Compromis entre Efficacité et Accélération dans les Systèmes Parallèles—Synthèse

A. Jebali et I. Chihi

Décembre 1997

Faculté des Sciences de Tunis

National School for Computer Sciences

Basé sur
Speedup Versus Efficiency in Parallel Systems
Derek Eager, John Zahorjan, et Edward Lazowska
IEEE Transactions on Computers, Vol. 38, NO. 3, MARS 1989.

1 Introduction

Les besoins en calculs des applications modernes croissent sans cesse. Qu'ils soient issus du domaine de l'ingénerie ou de la recherche, les problèmes résolus débouchent le plus souvent sur des problèmes encore plus complexes ou alimentent l'envie de résoudre d'autres dont, auparavant, la résolution n'était même pas envisageable.

En 1967, G. M. Amdahl publie un article dans lequel il remet en cause l'approche du processeur unique pour l'exécution de tâches de grande envergure. Depuis, le traitement parallèle est devenu une approche de plus en plus adoptée. Une exécution en parallèle consiste à structurer un programme en plusieurs sous-tâches aussi indépendantes que possible.

Dans une exécution séquentielle, la performance d'un système peut être décrite fidèlement par le taux d'instructions exécutées par le processeur et par le temps requis par le programme.

Dans un environnement parallèle, les choses sont plus compliquées. Il faut, en effet, tenir compte de facteurs supplémentaires tant matériels que logiciels tels le nombre de processeurs et la structuration du programme en tâches.

Dans l'évaluation d'un système parallèle deux mesures d'un intérêt majeur sont l'*accélération* et l'*efficacité*. L'accélération est définie, pour un nombre de processeurs n donné, comme le rapport du temps d'exécution séquentiel (sur un seul processeur) au temps écoulé lorsque n processeurs sont disponibles. Ainsi,

$$S(n) = \frac{T_1}{T_n}$$

L'efficacité est définie comme l'utilisation moyenne des n processeurs. L'accélération est reliée á l'efficacité par,

$$E(n) = \frac{S(n)}{n}$$

Le cas idéal serait d'avoir une accélération linéaire. Mais la concurrence pour l'obtention des ressources, les communications inter-processeurs et inter-processus qui croissent avec n rendent cette fin impossible à atteindre. Des tarvaux de Minsky et Papert ont montré qu'une accélération "typique" est de la forme,

$$S(n) = log(n)$$

Beaucoup de travaux ont étés menés afin d'améliorer les performances globale d'un sysème parallèle. Ces travaux couvrent la conception de compilateurs, l'interconnection des processeurs, etc.

2 Le problème traité

Ce travail traite le problème d'un point de vue très abstrait. Les auteurs étudient le compromis entre l'accélération et l'efficacité dans un cas très général sans oublier les considérations qui pourraient apparaître avec des systèmes réels.

- à quel point le *degrè de parallélisme moyen* peut-il affecter le compromis entre l'accélération et l'efficacité ?,

- à quel point l'accélération et l'efficacité peuvent-elles se dégrader simultanément ?,
- quelle configuration maximise-t-elle le gain en accélération par unité d'efficacité perdue ?,
- pour atteindre une accélération donnée qu'elle pénalité sur l'efficacité doit-on payer ?,
- comment l'accélération et l'efficacité évoluent-elles avec le nombre de processeurs disponibles ?,
- quel nombre de processeur utiliser pour maximiser les gains en accélération par unité d'efficacité ?

3 Objectifs

L'objectif de ce travail est d'établir des frontières (limites) de performance. Il serait plus intéressant d'avoir des évaluations exactes des performances si on avait tous les détails sur les paramètres. Le problème avec les réponses détaillées est qu'en pratique il n'est pas possible de collecter toutes les données nécéssaires pour faire des affirmations précises, de plus, les expressions de limites exprimées en fonction de peu de variables peuvent donner une vue approximative mais plus claire et plus simple du comportement du système. Par exemple, la loi d'Amdahl,

$$S(n) \leq \frac{1}{f+\frac{1-f}{n}},$$

qui n'utilise qu'un seul paramètre caractérisant le logiciel (f) et un seul paramètre caractérisant le matériel (n) donne une idée plus claire qu'une liste détaillée des accélérations.

