

P. KELLERT

**Modèles analytiques approximatifs d'un phénomène de blocage systématique dans une architecture multiprocesseur matricielle**

*RAIRO. Recherche opérationnelle*, tome 23, n° 4 (1989), p. 375-391

[http://www.numdam.org/item?id=RO\\_1989\\_\\_23\\_4\\_375\\_0](http://www.numdam.org/item?id=RO_1989__23_4_375_0)

© AFCET, 1989, tous droits réservés.

L'accès aux archives de la revue « RAIRO. Recherche opérationnelle » implique l'accord avec les conditions générales d'utilisation (<http://www.numdam.org/conditions>). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme  
Numérisation de documents anciens mathématiques  
<http://www.numdam.org/>

## MODÈLES ANALYTIQUES APPROXIMATIFS D'UN PHÉNOMÈNE DE BLOCAGE SYSTÉMATIQUE DANS UNE ARCHITECTURE MULTIPROCESSEUR MATRICIELLE (\*)

par P. KELLERT <sup>(1)</sup>

*Résumé. — Le système que nous étudions est une architecture multiprocesseur comportant un ensemble de modules mémoire identiques qui sont systématiquement bloqués durant un certain temps après chaque accès. La connexion entre les processeurs et les mémoires est assurée par un commutateur à point central.*

*Nous proposons, dans cet article, deux modèles analytiques approximatifs qui prennent en compte à la fois les conflits d'accès mémoire et le phénomène de blocage systématique. L'outil mathématique retenu pour la modélisation de ce système est le théorème BCMP.*

Mots clés : Architecture multiprocesseur; réseaux de files d'attente; blocage; BCMP.

*Abstract. — The proposed system is a multiprocessor system which consists of several identical memory modules systematically blocked for a certain amount of time following each access. Each processor can access to any other memory module through a connecting network.*

*In this paper we propose two approximate analytic models which take into account both the memory interference and the systematic memory blocking. The mathematical tool chosen for modelling this system is the BCMP theorem.*

Keywords : Multiprocessor system; queueing networks; blocking; BCMP.

### 1. INTRODUCTION

Parmi tous les systèmes qui appartiennent à la classe des systèmes discrets à partage de ressources, certains ont des ressources qui peuvent être temporairement indisponibles pour les entités qui se les partagent. On dit que de tels

---

(\*) Reçu novembre 1988.

(<sup>1</sup>) Laboratoire d'Informatique, Université Blaise-Pascal, Clermont-II, 63177, Aubière Cedex, France.

systèmes sont soumis à des phénomènes de blocage de certaines de leurs ressources.

La modélisation et l'évaluation des performances de telles architectures a fait l'objet de nombreux travaux qui reposent soit sur l'analyse markovienne à changements d'état discrets, soit sur des méthodes mathématiques approximatives. Les performances de ces systèmes ne peuvent être évaluées directement avec le théorème BCMP [1].

Nous évaluons les performances d'une architecture multiprocesseur matricielle composée d'un ensemble de  $n$  processeurs noté  $\{Pc_1, Pc_2, \dots, Pc_n\}$  <sup>(2)</sup> et d'un ensemble de  $m$  modules mémoire noté  $\{Mp_1, Mp_2, \dots, Mp_m\}$ . Nous proposons deux approches nommées respectivement approche *directe* et approche *itérative* mises en œuvre avec des modèles à réseau de files d'attente qui respectent les hypothèses du théorème BCMP tout en tenant compte des phénomènes de blocage des modules mémoire. Nous obtenons des résultats approximatifs mais suffisants en des temps de calcul très courts. Ces résultats sont validés soit par rapport à des résultats calculés avec l'analyse markovienne, soit par rapport à des résultats estimés par simulation à événements discrets.

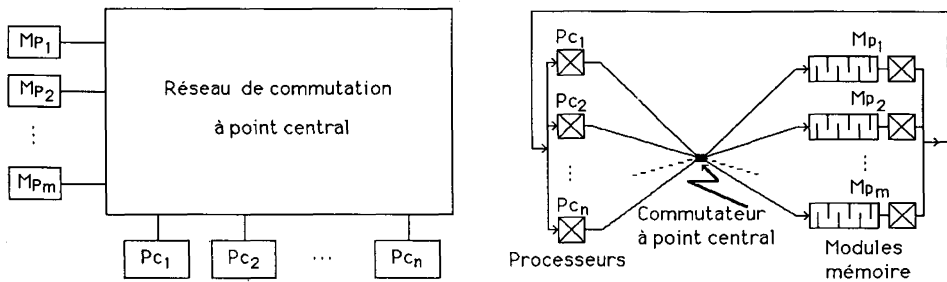
## 2. DESCRIPTION DU SYSTÈME MULTIPROCESSEUR

Ce système comporte  $n$  processeurs homogènes, une mémoire centrale décomposée en  $m$  modules mémoire identiques et indépendants les uns des autres qui ont la particularité d'être bloqués après chaque accès. Ils sont donc indisponibles, durant un certain temps, pour les requêtes processeur : ce mécanisme est appelé Mécanique de Blocage Mémoire noté MBM dans la suite. Chaque processeur peut accéder à n'importe quel module mémoire via un réseau de connexion. La figure 1 donne un schéma de connexions d'un système comportant  $n$  processeurs et  $m$  modules mémoire. Un tel système est noté  $n \times m$ .

