Les techniques du swapp sur disque

William NIVOL - Luc QUONIAM - Albert LATELA

Une nouvelle nécessité : la gestion d'informations

L'avènement des techniques et du matériel informatique, permet aujourd'hui de gérer de grandes quantités d'informations. Que cette information soit de type numérique, binaire ou tout simplement ASCII (Americain Standard Code for Interchange Information), sa gestion nécessite la conception d'outils spécifiques, élaborés et qui permettent de traiter d'importantes quantités de données en un minimum de temps. **Capacité** et **rapidité**, deux critères essentiels qui permettent de donner à tout outil informatique une **réalité pratique** (bien qu'un troisième devienne de plus en plus important : la convivialité interface-utilisateur).

Cette information, en général, se compose en deux parties, les données textuelles et les données factuelles (1). Pour faciliter son traitement, il est fréquent de la transformer en un format unique. Les chiffres étant plus faciles à manipuler que des chaînes de caractères, il n'est pas étonnant que la plupart des méthodes de gestions et de validations résultent de traitements sur des données préalablement transformées en formats numériques.

La méthode du "swapp on disk" (swapping)

Pour travailler sur un nombre important de données, deux solutions se présentent:

- On peut gérer la totalité en mémoire de façon simultanée.
- Ou ne gérer qu'une petite partie des données à la fois, avec échange de celles-ci sur disque : c'est la méthode du **SWAPP ON DISK** (SWAPP sur disque) ou **SWAPPING**.

Les techniques de SWAPP sur disque existent depuis longtemps. Sur les mini et gros-ordinateurs (MAIN-FRAME), tels que sur VAX (@DIGITAL), IBM..., elles sont notamment utilisées pour gérer de façon virtuelle l'ensemble de la mémoire du computer. Sur les micro-ordinateurs, elles sont parfois employées pour gérer du texte au travers d'éditeurs de texte, des données provenant de tableurs...

Nous allons montrer comment réaliser différentes méthodes de SWAPP pour gérer un nombre infini de données en vue de réaliser différents calculs matriciels.

Il existe de nombreux logiciels de calculs numériques mais leurs utilisations sont souvent limitées par le temps de calcul et par le nombre de données maximum que l'on peut traiter.

Les techniques de SWAPPING apparaissent comme une solution pour augmenter ces capacités de traitements.

Bien que leurs mises en oeuvre soient compliquées, le principe général est simple: on ne travaille qu'avec un nombre d'octets constant (paquet) en mémoire. Une fois le traitement réalisé sur cet ensemble d'octets, il y a écriture sur disque des données traitées et lecture d'un nouveau paquet et ainsi de suite jusqu'à traiter la totalité des informations.

Cette méthode procure de nombreux avantages : puisqu'elle fonctionne par "**pages**" de données constantes, elle permet de développer des outils qui sont uniquement limités par la place disponible sur disque et non plus par la place mémoire.

En repoussant ainsi les limites de nos applications, nous verrons qu'il devient tout à fait possible de gérer de façon simple des fichiers de données comportant jusqu'à deux millions d'octets (sous Quick Basic 4.5 et PDS 7.0 @Microsoft).

Ce confort et cette souplesse permettent de récupérer de l'espace de travail dans la zone de mémoire vive et d'augmenter les performances de traitements sur chaque page de données.

Ainsi la limite de cette zone, imposée par le DOS (*Disk Operating System*) à **640 KO**, n'est plus un réel handicap. Cela permet d'éviter de se servir des différentes méthodes de gestion de mémoires hautes (celles situées au dessus des 640 KO) qui possèdent de nombreuses contraintes d'utilisation et de gestion et qui sont encore très onéreuses.

Matériels et méthodes

Les objectifs

Développer plusieurs méthodes de SWAPP sur disque qui permettent de réaliser des calculs sur des matrices de données numériques dont

le nombre maximum ne sera imposé que par les limites de la machine : temps de calcul, capacité de stockage.