4 Préliminaire

4.1 Modélisation du système

Les relations de précédence entre les sous-tâches sont représentées par le graphe de précédence habituel qui porte sur les nœuds le coût en temps de calcul de la sous-tâche correspondante. Une sous-tâche est une unité séquentielle indépendente.

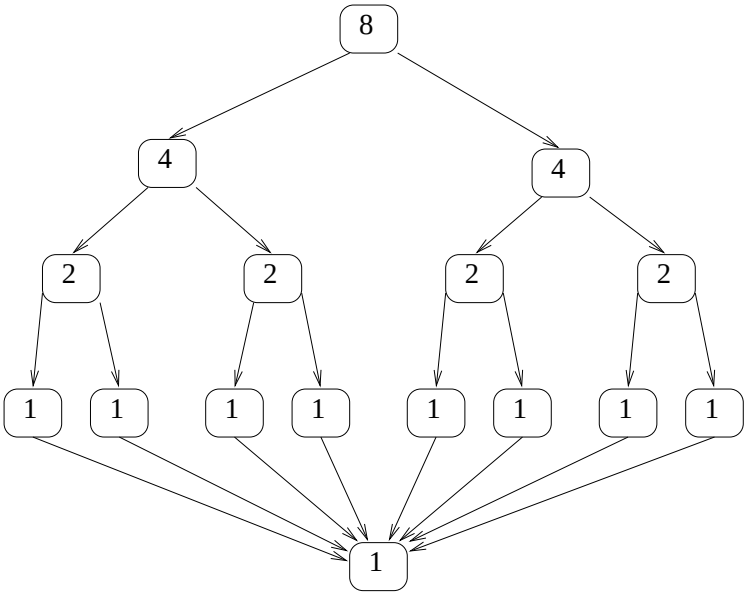

Fig. 1. Graphe de précédence d'un programme du type *Quick Sort*.

Dans ce qui suit nous présentons les hypothèses de travail: la composante matérielle est un ensemble de n processeurs identiques, le nombre de processeurs est constant le long de l'exécution et l'algorithme d'ordonnancement est du type *glouton*[1].

Certains résultats sont donnés pour la stratégie *processor sharing*[2].

Comme déjà cité, ce travail traite uniquement de l'influence de la structure logicielle sur les performances d'une application parallèle, on ne s'interressera donc pas aux influences de l'*overhead*[3] introduit par les communications et les synchronisations même si elles ne sont pas aussi négligeables. Pour ne pas donner des résultats basés sur des hypothèses aberrantes, les auteurs ont suivi une pratique commune dans ce genre de situation. En effet, les overheads introduits par les algorithmes d'ordonnancement et les protocoles de communication sont comptabilisés dans les temps de service. Les résultats donnés supposent que ces overheads sont fixes.

4.2 Le parallélisme moyen—définition

Le graphe de précédence est un moyen complet pour la représentation du parallélisme inhérent à une application. Une caractérisation plus simple et plus

[1] Une stratégie gloutonne ne laisse jamais un processeur libre tant qu'il y a des tâches prêtes.

[2] Dans cette déscipline, si k tâches sont prêtes et qu'il y ait n processeurs ($n < k$), chaque tâche reçoit un service à un taux $\frac{n}{k}$ fois celui qu'elle recevrait si un processeur lui était dédié.

[3] C'est la portion des ressources consommées par des mécanismes additionnels de contrôle et ne faisant pas partie des traitements utiles.

"compacte" du parallélisme serait préférable. Une telle caractérisation est celle donnée par Amdahl[4]: la fraction f de l'exécution qui est inhérement séquentielle.

La mesure proposée dans cette étude est le *parallélisme moyen*. Cette mesure peut être définie de quatre manières équivalentes:

1. le nombre moyen de processeurs actifs le long de l'exécution si on se donne un nombre infini de processeurs disponibles,
2. l'accélération si on se donne un nombre infini de processeurs disponibles,
3. le rapport du temps d'exécution total à celui d'un plus long chemin dans le graphe des tâches,
4. l'intersection entre les limites imposées par le matériel et celles imposées par le logiciel sur l'accélération (ceci est détaillé plus loin).