Si au moins deux requêtes veulent accéder au même module mémoire, un conflit va naître entre ces requêtes : le fonctionnement de ce système peut donc être modélisé à l'aide d'un réseau de files d'attente. La figure 2 montre un modèle à réseau de files d'attente pour un système  $n \times m$ .

---

(2) Nous utilisons, dans cet article, la notation de Bell et Newell [8] pour décrire l'organisation physique.

Figure 1. — Schéma d'un système multiprocesseur  $n \times m$ .Figure 2. — Réseau de files d'attente d'un système  $n \times m$ .

Les diverses techniques qui ont été proposées pour évaluer ce type de système reposent d'une part sur la théorie des chaînes de Markov à changements d'état discrets [3] et d'autre part sur des méthodes mathématiques approximatives [4, 5, 6]. Leur principale qualité est que les résultats calculés sont obtenus plus rapidement qu'avec l'analyse markovienne. Cependant, seuls les petits systèmes peuvent être étudiés avec ces techniques.

### 3. LES HYPOTHÈSES DE MODÉLISATION

La description exacte du fonctionnement d'un tel système étant trop complexe, certaines hypothèses simplificatrices sont proposées au niveau de ses différentes ressources.

#### 3.1. Hypothèses de fonctionnement des modules mémoire

Les paramètres qui caractérisent les performances d'une mémoire sont son temps d'accès noté  $TA$  et son temps de blocage  $TB$ . Cette dernière opération est effectuée systématiquement après chaque accès. Nous supposons que la durée des accès et que celle des blocages suivent chacune une loi générale de moyenne  $TA$  et  $TB$  respectivement. Les opérations de lecture/écriture sont supposées identiques.

#### 3.2. Hypothèses de fonctionnement des processeurs

On suppose que le temps moyen de traitement d'un processeur suit une loi exponentielle négative de moyenne noté  $TP$ . Le comportement d'un processeur est perçu comme une suite ordonnée d'une requête processeur suivie d'un traitement.

### 3.3. Hypothèses relatives au modèle d'accès

Le modèle d'accès des requêtes processeur aux modules mémoire est supposé être une suite d'épreuves de Bernoulli où la probabilité qu'un processeur émette une requête vers un module mémoire est constante et égale à  $1/m$ .

Nous construisons des modèles analytiques approximatifs qui prennent en compte le MBM. La validation des résultats obtenus est faite avec la méthode d'analyse markovienne (résultats exacts) si le nombre de couples processeur-mémoire est faible ( $\leq 4$ ), la simulation à événements discrets lorsque la méthode précédente n'est plus exploitable. Toutes ces méthodes sont mises en œuvre à l'aide du progiciel QNAP2 [2].

## 4. MODÈLES ANALYTIQUES DU MBM

Les réseaux de files d'attente que l'on peut étudier avec le théorème BCMP doivent avoir une solution stationnaire sous la « forme produit » [7]. Dans le système considéré, chaque module mémoire est bloqué durant une valeur moyenne de  $TB$  unité(s) de temps après chaque accès mémoire. Il est donc indisponible pour une éventuelle requête qui réside dans sa file d'attente même si celle-ci est en tête de la file. Ce phénomène peut être vu comme un blocage systématique de cette ressource après accès. Un tel système ne respecte pas les hypothèses du théorème BCMP.

Les différents cas à examiner sont  $TB \geq TA \geq TP$ ,  $TB \geq TP \geq TA$ ,  $TP \geq TB \geq TA$ ,  $TP \geq TA \geq TB$ ,  $TA \geq TB \geq TP$ ,  $TA \geq TP \geq TB$ . Les cas où  $TA \leq TB$  sont éliminés.

Les notations suivantes sont utilisées dans la suite :

- cproc : classe de clients modélisant l'activité des processeurs;
- cfic : classe de clients fictifs;
- NFIC : nombre total de clients fictifs;
- proc : station serveur infini visitée seulement par les clients de classe cproc;
- fic : station serveur infini visitée seulement par les clients de classe cfic;
- $Mc_i$  : station spécifique modélisant le module mémoire  $i$ , avec  $1 \leq i \leq m$ ;
- $\lambda_{cproc}$  : débit moyen de la classe cproc dans n'importe quel module mémoire;
- $\lambda_{cfic}$  : débit moyen de la classe cfic dans n'importe quel module mémoire.