Pour tester comparativement les performances des différentes méthodes de SWAPPING, nous utiliserons une matrice de dimensions raisonnables (1000 lignes et 60 colonnes) pour que ces méthodes puissent être comparées à la méthode classique de gestion de données (qui permet de lire et de mettre en mémoire la totalité des valeurs de la matrice, méthode n° 4).

Les différents types de calculs matriciels mis en oeuvre pour tester ces méthodes, sont principalement tous ceux nécessaires à la réalisation de calculs de valeurs et vecteurs propres (inversion et transposition de matrices, calcul de déterminants...) (2).

Les données numériques traitées pourront être en simple ou en double précision.

Les langages de programmation utilisés (Cf 2) permettent de stocker les données de la façon suivante :

- ☞ Un nombre en simple précision = 4 octets
- ☞ Un nombre en double précision = 8 octets

Choix du logiciel de programmation

Nous avons choisi de programmer avec des langages de programmation évolués **Quick Basic 4.5** (QB 4.5) et **Basic PDS 7.0** (Système de développement professionnel en basic).

Les raisons de ce choix :

Nature de l'existant

Il était primordial que nos applications restent homogènes avec d'autres outils d'analyses et de traitements de l'information, développés dans ces mêmes langages.

Langages à la fois simples et extrêmement performants

Bien que que le Basic offre l'avantage d'être un langage d'apprentissage simple, il reste le langage de prédilection pour les utilisateurs de l'informatique non informaticiens. Ce langage est un des plus performants au niveau des traitements de chaînes de caractères.

. Le Basic PDS 7.0 permet d'intégrer toutes les applications qui marchent avec Quick Basic 4.5, si bien que notre logiciel d'abord élaboré avec QB 4.5 sera ensuite "porté" sous PDS 7.0.

Ces deux langages de programmation sont parmi les plus évolués actuellement comme le montre leurs caractéristiques et performances (3) (4):

Caractéristiques communes (QB 4.5 et PDS 7.0)

- Gestion des nombres au format IEEE (Institut of Electrical and Electronics Engineers), avec support du coprocesseur arithmétique. En cas d'abscence, il est simulé.
- Entiers longs codés sur 32 bits (+- 2 147 483 647).
- Valeurs limites des nombres en simple précision :
  +- 3,402823 E +38 valeur maximale
  +- 1,201298 E -45 valeur minimale
- Valeurs limites des nombres en double précision :
  +- 1,7976931 D +309 valeur maximale
  +- 4,9406560 D -324 valeur minimale
- Traitement de chaînes de caractères de longueurs fixes.
- Longueur maximale de chaînes : 32 767 caractères.
- Nouvelles fonctions avancées de traitements des chaînes de caractères.
- Taille maximale des tableaux en gestion dynamique des données utilisateurs, celle de la mémoire.
- Nombre maximum de dimensions dans un tableau : 8
- Entrées/Sorties sur des fichiers binaires.
- Nombre maximum d'enregistrements dans des fichiers à accès direct : 2^{31} avec **32 767** octets au maximum par enregistrement.
- Taille maximale d'un fichier de données utilisateur, celle de l'espace disque disponible.
- Programmation de type procédurale et multi-modulaire.
- Possibilité de gérer plusieurs MODULES en mémoire, 64 Ko au maximum par module, nombre de modules illimité (fonction de la taille mémoire).
- Compatibilité avec la plupart des autres langages (C, Pascal, Fortran, Assembleur).
- Commandes de déboggage intégrées.
- Utilise la technologie du **Threaded P-Code** (compilateur incrémental à simple représentation), il s'agit d'un compilateur mémoire.

