UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À

L'UNIVERSITÉ DU QUÉBEC À CHICOUTIMI

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INGÉNIERIE

PAR

TUONG VINH HO

MODÉLISATION ET OPTIMISATION DE SYSTÈMES MULTIPROCESSEURS HIÉRARCHIQUES DANS UN CONTEXTE D'INTÉGRATION À TRÈS GRANDE ÉCHELLE

MAI 1994

Bibliothèque
Paul-Émile-Boulet

UQAC
Université du Québec
à Chicoutimi

Mise en garde/Advice

Afin de rendre accessible au plus grand nombre le résultat des travaux de recherche menés par ses étudiants gradués et dans l'esprit des règles qui régissent le dépôt et la diffusion des mémoires et thèses produits dans cette Institution, **l'Université du Québec à Chicoutimi (UQAC)** est fière de rendre accessible une version complète et gratuite de cette œuvre.

Motivated by a desire to make the results of its graduate students' research accessible to all, and in accordance with the rules governing the acceptation and diffusion of dissertations and theses in this Institution, the **Université du Québec à Chicoutimi (UQAC)** is proud to make a complete version of this work available at no cost to the reader.

L'auteur conserve néanmoins la propriété du droit d'auteur qui protège ce mémoire ou cette thèse. Ni le mémoire ou la thèse ni des extraits substantiels de ceux-ci ne peuvent être imprimés ou autrement reproduits sans son autorisation.

The author retains ownership of the copyright of this dissertation or thesis. Neither the dissertation or thesis, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

RÉSUMÉ

Afin de construire des systèmes d'ordinateurs de haute performance utilisant les technologies d'intégration à très grande échelle, une nouvelle architecture de systèmes multiprocesseurs hiérarchiques est proposée. Le présent projet consiste en la modélisation de la performance et l'optimisation de cette architecture. Une méthodologie permettant d'obtenir des modèles d'estimation de performance pour l'architecture hiérarchique est proposée. Le temps moyen requis par un message pour se rendre à sa destination (aussi appelé temps de réponse) est utilisé comme mesure de performance. Ces modèles permettent de prédire la performance d'architectures multiprocesseurs hiérarchiques avec une précision adéquate pour une vaste gamme de conditions de trafic et de configuration. Ces modèles permettent également d'optimiser la conception de systèmes hiérarchiques de grande dimension. En effet, puisque le temps de réponse d'un système affecte le temps d'exécution d'une application, la connaissance de la sensibilité de ce dernier en fonction de certains paramètres est essentielle pour la conception de systèmes parallèles de haute performance basés sur cette architecture.

REMERCIEMENTS

Je voudrais exprimer mes remerciements auprès du professeur Daniel Audet, mon directeur de recherche à l'Université du Québec à Chicoutimi, qui par ses questions pertinentes, son ouverture d'esprit et ses conseils m'a permis de réaliser ce travail. Je le remercie particulièrement pour avoir corrigé mon mémoire.

Je remercie également tous les membres d'ERMETIS (Équipe de Recherche en Microélectronique et Traitement Informatique des Signaux) pour leur soutien technique.

Je remercie le programme canadien de bourses de la Francophonie qui m'a attribué une bourse d'études pendant la durée de ma maîtrise.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION

L'évaluation des performances de nouveaux types de systèmes multiprocesseurs utilisant les technologies d'intégration à très grande échelle est une tâche essentielle pour les architectes d'ordinateurs parallèles de haute performance. Une telle évaluation permet, entre autres, de prédire si certains choix architecturaux sont plus avantageux que d'autres. Elle permet ainsi l'optimisation de l'efficacité de ces derniers avant même d'entreprendre leur construction. Le projet proposé consiste en la modélisation de la performance et l'optimisation d'une nouvelle architecture de systèmes multiprocesseurs hiérarchiques. Un des objectifs du projet consiste à caractériser l'efficacité du réseau de communication de cette architecture à l'aide de modèles. Afin d'étudier le comportement de ces systèmes en détail ainsi que pour

valider les modèles analytiques développés, un simulateur de systèmes hiérarchiques a été construit. Notons que le but visé lors du développement des modèles analytiques est de permettre des analyses plus approfondies sans avoir à recourir à de longues simulations. Cette architecture hiérarchique permet de construire des systèmes multiprocesseurs puissants pouvant regrouper plusieurs milliers de processeurs. Notons que dans de tels systèmes la tolérance aux défaillances peut être réalisée à un coût relativement faible en comparaison avec d'autres architectures.

Dans le cadre de ce projet, une méthodologie permettant d'obtenir des modèles d'estimation de performance pour l'architecture précédemment décrite est proposée. Le temps de réponse du système est utilisé comme indice de performance. Ces modèles permettent de prédire la performance d'architectures multiprocesseurs hiérarchiques avec une précision adéquate pour une vaste gamme de conditions de trafic et de configuration. Ces modèles permettent également d'optimiser la conception de systèmes hiérarchiques de grande dimension. En effet, puisque le temps de réponse d'un système affecte le temps d'exécution d'une application, la connaissance de la sensibilité de ce dernier en fonction de certains paramètres est essentielle pour la conception de systèmes parallèles de haute performance basés sur cette architecture.

Le présent mémoire est organisé comme suit: le chapitre 1 présente une vue d'ensemble de la modélisation de la performance de systèmes informatiques. Le chapitre 2 se veut une brève introduction à la théorie des files d'attente appliquée à la modélisation de la performance. Le chapitre 3 est, quant à lui, consacré au développement de modèles analytiques de la performance pour les systèmes hiérarchiques. Un simulateur de systèmes multiprocesseurs hiérarchiques est par la suite présenté au chapitre 4. La validation des modèles développés et l'analyse de la performance de certaines architectures hiérarchiques sont discutées au chapitre 5. Finalement, on retrouve au chapitre 6 la conclusion de ce projet ainsi que certaines perspectives de recherche future.

CHAPITRE 1

INTRODUCTION À LA MODÉLISATION DE LA PERFORMANCE DES SYSTÈMES INFORMATIQUES

1.1 Rôle de la modélisation de la performance

De nos jours, certains systèmes informatiques sont si complexes que l'intuition seule n'est pas suffisante pour prédire leur performance. Ainsi divers outils, tels la simulation et la modélisation mathématique jouent un rôle important dans la conception de ces systèmes. La performance est un facteur clé dont il faut tenir compte. Plusieurs méthodes peuvent être utilisées pour évaluer cette performance. Par exemple, on peut utiliser des moniteurs matériels et/ou logiciels, soit dans un environnement d'utilisateur, soit sous des conditions contrôlées (banc d'essai). Notons que les moniteurs matériels sont des appareils et les moniteurs

logiciels sont des ensembles de programmes servant à la mesure des performances d'un système. Cependant, pour un système informatique dont la performance ne peut être mesurée avant que les phases de conception et de développement soient terminées, il est nécessaire de faire des prédictions de performance basées sur mesures statistiques en utilisant des modèles. En plus des prédictions quantitatives, la modélisation permet également de mieux comprendre la structure et le comportement du système au cours du développement. Ceci peut ultimement aider à découvrir et à corriger certains défauts de conception.

Plusieurs outils mathématiques ont été développés afin d'obtenir des mesures (indices) de performance tels que le taux d'utilisation, le débit, le temps de réponse, etc. Dans le reste de ce chapitre, nous allons faire une brève introduction aux divers outils mathématiques existants ainsi qu'aux techniques de simulation.

1.2 Modèles analytiques de la performance

Un système d'ordinateurs peut être considéré comme un ensemble de ressources matérielles (unités centrales, mémoires, etc.) et logicielles (programmes

de système, compilateurs, etc.). Les performances d'un système dépendent directement de l'organisation et de la gestion de ces ressources. Une tâche (un élément du programme implanté dans la mémoire de l'ordinateur) est le traitement d'une transaction. Généralement, seule une tâche peut utiliser une ressource à un instant donné. Ainsi, toutes les autres tâches devant utiliser cette ressource doivent attendre dans une file d'attente. La théorie des files d'attente est particulièrement bien adaptée à l'analyse des performances de systèmes informatiques. À titre d'exemple, il est possible, avec cette théorie, d'estimer le temps que les tâches doivent attendre dans diverses files d'un système. Ces temps peuvent ensuite être analysés afin de prédire le temps de réponse du système. Notons que le temps de réponse est généralement un indice important de la performance d'un système. Une introduction de la théorie des files d'attente sera présentée au chapitre 2.

On notera qu'il existe une autre technique très répandue permettant d'établir des modèles analytiques de la performance de systèmes informatiques. Cette technique est basée sur l'utilisation de réseaux de Pétri ("Petri net"). Il est possible d'obtenir plus de détails sur cette technique en se référant à un livre écrit par Ajmone Marsan et al. [13].

1.3 Modèles de simulation de la performance

La simulation est une technique importante de modélisation des systèmes informatiques. Il existe deux principales méthodes de simulation, à savoir les méthodes en temps continu (analogique) et celles en temps discret. Cependant, dans le présent projet, nous ne nous intersserons qu'aux méthodes discrètes, c'est-à-dire, à la simulation à événement discrets.

Le principe de base d'une simulation est de suivre tous les changements d'un phénomène (décrit à l'aide d'un modèle) évoluant dans le temps. L'opération de base d'un simulateur de ce type consiste à changer la valeur des variables qui définissent l'état du système au fur et à mesure que le temps de simulation s'écoule. Dans la simulation à événements discrets, l'espace d'état du système considéré est fini ou dénombrable et chaque changement d'état (événement) se produit de façon discrète. Ces instants de changement d'état sont appelés occurrences d'événement. Un événement peut être l'arrivée d'une transaction ou la réalisation d'un service. Dans un contexte général, ces événements se produisent de façon aléatoire. Ainsi, les techniques d'estimation statistique sont alors utiles dans l'analyse d'un système.

Notons finalement que le choix d'un langage de simulation constitue un élément important lors du développement d'un simulateur. Celui-ci a une influence significative sur le temps de développement d'un modèle. Les langages de simulation spécialisées ou les logiciels de simulation exigent moins de temps de développement et facilitent plusieurs tâches communes telles que la vérification et les analyses statistiques. En contrepartie, les langages de programmation généraux sont plus flexibles et fournissent plus de contrôle sur l'efficacité et sur le temps d'exécution des simulations.

1.4 Validation des modèles de la performance

Le développement d'un modèle de performance consiste à caractériser les ressources matérielles et logicielles qui composent le système par des paramètres. Le choix de ces paramètres et une manière de les représenter sont à la base de la modélisation. Notons qu'au cours du développement de modèles de performance, plusieurs hypothèses peuvent être introduites afin de simplifier la complexité d'un système. La validation de ces modèles devient donc essentielle. Souvent plusieurs modèles ayant différents niveaux de détail sont développés en vue de représenter un système. Par exemple, il peut y avoir un modèle de simulation très détaillé et un

modèle analytique plus simple. Si le système n'a pas encore été réalisé, alors la validité du modèle plus simple pourrait être obtenue en comparant ses prédictions avec celles obtenues avec le modèle plus détaillé.

1.5. Résumé

Dans les sections précédentes, une vue d'ensemble de la modélisation des systèmes informatiques a été présentée. Il s'agit généralement de techniques relativement bien connues, parmi lesquelles on retrouve des modèles analytiques et des modèles de simulation. Ces techniques présentent chacune des avantages et des inconvénients dont la comparaison peut guider l'utilisateur. Les modèles analytiques permettent d'obtenir rapidement des prédictions mais elles sont moins précises que les modèles de simulation qui eux donnent des prédictions plus détaillées mais, en contrepartie, exigent plus de temps de simulation.

Une méthodologie générale pour la modélisation des systèmes informatiques pourrait donc schématiquement être composée des étapes suivantes:

- développement d'un modèle analytique du système,

- développement d'un modèle de simulation du système,

- validation de ces modèles en comparant leurs prédictions.

Cette approche sera utilisée en vue de la modélisation de systèmes multiprocesseurs hiérarchiques dans ce qui suit.

CHAPITRE 2

INTRODUCTION IE LA THÉORIE DES DES FILES D'ATTENTE APPLIQUÉE AUX SYSTÈMES INFORMATIQUES

Le but de ce chapitre est de réaliser un survol de la théorie des files d'attente et de discuter brièvement des problèmes de la modélisation de performance de systèmes informatiques.

2.1 Définition d'un système des files d'attente

Pour la plupart d'entre nous, l'observation de files d'attente est quasi quotidienne. En effet, très souvent des individus doivent former une file afin d'attendre un "service" qui nécessite un temps de traitement appelé "temps de

service". Les individus formant une file sont appelés, dans un contexte plus général, des "clients". On a coutume d'appeler "file d'attente" l'ensemble des clients qui attendent d'être servis, à l'exclusion de celui qui est en train de recevoir le service. On nomme système d'attente, l'ensemble des clients qui font la file et celui qui reçoit le service. Dans la figure 2.1, on montre les éléments d'un système de files d'attente ouvert. À l'entrée du système, il y a une population (c'est-à-dire une source de clients). Ces clients désirent un certain service (par exemple, la transmission d'un message, le traitement d'une demande, ou le service d'une demande entrée/sortie). Dans le centre de service, il peut y avoir un ou plusieurs serveurs. Si tous les serveurs sont occupés lorsqu'un client entre dans le système, le client joint une file jusqu'à ce qu'un serveur soit disponible. Des notions de base de la théorie des files d'attente sont présentées dans la section suivante.

2.2 Spécification d'un système de files d'attente

Les notions de base suivantes [2] sont nécessaires pour aborder l'étude des files d'attente et déterminer le temps de réponse d'un système informatique.

a) Le taux d'arrivée moyen des clients (λ): Cette notion peut se définir comme étant le nombre de clients arrivant en moyenne par unité de temps, c'est-à-dire $\lambda=1/E(t)$, où $E(t)$ est le temps entre les arrivées.

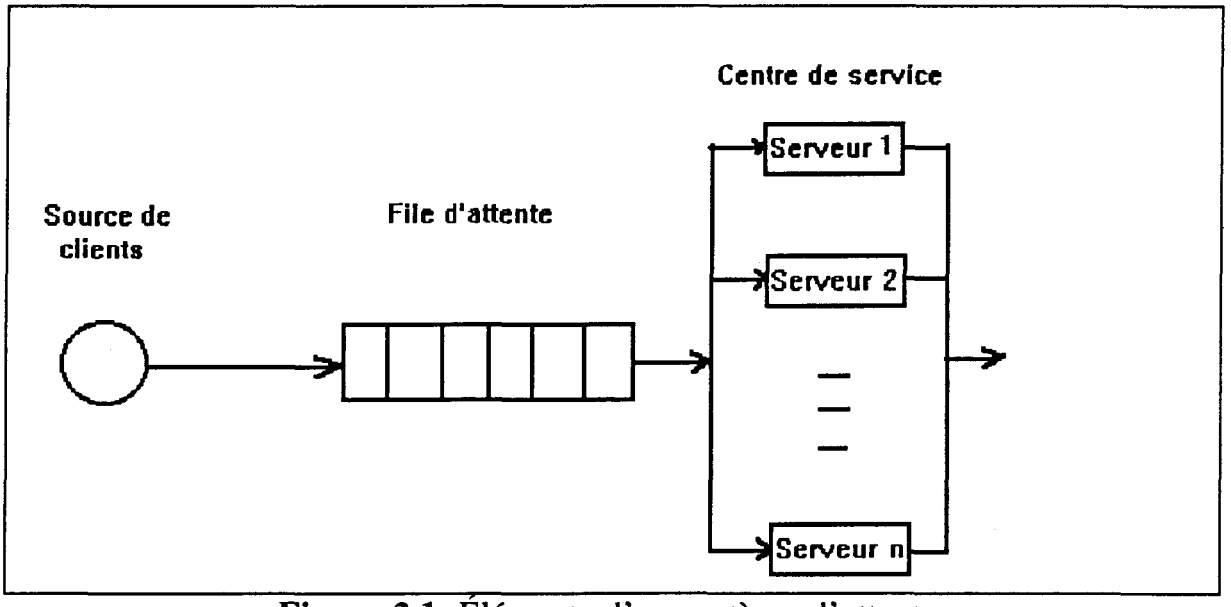

Figure 2.1. Éléments d'un système d'attente.

b) Le taux de service moyen (μ): C'est le taux moyen auquel on peut servir les clients par unité de temps, $\mu= 1/E(s)$, où $E(s)$ est la durée moyenne du service exécuté par le serveur.

c) La longueur moyenne de file d'attente (L_q)**:** C'est le nombre moyen de clients dans la file lorsque le système est dans un état stationnaire.

d) Le temps d'attente moyen (W): C'est le temps moyen qu'un client doit attendre avant d'être servi.

e) Le débit du système (λ_T): C'est le débit moyen d'un système informatique mesuré en événements ou interactions par unité de temps.

f) Le taux d'utilisation (ϱ): C'est la probabilité qu'un serveur donné soit occupé.

g) Loi de Little ("Little's law"): La loi de Little permet de relier les indices de la performance d'un système.

$$L_q = \lambda W.$$

h) Les distributions statistiques: Le processus d'entrée et le méchanisme des services suivent des distributions statistiques. La distribution la plus utilisée est la distribution exponentielle. Autres distributions, telles que la distribution d'Erlang et la distribution hyperexponentielle, sont également utilisées dans la simulation des files d'attente.

Nous proposons dans les pages qui suivent quelques exemples concrets d'application de la théorie des files d'attente pour l'évaluation des caractéristiques de systèmes informatiques.

2.3 Application d'un modèle ouvert de files d'attente

Le système d'attente le plus simple est appelé M/M/1. Cette appellation caractérise un système ayant des arrivées et un temps de service qui suivent une distribution exponentielle (M/M) et n'ayant qu'un seul serveur (/1). Il est considéré ouvert puisque les clients entrant dans le système proviennent de l'extérieur, reçoivent ensuite des services et sortent du système. Les systèmes fermés, c'est-à-dire, ceux dont les clients ne sortent jamais, seront considérés plus tard. Nous présentons ici un exemple montrant comment utiliser la file M/M/1 pour évaluer le temps de réponse d'un ordinateur.

Soit un ordinateur central d'un système informatique. À cet ordinateur sont connectés n terminaux conformément au schéma de la figure 2.2. Les terminaux peuvent être éloignés géographiquement et être aussi bien des écrans de visualisation que des terminaux lourds.

Supposons que le temps de traitement d'une tâche sur l'unité centrale de l'ordinateur suive une distribution exponentielle [7] et que sa moyenne soit de 500

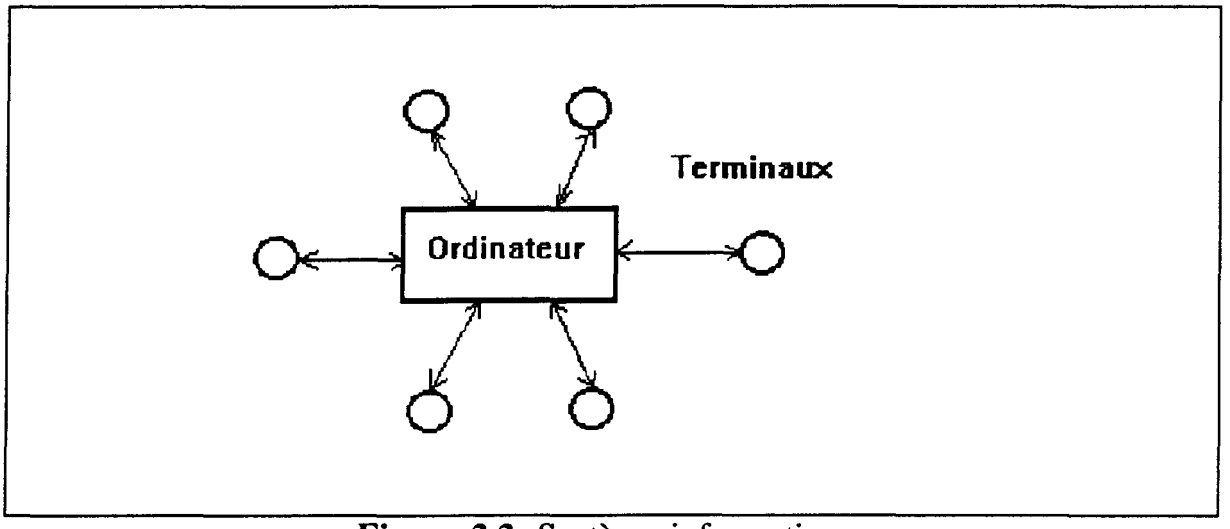

Figure 2.2. Système informatique.

ms. L'intervalle de temps entre deux requêtes provenant d'un même terminal est supposé suivre également une loi exponentielle de moyenne égale à 20s. Le schéma représentant ce système est présenté à la figure 2.3. C'est une file M/M/1 ayant un taux moyen de service $\mu = 1/0.5 = 2$ requêtes par seconde et un taux moyen d'arrivée $\lambda = (1/20)*n$ puisque la superposition de flots indépendants suivant une loi de Poisson (exponentielle) suit également une loi de Poisson.