La modélisation du phénomène de blocage systématique des modules mémoire est réalisée à l'aide du réseau de files d'attente montré par la figure 3

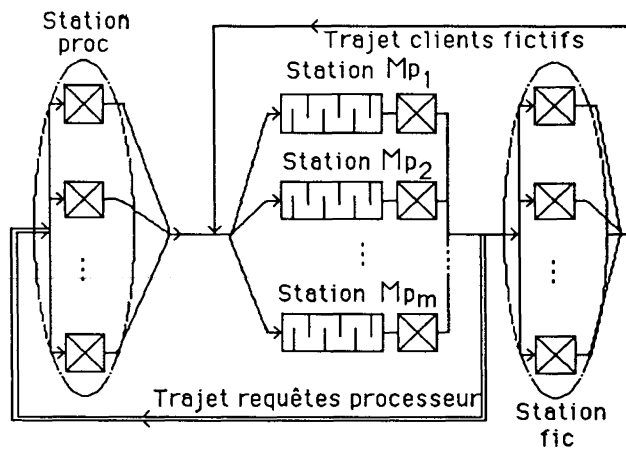


Figure 3. — Réseau de files d'attente associé au MBM et autorisant la mise en œuvre du théorème BCMP.

et dont les caractéristiques sont les suivantes :

- la station proc, où chaque service suit une loi générale de moyenne TP, est initialisée avec  $n$  clients de classe cproc. Toutes les activités sont supposées identiques;

- $m$  stations identiques, appliquant une discipline de gestion de file « processeur partagé », représentant les  $m$  modules mémoire. Chaque station comporte un serveur dont la loi des services suit une loi générale dont la moyenne est fonction de la classe du client;

- pour modéliser le phénomène de blocage, nous introduisons NFIC clients fictifs de classe cfic chargés de retarder « correctement » les clients de classe cproc dans les modules mémoire et qui visitent, après chaque accès, la station fic dont le rôle est de réguler le flot des clients fictifs dans les modules mémoire. Dans cette station, chaque service suit une loi générale de moyenne TRETARD.

Un tel réseau vérifie les hypothèses du théorème BCMP. De plus, puisque ce réseau est fermé, la solution stationnaire sous forme produit existe toujours. Nous étudions les cas précédents avec deux approches différentes.

#### 4.1. Approche par processus itératif

L'égalité des nombres moyens de passages par classe et pour chaque module mémoire est une contrainte à satisfaire. Puisque les modules mémoire sont identiques et que le modèle d'accès aux modules mémoire est uniforme, il suffit de vérifier cette contrainte pour un module mémoire. La contrainte fixée s'écrit pour un module mémoire :

$$|\lambda_{\text{cproc}} - \lambda_{\text{cfic}}| < \varepsilon \quad (4.1.1)$$

où  $\varepsilon$  est un réel donné positif.

Pour déterminer la valeur de TRETARD qui satisfait (4.1.1), nous proposons l'algorithme itératif suivant :

```

pas 1 :  initialisation de  $t_1 : = 0$ ;
           $t_2 : = b * \text{MAX}(\text{TA}, \text{TP})$ ; (où  $b$  est une constante réelle fixée)
          choix de  $\varepsilon$ ;
pas 2 :  calcul de TRETARD :  $=(t_1 + t_2)/2$ ; résolution par méthode
          analytique;
pas 3 :  Calcul de diff :  $=\lambda_{\text{cproc}} - \lambda_{\text{cfic}}$  pour un module mémoire;
pas 4 :  si  $|\text{diff}| < \varepsilon$  alors l'algorithme s'arrête
          sinon si  $\text{diff} < 0$  alors  $t_1 : = \text{TRETARD}$ 
                               sinon  $t_2 : = \text{TRETARD}$ ;
          fsi;
          aller au pas 2;
          fsi;
```

Si  $t_1$  et  $t_2$  sont correctement initialisés, la valeur de TRETARD satisfaisant (4.1.1) peut toujours être théoriquement déterminée.

Dans le cas de l'approche itérative, il est possible de considérer la sous chaîne cfic comme étant ouverte, c'est-à-dire que la station nommée fic devient une source qui génère des clients fictifs dans les modules mémoire, ces clients quittant définitivement le réseau à leur sortie d'un module mémoire. Cette modification est possible puisqu'on travaille sur des flots de clients et qu'on n'impose pas que le nombre moyen de clients fictifs qui résident dans les modules mémoire à l'état stationnaire soit entier. Notons de plus que les temps de calcul sont considérablement diminués puisque l'on traite un réseau mixte donc deux sous-chaînes indépendantes (il n'y a pas de changement de classe des clients). Il faudra assurer la stabilité du réseau par un choix correct de la borne  $t_1$ .

