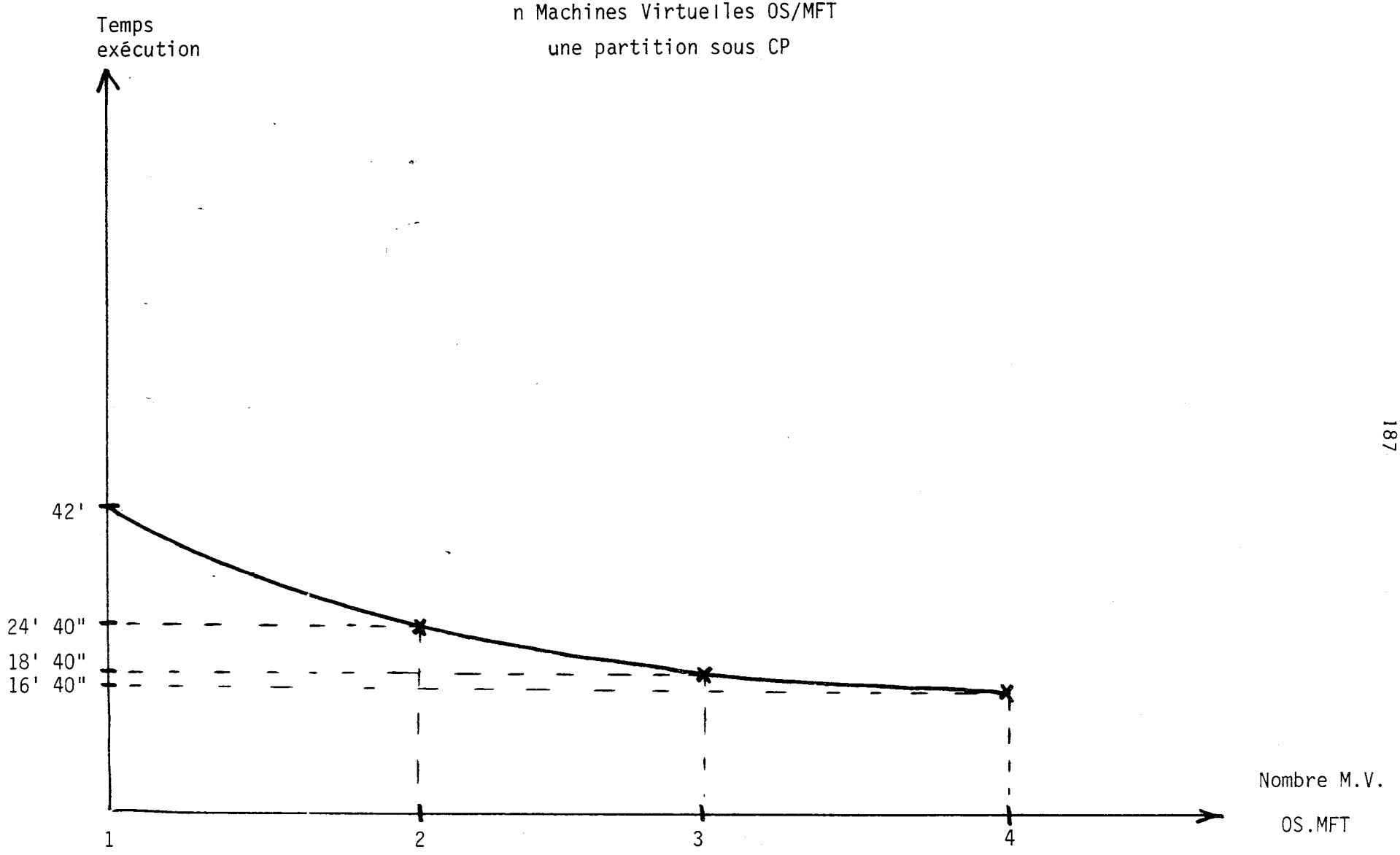

n Machines Virtuelles OS/MFT
une partition sous CP
Temps
exécution
42'
24' 40"
18' 40"
16' 40"
1
2
3
4
Nombre M.V.
OS.MFT