Nous avons représenté sur la figure 2.4 le temps de réponse pour une tâche en fonction du nombre de terminaux *n*.

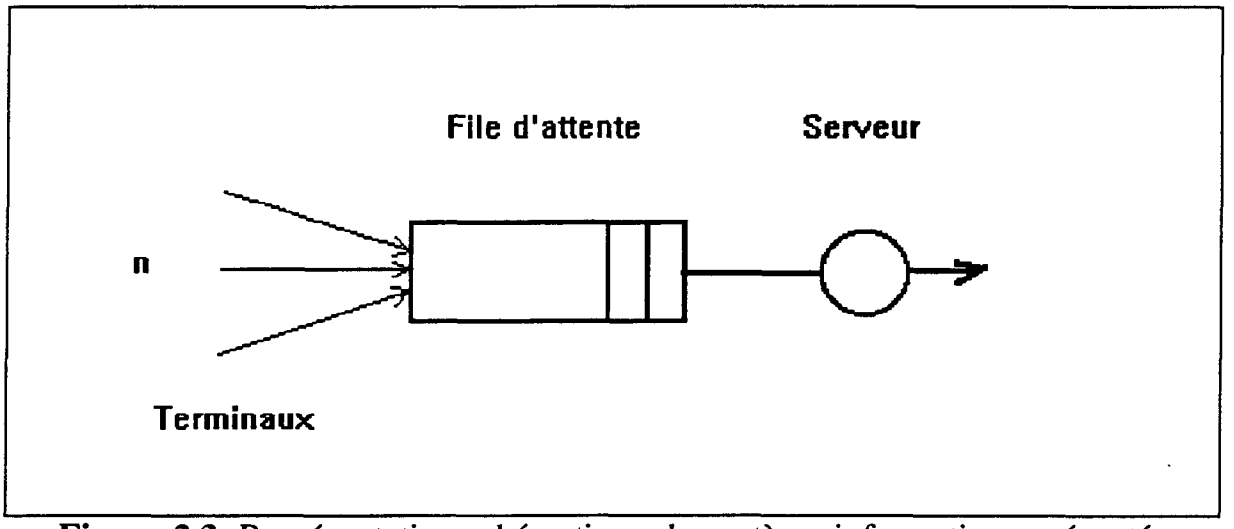

Figure 2.3. Représentation schématique du système informatique présenté à la figure 2.2.

Le temps de réponse de ce système est donné par la relation [2]:

$$W = \frac{1}{\mu(1-\varrho)} \qquad où \qquad \varrho = \frac{\lambda}{\mu} = \frac{n}{40}. \tag{2.1}$$

Pour que le système puisse effectuer toutes les tâches, il faut que $\varrho < 1$. On constate ici que le nombre de terminaux doit être strictement inférieur à 40 pour que W soit fini.

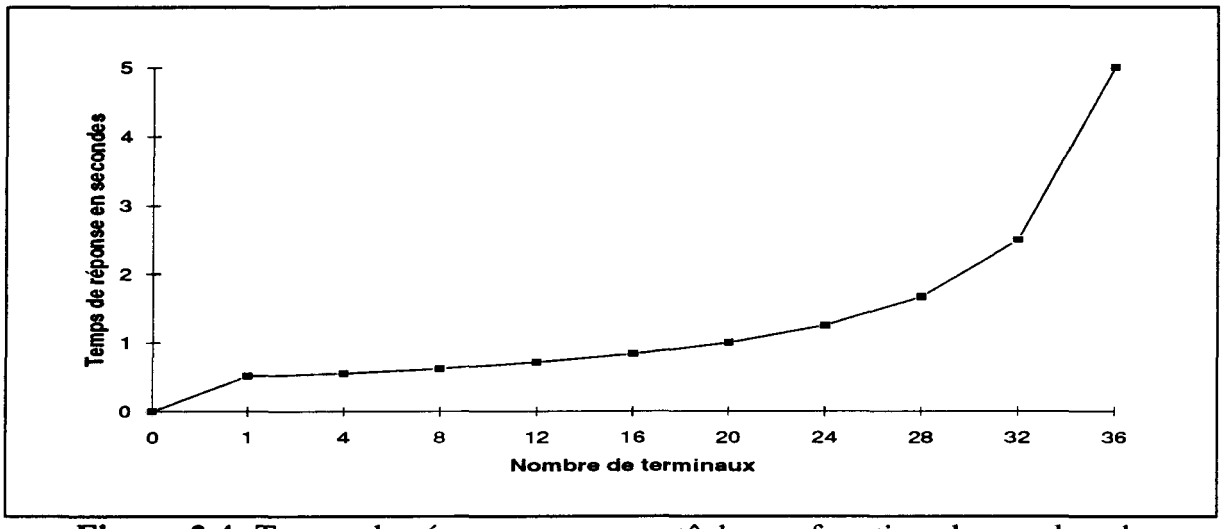

Figure 2.4. Temps de réponse pour une tâche en fonction du nombre de terminaux.

2.4 Application d'un modèle fermé de files d'attente

Dans l'exemple précédent, nous avons utilisé un modèle de files d'attente ouvert avec un nombre infini de clients. Cependant, il existe peu de systèmes qui soient réellement ouverts. Nous allons considérer des modèles de files d'attente plus réalistes, soient des modèles fermés. Dans ces systèmes, aucun client n'entre ni ne sort. De plus, un nombre fini de clients circulent dans ces systèmes.

Pour étudier les systèmes d'ordinateurs à temps partagé, un modèle relativement général est souvent utilisé [2, 7, 9]. Dans ce système, le processeur

central possède un temps de service qui suit une loi générale (arbitraire) et la discipline "premier arrivé, premier servi" est employée dans la file d'attente (comme dans le cas d'un système M/M/1). On considère dans ce modèle que la capacité de traitement de l'unité centrale (CPU) est divisé également entre les requêtes à servir. Ainsi, s'il y a *n* interactions attendant un service de l'unité centrale, chacune d'entre elles serait donc servie à un taux de μ/n clients par unité de temps, où μ est le taux de service d'unité centrale.

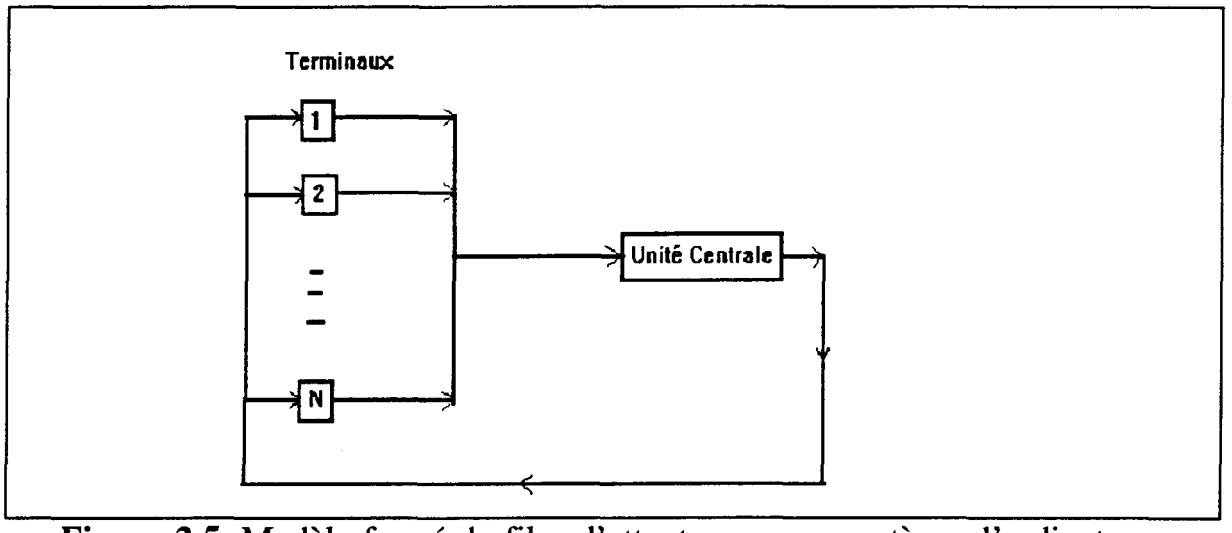

Figure 2.5. Modèle fermé de files d'attente pour un système d'ordinateurs interactifs.

Les équations qui décrivent ce modèle sont données à la table 2.1.

Table 2.1. Modèle fermé de files d'attente pour systèmes d'ordinateurs interactifs dans la figure 2.5.

Temps de réponse moyen:

$$W = \frac{N\,E(s)}{1-p_0} - E(t) \qquad \text{(A)}$$

Débit en interactions par unité de temps:

$$\lambda_T = \frac{\varrho}{E(s)} \qquad \text{(B)}$$

Dans ces équations

- E(s) = temps de service moyen par interaction,
- p_0 = probabilité que le système soit dans un état oisif,

$$p_0 = \left[\sum_{n=0}^{N} \frac{N!}{(N-n)!}\left(\frac{E(s)}{E(t)}\right)^n\right]^{-1} \qquad \text{(C)}$$

- ϱ = taux d'utilisation de l'unité centrale,

$$\rho = 1 - p_0 \qquad \text{(D)}$$

- E(t) = temps moyen entre deux requêtes successives à un terminal (temps de "réflexion").

Nous allons maintenant considérer un exemple concret. Soit un système d'ordinateurs interactifs à temps partagé ayant 20 terminaux actifs, un temps de "réflexion" moyen de 3 secondes, un taux de service d'unité centrale (CPU) de 500,000 instructions par seconde (0.5 MIPS). Considerons, de plus, qu'une demande de service exige en moyenne 100,000 instructions. On désire trouver le temps de réponse moyen W, le débit moyen λ_T, le taux d'utilisation d'unité centrale ϱ et le nombre de demandes en attente dans l'unité centrale L_q.

Puisque chaque demande de service a besoin d'exécuter en moyenne 100,000 instructions, on a

E(s) = 100000 / 500000 = 0.2 secondes.

Par l'équation C (table 2.1), p_0, la probabilité que l'unité centrale soit en état oisif, est donnée par

$$p_0 = \left[\sum_{n=0}^{20} \frac{20!}{(20-n)!}\left(\frac{0.2}{3}\right)^n\right]^{-1}$$

$$p_0 = 0.045593216$$

Ensuite, l'équation A (table 2.1) permet d'obtenir le temps de réponse moyen W

$$W = \frac{20\ 0.2}{1-0.045593216} - 3 = 1.191\ \textit{secondes}$$

Le taux d'utilisation de l'unité centrale ϱ, est donné par

$$\varrho = 1 - p_0 = 1 - 0.045593216 = 0.9544.$$

Le débit moyen du système est donc égal à:

$$\lambda_T = \frac{\varrho}{E(s)} = \frac{0.9544}{0.2} = 4.722 \textit{ interactions par seconde.}$$

Par la loi de Little, le nombre moyen de demandes en attente dans l'unité centrale est:

$$L_q = \lambda_T W = 1.191 * 4.722 = 5.68 \text{ demandes.}$$

2.5 Résumé

Dans ce chapitre, nous avons effecté un bref survol de la théorie des files d'attente. Nous avons montré comment cette dernière peut être utilisée pour la modélisation de la performance des systèmes informatiques en utilisant des exemples concrets. Ces notions de base serviront aux lecteurs lors des prochains chapitres.

CHAPITRE 3

MODÉLISATION DE SYSTÈMES MULTIPROCESSEURS HIÉRARCHIQUES

3.1 Introduction

Ce chapitre est consacré au développement de modèles analytiques de performance d'une architecture multiprocesseur hiérarchique. Un des objectifs du présent projet de recherche consistait à déterminer l'efficacité du réseau d'interconnexion de cette architecture. Le temps de réponse introduit par le protocole d'échange de données et par la structure du réseau d'interconnexion constitue un indice de la performance d'un réseau de communication. Ce temps de réponse est généralement exprimé sous la forme d'un délai moyen qui dépend de plusieurs facteurs dont le nombre de niveaux du réseau d'interconnexion, le nombre de processeurs dans le système et la fraction du temps d'exécution consacrée aux

activités de communication. En modifiant certains de ces paramètres, nous pouvons analyser la performance d'un réseau d'interconnexion pour une vaste gamme de conditions de trafic et de configuration.

Afin d'évaluer le temps de réponse moyen d'un système hiérarchique, nous allons développer des modèles analytiques basés sur des modèles de réseaux de files d'attente. Nous introduirons donc des hypothèses et des contraintes qui permettront de caractériser les systèmes multiprocesseurs hiérarchiques. Dans les sections suivantes, nous allons présenter en détail le développement de ces modèles.

3.2 Architecture des systèmes multiprocesseurs hiérarchiques

Dans les systèmes multiprocesseurs hiérarchiques [8, 17], l'échange de données s'effectue à travers un réseau d'interconnexion basé sur une hiérarchie. Ce réseau est constitué de trois types de bus appelés respectivement bus de bas niveau, bus de niveau intermédiaire et bus de haut niveau. Dans une des configurations les plus simples, ces bus sont interconnectés de façon à former un arbre binaire. Les aiguilleurs utilisés pour relier ces bus n'acceptent que certains messages en se basant

sur le contenu d'un champ d'adresse inclus dans ces derniers. Les processeurs sont connectés au réseau d'interconnexion par des bus de bas niveau. Chaque processeur est aussi relié à une mémoire locale qui lui est propre. Des exemples de configuration hiérarchique à 2 et à 4 niveaux sont illustrées aux figures 3.1 et 3.2.

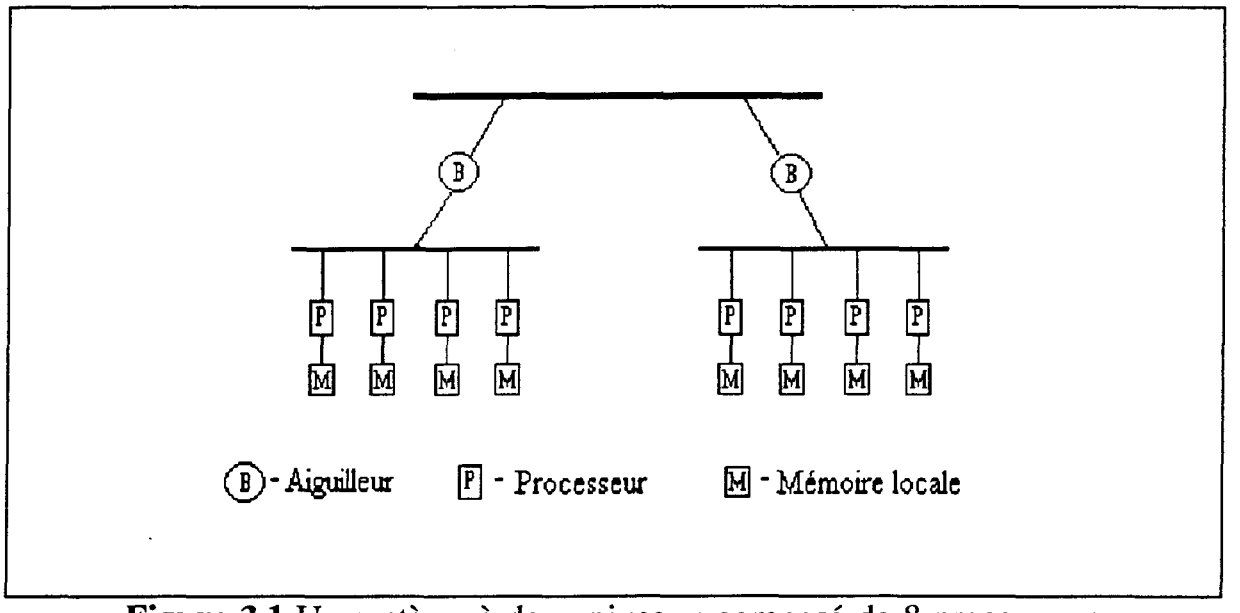

Figure 3.1 Un système à deux niveaux composé de 8 processeurs

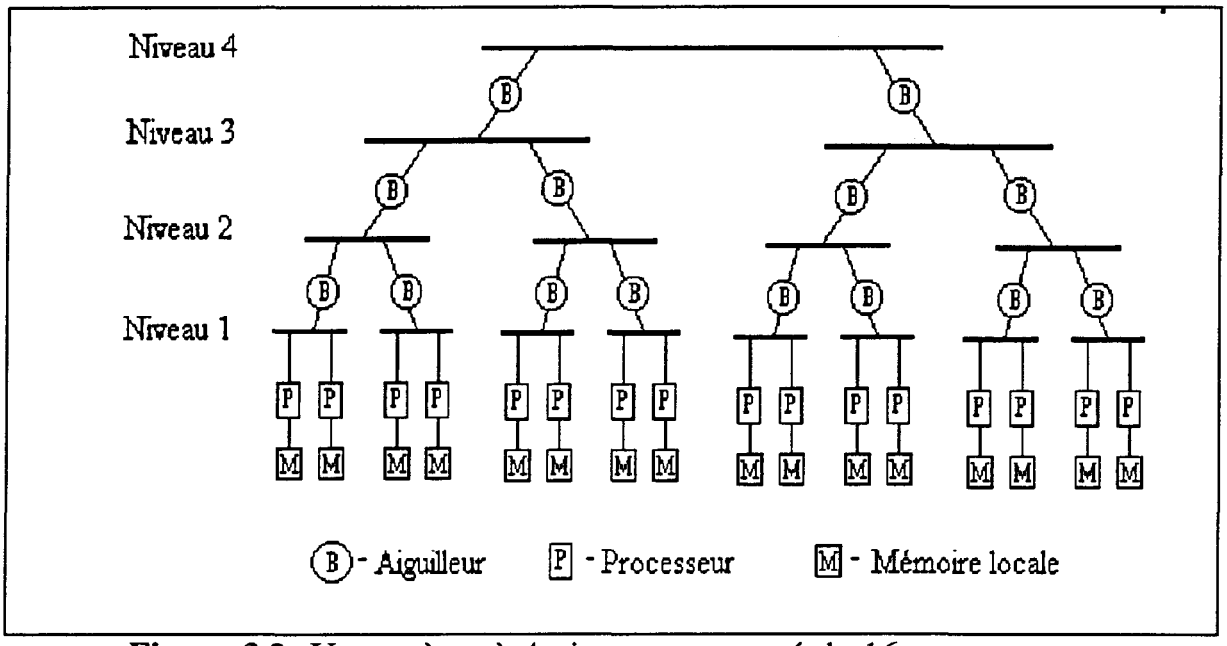

Figure 3.2 Un système à 4 niveaux composé de 16 processeurs

3.3 Modèle analytique de performance pour des architectures hiérarchiques

Il est possible de construire un modèle de files d'attente pour l'analyse de la performance d'architectures hiérarchiques. Un exemple de ce modèle pour un système à 3 niveaux est illustré à la figure 3.3. Dans cette figure, chaque bus est représenté par une file d'attente. Une certaine similitude existe entre ce réseau de files d'attente et un réseau de type fermé [2, 9] ayant M clients (M représentant le

nombre total de processeurs dans le système) et où les ressources (bus) sont passives. En effet, ce modèle peut décrire précisément le réseau d'interconnexion hiérarchique dans le cas où le trafic est élevé car le nombre de messages circulant dans le réseau est alors approximativement constant. Cependant, lorsque le trafic est faible, le comportement d'un réseau fermé sera différent de celui du système réel.

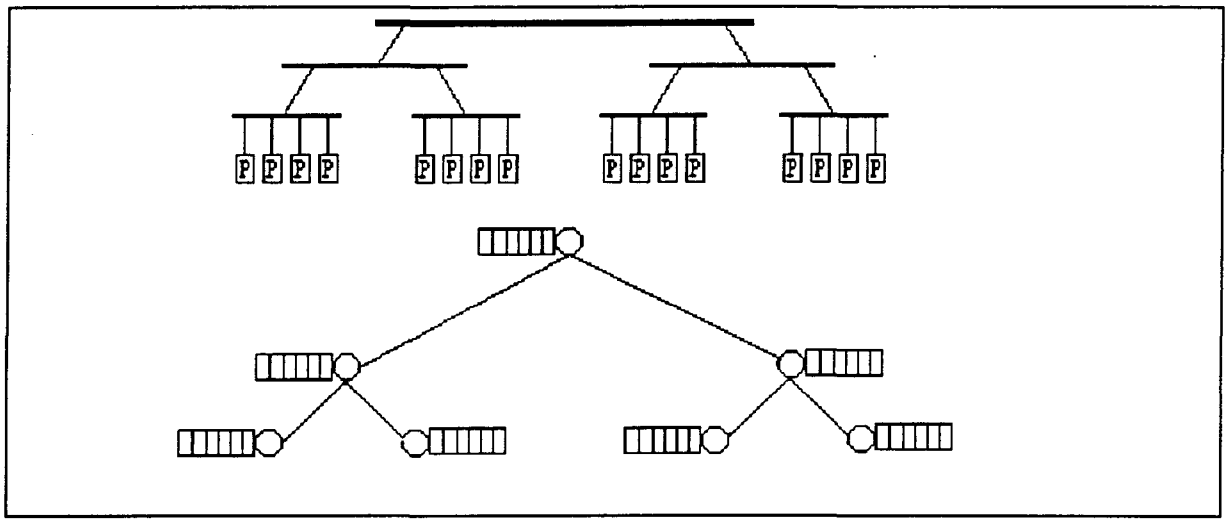

Figure 3.3 Réseau de files d'attente pour un système à trois niveaux

Dans le système, un processeur ne peut être que dans l'un des trois états suivants:

a) *état de traitement*: le processeur exécute des instructions en n'utilisant que sa propre mémoire.

b) *état de communication*: le processeur échange des données avec un autre processeur en émettant un message.

c) *état d'attente*: le processeur attend un accusé réception après avoir émis un message.