Il est possible de voir une limitation de l'accélération causée par le matériel et exprimée en fonction du nombre de processeurs n. Elle n'est atteinte que si tous les processeurs sont maintenus actifs le long de l'exécution.

Une autre limitation, simple à voir, est imposée par la structure du programme. En effet, pour n'importe quel nombre de processeurs, le temps d'exécution est au moins la longueur d'un plus long chemin dans le graphe des tâches, d'où l'accélération est au plus le rapport du temps d'exécution total à la longueur d'un plus long chemin. Dans Fig. 2, une illustration de ces limitations matérielles et logicielles est présentée.

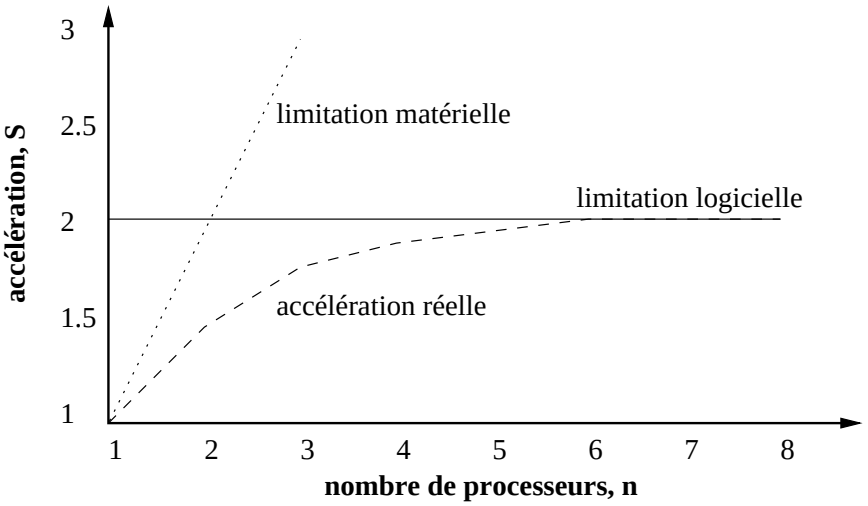

Fig. 2. Limitations sur l'accélération pour l'exemple en Fig. 1.

Pour le calcul de ce parallélisme moyen, on peut procéder analytiquement en utilisant le graphe des tâches ou pratiquement en tournant l'application sur

[4]Voir *Validity of the single processor approach to achieving large-scale computing capabilities* dans Proc. AFIPS, vol. 30, 1967.

un nombre suffisament grand de processeurs. Tout comme la mesure simple d'Amdahl, le plus intéressant n'est pas la manière de mesurer mais le pouvoir de caractériser succintement le parallélisme inhérent.

5 Des limites inférieures de l'accélération et de l'efficacité

Après avoir établi (grossièrement) des limites supérieures sur l'accélération, nous allons dans cette section proposer des limites inférieures.

5.1 Théorie

THÉORÈME 1. (des limites inférieures sur l'accélération et l'efficacité)

Soient A le parallélisme moyen d'une structure logicielle, $S(n)$ l'accélération avec n processeurs, et $E(n)$ l'efficacité avec n processeurs. Pour tout ordonnancement glouton,

$$S(n) \geq \frac{nA}{n+A-1},$$

et

$$E(n) \geq \frac{A}{n+A-1}.$$

Ces bornes sont atteintes.

PREUVE DU THÉORÈME 1[5].

Notons par T_∞ le temps d'exéxcution si un nombre illimité de processeurs est disponible. Il en suit (d'après la définition 1. du parallélisme moyen) que le temps d'exécution sur un seul processeur est en moyenne $T_\infty A$.