Figure 24

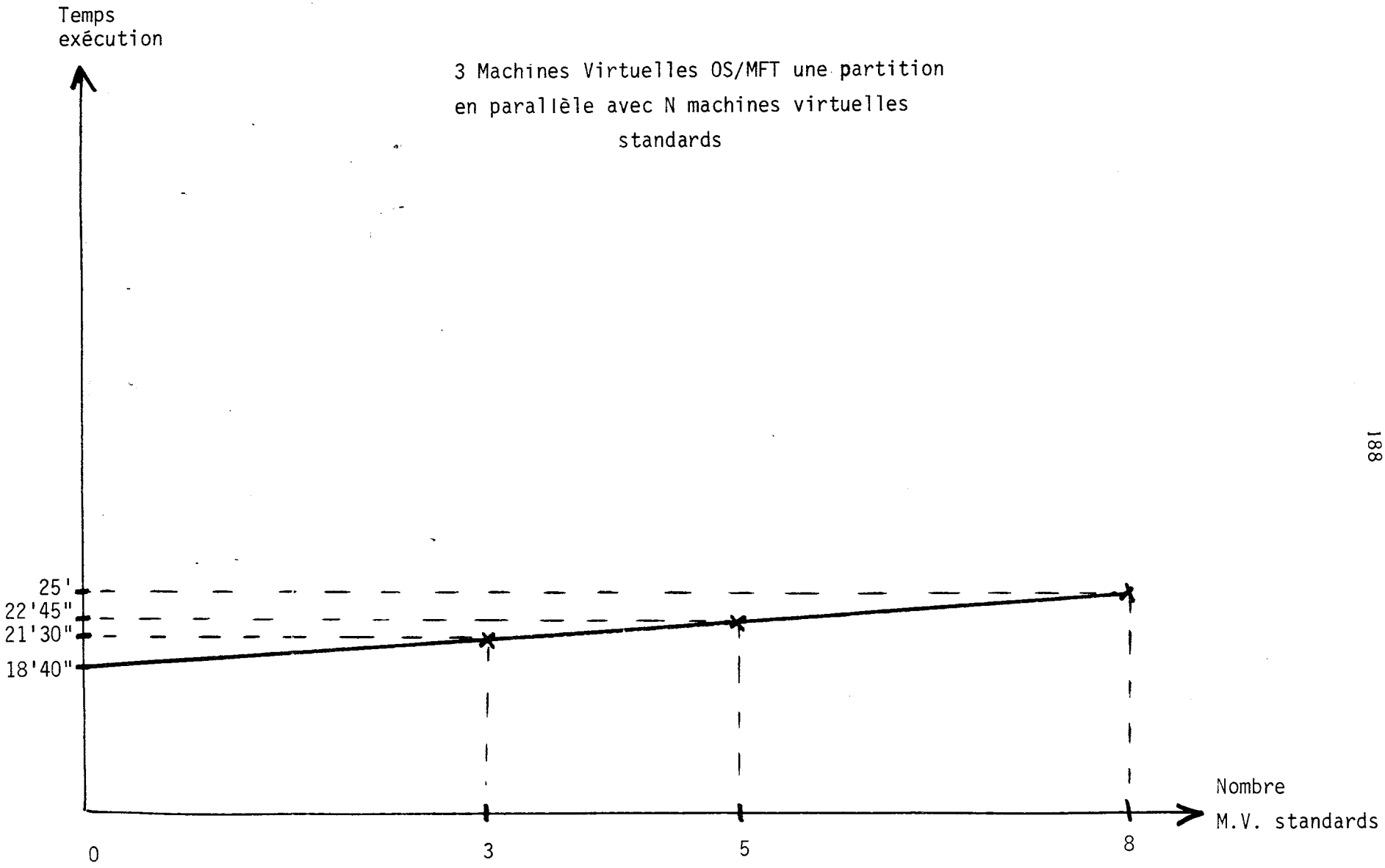

Temps
exécution
3 Machines Virtuelles OS/MFT une partition
en parallèle avec N machines virtuelles
standards
25'
22'45"
21'30"
18'40"
0
3
5
8
Nombre
M.V. standards

Figure 25

2- Performances du système OS/MFT sous CP

a- Machines virtuelles OS/MFT seules utilisatrices de CP

Le meilleur temps d'exécution du batch avec impression des résultats dans le spooling de CP a été obtenu en activant en parallèle 4 machines virtuelles OS/MFT :

16' 40"

Ce temps est supérieur de 5' 40" (soit de 33%) au temps d'exécution du batch par le système de multiprogrammation sur la machine réelle. (10' 53" sans impression réelle des résultats). Bien que cette expérience ait été réalisée avec des systèmes MFT, dont les caractéristiques sont proches de celles du système MVT, il est encore délicat de comparer les performances des systèmes OS/MFT sous CP et celles du système OS/MVT sur la machine réelle.

b- Machines virtuelles OS/MFT en parallèle avec des machines virtuelles standards.