L'objectif de notre analyse sera de déterminer le temps moyen qu'un processeur devra attendre avant de recevoir un accusé réception (lorsque ce dernier envoie un message). Nous définirons ce temps comme le temps de réponse du système. Certaines hypothèses d'ordre général doivent être précisées concernant l'opération du système.

3.3.1 Hypothèses concernant le fonctionnement du réseau

À travers le réseau d'interconnexion, les processeurs peuvent communiquer entre eux. Afin d'effectuer des transmissions fiables, un protocole de communication doit exister. Pour se faire, nous supposons que ce dernier exige un accusé réception pour chaque message. De plus, nous considérerons qu'après avoir expédié un message, un processeur émetteur ne peut reprendre le traitement local que lorsqu'il a reçu l'accusé réception correspondant. Évidement, l'attente d'un accusé réception affecte la performance du réseau mais c'est généralement le prix à payer pour

obtenir un réseau de communication fiable. Notons que la stratégie de routage peut affecter la performance d'un réseau d'interconnexion. Dans ce travail, nous considérerons que les messages sont fragmentés en paquets émis indépendamment et que la structure du réseau est telle qu'il n'existe qu'un seul chemin entre la source et la destination. Des conflits d'accès peuvent survenir en raison du partage des liens de communication (bus). Ainsi, les processeurs devront attendre afin de pouvoir transmettre leur message créant ainsi des files d'attente.

3.3.2 Hypothèses additionnelles

Des hypothèses additionnelles doivent être posées afin de pouvoir développer un modèle du système.

1) Les processeurs effectuent une activité qui n'exige que des accès à leur mémoire locale.
2) De temps à autre, les processeurs échangent des données en émettant des messages.
3) Un processeur recevant un message envoie immédiatement un accusé réception (sans aucun délai).

4) La durée d'accès à un bus est une variable indépendante, aléatoire et exponentiellement distribuée ayant une moyenne égale à $1/\mu_i$ pour le $i^{ième}$ bus. Notons que, dans un système réel, cette durée peut être constante car les messages peuvent avoir la même longueur.

5) L'intervalle entre les requêtes d'accès consécutives est une variable indépendante, aléatoire et exponentiellement distribuée ayant une moyenne égale à $1/\lambda_i$ pour le $i^{ième}$ processeur.

6) Lorsqu'un bus devient inoccupé (oisif), la prochaine requête à pouvoir utiliser ce bus est sélectionnée aléatoirement parmi toutes les têtes de files d'attente associées à ce bus.

Les hypothèses précédentes permettent de construire un modèle de type markovien. Cependant, ceci ne garantit pas qu'une solution simple puisse être obtenue. En effet, dans ces modèles, le nombre d'états explose littéralement lorsque le nombre de composants du système est élevé [11, 12]. L'analyse devient rapidement difficile même pour de petits systèmes. Afin de simplifier notre modèle de réseau de files d'attente, nous allons introduire deux hypothèses supplémentaires.

7) Les processeurs ont tous le même taux d'accès λ et la même durée d'accès ($1/\mu$) au bus auquel ils sont rattachés.

8) Une distribution uniforme de messages sera tout d'abord considérée.

L'hypothèse 7 se justifie par le fait que souvent, dans une application parallèle, les processeurs exécutent le même programme sur des données différentes. L'hypothèse 8, quant à elle, implique que toutes les requêtes d'accès (messages) provenant d'un processeur peuvent être dirigées vers n'importe quel autre processeur avec une probabilité égale à $1/(M-1)$, où M est le nombre total de processeurs dans le système. Cette hypothèse est un cas particulier qui permettra par la suite d'analyser des situations plus générales.

Ces deux hypothèses ne permettent cependant que d'analyser des systèmes relativement simples [11, 12]. Nous allons donc introduire, à la section suivante, d'autres hypothèses qui permettront de réduire la taille du réseau de files d'attente et ainsi traiter des cas plus complexes.

3.4 Modèle analytique approximatif

3.4.1 Hypothèses simplificatrices

Puisque le réseau d'interconnexion d'un système hiérarchique peut avoir plusieurs niveaux, nous posons les deux hypothèses simplificatrices suivantes pour réduire la taille du réseau de files d'attente.

9) Tous les bus situés sur un même niveau seront considérés égaux.

10) Toutes les stations de service seront considérées comme une station centrale à laquelle sont reliés un certain nombre de terminaux appelés "sources de requête". Chaque terminal peut générer des requêtes d'accès à la station. Pour la station située au niveau le plus haut, il y aura 2 terminaux, correspondants aux deux bus qui y sont connectés dans le réseau réel. De même, pour chaque station aux niveaux intermédiaires, il y aura trois terminaux si le réseau considéré est une arbre binaire. Le nombre de terminaux de chaque station au niveau le plus bas

(L) est déterminé en divisant le nombre de sources de requête restantes par le nombre de bus de bas niveau (K), c'est-à-dire

$$L = (M - 2*H - 3*I) / K$$

où

M: le nombre total de processeurs dans le système,

H: le nombre de bus au niveau le plus haut (il est égal à 1 dans ce cas),

I: le nombre de bus aux niveaux intermédiaires.

L'hypothèse 9 nous permet de supposer que les bus d'un même niveau ont tous la même longueur de file d'attente. Ces bus pourraient ainsi être représentés par une seule station de service n'ayant qu'une seule file d'attente. Tous les paramètres importants d'une station pourraient être déterminés en calculant des valeurs moyennes. Notre réseau pourrait ainsi être analysé en ne considérant que N stations de service correspondant aux N niveaux du système (figure 3.4).

Avec ces hypothèses additionnelles, le système modélisé devient en fait linéaire (Figure 3.4) et n'est plus hiérarchique. Pour simplifier encore ce dernier, on peut aplanir la hiérarchie pour obtenir la structure illustrée à la figure 3.5. Cette

transformation implique que le temps requis par un message pour traverser le réseau sera sous-estimé car le chemin parcouru dans ce réseau simplifié sera toujours plus court que dans le système réel. Cependant, ceci nous permettra de calculer plus facilement la longueur moyenne des files d'attente du réseau. En effet, le système simplifié est en fait le modèle du serveur central abondamment traité dans la littérature [2, 4, 9].

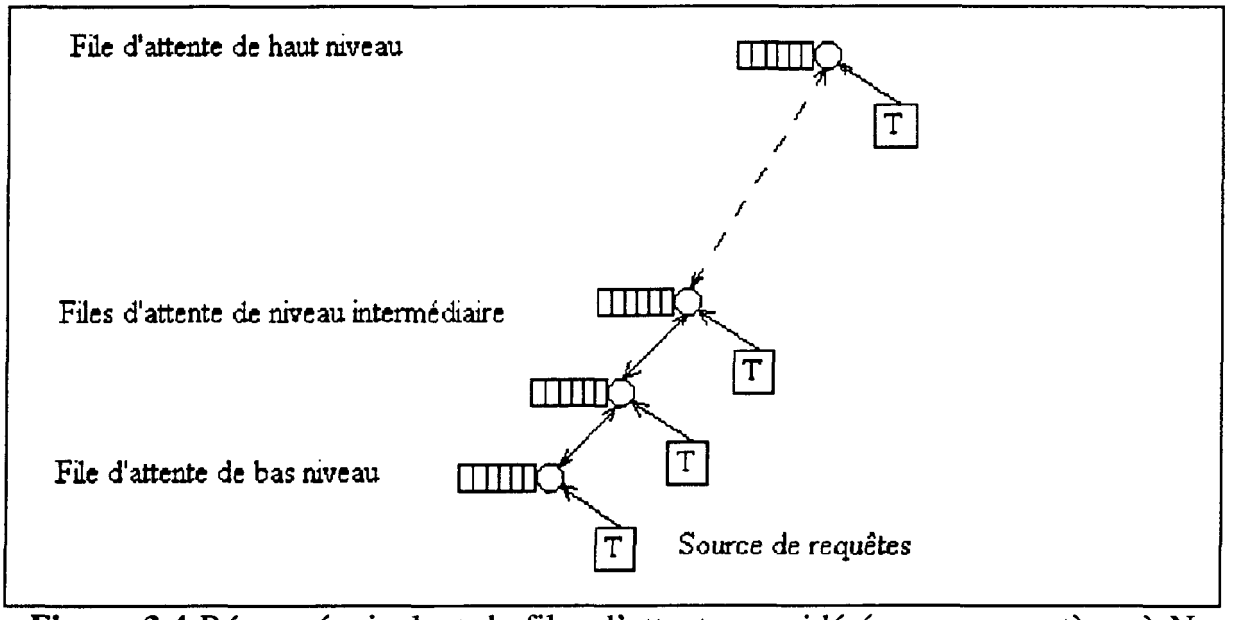

Figure 3.4 Réseau équivalent de files d'attente considéré pour un système à N niveaux

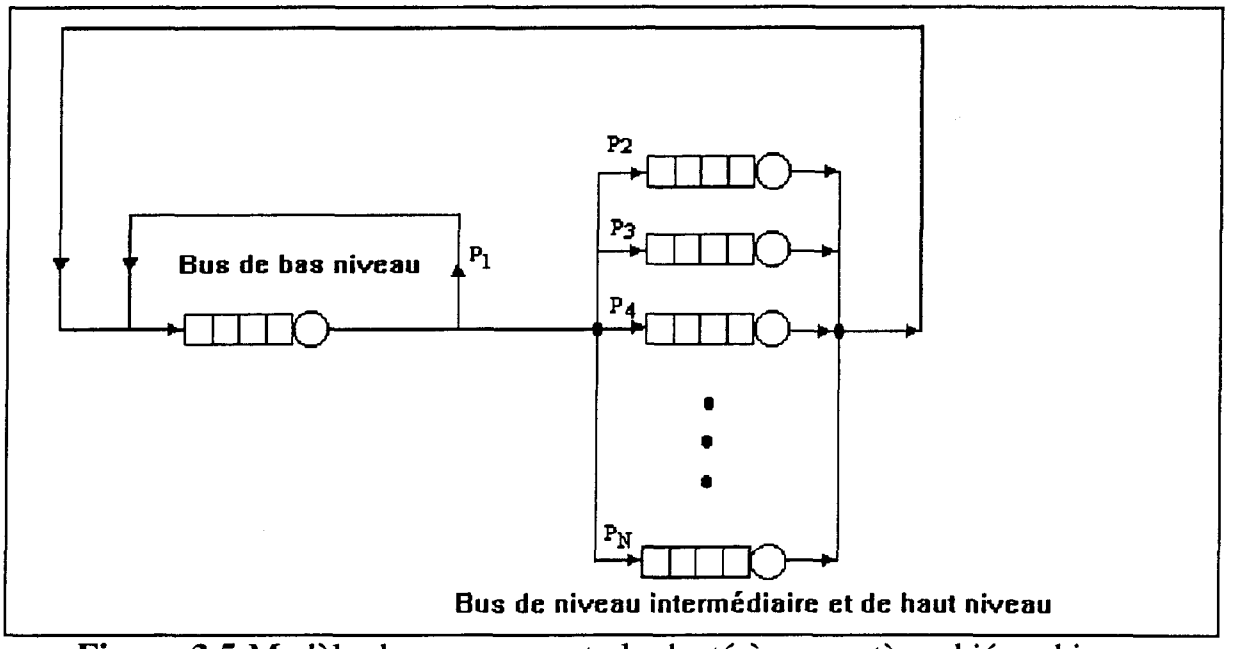

Figure 3.5 Modèle de serveur central adapté à un système hiérarchique à N niveaux

3.4.2 Modèle du serveur central

Soit p_i la probabilité qu'une requête d'accès soit effectuée à la $i^{ième}$ station. Si $(k_1, k_2, ..., k_N)$ représente l'état du système, où k_i est le nombre de clients de la $i^{ième}$ file d'attente (en attente et en service), Buzen [3,4] a montré que la probabilité $P(k_1, k_2, ..., k_N)$ que le système soit dans l'état $(k_1, k_2, ..., k_N)$ est donnée par

$$P(k_1, k_2, \ldots, k_N) = \frac{1}{G(K)} \prod_{i=2}^{N} \left(\frac{\mu_1 p_i}{\mu_i}\right)^{k_i} \qquad (3.1)$$

pour tout état (k_1, k_2, ..., k_N) tel que

$$\sum_{i=1}^{N} k_i = K \qquad (3.2)$$

où $k_i \geq 0$ et K est le nombre total de clients dans le système. Les termes G(k) sont des constantes de normalisation définies de sorte que la somme des probabilités P(k_1, k_2, ..., k_N) soit égale à 1. Un algorithme développé par Buzen permettant de calculer G(0), G(1), ..., G(K) est montré à la table 3.1 [3]. Les équations du modèle de file d'attente d'un serveur central sont données à la table 3.2. Notons que, pour une application typique, le temps entre les requêtes d'accès E(t) (table 3.2) peut être déterminé à partir du pourcentage de temps d'exécution d'un programme consacré aux activités de communication (α). En effet, si l'on suppose qu'une opération de communication ne requiert seulement qu'une unité de temps (lorsqu'il n'y a pas de conflit d'accès au lien), on peut définir α tel que

$$\alpha = \frac{\textit{temps associés aux opérations de communication}}{\textit{temps requis pour exécuter le programme}} = \frac{n}{N}$$

où n est le nombre d'opérations de communication et N est le nombre total de cycles requis compléter l'application (lorsqu'il n'y a pas de conflit d'accès). Le temps de "réflexion" E(t) peut alors être obtenu en calculant

$$E(t) = \frac{N-n}{n} = \frac{1-n/N}{n/N} = \frac{1-\alpha}{\alpha} \qquad (3.3)$$

Dans ce qui suit, nous utiliserons ce paramètre pour analyser le comportement de systèmes hiérarchiques sous diverses conditions de trafic.

La procédure précédente permet d'obtenir la longueur moyenne de la file d'attente de chacune des stations. En utilisant ces valeurs, le temps de réponse moyen du système peut être facilement obtenu. Le temps de réponse moyen est défini comme le temps total qu'un message doit passer dans le réseau de files d'attente, pour atteindre le processeur destinataire, auquel on doit ajouter le temps requis pour le retour de l'accusé réception correspondant. L'exemple de la section suivante montre comment calculer ce temps de réponse moyen pour un système à trois niveaux.

Table 3.1 Algorithme de Buzen

Étant donné les paramètres μ_i et p_i du modèle du serveur central de la figure 3.5, l'algorithme suivant permet d'obtenir les paramètres de normalisation G(m), où $0 \leq m \leq K$ et où K représente le nombre total de processeurs du système.

Étape 1: Assigner une valeur à tous les x_i:

Soit $x_1 = 1$ et $x_i = \mu_1 p_i / \mu_i$ pour i=2,3,..., N.

Étape 2: Assigner une valeur initiale aux paramètres g(K,1):

Soit g(k,1) = 1 pour k=0,1,...,K et g(0,n)=1 pour n=1,2,...,N.

Étape 3: Initialiser k:

k = 1.

Étape 4: Calculer la $k^{ième}$ ligne de la matrice g(i,j) à l'aide de:

$g(k,n) = g(k,n-1) + x_n g(k-1,n)$, n=2,3,...,N.

Étape 5: Incrémenter k:

k = k+1.

Étape 6: Si $k \leq K$, retourner à l'*Étape 4*. Sinon, aller à l'*Étape 7*.

Étape 7: Les paramètres G(m) sont tirés de la $N^{ième}$ colonne , soit g(m,N) = G(m) pour m=0,1,.....,K.

Table 3.2 Équations stationnaires du modèle de files d'attente du serveur central.

Calculer G(0), G(1),...., G(K) en utilisant l'algorithme de Buzen[3].

Le taux d'utilisation de chaque station est obtenu par

$$\rho_i = \begin{cases} \dfrac{G(K-1)}{G(K)}, & i=1 \\ \dfrac{\mu_1 \rho_1 p_i}{\mu_i}, & i=2,3,\ldots,N \end{cases} \qquad \text{(A)}$$

Le débit moyen de chaque station λ_i est donné par

$$\lambda_i = \mu_i \rho_i p_i \qquad \text{(B)}$$

Avec les hypothèses (9) et (10), la longueur moyenne de la file d'attente à chaque station de service est donnée par

$$L_{qi} = L_i - \lambda_i * E(t) \qquad \text{(C)}$$

où

L_{qi} : longueur de la file d'attente à la $i^{ième}$ station,
L_i : nombre de terminaux à la $i^{ième}$ station,
λ_i : débit de la $i^{ième}$ station,
E(t): temps de "réflexion".

Le temps d'attente à chaque station W_i est obtenu avec

$$W_i = L_{qi} / \mu_i \qquad \text{(D)}$$

3.5 Exemple d'application du modèle approximatif développé

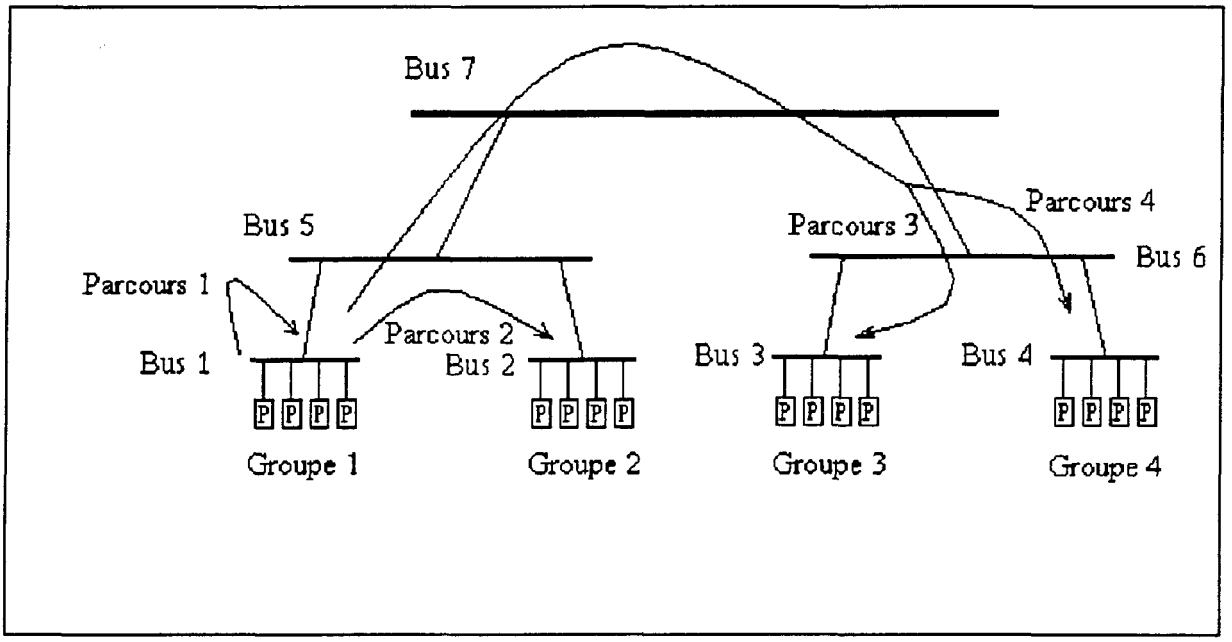

Figure 3.6 Les parcours de communication dans un système à 3 niveaux

Nous allons montrer ici comment calculer le temps de réponse moyen d'un système à trois niveaux composé de 20 processeurs. Pour calculer les probabilités p_i, nous déterminons premièrement le nombre de requêtes dirigées vers chaque station du système. Dans ce système à trois niveaux, les messages provenant d'un processeur donné peuvent emprunter 4 parcours. Les quatre parcours possibles pour les processeurs du groupe 1 sont illustrés à la figure 3.6. Soit X le nombre total de messages émis par un groupe de processeurs. Ici, X sera égal au nombre de processeurs de chaque groupe. En utilisant les hypothèses 7 et 8, le nombre de

messages provenant du groupe 1 peut être estimé comme suit: X/4 messages demeurent sur le bus 1; 3X/4 messages sont dirigés vers le bus numéro 5; parmi ces 3X/4 messages, 2X/4 passent par les bus 6 et 7 en direction des bus 3 et 4, et X/4 messages sont dirigés vers le bus numéro 2. La distribution des destinations est la même pour les groupes 2, 3 et 4. Ces derniers peuvent donc être analysés de la même façon. Ainsi, pour le système à trois niveaux, le nombre total de requêtes à chaque station est alors:

- nombre de requêtes à la station de bas niveau = X,
- nombre de requêtes aux stations de niveau intermédiaire = 2.5X,
- nombre de requêtes à la station de haut niveau = 2X.