## 4.2. Approche directe

Avec cette approche, nous nous intéressons à la détermination du temps moyen de résidence d'un client de classe cfic en dehors d'un module mémoire.

Le problème à résoudre consiste à :

- déterminer la valeur de NFIC;
- déterminer la valeur de TRETARD.

### 4.2.1. Détermination de la valeur de NFIC

Le modèle d'accès des requêtes processeur étant équiprobable, il faut que le modèle d'accès des clients fictifs soit équiprobable. Si on suppose que TRETARD tient compte du fait qu'un client fictif est associé à un client de classe cproc, le nombre nécessaire et suffisant, NFIC, de clients doit être égal à  $n$  pour que cette solution approche correctement le MBM.

### 4.2.2. Détermination d'une expression de TRETARD

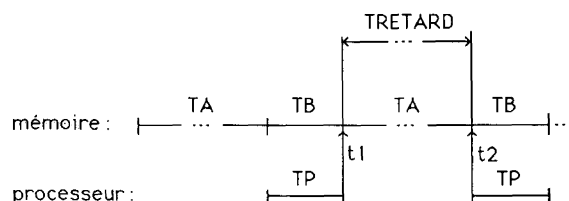
Considérons un système composé d'un processeur et d'un module mémoire. Les trois cas auxquels nous nous intéressons sont  $TP=TB$ ,  $TP>TB$  et  $TP<TB$ .

*Cas  $TP=TB$*

A la fin du blocage (instant  $t_1$ ), le client de classe cfic doit quitter le module mémoire et n'y revenir que pour commencer un nouveau blocage (instant  $t_2$ ). Le temps moyen TRETARD de résidence d'un client de classe cfic hors du module mémoire est donc donné par :

$$TRETARD = t_2 - t_1 = TA.$$

*Cas  $TP=TB$ :*



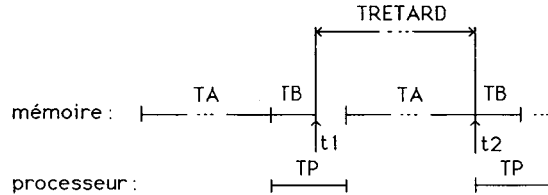


*Cas  $TP > TB$*

Le client de classe cfic doit quitter le module mémoire à la fin du temps  $TB$  (instant  $t_1$ ) et n'y revenir que pour commencer un nouveau blocage (instant  $t_2$ ). L'expression de TRETARD est donc :

$$TRETARD = t_2 - t_1 = TA + TP - TB.$$

*Cas  $TP > TB$ :*

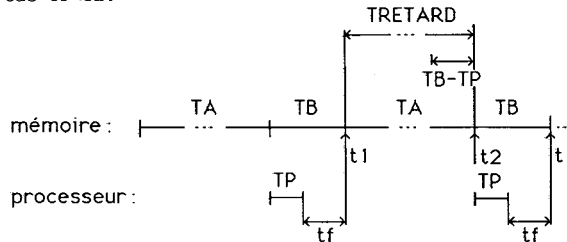


*Cas  $TP < TB$*

A chaque fois que le client fictif visite le module mémoire, la requête processeur subit un temps moyen d'attente, noté  $tf = TB - TP$ , dû au phénomène de blocage. A l'instant  $t_1$ , le client de classe cfic quitte la mémoire et n'y revient qu'à l'instant  $t_2$ , ce client ne devant pas être bloqué par la requête processeur qui lui est associée. D'où :

$$TRETARD = t_2 - t_1 = TA$$

*Cas  $TP < TB$ :*



L'expression de TRETARD s'écrit donc, pour tous les cas :

$$TRETARD = TA + \text{MAX}(TP - TB, 0) \quad (4.3)$$

quels que soient  $TP > 0$ ,  $TB > 0$  avec  $NFIC = n$ .

L'expression de TRETARD ne dépend pas des valeurs de  $n$  et de  $m$  car si un client de classe cproc réside en file d'attente  $T$  unités de temps à cause d'autres clients de classe cproc et donc d'autres clients de classe cfic, et si on

ne veut pas que les départs et les arrivées d'un client de classe cfic soient « décalées » par rapport à son repère absolu qui est le client de classe cproc associé, la requête fictive doit aussi subir un retard égal à  $T$ .

La généralisation de ce raisonnement peut s'exprimer comme suit : à chaque client de classe cproc, ou client « maître », est associé un client de classe cfic, ou client « esclave ». Le nombre d'esclaves qui circulent dans le réseau est donc égal au nombre de maîtres. Un système  $1 \times 1$  définit totalement le comportement moyen d'un esclave vis-à-vis de son maître. Pour un système quelconque, le temps de résidence en file d'attente d'un client esclave doit être égal au temps de résidence de son maître en file d'attente, cette attente résultant des conflits entre maîtres dus au partage des différents modules mémoire.