Supposons maintenant que l'exécution se fait sur n processeurs avec une discipline d'ordonnancement gloutonne. Par nature de cette discipline, on peut affirmer que le cumul des temps d'exécution sur tous les processeurs est $T_\infty A$. Le temps total est donc donné par,

$$\frac{T_\infty A + I(n)}{n}$$

où $I(n)$ est le temps mort sommé sur les n processeurs durant l'exécution.

[5]Cette preuve sera particulièrement détaillée pour son intérêt didactique.

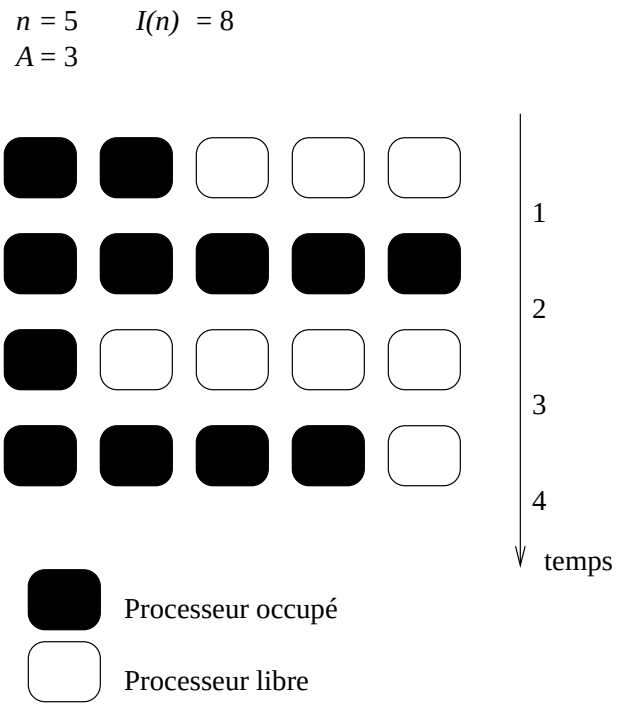

Fig. 3. Illustration du calcul de $I(n)$.

Ainsi, par la définition même de l'accélération, on trouve que $S(n) = \frac{nA}{\frac{A+I(n)}{T_\infty}}$. Pour établir notre résultat, il suffira donc de monter que $I(n) \leq T_\infty(n-1)$. En effet, comme la stratégie est gloutonne, il ne peut y avoir plus que $n-1$ processeurs libres à un instant donné de la période T_∞. Donc le cumul du temps pendant lequel les processeurs sont libres ($I(n)$) est au plus $T_\infty(n-1)$.

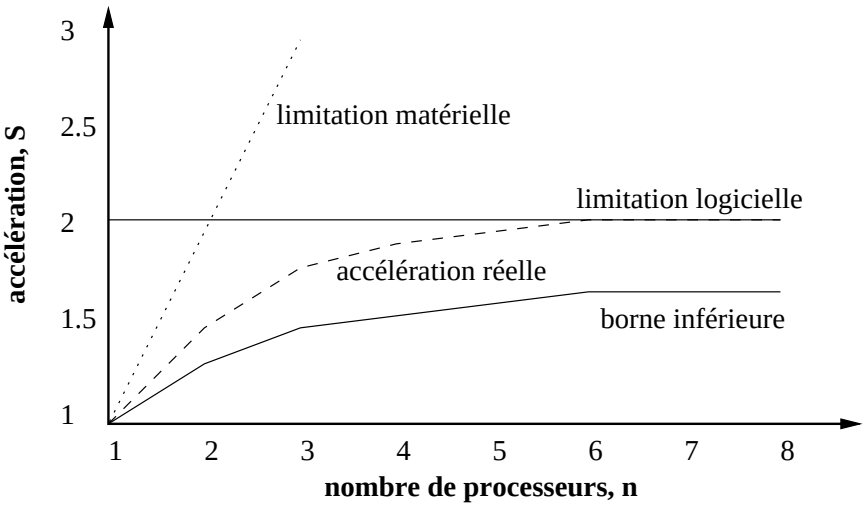

Fig. 4. La borne inférieure.