En généralisant pour un système à N niveaux, le nombre de requêtes peut être obtenu par la relation suivante:

- pour la station de bas niveau:

  nombre de requêtes = X,

- pour les stations de niveau intermédiaire:

  nombre de requêtes = $2^{j-N}(2N - 2^{j-1} - 2^{j-2})*X$,

- pour la station de haut niveau:

  nombre de requêtes = $2^{N-1}/2*X$,

où:

- X: nombre de processeurs sur chaque bus de bas niveau,
- N: nombre total de niveaux du système,
- *j*: niveau de la station et $1 \leq j \leq N$.

Comme la somme de probabilités p_i doit être égale à 1 ($\sum p_i=1$), nous pouvons déterminer les p_i en utilisant les différents nombres de requêtes:

$$p_i = V_i / V_{total}$$

où:

- V_i: nombre de requêtes à la $i^{ième}$ station,
- V_{total}: somme totale de toutes les requêtes dans le réseau.

Pour notre exemple, nous pouvons aisément obtenir ces probabilités comme suit:

- $p_1 = X / (X+2.5X+2X) = 0.182$,
- $p_2 = 2.5X / (X+2.5X+2X) = 0.455$,
- $p_3 = 2X / (X+2.5X+2X) = 0.364$.

En utilisant l'algorithme de Buzen (Table 3.1) et l'équation 1 de la table 3.2, nous obtenons les taux d'utilisation $\varrho_1=0.55$, $\varrho_2=0.25$ et $\varrho_3=0.20$ respectivement associés aux stations de niveau bas, intermédiaire et haut. Les débits moyens sont alors

obtenus par l'équation B (table 3.2), soit : λ_1=0.100, λ_2=0.114 et λ_3=0.073, respectivement. Notons que, dans cet exemple, μ_i est égal à 1. Nous calculerons les longueurs de files d'attente pour un pourcentage de communication α=50%. Ceci implique que E(t)=1. En utilisant l'équation C (table 3.2), les longueurs moyennes des files d'attente des stations de niveau bas, intermédiaire et haut sont respectivement L_{q1}=2.90, L_{q2}=2.886, et L_{q3}=1.927. Avec ces valeurs, nous pouvons calculer le temps de réponse moyen du système. En effet, celles-ci nous permettent de déterminer le temps de réponse moyen de chaque station i (i=1,2,3) en utilisant la loi de Little $t_i=L_{qi}/\mu_i$. Ainsi, la durée de chacun des quatre parcours de communication peut être calculée en additionnant les temps d'attente pour l'accès aux stations rencontrées. On multiplie ensuite ces résultats par un facteur de 2 afin de tenir compte du temps de retour de l'accusé réception. Dans notre exemple, ces temps de réponse moyens sont:

- $r_1 = 2t_1$,
- $r_2 = 4t_1 + 2t_2$,
- $r_3 = r_4 = 4t_1 + 4t_2 + 2t_3$,

où r_1, r_2, r_3, et r_4 sont les temps de réponse moyens correspondants aux parcours 1, 2, 3, et 4. Finalement, le temps de réponse moyen du système est donné par

$$R = (r_1 + r_2 + r_3 + r_4) / 4 = 19.3.$$

3.6 Résumé

Dans ce chapitre, nous avons présenté en détail une méthodologie permettant le développement de modèles analytiques de performance pour une architecture multiprocesseur hiérarchique. En introduisant des hypothèses et des contraintes, nous avons développé un modèle approximatif qui nous permet d'obtenir le temps de réponse des systèmes multiprocesseurs hiérarchiques. Nous avons aussi illustré par un exemple comment calculer le temps de réponse d'un système hiérarchique en utilisant ce modèle approximatif.

CHAPITRE 4

SIMULATION DE SYSTÈMES MULTIPROCESSEURS HIÉRARCHIQUES

4.1 Introduction

Dans les précédents chapitres, nous avons montré comment obtenir des modèles analytiques pour l'évaluation de la performance de systèmes multiprocesseurs hiérarchiques. Afin d'étudier le comportement de ces systèmes en détail ainsi que pour valider ces modèles analytiques, un simulateur de systèmes hiérarchiques a été développé. Ce simulateur a été construit en se basant sur le principe de la simulation par événements discrets. Il a été réalisé dans le langage de programmation C. À l'aide de ce simulateur, il est possible d'étudier le

comportement de diverses configurations de système hiérarchique. La structure de ce simulateur sera décrite dans la section suivante.

4.2 Structure du simulateur

4.2.1 Les éléments principaux du simulateur

Pour construire un simulateur, on doit tout d'abord modéliser le système réel. Le développement de ce modèle consiste à créer des éléments qui émuleront le comportement des composants du système. Les trois éléments principaux du système multiprocesseur hiérarchique sont l'élément "processeur", l'élément "bus " et l'élément "message".

L'élément "processeur": Chaque processeur du système est représenté par cet élément. Ce dernier est complètement caractérisé par:

- L'état du processeur: comme dans le système réel, un processeur peut être dans un des trois état suivants: état de traitement, état de communication et état d'attente.

- Un programme: chaque processeur exécute un programme. Le programme d'un processeur peut être différent de celui d'un autre processeur. Dans ces programmes, il existe deux classes d'instructions: les instructions de traitement, qui sont réalisées dans la mémoire propre du processeur, et les instructions de communication, qui effectuent une opération de communication avec un autre processeur. Le nombre d'instructions de communication peut être déterminé à partir du paramètre α représentant la fraction du temps d'exécution consacrée aux activités de communication. Pour des raisons de généralité, la destination des messages est choisie aléatoirement. La distribution de ces messages peut être uniforme dans le cas le plus simple. Dans le présent simulateur, cette distribution peut être modifiée en fixant un pourcentage de messages locaux (β). Ce paramètre indique quel pourcentage des messages expédiés n'utilise que les bus de bas niveau (par exemple, parcours 1 à la figure 3.6).
- Le temps de traitement moyen d'une instruction: Dans le présent simulateur, ce dernier est identique pour tous les processeurs et a été fixé à 1 cycle (en unité arbitraire).

L'élément "bus": Chaque bus du système est représenté par cet élément. Ainsi un élément "bus" est caractérisé par:

- Une file d'attente: lorsqu'un processeur désire communiquer avec un autre processeur, il envoie une requête de communication à ce dernier. Ces requêtes sont dirigées vers certaines files d'attente en fonction de leur destination. La "politique" de service de ces files d'attente est de type "premier arrivé, premier servi". Dans ce simulateur, une variable est utilisée pour enregistrer la longueur de la file d'attente à chaque cycle de simulation.
- Un temps de service (durée d'un message): Dans le présent simulateur, ce temps est identique pour toutes les stations et a été fixé à 1 cycle.

L'élément "message": Chaque message du système est représenté par cet élément. Ainsi un élément "message" est caractérisé par:

- Le numéro du processeur source qui envoie ce message.
- Le numéro du processeur destination à qui est expédié ce message.

En utilisant ces trois éléments, un simulateur fonctionnel de haut niveau a été développé.

4.2.2 Paramètres d'entrée et de sortie du simulateur

Les paramètres d'entrée suivants sont nécessaires pour la simulation de systèmes multiprocesseurs hiérarchiques dans un contexte général:

- Nombre de niveaux du réseau d'interconnexion (N): Ce paramètre peut être modifié pour étudier l'influence de la profondeur de la hiérarchie.
- Pourcentage de communication ($0\% \leq \alpha \leq 100\%$): Ce paramètre permet de contrôler l'intensité du trafic dans le réseau de communication.
- Pourcentage de messages locaux (β): En variant ce pourcentage, il est possible d'étudier la performance du réseau de communication en fonction de différentes distributions de messages et, conséquemment, de trafic.
- Nombre de processeurs (M): Ce paramètre détermine le nombre total de processeurs dans le système.

En ce qui concerne la sortie, ce simulateur fournit les valeurs suivantes:

- Longueur moyenne de la file d'attente (L_q) de chaque bus.
- Temps de réponse moyen (W): Ce temps de réponse est calculé à partir de la longueur de toutes les files d'attente du système.

4.2.3 Le principe de fonctionnement du simulateur

Le programme de simulation est divisé en deux blocs d'opérations, soit l'initialisation de l'état interne du système et le fonctionnement normal du système au cours de la simulation. Le principe de fonctionnement général du simulateur est illustré à la figure 4.1.

Initialisation de l'état du système

L'état du système est caractérisé par l'état des éléments "processeur" et "bus".

Pour les éléments "processeur", chacun est initialisé de manière à posséder son propre programme. Comme dans un système réel, chaque processeur peut avoir

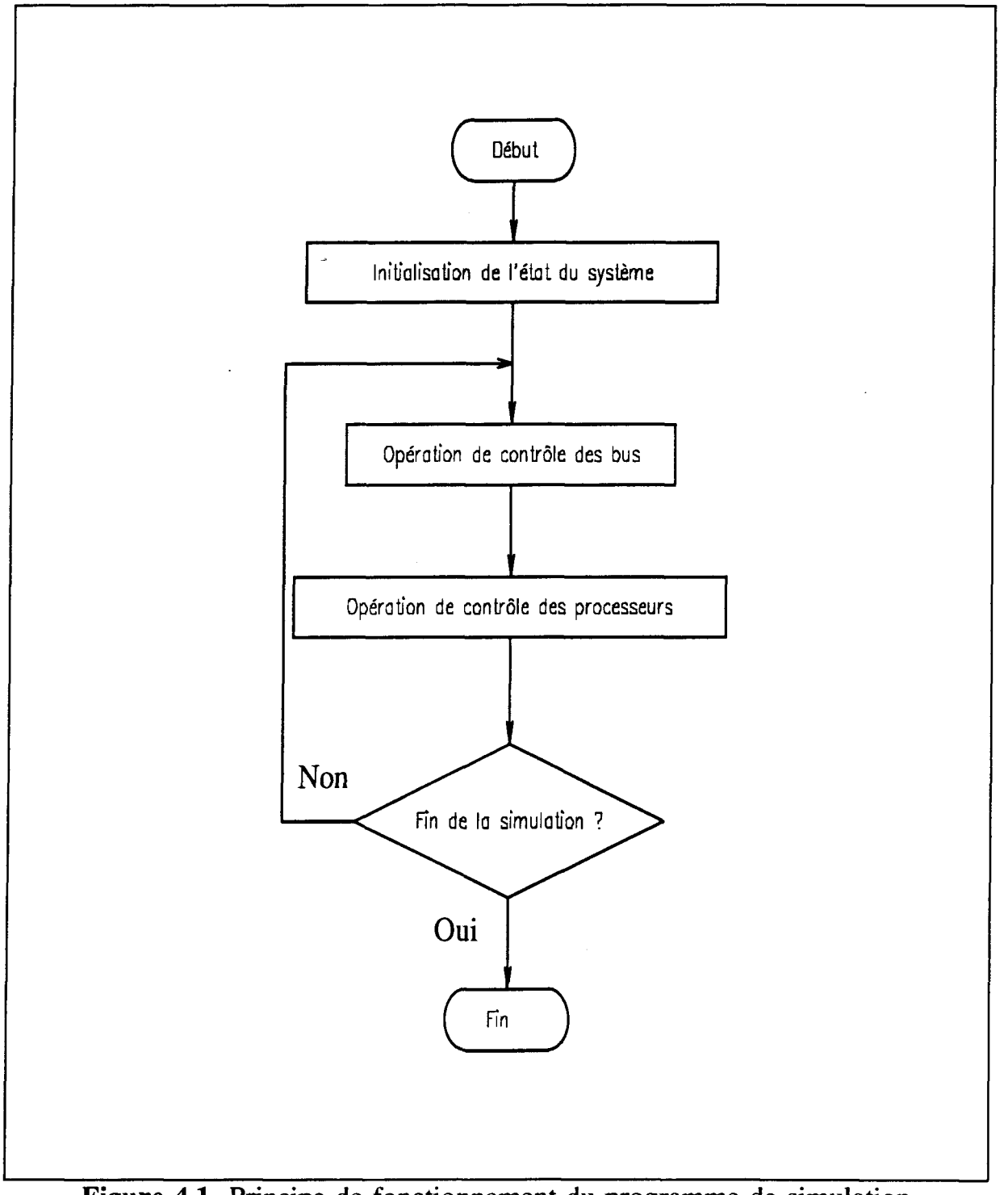

Figure 4.1 Principe de fonctionnement du programme de simulation

un programme différent des autres, on choisit aléatoirement la longueur du programme et on réserve l'espace-mémoire requis en conséquence. On détermine également le nombre d'instructions du programme qui constitueront des opérations de communication. Cette valeur dépend du pourcentage de communication (α) imposé lors du démarrage du simulateur. Notons que la position des instructions de communication dans le programme est également choisie aléatoirement pour tenir compte du fait que les programmes peuvent tous être différents. De plus, on choisit aléatoirement le numéro du processeur destination pour chacune des instructions de communication. La distribution des destinations dépend du pourcentage de messages locaux (β).

Pour les éléments "bus", le contenu de toutes les files d'attente est éliminé.

Fonctionnement du simulateur

Une boucle de simulation a été définie. Chaque itération simule en quelque sorte un cycle de l'horloge interne du système. Le nombre d'itérations de simulation est déterminé préalablement. À chaque itération, deux étapes de contrôle sont réalisées, soit le contrôle des processeurs et le contrôle des bus.

L'opération de contrôle des processeurs balaie tous les processeurs du système. L'état de chaque processeur dépend de l'instruction que ce dernier doit exécuter. Si cette instruction n'est pas une instruction de communication alors le simulateur passera au prochain processeur. De cette façon, on émule le comportement d'un processeur qui exécute une instruction dans sa propre mémoire. Le compteur d'instruction du programme sera donc incrémenté. Le processeur sera ainsi mis en état de traitement. Si l'instruction à exécuter est une instruction de communication, il faut tout d'abord s'assurer que le message puisse être acheminé sur le bus auquel est connecté le processeur. Le message ne peut être acheminé si ce bus est déjà occupé. Lorsque cette condition est réalisée, le processeur peut envoyer le message et sera alors mis en état de communication. Dans le cas contraire, le processeur doit attendre pour l'obtention du bus. L'incrémentation du compteur d'instruction ne peut se faire tant que le bus n'est pas libre. Après avoir émis le message, le processeur attendra (mode attente) jusqu'à ce qu'il reçoive l'accusé réception du processeur destination. Durant cette période, aucune autre instruction de son propre programme n'est réalisée.

L'opération de contrôle de bus balaie tous les bus pour vérifier si leur file d'attente contient des messages. Si c'est le cas, il faut alors s'assurer que le message

à la tête de la file d'attente soit acheminé. Ceci est réalisé selon la destination du message. La destination du message peut être un processeur ou un bus qui est connecté à cette file d'attente. Dans le premier cas (la destination est un processeur), le message sera transmis immédiatement à ce processeur. Dans l'autre cas, le message sera transmis à un bus connecté si la longueur de file d'attente de ce dernier n'a pas dépassé la longueur permise. Lorsqu'un message a été acheminé, il est retiré de la file d'attente. Cette dernière contient alors un message de moins.

Il est important de noter que la séquence de balayage des opérations de contrôle des processeurs et des bus est aléatoire de façon à ne privilégier aucun parcours de message, ce qui est conforme au comportement d'un système réel.

4.3 Validation du simulateur

La fiabilité d'un simulateur se mesure en comparant les résultats fournis par ce simulateur à ceux du système simulé. Dans le cas présent, il est impossible de comparer toutes les configurations de systèmes simulables avec des systèmes réels. Un certain nombre d'hypothèses ont été émises concernant le fonctionnement du

système simulé. La tâche de validation du simulateur consistait donc à vérifier si toutes ces hypothèses étaient respectées, et ce à chaque cycle d'exécution (itération par itération).

Pour réaliser cette validation, une étude attentive du comportement du simulateur sur un certain nombre de configurations a été accomplie. Il a été observé qu'à chaque instant des simulations, le comportement du simulateur était conforme aux hypothèses établies. Puisque la structure du simulateur a été bâtie dans l'optique de simuler des systèmes de taille variable, ce qui requiert un haut niveau de modularité dans le programme, on peut donc conclure que le simulateur reproduira fidèlement le comportement de toute autre configuration. Notons que certaines vérifications quantitatives seront présentées au prochain chapitre.

Il est important de noter que, sur des ordinateurs de type "SPARC Station 1+", le simulateur ne peut simuler que des systèmes hiérarchiques allant jusqu'à 10 niveaux. En effet, le simulateur exige beaucoup d'espace-mémoire ce qui limite la taille de la hiérarchie. De plus, notons que le temps de simulation dépend aussi de la taille du système à simuler. Ce dernier peut varier de quelques minutes à plusieurs heures.

4.4 Résumé

Dans ce chapitre, les principes généraux de fonctionnement d'un simulateur de systèmes multiprocesseurs hiérarchiques sont présentés. Ce simulateur permet d'étudier de façon détaillée le comportement de ces systèmes sous diverses conditions à l'aide de paramètres de contrôle. Un guide de l'utilisation de ce simulateur est présenté en annexe.

CHAPITRE 5

ANALYSE DES RÉSULTATS

5.1 Introduction

Dans ce chapitre, nous allons analyser les résultats obtenus au cours de ce travail. Nous aborderons tout d'abord la validation des modèles analytiques approximatifs développés précédemment. Ces modèles seront ensuite utilisés pour analyser la performance de systèmes multiprocesseurs hiérarchiques sous diverses configurations du réseau d'interconnexion ainsi que sous diverses conditions de trafic. Notons que le but visé lors du développement de ces modèles est de permettre des analyses plus approfondies sans avoir à recourir à de longues simulations. L'optimisation de systèmes hiérarchiques sera également abordé. Ainsi une étude visant à maximiser le rapport "performance / coût " sera réalisée.

5.2 Validation des modèles développés

5.2.1 Distribution uniforme des messages

L'exactitude des modèles peut être déterminée en comparant leurs prédictions avec celles du simulateur. Dans cette section, nous comparerons les temps de réponse moyens obtenus à partir des modèles analytiques approximatifs avec ceux fournis par le simulateur lorsque la distribution des messages est uniforme. Une analyse de systèmes binaires ayant de 2 à 5 niveaux a été réalisée en modifiant les pourcentages de communication (α). Les figures 5.1 à 5.4 montrent les résultats obtenus. Notons que, pour ces figures, la distribution des messages est uniforme ($\beta = 1/2^{N-1}$).

Pour mesurer l'exactitude du modèle analytique, on doit considérer la différence entre le temps d'exécution réel ($T_{réel}$) et le temps d'exécution estimé ($T_{estimé}$). Notons que mesurer l'exactitude des modèles en se basant uniquement sur la différence entre les temps d'attente moyens ne reflète pas correctement la réalité. En effet, une différence de 50% entre le temps de réponse estimé et le temps de réponse réel n'aura qu'un faible effet sur la valeur estimée du temps d'exécution (par

rapport à la valeur réelle) si le temps de réponse est petit. Il est donc préférable de mesurer l'exactitude à l'aide de l'expression suivante:

$$Erreur = \frac{|T_{réel} - T_{estimé}|}{T_{réel}} \qquad (5.1)$$

où $T_{réel}$ et $T_{estimé}$ sont obtenus avec les relations

$$T_{réel} = \bar{n}[(1-\alpha)t_a + \alpha(t_b + t_{réponse\ réel})]$$

$$T_{estimé} = \bar{n}[(1-\alpha)t_a + \alpha(t_b + t_{réponse\ estimé})]$$

Dans ces dernières,

- t_a et t_b représentent respectivement le nombre de cycles requis pour une opération de traitement et le nombre de cycles requis pour une opération de communication,
- α est le pourcentage du temps d'exécution consacré aux opérations de communication (excluant tout conflit d'accès aux liens de communication),
- $\bar{n}$ représente le nombre moyen d'opérations exécutées par chaque processeur afin de compléter leur tâche.