L'expression (4.3) n'est valable que si le comportement d'un processeur est perçu comme une suite ordonnée d'une requête mémoire suivie d'un traitement par le processeur. Si à la fin d'un accès mémoire, la requête processeur ne retourne pas forcément à son processeur, c'est-à-dire que s'il existe un réel  $\alpha$  avec  $0 < \alpha \leq 1$  tel que :

- la requête processeur retourne à son processeur avec la probabilité  $\alpha$ ;
- la requête processeur retourne faire un accès mémoire avec la probabilité  $(1 - \alpha)$  et un module mémoire est choisi de manière équiprobable,

alors le comportement d'un processeur n'est plus perçu comme une suite ordonnée d'une requête mémoire suivie d'un traitement par le processeur. Un raisonnement identique au précédent conduit à l'expression suivante :

$$T_{\text{RETARD}} = T_A + \alpha * \text{MAX}(T_P - T_B, 0).$$

## 5. COMPARAISON DES DIFFÉRENTS RÉSULTATS

Nous donnons des résultats pour le cas où  $T_P < T_B < T_A$ , tous les autres cas étant étudiés dans [9]. Nous comparons aussi des résultats obtenus avec des résultats proposés par [3].

### 5.1. Cas $T_A > T_B > T_P$

Les différentes valeurs fixées sont  $T_A = 1,2$ ;  $T_B = 0,45$ ;  $T_P = 0,1$ . Le critère de performance choisi est ici le débit en instruction(s) par unité de temps des modules mémoire. Les résultats calculés avec QNAP2 sont donnés par le tableau de la figure 4. On constate que les deux méthodes donnent des

| Couple<br>n x m | Markov | Simulation | Analytique<br>direct | Analytique<br>itératif |
|-----------------|--------|------------|----------------------|------------------------|
| 1x1             | 0.5995 | ---        | 0.5790               | 0.5830                 |
| 1x2             | 0.6644 | ---        | 0.6437               | 0.6633                 |
| 1x3             | 0.6933 | ---        | 0.6686               | 0.6952                 |
| 1x4             | 0.7096 | ---        | 0.6818               | 0.7123                 |
| 2x1             | 0.6060 | ---        | 0.6057               | 0.6055                 |
| 2x2             | 0.8622 | ---        | 0.8776               | 0.8728                 |
| 2x4             | 1.0992 | ---        | 1.1050               | 1.1160                 |
| 2x6             | 1.2126 | ---        | 1.2040               | 1.2300                 |
| 3x1             | 0.6061 | ---        | 0.6061               | 0.6060                 |
| 4x1             | 0.6061 | ---        | 0.6061               | 0.6061                 |
| 4x2             | 1.0088 | ---        | 1.0410               | 1.0206                 |
| 4x4             | 1.5132 | ---        | 1.5740               | 1.5432                 |
| 4x8             | ---    | 2.0203     | 2.0670               | 2.0640                 |
| 6x2             | 1.0686 | ---        | 1.1000               | 1.0786                 |
| 6x8             | ---    | 2.5100     | 2.6130               | 2.5660                 |
| 8x4             | ---    | 1.8606     | 1.9540               | 1.8940                 |
| 8x6             | ---    | 2.4230     | 2.5530               | 2.4730                 |
| 8x8             | ---    | 2.8420     | 2.9980               | 2.9170                 |
| 8x16            | ---    | 3.8950     | 4.0050               | 3.9770                 |
| 16x8            | ---    | 3.5800     | 3.7950               | 3.6570                 |
| 16x16           | ---    | 5.5539     | 5.8600               | 5.6780                 |
| 16x32           | ---    | 8.4410     | 7.7610               | 7.8100                 |
| 32x16           | ---    | 7.5830     | 6.3660               | 7.1910                 |
| 32x32           | ---    | 12.0600    | 10.7000              | 11.2100                |
| 32x64           | ---    | 16.7900    | 15.3700              | 15.4800                |
| 64x32           | ---    | 14.9900    | 12.6200              | 14.2600                |
| 64x64           | ---    | 24.1700    | 21.2400              | 24.4000                |

Figure 4. — Cas TA &gt; TB &gt; TP.

résultats proches de ceux calculés avec la méthode d'analyse markovienne et de ceux estimés par stimulation à événements discrets. Les résultats les moins satisfaisants sont obtenus lorsque les modules mémoire sont soumis à une faible charge. Dans le cas où le système est en surcharge, les résultats sont très satisfaisants d'autant qu'ils sont pessimistes. Enfin, rappelons que les temps de calcul pour la méthode d'analyse markovienne discrète et pour la méthode de simulation à événements discrets sont prohibitifs par rapport à ceux des deux approches.