On prouve que la limite de l'accélération est atteinte par un exemple. Considérons une application qui est formée d'une sous-tâche suivie par $kn+1$ sous-tâches ($kn > A$). Le temps d'exécution de la première sous-tâche est $T_\infty \frac{kn+1-A}{kn}$, celui des autres est de $T_\infty(A-1)/kn$. L'accélération est donc donnée par,

$$S(n) = \frac{(kn+1)T_\infty \frac{A-1}{kn} + T_\infty \frac{kn+1-A}{kn}}{T_\infty \frac{kn+1-A}{kn} + (kn+1)T_\infty \frac{A-1}{kn}}.$$

Ce qui démontre notre théorème (moyennant une petite simplification).

Notons là que si $n \ll A$, alors $S(n) \to n$, et si $n \gg A$, alors $S(n) \to A$. Ce qui veut dire en quelque sorte que l'accélération est bornée par A.

Le théorème 1 est énoncé pour n'importe qu'elle discipline d'ordonnancement gloutonne aussi mal conçue qu'elle soit. Le théorème suivant donne un résultat plus raisonable pour des disciplines plus réalistes. On verra que ceci permettera de réduire la bande cadrant l'acélération mais exige, en plus de la supposition d'une discipline spécifique, plus d'informations telles le *degré de parallélisme maximal*[6].

THÉORÈME 2.

Soient A le degrè de parallélisme moyen, m_{max} le degrè de parallélisme maximal, $S(n)$ l'accélération avec n processeurs, et $E(n)$ l'efficacité avec n processeurs. L'ordonnancement utilisé est du type processor sharing

$$S(n) \geq min\left(A, \frac{nA}{n + A - 1 - \frac{(n-1)(A-1)}{m_{max}-1}}\right)$$

[6]Le nombre maximal de processeurs occupés si un nombre illimité est disponible.

et

$$E(n) \geq min\left(\frac{A}{n}, \frac{A}{n + A - 1 - \frac{(n-1)(A-1)}{m_{max}-1}}\right)$$

Ces bornes sont atteintes. La fonction *min* sert à éviter que l'accélération aille au delà de A si plus que m_{max} processeurs sont disponibles.

Notons là que si on ne peut pas avoir des garanties sur un degré maximal de parallélisme (il peut être arbitrairement élevé, i.e. $m_{max} = \infty$), se résultat se ramène à celui du théorème 1.

5.2 Application

5.2.1 Borne inférieure pour la pire situation

Au fur et à mesure que le nombre de processeurs augmente, toute augmentation de l'accélération est accompagée par une diminution de l'efficacité. Mais, on peut se demander s'il y a des situations où, à la fois, l'efficacité et l'accélération sont médiocres.

COROLLAIRE 1.1.

Pour toute discipline gloutonne, toute structure logicielle, et tout nombre de processeurs,

$$\frac{E(n) + S(n)}{A} > 1$$

Ainsi, si un système parallèle est utilisé à 20%, l'accélération qu'il atteint est d'au moins 80%.

5.2.2 Pénalité sur l'efficacité impliquée par un choix de l'accélération

COROLLAIRE 1.2.

Pour toute structure logicielle non séquentielle ($A > 1$) et tout ordonnancement glouton,

$$E(n) \geq \frac{A - S(n)}{A - 1}$$

Remarquons que pour les faibles valeurs de S, la pénalité en efficacité est minime, mais dès que S s'approche de la borne supérieure (A) l'efficacité tend à décroitre rapidement.

5.2.3 Importance du degré de parallélisme moyen

À quel point ce degré et le nombre de processeurs peuvent-ils réduire la marge de limitation de l'efficacité et de l'accélération ?. Qu'ajouterait la connaissance de m_{max} et de f ?

Nous allons tout d'abord éstimer la précision de la limitation proposée.

COROLLAIRE 1.3.

Pour tout ordonnancement glouton, l'éstimateur

$$\widehat{S}(n) = 2\frac{min\,(n, A)\,\frac{nA}{n+A-1}}{min\,(n, A) + \frac{nA}{n+A-1}}$$

a une erreur relative de 34% i.e. $\frac{\left|\widehat{S}(n)-S(n)\right|}{S(n)} < 0.34$.