Caractéristiques de PDS 7.0

- Gestion de la mémoire paginée E.M.S. (expanded Memory Specification), pour le code utilisateur et les tableaux.
- Addition du support de la mémoire étendue (X.M.S., Extanded Memory Specification) pour l'éxécuteur basic
- Taille maximale des programmes 16 Mo.
- Taille des fichiers éxécutables réduit de 68 % par rapport au Quick Basic 4.5.
- Déplacement des chaînes de caractères (string data) en mémoire lointaine (far memory ou far heap).
- Vitesses des Entrées/Sorties (Input/Output) écran et fichiers augmentées de 45 %.
- Tableaux statiques (static arrays) dans les enregistrements.
- Calculs matriciels 5 fois plus rapides par rapport au Quick Basic 4.5.
- Vitesses d'exécution des fonctions mathématiques simples plus rapides (**entre 1000 et 2000 fois** plus par rapport au Quick Basic 4.5)

Principes de programmation

Principe

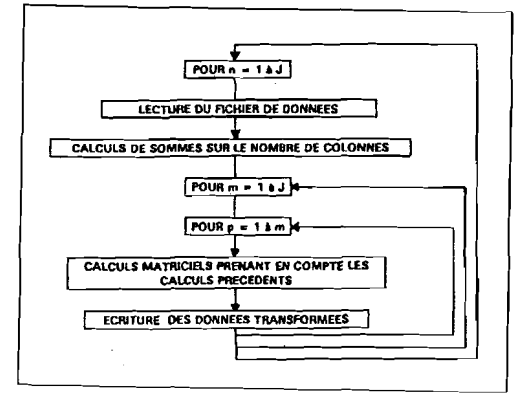
Trois méthodes de SWAPP sur disque ont été réalisées. Bien que différentes, elles sont toutes basées sur le même principe :

- Lecture d'1 ensemble de données
- Traitement
- Ecriture sur disque des données traitées

Les données numériques ne sont pas directement stockées dans des fichiers sous forme matricielle, mais selon un format de stockage propre aux fichiers *"à accès direct"* du Basic. Elles sont répertoriées dans un ordre préétabli et sont écrites en *binaire condensé* (méthode optimisée de stockage de données). Cette technique permet de travailler avec des valeurs qui sont toutes codées sur un nombre d'octets constant. Ceci présente l'avantage d'obtenir un gain de place sur disque et de pouvoir traiter ces données comme des chaines de caractères. Lorsque l'on veut traiter des valeurs numériques stockées dans des fichiers à accès direct, il faut utiliser des fonctions de conversion pour transformer ces données du format binaire condensé (il s'agit en fait de chaînes de caractères) en format numérique.

b) Les calculs testés

Les calculs testés sont réalisés sur la matrice triangulaire de la matrice de départ. Soit A(I, J) cette matrice (I : nombre de lignes, J : nombre de colonnes) :



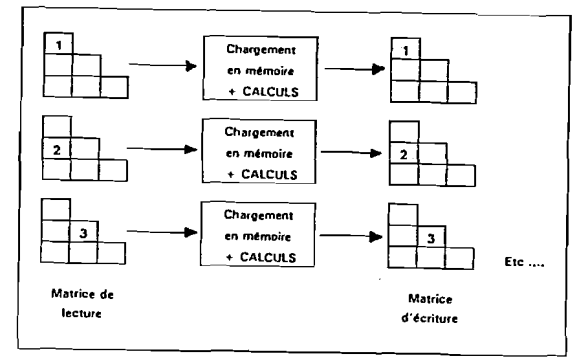
Algorithmes et méthodes de programmation

1ère méthode :

On traite l'ensemble des données, valeur par valeur, chacune étant interprétée comme une chaîne de caractères et convertie en valeur numérique avant les calculs.

Principe :

1-lecture de la valeur (sous forme chaîne) dans le fichier de données.
2-Conversion de la chaîne de caractères en valeur numérique.
3-Calculs matriciels.
4-Conversion de cette nouvelle valeur en chaîne de caractères.
5-Ecriture de cette chaîne dans le fichier de données.