En activant en parallèle :

- 3 machines virtuelles OS/MFT
- 5 machines virtuelles standards
- 1 machine mesure

Le temps d'exécution total du batch avec impression des résultats dans le spooling de CP est de :

22' 45"

soit de 11'45" (ou 51%) supérieur au meilleur temps d'exécution du batch par le système de multiprogrammation OS/MVT sur la machine réelle.

3- Charge apportée à CP

Pour la gestion d'une machine virtuelle OS/MFT, le temps CPU passé dans CP est égal à 30 % du temps CPU total utilisé. Ce pourcentage est intermédiaire entre celui obtenu pour la gestion d'une machine virtuelle OS/MVT et celui de la gestion d'une machine virtuelle OS/PCP.

La charge apportée à CP par l'activation de 3 machines virtuelles OS/MFT et de 5 machines virtuelles standards est inférieure à la charge habituelle du système CP.

La machine virtuelle mesure nous donne pour ce fonctionnement les résultats suivants :

- Temps "problème" : 65,28 %
- Temps WAIT : 9 %
- Temps passé dans CP : 25,72 %

III - CONCLUSIONS

Un ensemble de machines virtuelles fonctionnant sous CP avec le système OS de monoprogrammation PCP donne des résultats nettement plus satisfaisants que ceux obtenus avec une seule machine virtuelle fonctionnant avec le système OS de multiprogrammation MVT.

Ce résultat était prévisible; il est toutefois intéressant de souligner l'efficacité avec laquelle CP gère 3 machines virtuelles fonctionnant avec le système OS PCP (performances voisines de celles du système MVT sur la machine réelle). Cependant il est à noter que le système PCP n'est pas utilisable, seul, en exploitation.

Il ne permet pas le contrôle :

- du temps d'exécution d'un JOB
- de la quantité de lignes écrites ou de cartes perforées.

Notons encore que les machines virtuelles OS/PCP ne constituent pas, pour CP une charge aussi importante qu'une seule machine virtuelle OS/MVT. (Rappelons la baisse considérable des performances de MVT lorsque la charge de CP augmente.).

Les résultats obtenus sous CP avec le système OS/MFT sont intermédiaires entre ceux obtenus par le système OS/PCP et ceux obtenus par le système OS/MVT.

C H A P I T R E VI

CONCLUSION

L'utilisation d'un système de multiprogrammation (MVT) fonctionnant dans un environnement paginé (CP) conduit à une perte de performance d'environ 45 %.

En tenant compte des hypothèses posées :

- Temps de fonctionnement accordé au système batch de multiprogrammation pendant la journée : 3 heures

- Temps réel de fonctionnement compte tenu de la mise en place des systèmes : 2 h 30 mn.

Nous obtenons en 3 heures de fonctionnement sous CP l'équivalent de 1 h 40 mn de machine réelle. Nous notons donc une diminution de 25 % de la quantité de travaux exécutés en 3 heures.

Nous pouvons donc conclure que cette solution n'est pas satisfaisante et ne justifie pas l'utilisation en continu du système de partage de temps pendant la journée.

Rappelons cependant les avantages que présenterait cette solution :

- possibilité d'extension de la session batch sous CP pour les travaux n'utilisant pas de bibliothèques privées
- possibilité de préparer sous CP l'enregistrement des travaux batch (lecteur virtuel) et de continuer l'impression des résultats après l'arrêt de l'exploitation batch (imprimante virtuelle)
- possibilité de continuer les travaux en mode conversationnel pendant la session batch pour certains utilisateurs privilégiés.

Par contre l'utilisation d'un système séquentiel (PCP), fonctionnant dans un environnement paginé (CP) donne des résultats très intéressants: (peu de perte de performance avec 4 PCP sous CP). Mais, rappelons que l'utilisation dans une exploitation de ce système ne permet pas le contrôle :

- du temps d'exécution d'un JOB
- de la quantité de lignes écrites ou de cartes perforées.

La différence essentielle entre les systèmes MVT et PCP est la gestion mémoire (complexe sous MVT très simple sous PCP). Il est donc important d'employer sous CP un système utilisant une gestion mémoire simplifiée, voire spécialement conçue pour la pagination.