*Remarque :* Le symbole --- du tableau de la figure 4 signifie que les résultats correspondants ne sont pas calculés soit parce que l'on dispose des résultats exacts avec Markov et donc on n'utilise pas ceux de simulation, soit parce qu'avec Markov ces résultats ne peuvent être calculés (problèmes de place en mémoire, temps de calcul prohibitifs, ...).

## 5.2. Comparaison avec les résultats de [3] (Cas $TP > TB$ )

Ils sont montrés par le tableau de la figure 5. On considère ici qu'à la fin d'un accès mémoire, la requête retourne à son processeur avec la probabilité  $\alpha$  sinon elle retourne à l'un des  $m$  modules mémoire (choisi de manière uniforme).

Le critère de performance choisi est le nombre moyen de mémoire(s) occupée(s). Les résultats de simulation de [3] sont donnés avec un intervalle de confiance à 90 %. Les résultats de simulation estimés avec QNAP2 ont un intervalle de confiance à 95 % calculé à chaque fois avec dix replications. Notons que les résultats obtenus avec la méthode itérative sont pratiquement tous à l'intérieur des intervalles de confiance calculés avec QNAP2 bien que les lois des temps de service des modules mémoire soient des lois constantes dans le modèle de simulation.

*Remarque :* Le symbole ... du tableau de la figure 5 signifie que les résultats correspondants ne sont pas disponibles dans [3].

## 6. COMPARAISON DE L'APPROCHE ITÉRATIVE ET DE L'APPROCHE DIRECTE

L'approche itérative impose l'égalité  $\lambda_{cproc} = \lambda_{cfic}$ . L'approximation faite dans cette approche est que les clients fictifs ne sont pas prioritaires lorsqu'ils visitent un module mémoire. Il en résulte que le temps de résidence d'un client fictif dans un module mémoire est surestimé : la méthode itérative a donc tendance à surestimer le nombre moyen de mémoire(s) occupée(s).

Quant au modèle analytique direct, il impose à chaque client fictif un temps moyen de résidence en dehors d'un module mémoire égal à :  $TRETARD = TA + \text{MAX}(TP - TB, 0)$ . Cette condition est vraie en moyenne, mais puisque la discipline de gestion d'une file d'un module mémoire n'est pas du type « priorité », il n'y a aucune raison que l'on ait  $\lambda_{cproc} = \lambda_{cfic}$ .

La figure 6 schématise la tendance pour les temps de calcul avec les différentes méthodes. L'axe des abscisses représente le nombre de couples processeur-mémoire tandis qu'en ordonnée sont portés les temps de calcul.

| nxm | $\alpha$ | [3]    | [3]        |  | QNAP2      |  | QNAP2  | QNAP2    |
|-----|----------|--------|------------|--|------------|--|--------|----------|
|     |          | Markov | Simulation |  | Simulation |  | direct | itératif |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |
|     |          |        |            |  |            |  |        |          |

Figure 5. — Comparaison avec les résultats de [3] (TP &gt; TB).

## 7. LIMITES DES DEUX APPROCHES

### 7.1. Limites de l'approche itérative

Avec cette approche tous les modules mémoire doivent être identiques (même temps de service pour chaque classe, même politique de gestion, ...). Pour montrer les limites de l'approche itérative, on introduit un réseau noté

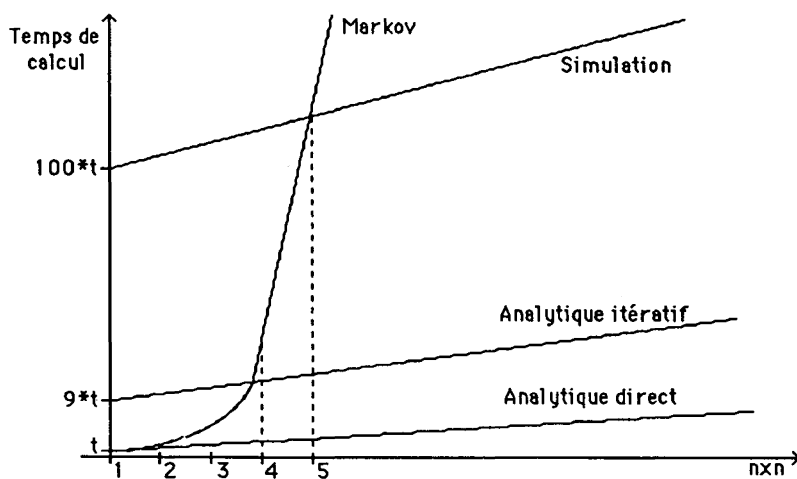


Figure 6. — Tendance pour les calculs avec les différentes méthodes.