PREUVE DU COROLLAIRE 1.3.

L'erreur de cet éstimateur est maximale aux bornes, d'où

$$S_{max}(n) = \frac{min\,(n, A) - \frac{nA}{n+A-1}}{min\,(n, A) + \frac{nA}{n+A-1}}$$

cette expression est maximale pour $n = A$ et $S_{max}(A) < \frac{1}{3}$.

On en conclut que l'ajout des informations supplémentaires au degré de parallélisme moyen n'améliore pas l'éstimation des bornes. Le degré de parallélisme maximal sert à réduire la borne inférieure de l'accélération.

5.2.4 Effet de la connaissance de *f*

COROLLAIRE 2.1.

Soient A le degré de parallélisme moyen, f la fraction séquentielle du programme, et $S(n)$ l'accélération avec n processeurs. L'ordonnancement est du type processor sharing et $n \geq 2$

$$S(n) \geq \frac{nA}{n + A - 1 - (1 - fA)}$$

Pour les grandes valeurs de n ou A, la connaissance de f n'apporte pas grand chose.

COROLLAIRE 2.2.

Soient A le degré de parallélisme moyen, f la fraction séquentielle du programme, et $S(n)$ l'accélération avec n processeurs. L'ordonnancement est du type processor sharing,

$$S(n) \leq min\left(\frac{n}{1 + f(n-1)}, A\right)$$

Nous remarquons là aussi que pour les faibles valeurs de f, l'amélioration est minime. Lorsque $f \to 1$, l'amélioration augmente et la borne supérieure s'approche de la borne inférieure donnée par le théorème 1. Pour f et A fixes, les meilleures améliorations sont obtenues pour n proche de A.

5.3 En résumé

Le degré de parallélisme moyen détermine significativement le compromis entre efficacité et accélération. La connaissance de ce paramètre et du nombre de processeurs permet de donner une éstimation de l'encadrement des grandeurs en question à 34% près. La connaissance d'autres paramètres du système (tels m_{max}, ou *f)* n'ajoute pas grand chose quant à la précision des bornes données. Il sont d'un intérêt seulement dans des conditions de parallĺismes très contraignantes.

6 Effet de l'ajout de processeurs

L'augmentation du nombre de processeurs a nécessairement des effets sur le coût (décroissement de l'efficacité) et le gain (augmentation de l'accélération).

Le théorème 3 étudie les bornes de variation de l'accélération et de l'efficacité pour une augmentation d'un facteur k du nombre de processeurs et l'exprime en fonction de A degré du parallélisme moyen.

THÉORÉME 3.

Avec la discipline de partage de processeurs l'effet de l'augmentation du nombre des processeurs de n à kn est comme suit:

$$max\left(1, \frac{kA}{(k-1)n+A}\right) \leq \frac{S(kn)}{S(n)} \leq min\left(1+(A-1)\frac{k-1}{kn-1}, k\right)$$

et

$$max\left(\frac{1}{k}, \frac{A}{(k-1)n+A}\right) \leq \frac{E(kn)}{E(n)} \leq min\left(\frac{1}{k}+(A-1)\frac{1-\frac{1}{k}}{kn-1}, 1\right)$$

Ces limites peuvent être atteintes.

Pour n, le théorème fournie les bornes de l'accélération avec k processeurs car le facteur avec lequel elle augmente est S elle même. Ces bornes sont donc consistantes avec celles vues auparavant.

Notons que lorsque $k \to \infty$ avec n et A constants, $S \to A$ la borne supérieur de l'accélération.

On considère maintenant quelques cas particuliers pour n et kn.