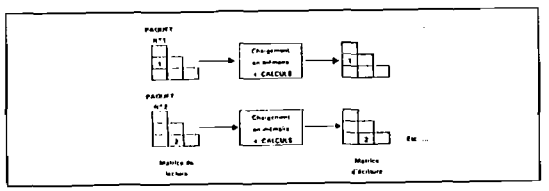
Et ainsi de suite jusqu'à traiter l'ensemble des données.
Comme nous travaillons sans aucun tableau (pour traiter un nombre infini de données), il faut élaborer une relation permettant de déterminer le numéro d'enregistrement et la position dans le fichier de la valeur à extraire en fonction des indices de calculs I et J.
N° enregistrement = (P - 1) * J + m

Pour trouver cette valeur dans le fichier il suffit de se positionner à l'adresse fournie par le numéro d'enregistrement et lire sur un nombre d'octets correspondant à la longueur d'enregistrement du fichier :
en simple précision : longueur = 4 octets
en double précision : longueur = 8 octets

2ème méthode :
On traite l'ensemble des données, **paquet par paquet**, chacun étant **interprété comme une chaîne de caractères** puis converti en valeur numérique avant d'effectuer les calculs.

Le nombre d'octets constituant ces paquets est fonction de la précision des données (simple ou double) et du nombre d'éléments maximum composant la matrice.

Principe :
1 - Définition du nombre d'octets constituant chaque paquet
2 - Lecture du paquet d'octets (sous forme de chaîne) dans le fichier.
3 - Pour l'ensemble de cette chaîne, extraction du morceau de chaîne correspondant à chaque valeur, conversion en numérique, calculs sur cette valeur, conversion en chaîne, reconstitution de la chaîne de caractères initiale.
4 - Ecriture de cette chaîne (paquet) dans le fichier de valeurs.
Et ainsi de suite jusqu'à traiter l'ensemble des données.



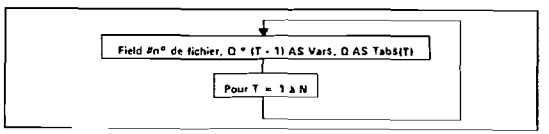
Cette fois, il faut élaborer une formule qui permette de repérer la position de la valeur à traiter parmi toutes celles extraites.
Ipos = ((NuLect - (Nuenr -1) * (Noct / Q))) - Q - 1
Avec :
Nuenr : le numéro d'enregistrement du paquet à extraire :
Nuenr = INT ((Nulect - 1) / Noct / Q)) + 1
Q : le nombre d'octets sur lesquels sont stockées les données (4 ou 8 octets)
Noct : le nombre d'octets constituant chaque paquet :
Noct = Q * nombre de données du paquet
Nulect : le numéro de lecture des valeurs extraites dans la matrice (il s'agit de leur position par rapport aux indices de boucles I et J) :
NuLect = (p -1) * J + m

3ème méthode :
On traite l'ensemble des données, paquet par paquet, mais chacun représente un tableau de chaînes de caractères.

Principe :
La méthode est identique à la précédente, à l'exception du fait que l'on ne travaille plus qu'avec une seule chaîne de caractères mais avec un tableau de chaînes où chaque élément est constitué par une valeur du paquet extrait.

Ceci n'est possible que si l'on utilise la technique des ***FIELD TABLEAUX (5)***, qui permet de réserver une zone tampon dans la mémoire pour la lecture et l'écriture des données sur fichier dans le cas de tableaux.

Le seul changement par rapport à la méthode précédente, intervient dans la détermination de cette zone qui est fonction du nombre d'octets constituant chaque paquet et de la précision des valeurs numériques :



Avec :
N : le nombre de valeurs constituant chaque paquet de données.
Q : le type de précision de données (simple : 4 , double : 8).
Var$: la variable servant de pointeur dans la zone tampon.
Tab$() : le tableau d'éléments chaînes correspondant aux valeurs extraites

Description technique du matériel utilisé.