$R$  constitué d'un réseau  $R'$  qui comprend  $m$  modules mémoire et la station fictive fic. La figure 7 montre le réseau  $R$ .

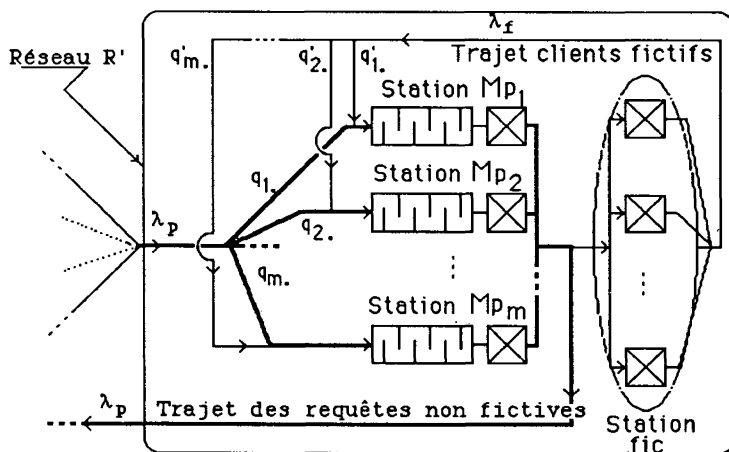


Figure 7. — Réseau  $R$  formé d'un réseau BCMP et d'un réseau  $R'$  qui exploite le MBM.

Soient :

- $K$  : le nombre de classes de clients non fictifs;
- $C = \{c_1, c_2, \dots, c_K\}$  l'ensemble des classes des clients non fictifs qui visitent les modules mémoire;
- $\text{Card}(c_i)$  : le nombre de clients de classe  $c_i$  qui visitent les modules mémoire, avec  $1 \leq i \leq K$ ;
- $\lambda_p$  : le flux des clients qui se présentent devant les modules mémoire et dont la classe appartient à l'ensemble  $C$ ;
- $\lambda_f$  : le flux des clients fictifs qui se présentent devant les modules mémoire;
- $\lambda_{M_{c_i}, C}$  : le débit moyen des clients non fictifs du module mémoire  $i$ , avec  $1 \leq i \leq m$ ;
- $\lambda_{M_{c_i}, \text{fic}}$  : le débit moyen des clients fictifs du module mémoire  $i$ , avec  $1 \leq i \leq m$ ;
- $q_{ij}$  : la proportion du flot  $\lambda_p$  composée de clients de classe  $c_j$  qui accèdent au module mémoire  $i$ , avec  $1 \leq i \leq m$  et  $1 \leq j \leq K$ ;
- $q_i$  : la proportion du flot  $\lambda_p$  des clients qui accèdent au module mémoire  $i$ , avec  $1 \leq i \leq m$ , soit :

$$q_i = \sum_{j=1}^{j=K} q_{ij} \quad \text{et} \quad \sum_{i=1}^{i=m} q_i = 1$$

- $q'_i$  : la proportion du flot  $\lambda_f$  des clients fictifs accédant au module mémoire  $i$ , avec  $1 \leq i \leq m$ .

Avec ces notations, le nombre total de clients fictifs qui visitent la station fictive notée fic s'écrit :

$$\text{NFIC} = \sum_{i=1}^{i=K} \text{Card}(c_i).$$

Nous supposons que le modèle d'accès au module mémoire reste de type bernoullien mais n'est plus forcément uniforme, c'est-à-dire :

$$\lambda_{M_{c_i}, C} = q_i \cdot \lambda_p.$$

Si le modèle d'accès des clients fictifs est identique à celui des clients dont la classe appartient à l'ensemble  $C$ , c'est-à-dire si :

(7.1.1)  $q_i = q'_i$ , quel que soit  $i$ , avec  $1 \leq i \leq m$ , alors une condition nécessaire et suffisante pour que l'approche itérative reste correcte est que  $\lambda_p = \lambda_f$ . (7.1.2).

En effet, si :

(7.1.3)  $\lambda_{M_{c_i}, C} = \lambda_{M_{c_i}, \text{fic}}$  quel que soit  $i$ , avec  $1 \leq i \leq m$   
alors :

(7.1.3)  $\Leftrightarrow q_i * \lambda_p = q'_i * \lambda_f \Leftrightarrow \lambda_p = \lambda_f$  puisque  $q_i = q'_i$ , quel que soit  $i$ , avec  $1 \leq i \leq m$ .

En particulier si l'égalité (7.1.3) est vraie pour un module, elle l'est pour tous les autres puisque nous raisonnons sur des proportions.

Les conditions (7.1.1) et (7.1.2) sont nécessaires puisqu'il y a blocage systématique des requêtes processeur et elles sont suffisantes (pour l'approche itérative) d'après les hypothèses.