- pour n relativement petit devant A, le facteur avec lequel augmente l'accélération est linéaire avec le nombre de processeurs, par exemple si $n = \frac{A}{9}$ doubler n engendre au minimum une augmentation de l'accélération de 80%,
- d'autre part lorsque n augmente en dépassant A, les bornes de changement convergent vers la borne inférieur 1. Prenons par exemple un nombre de processeurs égale à A grand; doubler le nombre de processeurs de A à $2A$ implique une augmentation de l'accélération d'au plus 50%, doubler une autre fois n de $2A$ à $4A$ engendre au maximum un facteur de 25%. À la i^{ieme} étape le facteur est au maximum de $\frac{100}{2i}$%,

- comme il a été évoqué ulérieurement on est plus incertain lorsque le nombre de processeurs est proche de A. Par exemple pour $n = \frac{2}{3}A$ (A grand), doubler le nombre de processeurs peut donner une variation de l'accélération de 20 à 75%.

7 Optimisation de l'efficacité et du temps d'exécution

Il est important de déterminer le point optimal de l'efficacité et du temps d'exécution. Ce point indique le nombre optimal de processeurs. Cette étude est motivée par ces deux points de vues:

Sous un premier angle l'efficacité est vue comme le bénéfice à tirer des processeurs, plus l'efficacité est grande meilleur est le bénéfice, le coût a payer ici est le temps d'exécution.

En regardant le problème d'un autre angle on considère que le temps d'exécution est le bénéfice à atteindre, plus il est petit meilleur est le bénéfice.

Le profil temps d'exécution-efficacité est un graphe représentant pour différentes valeurs de n, nombre de processeurs, l'efficacité et le temps d'exécution obtenus.

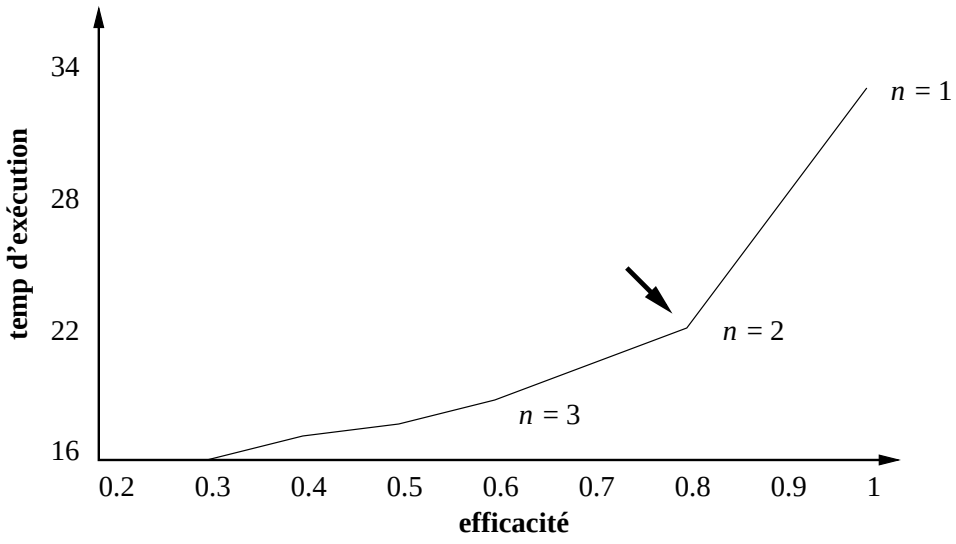

Fig. 4. Courbe temps d'exécution en fonction de l'efficacité pour l'exemple en Fig. 1.

L'optimum de ce profil est obtenu pour $\frac{E(n)}{T(n)}$ maximum, ce qui correspond aux deux points de vues cités ci-dessus.

7.1 Localisation du point optimal

Dans cette section on étudie l'existence et la localisation de ce point.

Il est permis d'utiliser des nombres non entiers. Les résultats comportant des nombres non entiers sont plus généraux et peuvent être ramener au cas entier. En effet un nombre non entier de processeurs peut être vu comme le résultat du partage d'un processeur entre deux jobs.

THÉORÉME 4.