Appareil : XT/286 (compatible PC) Western digital
Carte mère : Western Digital
Processeur : 286 Intel (12,5 Mhz)
Coprocesseur : 287 Intel (8 Mhz)
Bus SCSI, géré par carte Future Domaine TMC 860
Disque dur : Control Data 190 Mo, 94 221 209, temps d'accès moyen 18 ms
Ports séries : 2
Ports parallèles : 1
Unités de disques : 9; A-I
Mode vidéo utilisé : texte, 25 * 80 couleur
Adaptateur d'affichage vidéo : Enhanced Graphics (EGA), 256 Ko
Waite state : 0
Système d'exploitation : DOS 4.0
Le DOS indique 640 Ko de mémoire :104 Ko utilisés par le DOS et les programmes résidents
536 Ko disponibles pour les programmes d'applications
La recherche de la mémoire active rapporte :
640 Ko de mémoire principale
32 Ko de mémoire d'affichage
3072 Ko de mémoire étendue
Indice de traitement (IT) / à l'IBM / XT = 12,3
Indice de disque (ID) / à l'IBM / XT = 6,3
Indice de performance (IP) / à l'IBM / XT = 10,3
Source : SI - information, édition avancée 4.50 (c) Copr 1987 - 88 Peter Norton

Les résultats (obtenus avec QB 4.5)

Les temps de calculs des trois méthodes proposées sont comparés à celui obtenu par la méthode classique de gestion de données (méthode n° 4). Elles sont utilisées avec des utilitaires du DOS qui permettent de diminuer les temps d'accès à la mémoire et au disque (6) :

RAMDRIVE.SYS : driver qui permet la gestion d'un ou plusieurs disques virtuels en mémoire.

SMARTDRV.SYS : driver qui permet la création d'une antémémoire de disques pour les ordinateurs équipés d'un disque dur et d'une mémoire étendue ou paginée, diminuant le temps d'accès à la lecture.

Les vitesses de travail :
Caractéristiques :

Taille de l'antémémoire (Smartdrv) : 1 Mo Nombres testés en simple précision
Taille du disque virtuel (Ramdrive) : 1 Mo Vitesses calculées en seconde
Taille de la matrice : 1000 * 60

Méthodes utilisées	Principe d'accès aux données			Nb. d'octets traités simultanément	Nb. d'accès disque (lect. + écrit.)
	Fichier (Tps en sec.)	Fichier + Smartdrv (Tps en sec.)	Fichier + Ramdrv (Tps en sec.)		
Méthode n°1	85000	75000	5000	4	3600000
Méthode n°2	12000	11000	9000	20000	7200
Méthode n°3	7000	5000	4000	20000	7200
Méthode n°4	1000	1000	1000	240000	2

La technique de swapp la plus efficace est celle développée au moyen des *FIELD TABLEAUX* (méthode n° 3), la moins rentable est celle qui traite les données une par une (méthode n° 1).

Les différences de temps entre les techniques 2 et 3 s'expliquent par les traitements sur les chaînes de caractères mis en oeuvre au cours de la méthode n° 2 et qui sont relativement lents.

Toutefois, l'expérience montre que lorsque l'on disposera d'un nombre restreint de valeurs, il sera préférable d'utiliser la méthode de gestion de données en mémoire, **classique**, les techniques de SWAPP sont moins rapides (les accès disques étant plus lents que les accès mémoire) mais présentent l'énorme avantage de travailler avec un nombre illimité de données.

Afin de rendre plus performantes ces méthodes, il est conseillé de réorganiser le disque dur avant les traitements. Des essais ont pu montrer des différences de performances variant de 1 à 5.

CONCLUSION

Cette étude permet de montrer que les méthodes de SWAPP sur disque peuvent être utilisées de façon simple sur micro-ordinateur et que de grandes quantités de données peuvent être gérées lorsque les capacités de stockages et le temps de calcul ne sont pas des contraintes discriminantes pour l'utilisateur.

Gestion des données par la mémoire E.M.S.
(Extended Memory Specification, ou mémoire paginée)

L'avantage de l'E.M.S est de pouvoir augmenter le nombre maximum de données gérables de façon simultanée en mémoire. Il est évident que pour l'utilisateur possédant une mémoire E.M.S., les techniques de SWAPP ne présentent plus beaucoup d'intérêt.