Pour simplifier les équations, nous supposerons qu'une opération de

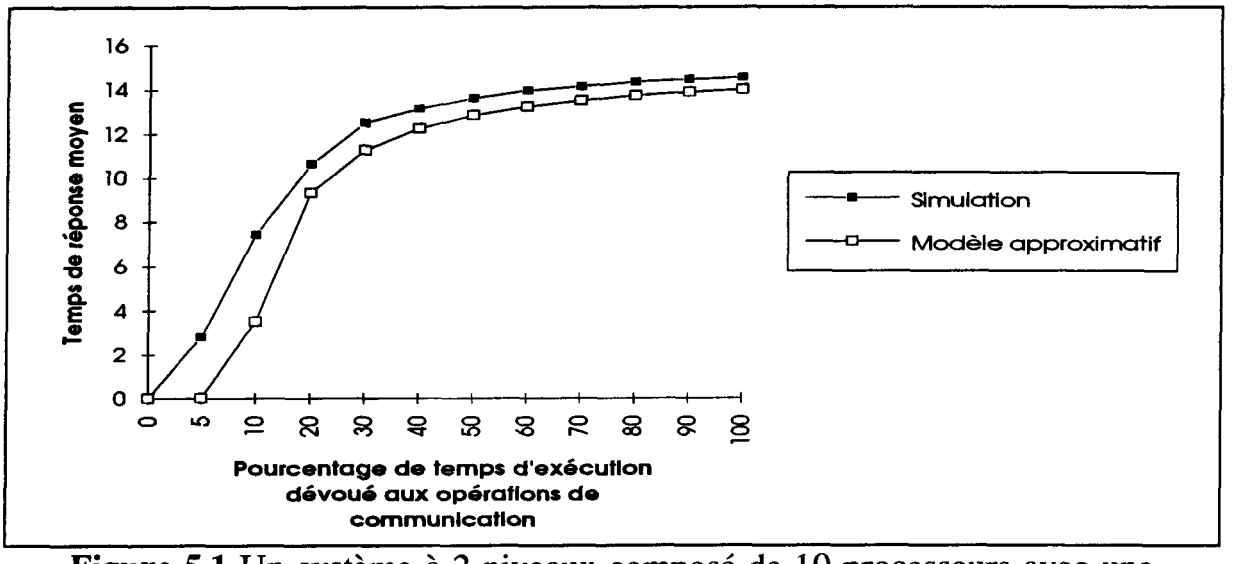

Figure 5.1 Un système à 2 niveaux composé de 10 processeurs avec une distribution de messages uniforme.

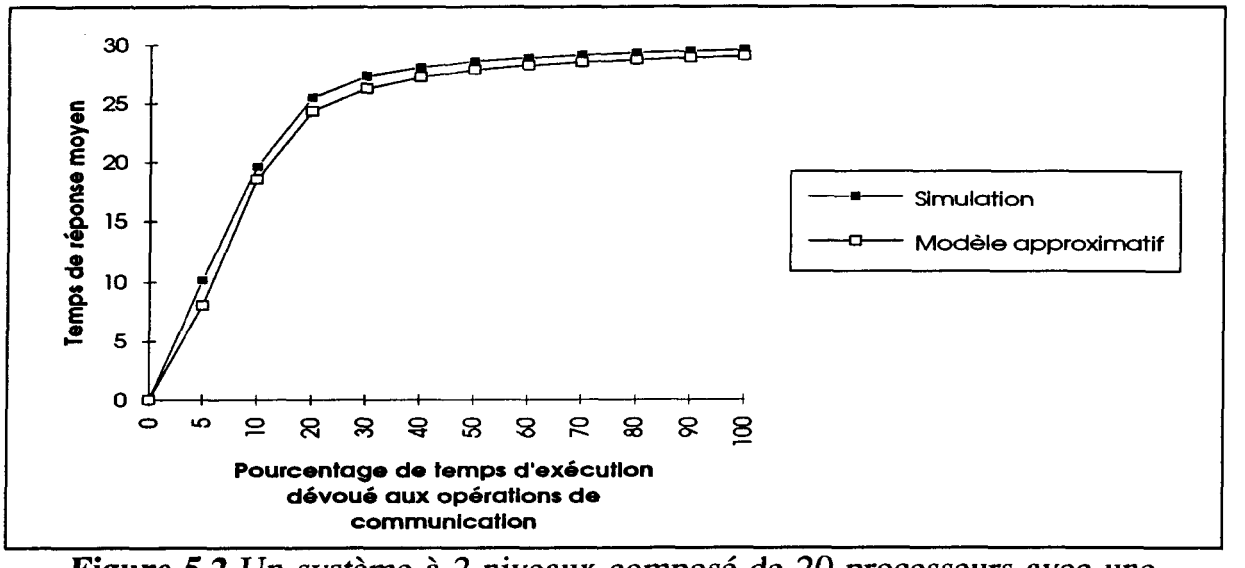

Figure 5.2 Un système à 2 niveaux composé de 20 processeurs avec une distribution de messages uniforme.

communication requiert une unité de temps lorsqu'aucun conflit d'accès ou retard dû à la structure du réseau de communication ne survient. De plus, en toute

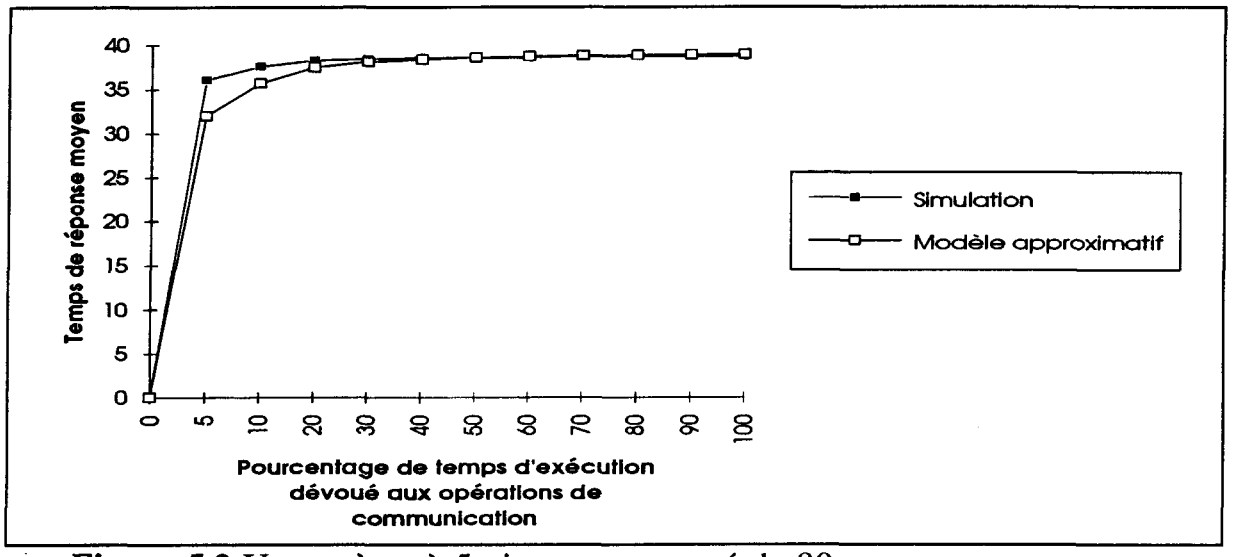

Figure 5.3 Un système à 5 niveaux composé de 80 processeurs avec une distribution de messages uniforme.

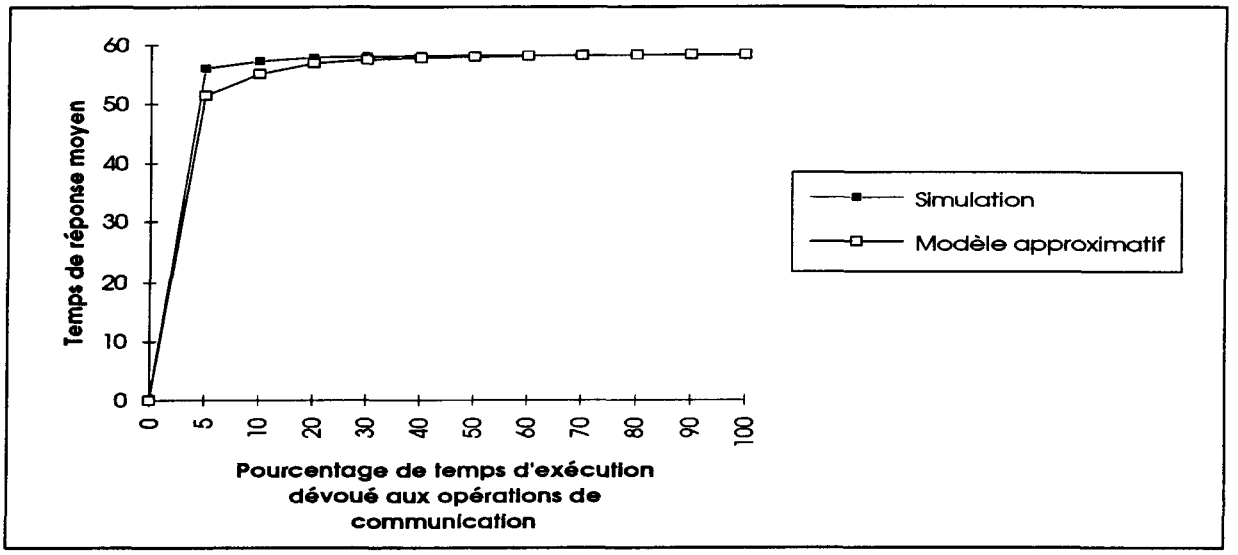

Figure 5.4 Un système à 5 niveaux composé de 160 processeurs avec une distribution de messages uniforme.

généralité, nous pouvons également supposer que t_a est aussi égal à 1, ce dernier étant multiplié par un facteur $(1-\alpha)$ qui, lui, variera. Notons que le terme $\tilde{n}$ disparaît

lors du calcul de l'exactitude. L'exactitude associée à chacun des points des figures 5.1, 5.2, 5.3 et 5.4 est présentée aux figures 5.5 et 5.6.

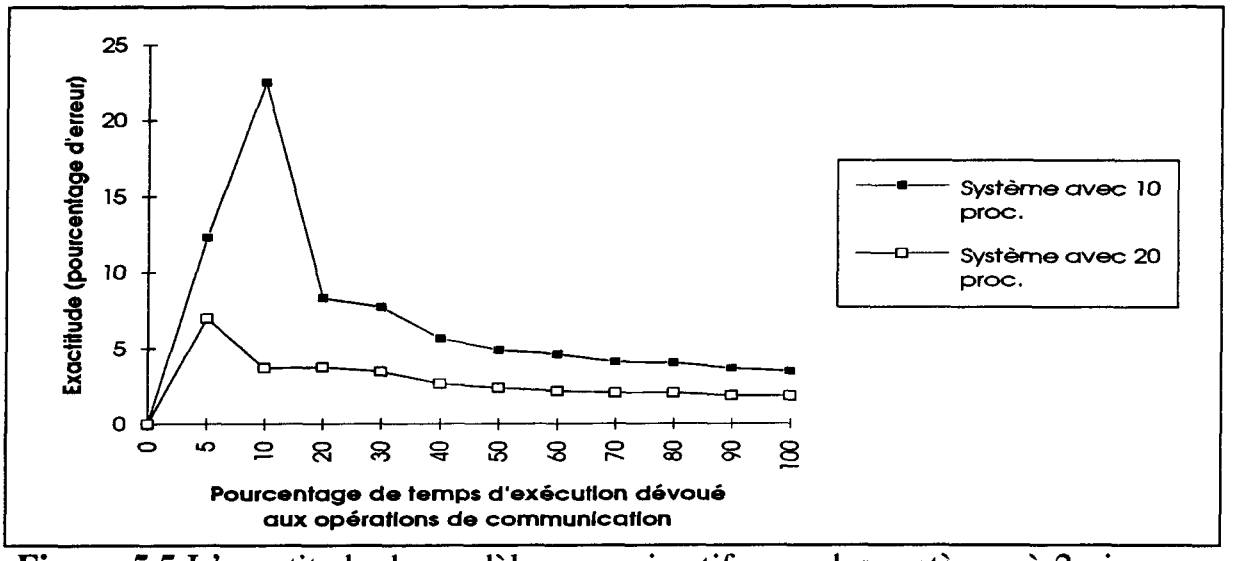

Figure 5.5 L'exactitude du modèle approximatif pour des systèmes à 2 niveaux.

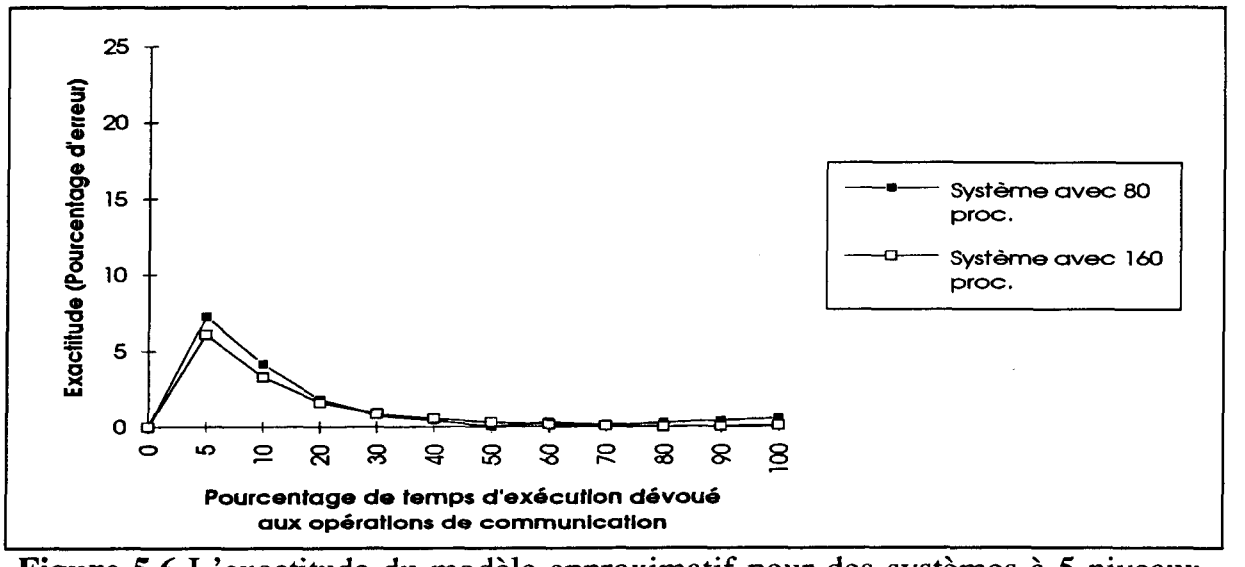

Figure 5.6 L'exactitude du modèle approximatif pour des systèmes à 5 niveaux.

En comparant les résultats obtenus à partir du modèle avec ceux obtenus par simulation, nous pouvons faire les observations suivantes:

1) Le pourcentage d'erreur du modèle approximatif est inférieur à 10% pour une vaste gamme de conditions de trafic (α), à l'exclusion du cas d'un système à 2 niveaux composé de 10 processeurs lorsque $\alpha < 20\%$. Cette situation peut être expliquée de la façon suivante. Dans les systèmes composés d'un petit nombre de processeurs, chacun exécutant une application où α est petit, le trafic engendré par les messages est faible. En conséquence, le réseau se comportera différemment d'un système fermé de files d'attente tel que supposé initialement. Par contre, lorsque le nombre de processeurs du système et le pourcentage de communication sont élevés, le modèle approximatif décrit bien le réseau puisque le nombre de messages dans le réseau devient approximativement constant et égal au nombre de processeurs. Ces deux comportements peuvent être observés lorsque nous examinons l'influence du pourcentage de communication sur la performance de systèmes (Figures 5.1, 5.2, 5.3 et 5.4).

2) Lorsque le nombre de processeurs dans le système augmente, l'exactitude du modèle approximatif augmente aussi (Figures 5.5, 5.6). Cette caractéristique est

particulièrement intéressante car, en général, il est plus utile de modéliser le comportement de systèmes hiérarchiques de grande dimension, ceux-ci étant plus complexes à analyser.

Il est possible de vérifier à l'aide d'un calcul simple la justesse du modèle de même que du simulateur. Nous considérons le cas d'un système composé de 10 processeurs à 2 niveaux. En effet, lorsque α=100%, on peut aisément constater que la longueur de la file d'attente de chaque bus de bas niveau est égale à 4 et que la longueur de la file d'attente du bus de haut niveau est égale à 2. Donc, le temps de réponse moyen de ce système doit être égal à:

$$0.5\ (4 + 4) + 0.5\ (4 + 2 + 4 + 4 + 2 + 4) = 14.$$

Notons que les facteurs 0.5 sont en fait les probabilités associées à chacun des parcours possibles. On observe à la figure 5.1 que le temps d'attente atteint cette valeur lorsque α=100%. Les observations précédentes nous permettent de conclure que le modèle développé fournit le temps de réponse des systèmes hiérarchiques avec une précision adéquate lorsque la distribution des messages est uniforme.

5.2.2 Distribution non-uniforme des messages

Dans les précédentes discussions, nous avons utilisé le modèle approximatif de performance en se basant sur quelques hypothèses simples. Certaines de ces hypothèses peuvent être modifiées afin d'étudier une gamme plus vaste de configurations et de conditions de trafic. Nous examinerons ici l'influence de l'hypothèse 8 affectant la distribution des messages et mesurerons l'exactitude du modèle approximatif pour des systèmes complexes.

Plusieurs types de distribution de message peuvent être considérés. Cependant, nous ne nous intéresserons qu'à un type particulier afin de montrer comment les autres peuvent être analysés. Ainsi, nous modifierons la distribution des messages en imposant que les messages en provenance d'un processeur associé à un groupe donné soient dirigés vers les autres processeurs de ce groupe avec un pourcentage β. Par exemple, on peut imposer que 25% de tous les messages émis par les processeurs du groupe 1 de la figure 3.6 soient dirigés vers les autres processeurs de ce groupe, et ce de manière uniforme. Pour des raisons de simplicité, on peut ensuite supposer que le reste des messages est uniformément distribués. Le

reste des messages sera dirigé vers les autres groupes avec une probabilité égale à $(1-\beta)/(L-1)$ pour chacun des L-1 autres groupes.

Une modification de la distribution des messages entraîne nécessairement une modification des probabilités p_i. Ces probabilités sont déterminées de la même façon que celle montrée à la section 3.4. Comme on peut l'anticiper, la distribution des messages aura une influence significative sur la performance du système. Les résultats obtenus en utilisant le modèle approximatif ont été comparés à des résultats de simulation pour diverses valeurs de β de même que pour quelques configurations. Les figures 5.7, 5.8, 5.9 et 5.10 montrent le comportement du modèle approximatif pour le cas $\beta = 25\%$. L'exactitude du modèle approximatif pour des systèmes à 2 et à 5 niveaux est présentée aux figures 5.11 et 5.12. Pour la plupart des cas, l'exactitude du modèle approximatif est inférieure à 10%. Les deux observations notées dans le cas d'une distribution uniforme s'appliquent également ici.

À partir de ces résultats, nous pouvons conclure que le modèle approximatif développé est un outil donnant suffisamment de précision pour étudier des systèmes multiprocesseurs hiérarchiques de grande dimension sur une vaste gamme de conditions de trafic.

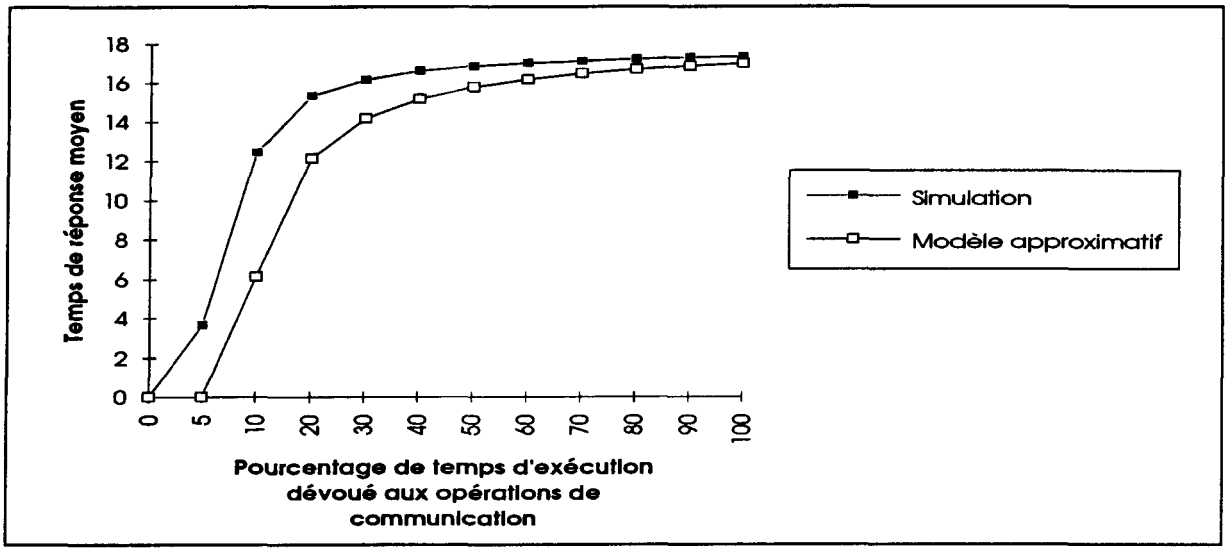

Figure 5.7 Un système à 2 niveaux composé de 10 processeurs où 25% des messages générés sont des messages locaux.