Pour améliorer ce fonctionnement, deux solutions peuvent être adoptées :

1- L'écriture d'un système CP spécialisé et simplifié avec :

- simplification du "dispatch" pour cet utilisateur
- création de tables adresse virtuelle adresse réelle
- suppression de la translation des CCW en fixant des pages aux adresses réelles (sauf page 0)
- simulation rapide des instructions privilégiées (LPSW SSM ICK SSK)
- réflexion rapide des interruptions programme
- " " " SVC

2- L'utilisation d'un système OS plus simple dans la machine virtuelle

Ce système pouvant à la limite être séquentiel.

La multiprogrammation peut alors être obtenue par l'activation en parallèle de plusieurs machines virtuelles fonctionnant avec ce système séquentiel (PCP modifié pour être utilisable en exploitation). Cette solution posant en plus le problème de la compatibilité entre plusieurs systèmes d'exploitation.

Le problème de gestion de machine virtuelle existe donc sous CP d'autant plus que la simulation de la machine réelle n'est pas absolue. Il est en effet possible de bloquer le mécanisme de pagination en commandant l'écriture (sur bande magnétique) d'un bloc d'informations de taille supérieure à la taille de la mémoire réelle que peut utiliser CP pour la pagination. La mémoire virtuelle étant bloquée en mémoire réelle pour la réalisation de l'opération d'entrée-sortie.

Réalisons l'expérience suivante :

Hypothèses :

- taille de la mémoire réelle: 1024K octets
- taille de la mémoire virtuelle de la machine OS: 1024K.

Exécutons un programme construisant dynamiquement des programmes canaux en fonction d'un paramètre :

N= nombre de programmes canaux désirés

Chaque programme canal construit, a pour rôle d'écrire sur bande magnétique un bloc de 32760 octets avec chaînage des données. Réalisons une opération d'entrée-sortie pour écrire un bloc de N x 32760 octets.

Le système CP se bloque lorsque N=25, c'est-à-dire lorsque la taille du bloc devient supérieure à 800 000 octets. (780 K octets).
Il est à noter que de tels programmes sont cependant très exceptionnels sur la machine réelle.

La solution proposée par IBM

permettant l'utilisation du concept de mémoire virtuelle pour un système batch de multiprogrammation (technique qui permet l'extension de la taille de la mémoire réelle) a été orientée vers les systèmes OS VS1 et OSVS2.

Ces systèmes sont les nouvelles versions des systèmes OS MFT et OS MVT avec une gestion mémoire complètement modifiée utilisant le concept de mémoire virtuelle avec :

- pagination et segmentation de la mémoire virtuelle
- séparation des données et des descripteurs de ces données (blocs de contrôle regroupés pour supprimer la recherche en liste existant dans le système MFT et MVT) afin de diminuer le nombre de fautes de pages
- concentration et organisation des programmes du système par page.

B I B L I O G R A P H I E

* * * *

I J.P.VIZIMET - G.LEVY

Paging experimentation.

II IBM S/360 GA226895-3

Component descriptions-2301 drum storage
2820 storage control.

III IBM S/360 GA263599-3

Component descriptions. 2314 access storage facility and 2844 auxiliary storage control.

IV IBM S/360 GH200466-6

Attached support processor system- Version 2
Systeme description.

V IBM S/360 GC286550-9

System programmer's guide

VI IBM S/360 GC286554-11

System generation.

VII IBM S/360 GY28-6660-8

Job management (PLM)

VIII IBM S/360 GY28-6659-3

MVT supervisor

IX IBM CP-67/CMS GH20-0859-0

User's guide

TABLE DES MATIERES

* * * * *

			Page
Chapitre I	:	INTRODUCTION	7
Chapitre II	:	DESCRIPTION DU MATERIEL	15
		I-Description de la machine-configuration	16
		II-Description des systèmes utilisés	18
		1- Système de multiprogrammation OS/MVT	18
		2- Système de partage de temps CP-67	26
		III-Description du batch test	30
		1- Caractéristiques d'un batch test	30
		2- Description du batch test utilisé	30
		3- Utilisation de la priorité d'activation	32
		4- Conditions d'exécution du batch test par le système OS/MVT	33
Chapitre III	:	ETUDE DU SYSTEME DE MULTIPROGRAMMATION OS/MVT SUR LA MACHINE REELLE	34
		I-Influence des modules résident en mémoire principale	39
		1- Détermination de ces modules : Expérience AMAP	40
		2- Influence des modules résident en mémoire principale	51
		3- Conclusions	53

Page

II-Influence du type de l'unité à accès direct supportant certains composants du système 54