Toutefois, le prix du "KiloRam" encore élevé aujourd'hui, redonne une utilité certaine aux techniques de SWAPP qui apparaissent comme une solution parmi d'autres.

Il est important de signaler que nos programmes d'abord élaborés en QB 4.5 puis en Basic PDS 7.0, ont été conçu de telle façon que les différentes méthodes de SWAPP peuvent utiliser la mémoire E.M.S lorsqu'elle est présente.

Bibliographie

1 **500 BANQUES DE DONNEES POUR L'ENTREPRISE**
TELEMATIQUE MAGAZINE, N° M204, 4-6, FEVRIER 1989

2 **STAT-ITCF**
COMMENT INTRODUIRE UN NOUVEAU MODULE DANS STAT-ITCF
I.T.C.F. (INSTITUT TECHNIQUE DES CEREALS ET DES FOURRAGES),
1985, 3-7, 15-17, 11-12, 19-28

3 **MICRO-SYSTEMES T 1508-108**
MAI 1990, N°108, 131-134

4 **MICROSOFT QUICK BASIC 4.5**
GUIDE D'UTILISATION ET MANUEL DE REFERENCE , 1.8-C26-C27
MICROSOFT, 1989

5 **LE LIVRE DU PC BASIC**
MICRO-APPLICATION, 7.212-7.215

6 **MICROSOFT MS-DOS**
GUIDE L'UTILISATEUR ET MANUEL DE REFERENCE, 4.1-4.72 - MICROSOFT, 1990

7 **GUIDE DU PROGRAMMEUR SOUS MS-DOS**
SYBEX, 293-386

8 **L'EDITION DES LIENS - MICROSOFT N°14**
GESTION DE LA MEMOIRE AVEC QUICK BASIC 4.5, 22-27
LES NOMBRES AU FORMAT IEEE, 9-13 - MICROSOFT, 1990

Oui, c'est français!

On nous a accusé - parfois peut-être avec raison - d'user dans cette publication, de termes impropres parce que non conformes aux usages, au dictionnaire, ... en un mot aux censeurs qui assurent la pureté et la conservation de notre belle langue nationale.

Régulièrement nous plaidons coupables, arguant pour notre défense que chaque profession, des médecins aux informaticiens en passant par les pilotes d'avions, avait son jargon et que l'emploi de celui-ci permettait d'être compris entre gens du même bord.

Aujourd'hui, nous sommes satisfaits de voir que l'AFNOR vient nous absoudre en consacrant par la publication dans le Glossaire des termes officiels de l'informatique (2ème édition - octobre 1989) quelques uns de nos termes favoris. Barbarismes d'hier, gallicismes d'aujourd'hui !

Nous en avons choisi quelques uns, parfois avec impertinence, d'autres fois pour leur pertinence, voire pour leur consonance. Afin que cette lecture soit enrichissante pour votre esprit, nous vous proposons de retrouver le terme anglais correspondant. Les solutions se trouvent quelque part, dans ce même numéro.

bus : Dispositif non bouclé destiné à assurer simultanément les transferts d'information entre différents sous-ensembles d'un système informatique selon des spécifications physiques et logiques communes.
Calculette : calculatrice électronique de petite dimension.

clicher : Recopier le contenu, à un instant déterminé, de tout ou partie d'une mémoire sur un autre support.

codet : Groupe d'éléments représentant, selon un code, une donnée élémentaire.

dévideur : Dérouleur de bande magnétique voué par construction à la création en continu de sauvegardes des informations contenues sur un disque.

forme : Ensemble de caractéristiques retenues pour représenter une entité en fonction du problème à résoudre.

implémenter : Réaliser la phase finale d'élaboration d'un système qui permet au matériel, aux logiciels et aux procédures d'entrer en fonction.

instaurer : Mettre dans un état actif[1].

invite : Message visuel ou sonore sollicitant, conformément à une