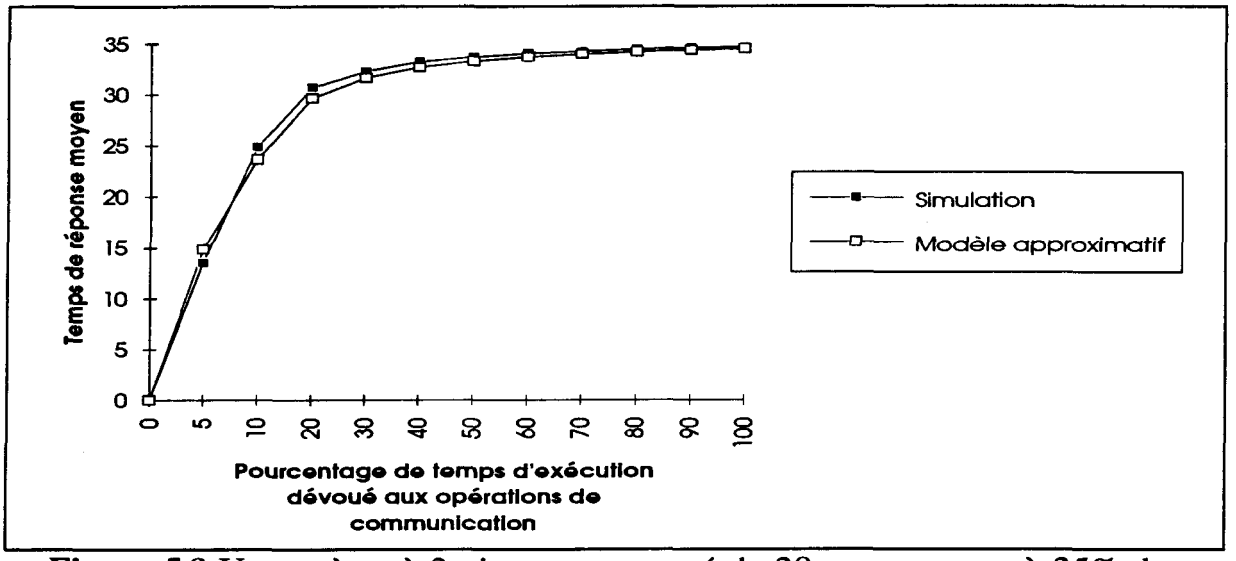

Figure 5.8 Un système à 2 niveaux composé de 20 processeurs où 25% des messages générés sont des messages locaux.

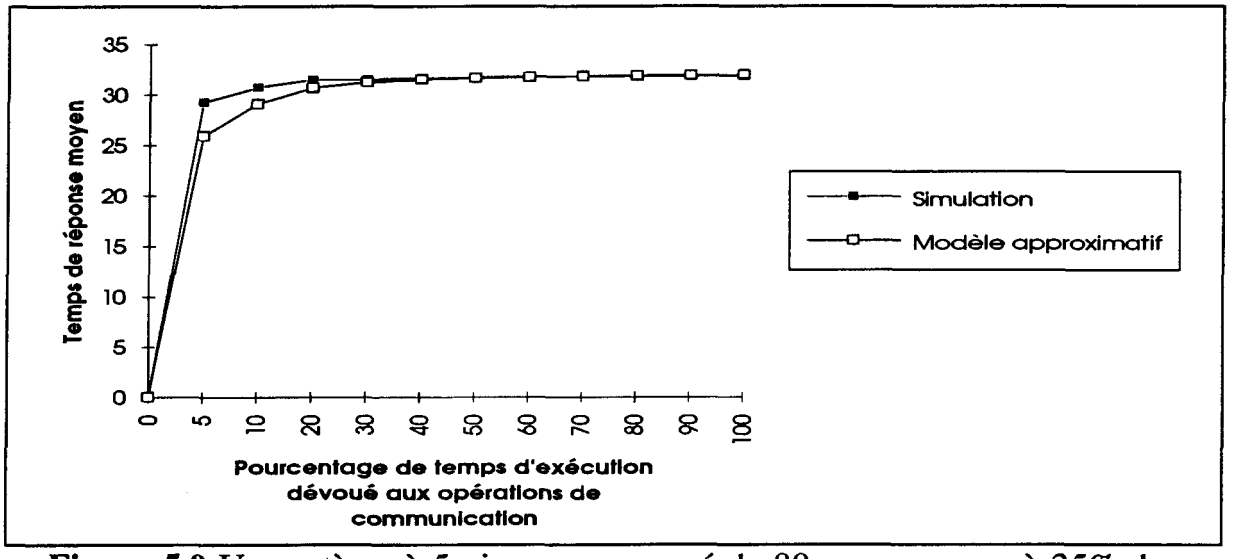

Figure 5.9 Un système à 5 niveaux composé de 80 processeurs où 25% des messages générés sont des messages locaux.

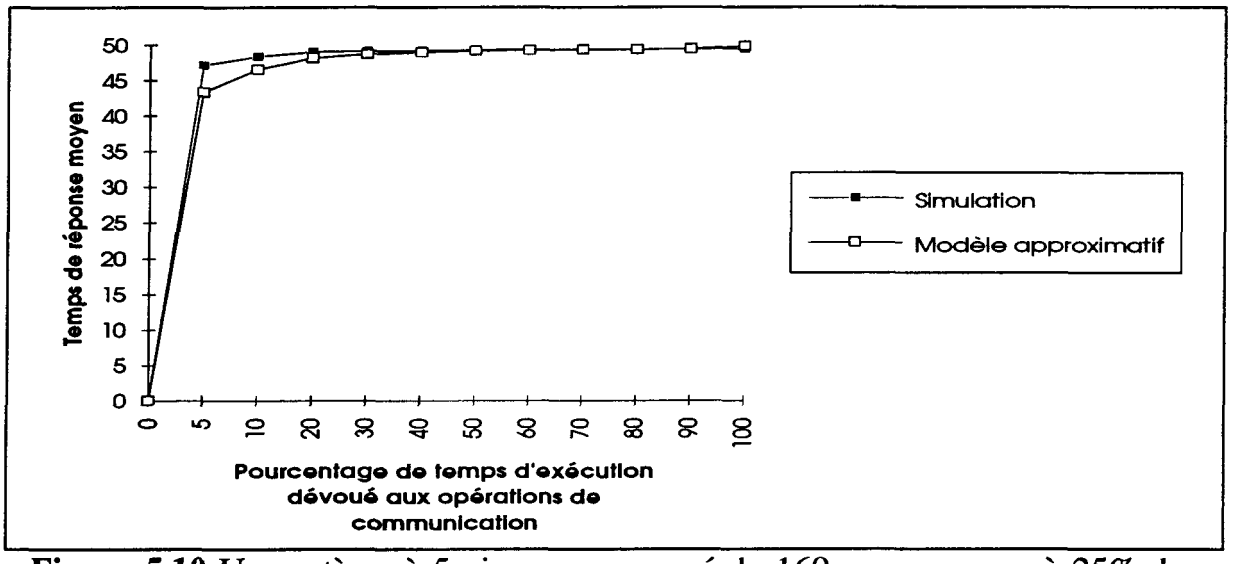

Figure 5.10 Un système à 5 niveaux composé de 160 processeurs où 25% des messages générés sont des messages locaux.

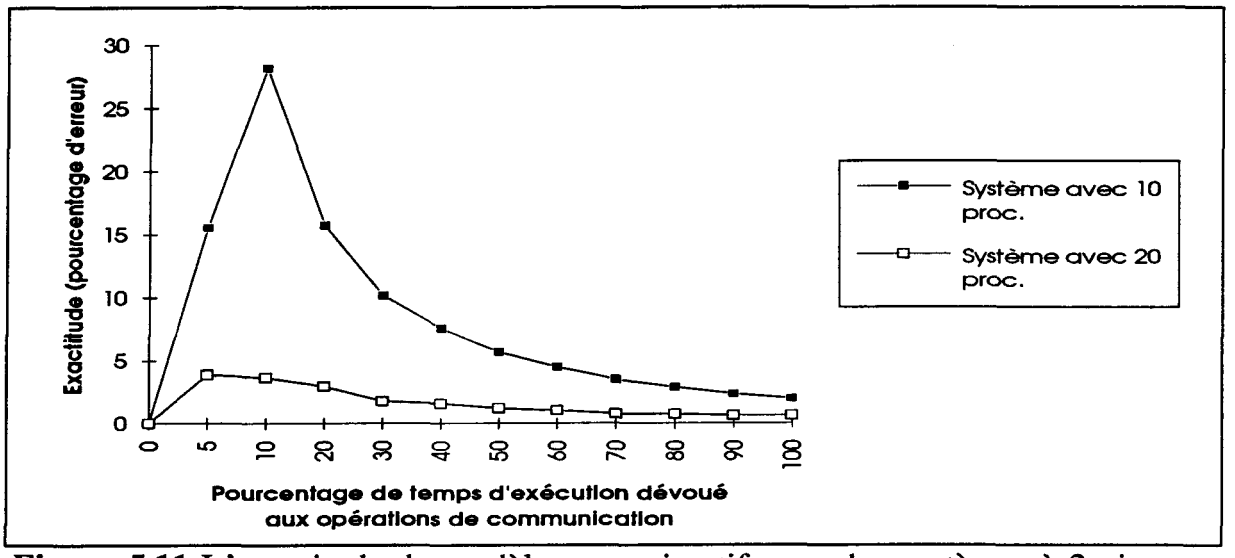

Figure 5.11 L'exactitude du modèle approximatif pour des systèmes à 2 niveaux où 25% des messages générés sont des messages locaux.

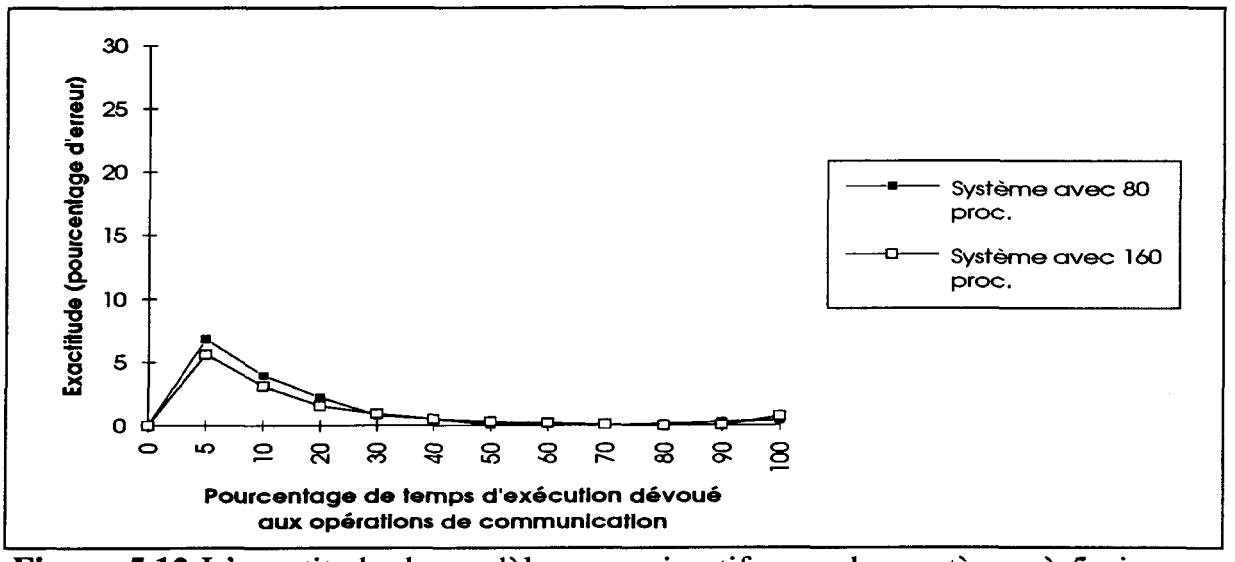

Figure 5.12 L'exactitude du modèle approximatif pour des systèmes à 5 niveaux où 25% des messages générés sont des messages locaux.

5.3 Analyse de la performance de systèmes hiérarchiques binaires

Dans cette sous-section, nous nous intéresserons plus particulièrement à l'aspect performance des systèmes hiérarchiques binaires dans deux situations différentes, soit lorsque la distribution des messages est uniforme et lorsqu'elle est non-uniforme.

A) Distribution de messages uniforme. Les figures 5.1, 5.2, 5.3 et 5.4 montraient le temps de réponse de certains systèmes en fonction du pourcentage (α) de temps d'exécution consacré aux activités de communication. Comme on pouvait s'y attendre, le temps de réponse augmente lorsqu'on augmente le pourcentage de communication (α). De plus, lorsque la taille du système s'accroit, le temps de réponse observé, lorsque α est petit, atteint plus rapidement une certaine limite supérieure. Notons que cette limite dépend du nombre de niveaux et du nombre de processeurs tel que montré à la section 5.2.1.

B) Distribution de messages non-uniforme. Les figures 5.13 et 5.14 présentent le temps de réponse obtenu avec le modèle approximatif pour diverses distributions de messages (caractérisées par β) en fonction du pourcentage de

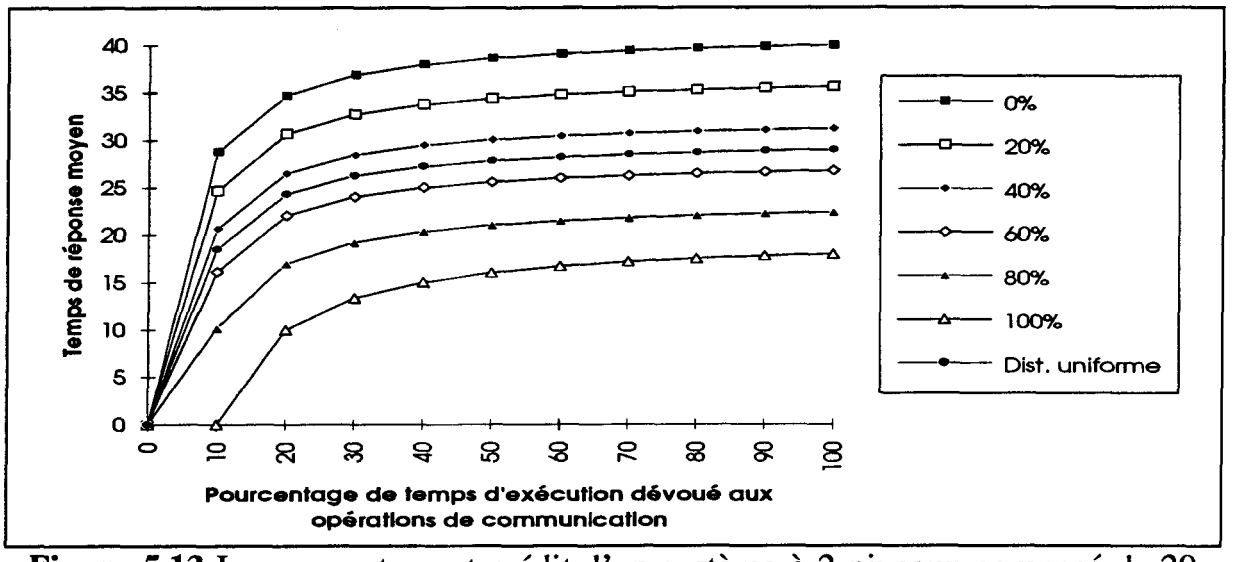

Figure 5.13 Le comportement prédit d'un système à 2 niveaux composé de 20 processeurs pour divers pourcentages de messages locaux.

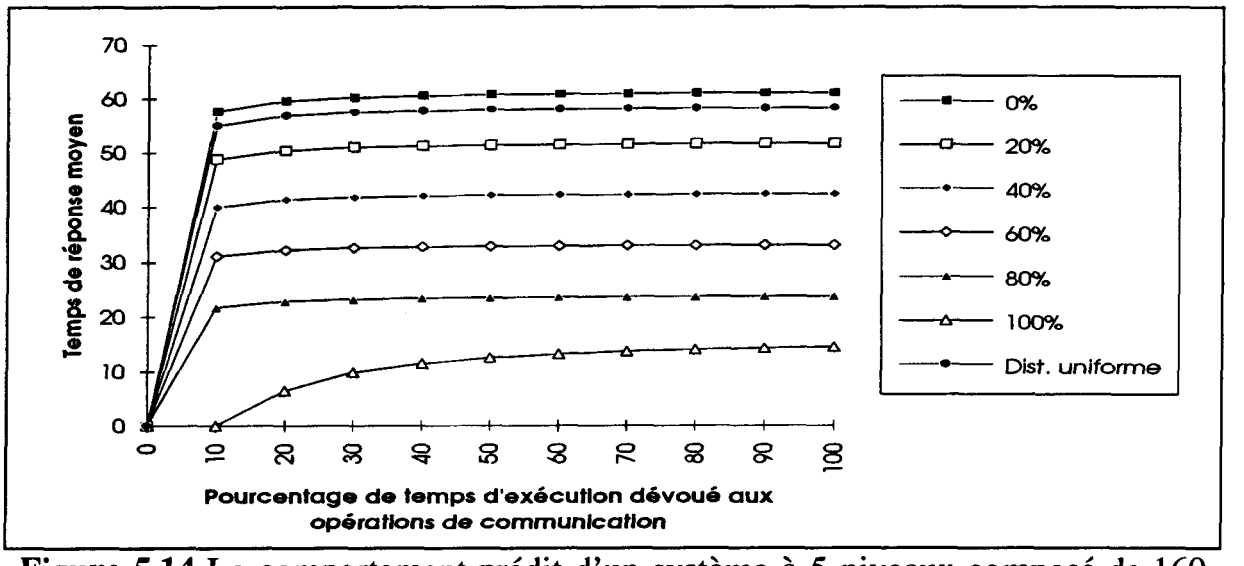

Figure 5.14 Le comportement prédit d'un système à 5 niveaux composé de 160 processeurs pour divers pourcentages de messages locaux.

communication (α). Les résultats montrent que le temps de réponse augmente lorsque le pourcentage de communication augmente et que, pour un même

pourcentage de communication, celui-ci diminue lorsque le pourcentage de messages locaux augmente. Ce dernier comportement pouvait être prévu. En effet, le pourcentage de messages locaux affecte le flux de messages à travers les bus du système. Il est normal que l'accusé réception parvienne plus rapidement à l'expéditeur si le message original n'utilise que le bus local de bas niveau. Ainsi, lorsque le pourcentage de messages locaux augmente, le temps de réponse du système doit diminuer, ce qui est effectivement observé aux figures 5.13 et 5.14.

5.4 Optimisation de systèmes hiérarchiques binaires

Un des principaux objectifs de ce travail consistait à utiliser le modèle développé pour optimiser des architectures multiprocesseurs. Dans cette section, nous considérerons donc le problème de l'optimisation d'architectures hiérarchiques binaires comme celles montrées aux figures 3.1 et 3.2. Le but visé par cette optimisation est de déterminer le nombre optimal de niveau du réseau de communication qui minimisera le temps de réponse d'un système utilisant un nombre de processeurs donné. Or, le modèle de performance développé nous permet de déterminer un réseau de communication optimal. Par exemple, pour construire un

système de 128 processeurs, les deux configurations suivantes sont possibles:

i) un système à 5 niveaux avec 8 processeurs connectés à chacun des 16 bus de bas niveau,

ii) un système à 2 niveaux avec 64 processeurs connectés à chacun des 2 bus de bas niveau.

Dans les paragraphes qui suivent, une méthode sera développée pour déterminer, par exemple, laquelle de ces deux configurations produira le temps de réponse minimal sous certaines conditions. Cette méthode nous permettra aussi d'introduire certains paramètres afin de comparer diverses configurations en fonction de leur coût d'implantation.

Soit M le nombre total de processeurs dans le système, et N le nombre de niveaux du réseau d'interconnexion. Dans un système binaire, nous aurons donc $M/2^{N-1}$ processeurs connectés à chaque bus de bas niveau. Notons que le nombre de niveaux du réseau doit être inférieur ou égal à $\log_2(M)+1$ (c.-à-d., $N \leq \log_2(M)+1$) afin d'assurer que le nombre de processeurs sur chacun des bus de bas niveau est égal ou supérieur à 1. Puisque l'un des paramètres dépend de l'autre, le problème d'optimisation se résumera à déterminer la valeur N qui minimisera le temps de réponse du système pour un nombre de processeurs donné.