Si p_m indique la proportion du temps où m processeurs sont occupés lorsque un nombre infini de processeurs est utilisé, m_{max} le degré maximum de parallélisme. Avec la discipline de partage de processeurs, le profil temps d'exécution-efficacité admet un unique point optimal donné par:

$$n = \frac{\sum_{m=\lfloor n \rfloor+1}^{m_{max}} p_m m}{\sum_{m=1}^{\lfloor n \rfloor} p_m}$$

n est borné par,

$$\frac{\sum_{m=n+1}^{m_{max}} p_m m}{\sum_{m=1}^{n} p_m} \leq n \leq \frac{\sum_{m=n}^{m_{max}} p_m m}{\sum_{m=1}^{n-1} p_m}$$

Etudions maintenant l'effet de l'utilisation d'un nombre de processeurs exactement égale à celui du point optimal.

D'abord nous définissons deux notions:

- utilisation du k^{ieme} dernier processeur: elle indique si n, le nombre de processeurs utilisés, est un surplus ou non. On adopte la discipline suivante d'affectation des processeurs: si on a s sous tâches éligibles avec $s < n$ on les affecte aux processeurs de 1 à s, même si cela implique la préemption d'une sous tâche s'exécutant sur un processeur numéroté supérieur à s et son affectation à un processeur inférieur à s. L'utilisation du k^{ieme} dernier processeur est définie comme étant la moyenne de l'utilisation du processeur numéro $n-k+1$. Notons qu'on peut ordonner les utilisations des processeurs en croissance avec leur numéros:

$\boldsymbol{p}$	1	$\cdots$	$n-k$	$n-k+1$	$\cdots$	n
$\boldsymbol{U}$	$U_1 <$	$\cdots$	$< U_{n-k} <$	$< U_{n-k+1} <$	$\cdots$	$< U_n$

- utilisation du k^{ieme} processeur additionnel: elle indique si n est suffisant ou pas. Elle est définie comme étant l'utilisation du $(k+n)^{ieme}$ processeur si on ajoute k processeurs. Notons aussi que cette quantité est inférieure à celles des processeurs $n+1$ à $n+k-1$.

THÉORÈME 5.

Si le nombre de processeurs disponibles est K optimum du profil temps d'exécution-efficacité:

- l'accélération atteinte est au moins 50% de l'accélération maximale,

- l'efficacité est au moins 50%,
- l'utilisation d'un processeur additionnel ne dépasse pas les 50%.

Ces bornes peuvent être atteintes.

Intuitivement le théorème 5 indique l'équilibre entre l'efficacité et l'accélération lorsque on opère au point optimal du profil temps d'exécution-efficacité.

7.2 Bornes du point optimal

La section précédente fournie le point optimal en utilisant une information complète sur le système, particulièrement p_m. Dans cette section on fournie ces bornes en utilisant simplement le degré de parallélisme moyen.

THÉORÉME 6. Avec la discipline de partage de processeurs et un nombre de processeurs K correspondant au point optimal du profil temps d'exécution-efficacité on satisfait:

$$\frac{A}{2} \leq K \leq 2A - 1$$

Les bornes peuvent être atteintes.

Il est possible de conclure à partir de ce théorème qu'on peut approcher K avec A. L'effet de l'utilisation de A processeurs est développé ci-dessous.

THÉORÉME 7. Avec la discipline de partage de processeurs et en utilisant A processeurs:

- l'accélération atteinte est au moins 50% de l'accélération maximale,
- l'efficacité est au moins 50%,
- l'utilisation du k^{ieme} processeur est au moins égale à $\frac{K}{K+A}$.
- l'utilisation du k^{ieme} processeur additionnel est inférieure à $\frac{A-1}{A+k-1}$.

On remarque que les bornes atteintes pour l'accélération et l'efficacité sont identiques à celles du théorème 5. Toutefois on n'a pas de garantie quant à la suffisance du nombre de processeurs utilisés.

8 Conclusion

La loi d'Amdahl est assez utile vue l'équilibre entre la simplicité et la précision qu'elle fournie. Le travail présenté dans ce rapport étudie les relations existantes entre l'accélération et l'efficacité en fonction du degré de parallélisme moyen. Les résultats obtenues sont en effet assez riches est assez simples. De tels résultats sont d'une utilité extrême pour la prédiction des performances d'un système parallèle.

En effet, on conclue que les systèmes parallèles sont "robustes" du fait qu'ils peuvent opérer à des performances intéressantes.