Si le réseau  $R$  n'est pas fermé, le serveur infini de la station fic est remplacé par une source génératrice de clients, la quantité NFIC ne sera donc plus entière. Dans ce cas, l'algorithme converge si et seulement si le réseau  $R$  est stable, c'est-à-dire qu'il faut vérifier, pour tout  $i$  :

$$\sum_{j=1}^{j=K} q_{ij} * \lambda_p / \mu_j + q'_{it} * \lambda_f / \mu_{\text{fic}} < 1$$

où  $\mu_j$  est le taux de service d'un client de classe  $c_j$  appartenant à  $C$  dans un module mémoire, et  $\mu_{\text{fic}}$  est le taux de service des clients fictifs dans un module mémoire.

La condition (7.1.3) s'écrit :

$$\sum_{j=1}^{j=K} q_{ij} * \lambda_p / \mu_j = q'_{it} * \lambda_p / \mu_{\text{fic}}$$

c'est-à-dire :

$$\sum_{j=1}^{j=K} q_{ij} * \lambda_p / \mu_j < 1/2 \quad \text{pour tout } i, \quad 1 \leq i \leq m$$



qui s'écrit :

$$\text{MAX}_{1 \leq i \leq m} \left( \sum_{j=1}^{j=K} q_{ij} * \lambda_p / \mu_j \right) < 1/2.$$

Les limites de l'approche itérative sont que toutes les ressources subissant un blocage soient identiques et isolées dans un sous-réseau unique, c'est-à-dire qu'il n'y a qu'un point d'entrée et qu'un point de sortie dans ce sous-réseau pour les clients dont la classe appartient à l'ensemble  $C$ . Notons que la seule condition imposée au réseau  $R$  dont on exclut le sous-réseau  $R'$  est qu'il respecte les hypothèses du théorème BCMP. Ainsi, dans notre étude, les processeurs pourraient être plus complexes (processeurs non homogènes, distinction entre les opérations de traitement, d'entrée-sortie, accès à des mémoires-caches, etc.).

## 7.2. Limites de l'approche directe

Avec cette approche, il ne suffit pas que toutes les ressources qui subissent le blocage soient identiques.

Il faut en plus :

- connaître le temps moyen de résidence d'un client non fictif dans le réseau  $R$  sans les stations mémoire et la station fictive ;
- que tous les clients non fictifs aient le même temps de service dans les stations mémoire.

Avec de telles contraintes, il nous semble *a priori* que l'approche directe n'est pas généralisable car trop particulière. Notons, cependant, qu'elle fournit des résultats acceptables avec des temps de calcul très courts.

## 8. CONCLUSION

Nous avons proposé deux approches approximatives, nommées approche itérative et approche directe, et avons fixé leurs limites. Bien que les résultats qu'elles fournissent ne soient qu'approchés, elles ont le mérite d'être simples et rapides à exploiter. Leurs résultats sont validés par comparaison avec des résultats obtenus : soit avec la méthode d'analyse markovienne, soit avec des simulations à événements discrets. Il apparaît que l'approche itérative est plus robuste que l'approche directe dont la qualité première est de fournir des résultats très rapidement.

## BIBLIOGRAPHIE

1. F. BASKETT, K. CHANDY, R. MUNTZ et F. PALACCIO, *Open, Closed and Mixed Networks or Queues with Different Classes of Customers*, J.A.C.M., vol. 22, n° 2, avril 1975.
2. D. POTIER, *New Users Introduction to QNAP2*, Rapport technique INRIA, n° 40, oct. 1983.
3. D. P. BHANDARKAR, *Analysis of Memory Interference in Multiprocessors*, I.E.E.E. Trans. Comput., vol. C-24, sept. 1975, p. 897-908.
4. F. BASKETT et A. J. SMITH, *Interference in Multiprocessor Computer System with Interleaved Memory*, Commun. Ass. Comput. Mach, juin 1976.
5. C. H. HOOGENDOORN, *A General Model for Memory Interference in Multiprocessors*, IEEE Trans. Comput., vol. C-26, oct. 1977, p. 998-1005.
6. B. R. RAU, *Interleaved Memory Bandwidth in a Model of a Multiprocessor Computer System*, I.E.E.E. Trans. Comput., vol. C-28, n° 9, sept. 1979, p. 678-681.
7. G. PUJOLLE, *Réseaux de files d'attente à forme produit*, Rapport de Recherche I.N.R.I.A., n° 19, mai 1980.
8. C. G. BELL et A. NEWELL, *Computer Structures: Readings and Examples*, New York, McGraw-Hill, 1971.
9. P. KELLERT, *Modélisation de systèmes complexes avec QNAP2*, Thèse de Docteur-Ingénieur, Clermont-Ferrand, janvier 1987.