Avec le modèle développé, il est relativement facile de déterminer le réseau optimal en examinant toutes les configurations possibles du système. Par exemple, examinons deux systèmes hiérarchiques binaires. Ces deux systèmes sont composés respectivement de 128 et de 192 processeurs. Traçons le temps de réponse de ces systèmes pour certains nombres de niveaux (N) et ainsi que pour certains pourcentages de messages locaux (β). Pour simplifier la discussion, le pourcentage de communication (α) sera fixé à 100%. Nous devons mentionner que des comportements analogues à ceux observés pour α=100% sont aussi observés pour $0\% \leq \alpha < 100\%$. Les figures 5.15, 5.16 et 5.17 montrent les temps de réponse moyens estimé et réel (simulation) pour toutes les configurations permises.

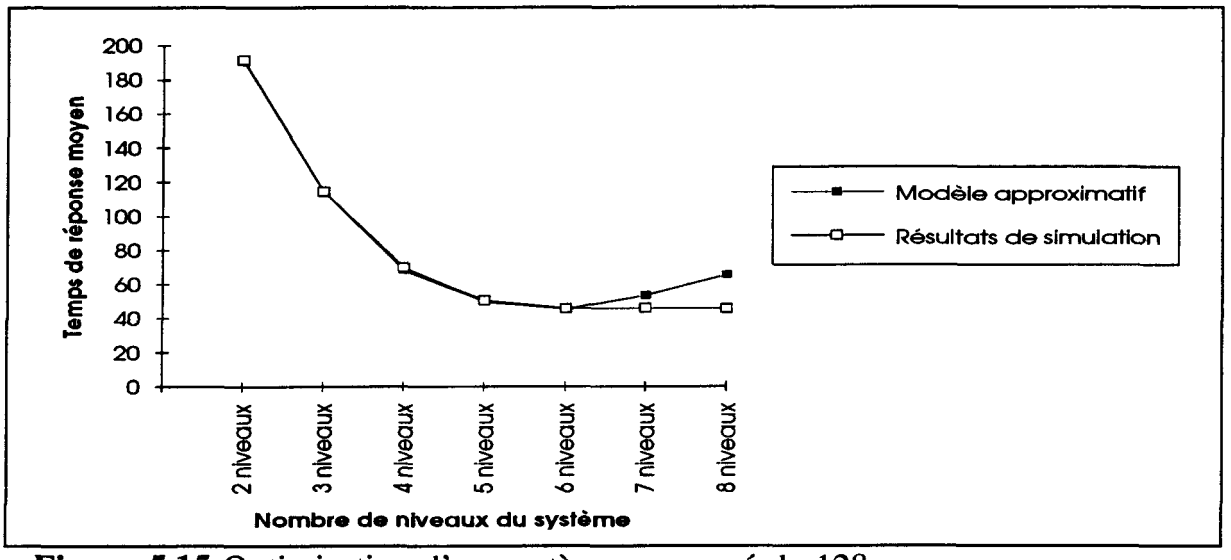

Figure 5.15 Optimisation d'un système composé de 128 processeurs avec une distributions (β) de messages uniforme et α=100%.

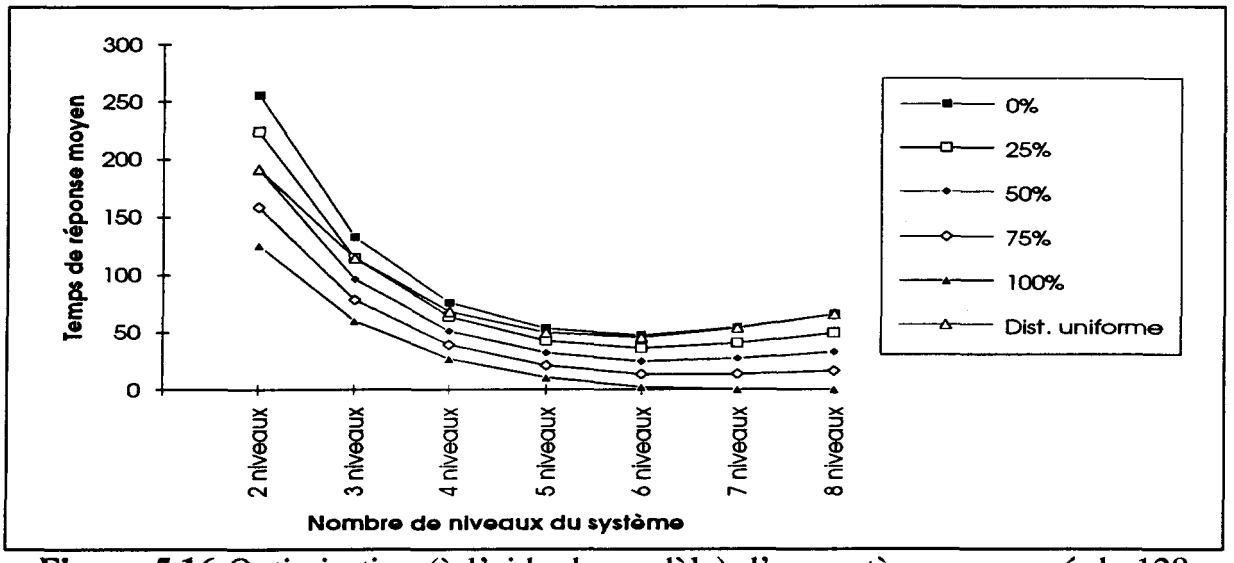

Figure 5.16 Optimisation (à l'aide du modèle) d'un système composé de 128 processeurs avec différentes distributions (β) de messages et α=100%.

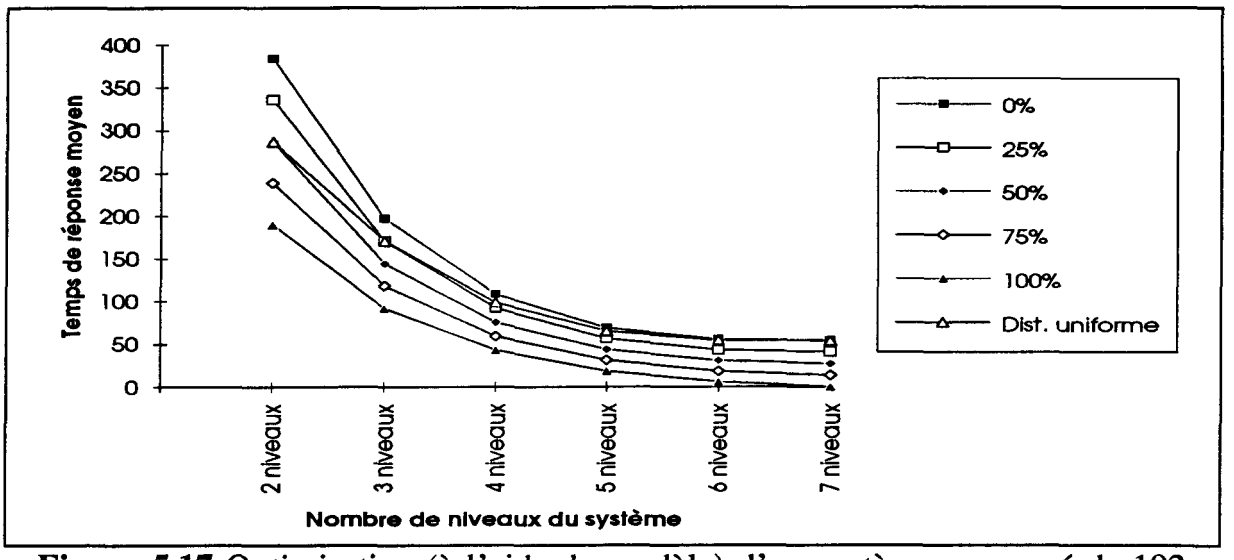

Figure 5.17 Optimisation (à l'aide du modèle) d'un système composé de 192 processeurs avec différentes distributions (β) de messages et α=100%.

La figure 5.15 montre les résultat obtenus pour un système de 128 processeurs. Le temps de réponse diminue lorsque le nombre de niveaux du système augmente. On peut expliquer cette observation de la façon suivante. En effet, plus le système est profond, plus le nombre de bus augmente. Ainsi, le nombre de liens de communication dans le système augmente. Ceci permet de diminuer les conflits d'accès ainsi que la longueur de la file d'attente de chaque bus. Conséquemment, le temps de réponse du système diminue. Les figures 5.16 et 5.17 présentent les temps de réponse obtenus avec le modèle pour les systèmes de 128 processeurs et de 192 processeurs, et ceux pour différents pourcentages de messages locaux (β). On

observe que le temps de réponse diminue également lorsque le nombre de niveaux du système augmente. Ceci nous permet donc de conclure que la configuration fournissant le temps de réponse minimal est celle où le nombre de niveaux est maximal.

Notons que le nombre de bus dans un système est égal à 2^N-1, où N est le nombre de niveaux du système. Or, le nombre de bus augmente de façon importante lorsque le nombre de niveaux augmente. Ceci aura une influence significative sur le coût d'implantation du système. Nous reviendrons sur ce sujet à la section 5.6.

5.5 Analyse de systèmes hiérarchiques quaternaires

Jusqu'à présent, nous avons étudié la performance des architectures hiérarchiques binaires. Ce type d'architecture n'est qu'une des classes de configuration possibles. Bien qu'il soit impossible de les étudier toutes, nous nous intéresserons à une autre classe de configuration qui nous permettra de tirer des conclusions plus générales, soit les réseaux dits "quaternaires". La figure 5.18 montre un réseau quaternaire à 3 niveaux. On remarque que tous les bus de niveau

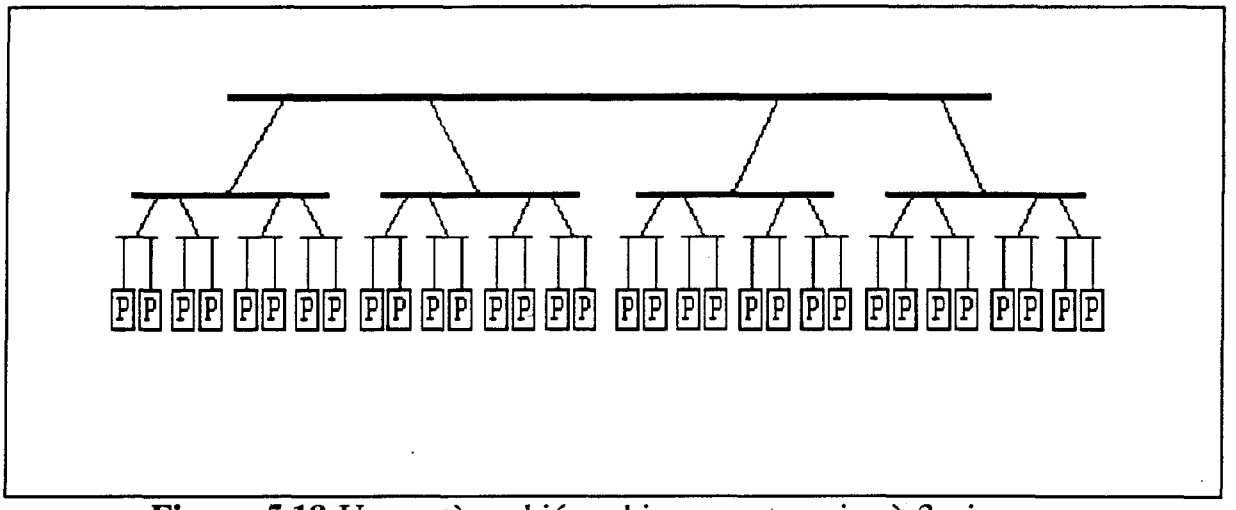

Figure 5.18 Un système hiérarchique quaternaire à 3 niveaux composé de 32 processeurs.

intermédiaire et celui de haut niveau sous-tendent 4 bus. Pour simplifier les analyses et les discussions, nous nous restreindrons au cas où la distribution de messages est uniforme. En utilisant le modèle approximatif développé précédemment, nous pouvons déterminer le temps de réponse d'un système quaternaire sous diverses conditions de trafic. En effet, il est relativement facile de généraliser la méthodologie développée au chapitre 3 car la configuration du réseau de communication ne modifiera que les probabilités de requête (p_i). Pour tester l'exactitude du modèle approximatif lorsqu'il est appliqué aux systèmes quaternaires, nous avons comparé les résultats du modèle approximatif et ceux obtenus par simulation. Ces résultats sont présentés aux figures 5.19 et 5.20. La figure 5.21 montre que l'erreur sur le temps d'exécution estimé (c'est-à-dire, l'exactitude du modèle) est bien inférieure

à 10% pour plusieurs valeurs de α. Nous avons aussi comparé la performance des systèmes binaires et des systèmes quaternaires (Figures 5.22 et 5.23). Comme on peut le remarquer, pour un nombre donné de processeurs, le réseau quaternaire possède un temps de réponse moindre que celui-ci du réseau binaire. Notons que le réseau quaternaire est cependant plus coûteux que le réseau binaire car il requiert plus de bus. Cette dernière observation nous amène à penser qu'il doit exister une façon de déterminer le meilleur choix de configuration en considérant à la fois le temps de réponse et le coût d'implantation. Ce sujet sera abordé à la section suivante.

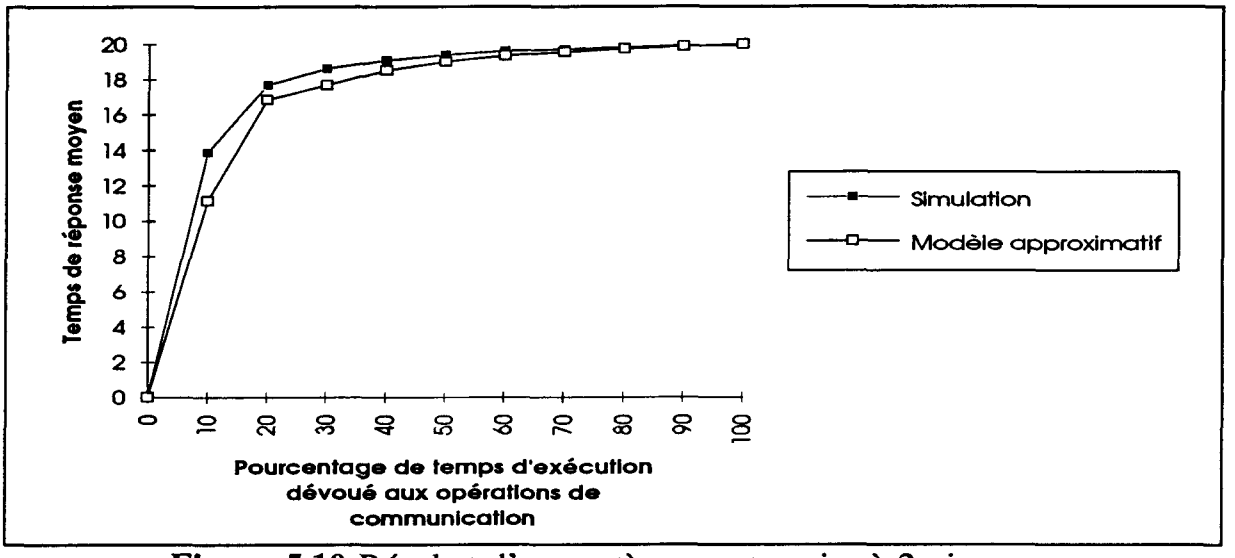

Figure 5.19 Résultat d'un système quaternaire à 2 niveaux composé de 20 processeurs.

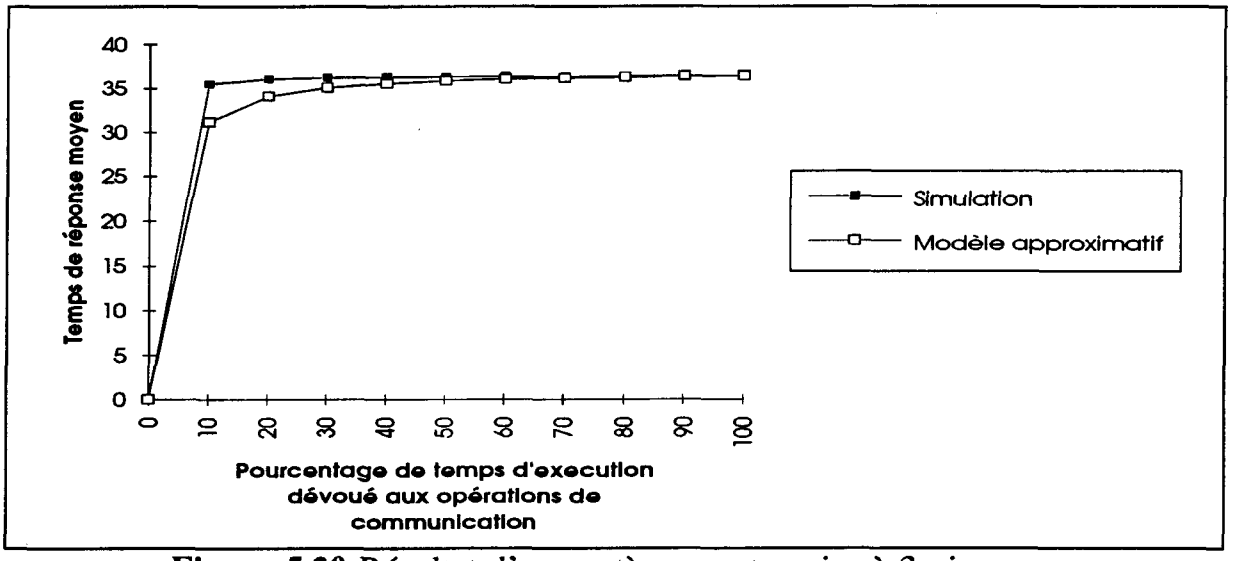

Figure 5.20 Résultat d'un système quaternaire à 3 niveaux composé de 80 processeurs.

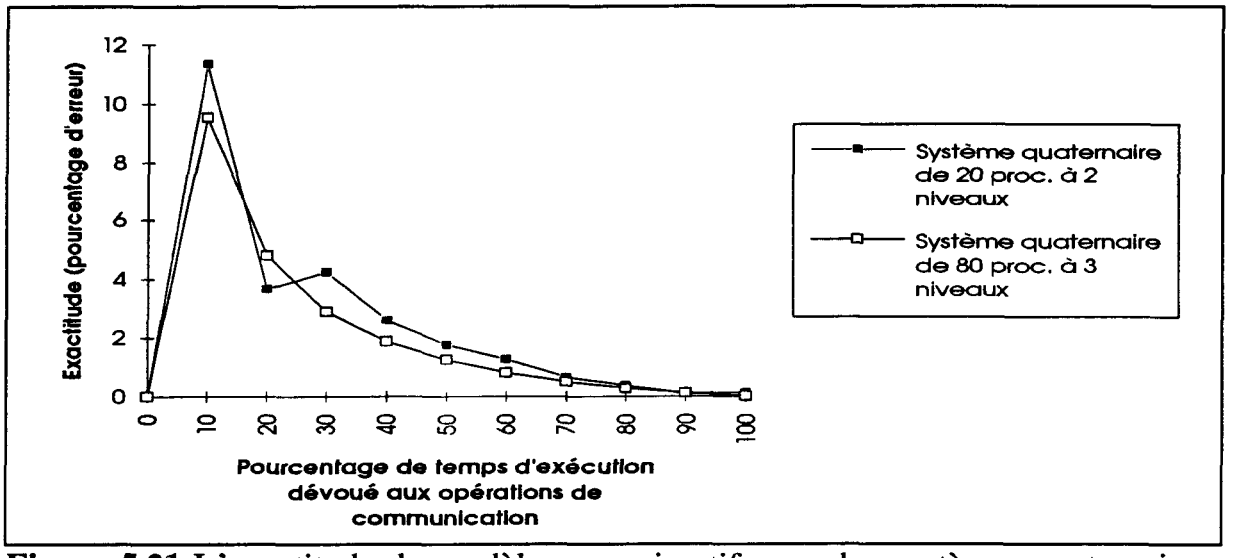

Figure 5.21 L'exactitude du modèle approximatif pour des systèmes quaternaires.