1- Etude de la position du volume système résident 55

2- Etude de la position de la file d'attente des travaux 57

3- Etude de la position des data sets de travail 58

4- Conclusions 61

III-Influence des paramètres du reader 62

1- Influence du type de l'unité de lecture des jobs 65

2- Influence de la taille des mémoires tampons du reader 66

3- Influence du nombre de mémoires tampons du reader 70

4- Influence du chaînage de la lecture 71

5- Conclusions 73

IV-Influence du nombre d'initiateurs 75

1- Expérience 75

2- Conclusions 78

3- Contre exemple de multiprogrammation 79

V-Conclusions 80

1- Tableau récapitulatif des résultats obtenus séparément 80

2- Choix des valeurs optimales des paramètres de MVT étudiés 80

3- Gain de temps d'exécution obtenu 81

Page

Chapitre IV : ETUDE DU SYSTEME DE MULTIPROGRAMMATION OS/MVT FONCTIONNANT DANS UNE MACHINE VIRTUELLE GEREE PAR LE SYSTEME DE PARTAGE DE TEMPS CP 82

PARTIE I : MACHINE VIRTUELLE OS/MVT SEULE UTILISATRICE DE CP 86

A- MODIFICATION DES PARAMETRES DE MVT 88

I- Influence des modules résident en mémoire virtuelle 88

1- Expérience 88

2- Conclusions 89

II- Influence du type de l'unité à accès direct supportant certains composants du système OS/MVT 91

1- Expérience 91

2- Conclusions 92

III- Influence des paramètres du reader 94

1- Influence du type de l'unité d'entrée des jobs 94

2- Influence de la taille de mémoires tampons du reader 95

3- Influence du nombre de mémoires tampons du reader 97

4- Influence du chaînage de la lecture 99

5- Conclusions 101

IV- Influence du nombre d'initiateurs 103

1- Expérience 103

2- Conclusions 104

V- Conclusions 106

1- Tableau récapitulatif des résultats obtenus séparément 106

Page

2- Choix des valeurs optimales des paramètres de MVT étudiés 106

3- Gain de temps d'exécution réel obtenu 107

4- Remarque: influence de CP sur l'ordre d'activation des JOBS. 107

B- MODIFICATION DES PARAMETRES DE CP 120

I- Influence de la taille de la mémoire réelle et de la mémoire virtuelle 120

1- Expérience 120

2- Conclusions 122

II- Utilisation du lecteur de cartes et de l'imprimante virtuels 125

1- Expérience 126

2- Conclusions 127

3- Remarque: suppression du spooling de OS/MVT en sortie 128

III- Verrouillage des pages les plus utilisées de la machine virtuelle OS/MVT en mémoire réelle 131

1- Détermination de ces pages 132

2- Influence du verrouillage de certaines pages 156

3- Conclusions 159

IV- Conclusions 161

1- Tableau récapitulatif des résultats obtenus séparément 161

2- Choix des valeurs optimales des paramètres de CP étudiés 161

3- Gain de temps d'exécution obtenu 162

Page

C- PERFORMANCE DE MVT SEUL UTILISATEUR DE CP 163

I- Evaluation 163
II- Justification 165
1- Temps global passé dans CP 165
2- Temps passé en recompilation de programmes canaux 165

PARTIE II : MACHINE VIRTUELLE OS/MVT EN PARALLELE AVEC N MACHINES VIRTUELLES STANDARDS 167

A- ETUDE DE LA CHARGE CROISSANTE DE CP 169

I- Détermination de N pour obtenir la charge habituelle de CP 169
II- Conclusions 174

B- PERFORMANCES DE MVT DANS CET ENVIRONNEMENT 175

Chapitre V : ESSAIS COMPARATIFS AVEC LES SYSTEMES OS PCP et MFT SOUS CP 176

I- Etude du système OS PCP sous CP 179
1- Etude de la variation du nombre de machines virtuelles OS/PCP et du nombre de machines virtuelles standards 179
2- Performance du système OS/PCP sous CP 183
3- Charge apportée à CP 184

II- Etude du système OS/MFT sous CP 185
1- Etude de la variation du nombre de machines virtuelles OS/MFT et du nombre de machines virtuelles standards 185

Page

2- Performances de OS/MFT sous CP 189

3- Charge apportée à CP 190

III- Conclusions 191

Chapitre VI : CONCLUSIONS 192

* * * * *

BIBLIOGRAPHIE 197

* * * * *

TABLE DES MATIERES 199

* * * * *