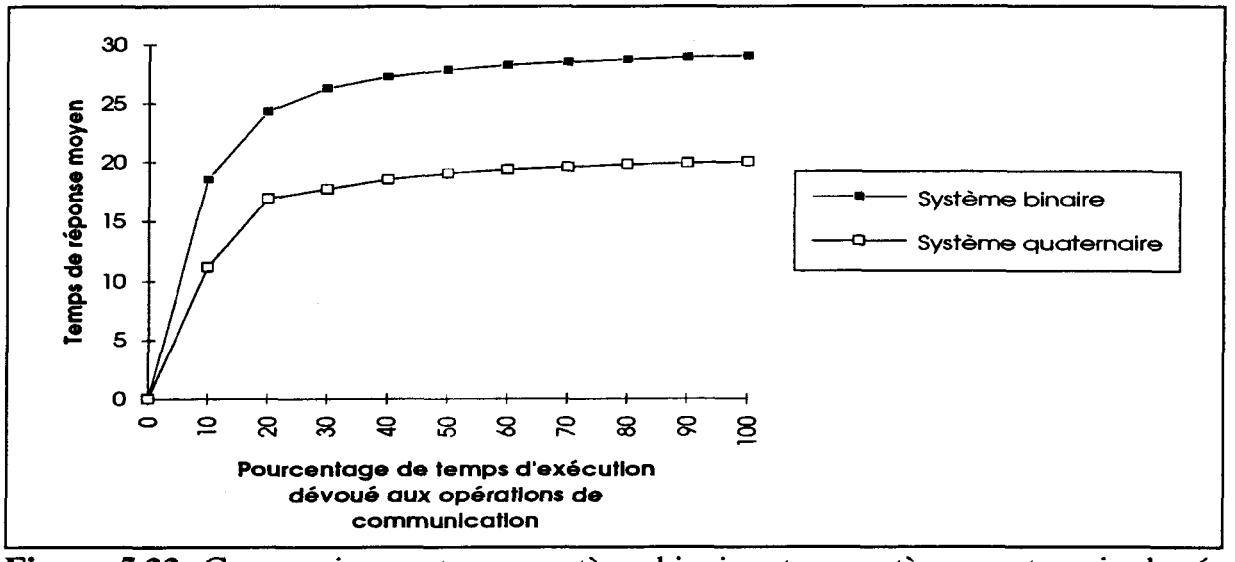

Figure 5.22 Comparaison entre un système binaire et un système quaternaire basés sur une hiérarchie à 2 niveaux, chacun composé de 20 processeurs.

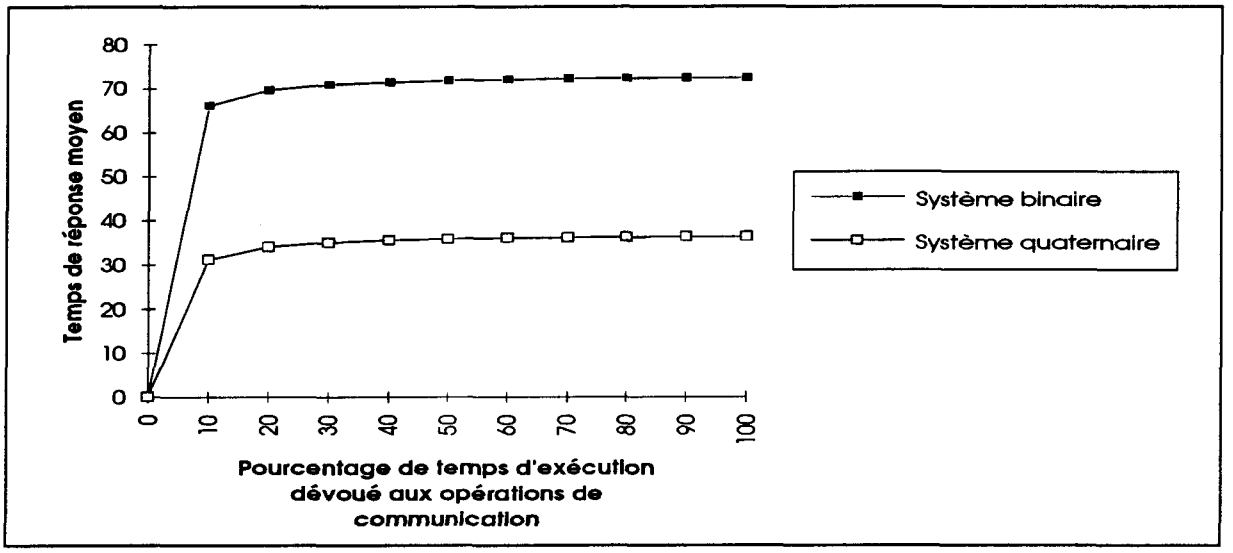

Figure 5.23 Comparaison entre un système binaire et un système quaternaire basés sur une hiérarchie à 3 niveaux, chacun composé de 80 processeurs.

5.6 Analyse coût - performance

Le modèle de performance développé précédemment peut être utilisé pour étudier l'espace de conception des systèmes hiérarchiques. Comme nous l'avons vu, les réseaux de communication hiérarchiques possédant un grand nombre de niveaux sont avantageux du point de vue du temps de réponse. Cependant, ceux-ci exigent un coût d'implantation plus élevé car le nombre de bus et d'aiguilleurs augmente exponentiellement avec le nombre de niveaux. Il serait très utile, entre autres, aux concepteurs de tels systèmes d'avoir un outil permettant de déterminer une configuration offrant le meilleur compromis entre la performance et le coût. Bien

qu'il soit possible de définir des critères relativement élaborés pour définir le "point optimal", nous montrerons comment il est possible d'établir une métrique simple. La procédure qui sera élaborée pourra, bien sûr, être appliquée à des métriques plus complexes. Nous limiterons donc volontairement notre analyse car l'espace de conception est très grand.

Nous introduirons trois hypothèses simplificatrices. Tout d'abord, nous ne considérerons que des hiérarchies binaires. En second lieu, nous considérerons que la dimension du système n'est pas restreinte. Autrement dit, la taille du système peut croître sans aucune contrainte. De plus, nous allons supposer que le coût du réseau d'interconnexion croît comme la fonction 2^i dans une hiérarchie binaire, où i représente le niveau. Cette troisième hypothèse implique que le coût du réseau est proportionnel au nombre de bus dans la hiérarchie.

Nous définirons une métrique Q (appelée coût de conception) telle que

$$Q = \frac{\textit{puissance de traitement}}{\textit{coût d'implantation}} \qquad (5.2)$$

La puissance de traitement théorique est simplement égale au nombre de processeurs M dans le système multiplié par leur pourcentage d'utilisation. Le pourcentage d'utilisation peut être obtenu à partir du rapport

$$\textit{Pourcentage d'utilisation} = \frac{T(t_{réponse} = 0)}{T(t_{réponse\ réel})} \tag{5.3}$$

Ce dernier peut être estimé par

$$\textit{Pourcentage d'utilisation} = \frac{T(t_{réponse} = 0)}{T(t_{réponse\ estimé})} \tag{5.4}$$

Pour estimer le coût d'implantation, il nous faut déterminer le coût du réseau d'interconnexion. En utilisant la troisième hypothèse, le coût d'implantation total du système est égal à

$$\textit{coût d'implantation} = \sum_{i=0}^{N-1} 2^i = 2^N - 1 \tag{5.5}$$

En substituant 5.4 et 5.5 dans 5.2, on obtient

$$Q = \frac{M\left\{\dfrac{(1-\alpha)t_a + \alpha t_b}{(1-\alpha)t_a + \alpha(t_b + t_{réponse\ estimé})}\right\}}{(2^N - 1) + k_1 M} \tag{5.6}$$

Dans cette équation, nous avons introduit un coefficient k_1 qui permet pondérer le coût des processeurs en fonction de celui du réseau.

En conservant toute la généralité, nous pouvons supposer que t_a et t_b sont tous deux égaux à 1. Ainsi, l'équation 5.6 devient

$$Q = \frac{M}{(1-\alpha t_{réponse\ estimé})(2^N-1 + k_1 M)} \qquad (5.7)$$

Les figures 5.24 et 5.25 montrent respectivement les valeurs de la métrique Q pour un système comportant 128 processeurs lorsque $k_1=0$ et $k_1=1$. Pour ces deux figures, la valeur de α a été fixée à 100%. Notons que la valeur optimale de Q est celle dont la valeur est la plus grande. Figure 5.24 montre que la configuration à 2 niveaux fournit la valeur maximale de la métrique Q. Cependant, lorsque $k_1=1$, cette métrique est maximale pour une configuration à 5 niveaux (Figure 5.25).

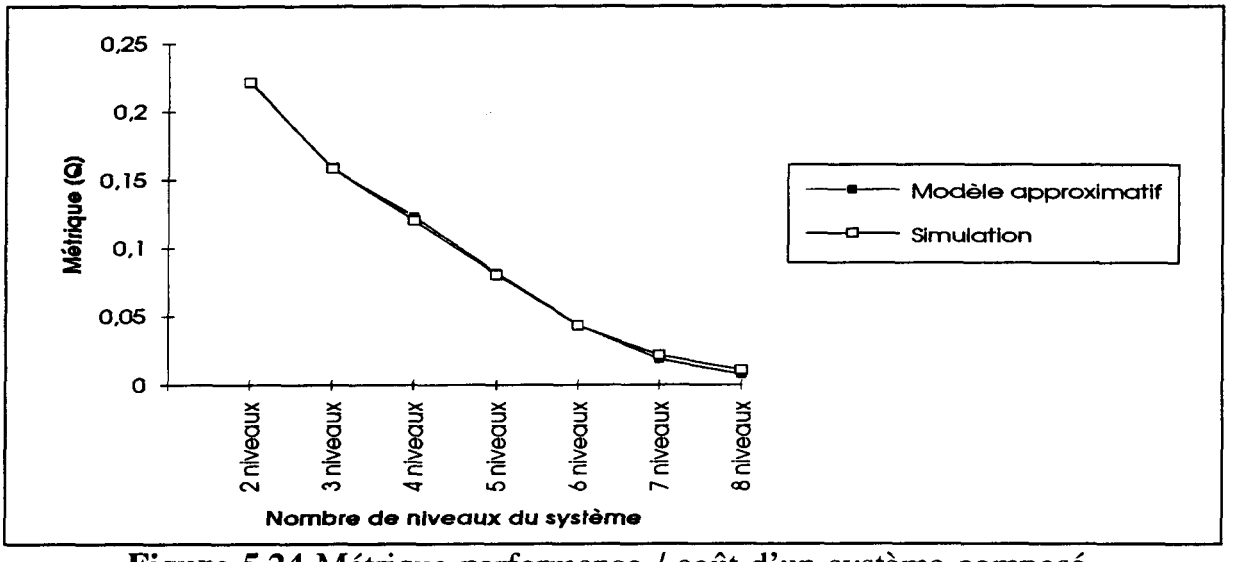

Figure 5.24 Métrique performance / coût d'un système composé de 128 processeurs avec $k_1=0$ et $\alpha=100\%$.

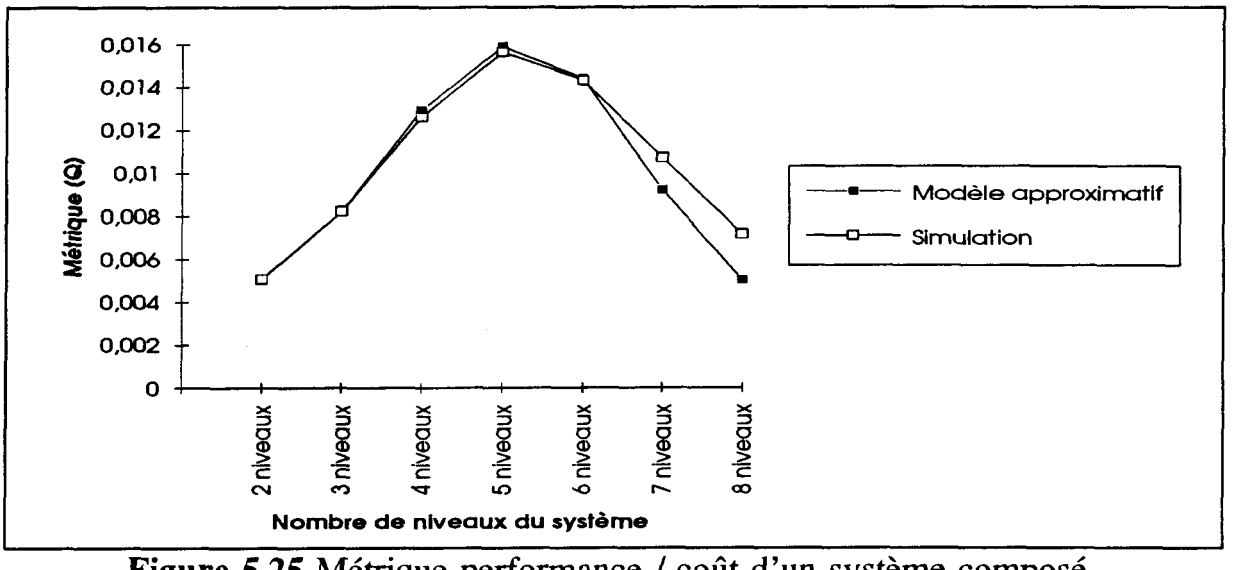

Figure 5.25 Métrique performance / coût d'un système composé de 128 processeurs avec $k_1=1$ et $\alpha=100\%$.

5.7 Résumé

Nous avons présenté les résultats obtenus au cours de ce travail. Nous avons montré que le modèle approximatif développé évalue la performance des systèmes multiprocesseurs hiérarchiques avec une précision adéquate pour une vaste gamme de configurations et de conditions de trafic. En utilisant ce modèle, nous avons analysé la performance des systèmes hiérarchiques ainsi que l'optimisation de ces systèmes. De plus, nous avons développé un outil basé sur le modèle développé qui nous permet d'étudier la maximisation du rapport "performance / coût" pour des systèmes hiérarchiques.

CHAPITRE 6

CONCLUSION

Dans ce mémoire, nous avons présenté une méthodologie permettant de modéliser la performance de systèmes informatiques. Nous avons ensuite utilisé cette méthodologie pour modéliser la performance d'une nouvelle architecture multiprocesseur hiérarchique afin d'étudier l'efficacité du réseau de communication de cette architecture. Le temps de réponse introduit par le protocole d'échange de données et par la structure du réseau de communication a été utilisé comme indice de performance pour le réseau de communication. Ce temps de réponse, généralement exprimé sous la forme d'un délai moyen, dépend de plusieurs facteurs dont le nombre de niveaux du réseau de communication, le nombre de processeurs

dans le système et la fraction du temps d'exécution consacrée aux activités de communication.

Certaines hypothèses ont été introduites afin de pouvoir caractériser les systèmes multiprocesseurs hiérarchiques. Nous avons ainsi développé un modèle analytique approximatif basé sur des modèles de réseaux de files d'attente en vue d'évaluer le temps de réponse moyen. Parallèlement, nous avons construit un simulateur fonctionnel de systèmes hiérarchiques. Le modèle approximatif a été validé à l'aide de ce simulateur ainsi que par une analyse simple. Plusieurs expériences ont été réalisées pour une vaste gamme de conditions de configuration et de trafic du réseau de communication. Les résultats obtenus montrent que le modèle approximatif estime la performance des systèmes hiérarchiques avec une bonne précision pour des systèmes de grande dimension.

L'optimisation de systèmes hiérarchiques a également été étudiée. Dans cette étude, le rapport "performance / coût" de ces systèmes a été considéré. Une métrique particulière a été développée et il a été montré que les modèles développés pouvaient être facilement mis à profit lors de la conception de systèmes parallèles de haute performance basés sur cette architecture.

Dans le présent projet, nous avons réalisé une analyse de la performance d'architectures hiérarchiques binaires et quaternaires. Pour des travaux futurs, les modèles développés permettent d'envisager l'analyse de structures plus complexes. Des résultats préliminaires, non-montrés dans ce travail, ont en effet permis d'appliquer un modèle approximatif à une architecture hybride. Une autre perspective intéressante consisterait à étudier l'implantation d'une architecture hiérarchique en utilisant les technologies d'intégration à très grande échelle (VLSI). Les contraintes physiques inhérentes aux technologies d'intégration (résistance, capacité) auraient sûrement une influence importante sur la performance du réseau de communication. Le modèle approximatif développé pourrait être très utile pour en optimiser l'efficacité selon différentes métriques.

ANNEXE

Bref guide d'utilisation du simulateur de systèmes multiprocesseurs hiérarchiques (TREE_BUS)

* **Paramètres d'entrée**: Ces paramètres permettent de modifier la configuration du système simulé. L'utilisateur doit modifier ces paramètres dans le programme-source (tree_bus.c) lorsqu'il désire modifier la configuration à simuler.

- NIVEAU_MAX: définit le nombre de niveaux du système
- N_BUS_MAX: définit le nombre total de bus du système
- N_LOW_BUS: définit le nombre de bus de bas niveau
- N_PROC_MAX: définit le nombre total de processeurs du système
- N_PROC_BAS: définit le nombre de processeurs à chaque bus de bas niveau

- Distribution de messages:

    + Distribution de messages uniforme:

        #undef NON_UNIFORM

    + Distribution de messages non-uniforme:

        #define NON_UNIFORM

        #define LOCAL_DISTRIBUTION < pourcentage de messages locaux >

*** Pour compiler le programme (tree_bus.c):**

cc -o tree_bus tree_bus.c -lm (sur SUN OS 4.1)

*** Pour exécuter le programme (tree_bus):**

tree_bus

*** Fichier de sortie: tree_bus.dat**

Tous les résultats statistiques sont retrouvés dans le fichier.

BIBLIOGRAPHIE

[1] Anup K. Ahluwalia, Mukesh Singhal, "Performance Analysis of the Communication Architecture of the Connection Machine", IEEE Trans. Parallel and Distributed Systems, Vol. 3, No. 6, Nov. 1992, pp. 728-738.

[2] Arnold O. Allen, "Queueing Models of Computer Systems", IEEE Comput., April 1980, pp. 13-24.

[3] J. P. Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers", Comm. ACM, Vol. 16, No. 9, Sept. 1973, pp. 527-531.

[4] J. P. Buzen, Queueing Network Models of Multiprogramming, Ph.D. Thesis, Div. of Engineering and Applied Physics, Harvard University, Cambridge, Mass., May 1971.

[5] Mercedes Granda, et al., "Performance Evaluation of Parallel Systems by Using Unbounded Generalized Stochastic Petri Nets", IEEE Trans. Software Engineering, Vol. 18, No. 1, Jan. 1992, pp. 55-71.

[6] Hwang, K., Briggs, F., Computer Architecture and Parallel Processing, McGraw-Hill, New York, 1984.

[7] Raj Jani, The Art of Computer System Performance Analysis, John Wiley & Sons Inc., 1991.

[8] R. Kermouche, Y. Savaria, D. Audet, "Harvest Model of an Integrated Hierarchical-Bus Architecture", Proc. IEEE Int. Conf. on WSI, San Francisco, Jan. 1994, pp. 69-78.

[9] L. Kleinrock, Queueing Systems, John Wiley, New York, 1976.

[10] Catherine A. Lamana, Wade H.Shan, "A Performance Study of the Hypercube Parallel Processor Architecture", Simulation, Mar. 1991, pp. 185-196.

[11] M. Ajmone Marsan, F. Gregoretti, "Memory Interference Models for a Multimicroprocessor System with a Shared Bus and a Single External Common Memory", EUROMICRO J., Feb. 1981, pp. 124 - 133.

[12] M. Ajmone Marsan, Mario Gerla, "Markov Models for Multiple Bus Multiprocessor Systems", IEEE Trans. Comput., Vol. c-31, No. 3, Mar. 1982, pp. 239-248.

[13] M. Ajmone Marsan, et al., Performance Models of Multiprocessor Systems, The Massachusetts Institute of Technology, 1986.

[14] T.N Mudge, J.P. Hayes, D.C. Winsor, "Multiple Bus Architectures", IEEE Comput., June 1987, pp. 42-48.

[15] T.N Mudge, H.B. Al-Sadoun, "A Semi-Markov Model for the Performance of Multiple-bus Systems", IEEE Trans. Comput., Oct. 1985, pp. 934-942.

[16] Ashani K. Ramani, et al., "A General Model for Performance Investigations of Priority Based Multiprocessor System", IEEE Trans. Compt., Vol. 41, No. 6, Jun. 1992, pp. 747-754.

[17] Y. Savaria, "Parallel Microprocessor Architecture", US Patent application 07/310 828 (also filed as PCT/CA90/0041).

[18] Harold S. Stone, High-Performance Computer Architecture, Addison Wesley, 1990.

[19] D. Towsley, "Approximate Models of Multiple Bus Microprocessor Systems", IEEE Comput., Mar. 1986, pp. 220-227.

[20] Dalibor F. Vraslovic et al., "Performance Prediction and Calibration for a Class of Multiprocessors", Trans. Comput., Vol. 37, No. 11, Nov. 1988, pp. 1353-1365.

[21] Donald C. Winsor, Trevor N. Mudge, "Analysis of Bus Hierarchies for Multiprocessors", The 15th Annual International Sym. on Comput. Architecture, Hawaii, May 30 1988, pp. 100-107.