Partage de tâches entre processeurs homogènes

Philippe Nain

Partage de tâches entre processeurs homogenes

Philippe Nain[1]

INRIA, Domaine de Voluceau, B.P. 105, F-78150 Le Chesnay, France

Résumé. Une méthode de partage de tâches entre deux processeurs homogènes interconnectés est proposée. Nous supposons que l'un des deux processeurs est plus rapide que l'autre et nous donnons un algorithme s'appuyant sur une modélisation du système, permettant de diminuer les temps d'exécution des tâches du processeur le plus lent en les sous-traitant au processeur le plus rapide. L'algorithme proposé tient compte à la fois de la charge du processeur le plus rapide et des temps de transmission. L'efficacité de l'algorithme est montré à travers une étude théorique et une étude expérimentale mettant en œuvre un PDP 11/03 et un PDP 11/34.

Task Sheduling Between two Homogeneous Processors

Summary. A method for task scheduling between two homogeneous interconnected processors is proposed. We assume that one of the processors is faster than the other and we develop an algorithm based on a model of the system, which minimizes the execution delay of tasks arriving to the slow processor by taking into account the load of the fastest processor and the transmission delay. The performance of the algorithm is investigated experimentally using a PDP 11/03 connected to a PDP 11/34.

1. Introduction

Dans un système distribué, les programmes de gestion des ressources ont pour fonction d'allouer des tâches aux différents processeurs du système.

L'objectif est de rendre le meilleur service aux usagers, l'allocation des tâches aux processeurs doit donc être optimisée. On distingue deux classes d'algorithmes sur lesquels sont basés les programmes de gestion des ressources: les *algorithmes statiques* et les *algorithmes dynamiques.*

Un premier type d'algorithme statique est celui qui tient compte uniquement des instructions contenues dans la requête, sans se préoccuper de l'état

[1] Ce travail a été réalisé au laboratoire de Recherche en Université Paris-Sud, F-91405 Orsay, France

0001-5903/83/0018/0449/$03.60

du système ce qui, en particulier, peut avoir comme effet de transmettre une requête à un processeur en panne.

Un deuxième type d'algorithme statique est celui qui assigne des requêtes aux processeurs en tenant compte des ressources demandées, de l'architecture du réseau et du flux dans le réseau [2, 6–9]. Les algorithmes dynamiques tiennent compte des ressources demandées par les requêtes, de l'architecture du réseau et de la *charge* des processeurs [1, 3, 10].

La difficulté de caractériser l'évolution temporelle du réseau explique le peu d'études réalisées sur les algorithmes dynamiques.

Nous développons ici, un *algorithme d'assignation dynamique* de requêtes aux processeurs.

Dans cet article, l'architecture considérée est la suivante: deux processeurs *homogènes* (disposant des mêmes ressources) sont interconnectés à travers leur bus; l'un des deux (en l'occurrence un PDP 11/34) noté $P2$ est d'une puissance de traitement et d'une vitesse d'accès aux mémoires secondaires considérablement plus élevées que l'autre (Fig. 1).

Les requêtes[1] arrivent au processeur le moins rapide (en l'occurrence un PDP 11/03) noté $P1$, par le biais d'une console ($P1$ est monoutilisateur); en fonction de la charge de $P2$, qui travaille en multiprogrammation, $P1$ traite la requête soit localement soit la transmet pour exécution à $P2$ qui lui transmettra ensuite le résultat.

Ce transfert reste transparent pour l'usager de $P1$. L'objectif est, bien entendu, de minimiser en moyenne le temps d'exécution effectif des requêtes de $P1$.

Une étude théorique a été effectuée en liaison avec une étude expérimentale. La technique utilisée est de mesurer le facteur de charge du PDP 11/34. Ce facteur mesuré η est alors comparé à un seuil optimal θ calculé préalablement. Un message est envoyé en temps réel à $P1$ par $P2$, pour lui indiquer tout franchissement de θ par η, par valeur inférieure ou supérieure.

Le seuil θ calculé par un modèle théorique tient compte du rapport de puissance effective des deux processeurs et des temps que nécessitent les transferts d'information.

Dans une première étape, nous proposons un modèle du système ($P1$, $P2$). Nous définissons ensuite un *critère de décision* permettant à $P1$ de sous-traiter ou non ses requêtes à $P2$, dans deux contextes différents:

a) $P1$ et $P2$ ne traitent pas des requêtes en parallèle,

b) $P1$ et $P2$ traitent des requêtes en parallèle.

Ce critère est optimisé de façon à minimiser les temps moyen de service et de réponse des requêtes a) et le débit de traitement des requêtes b).

Pour deux modes de fonctionnement du processeur $P2$ («TIME-SHARING» et «FIFO») nous indiquons des algorithmes permettant de déterminer le facteur de charge instantané de $P2$.

Enfin, nous développons une étude expérimentale attestant de l'efficacité de l'algorithme et du modèle.

---

[1] Les tâches de $P1$ seront appelées des requêtes

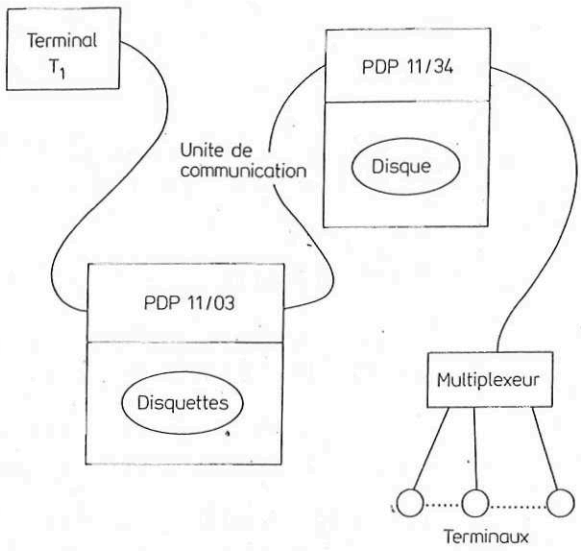


Fig. 1. Représentation schématique des PDP 11/03 et 11/34

2. Modélisation de (*P*1, *P*2)

Notations:

λ: taux d'arrivée des requêtes
S: v.a. des temps de service de $P1$, s sa moyenne
T: v.a. des temps de transmission aller et retour entre $P1$ et $P2$, t sa moyenne
η: v.a. du facteur de charge de $P2$, à valeurs dans $[1, \infty[$
F_η, F_S: fonctions de répartition de η et de S
f_η, f_S: densités de probabilité de η et de S

On suppose que S, T, η sont des v.a. deux à deux indépendantes. On définit la v.a. S' du temps passé par une requête dans $P2$ (temps de réponse de $P2$

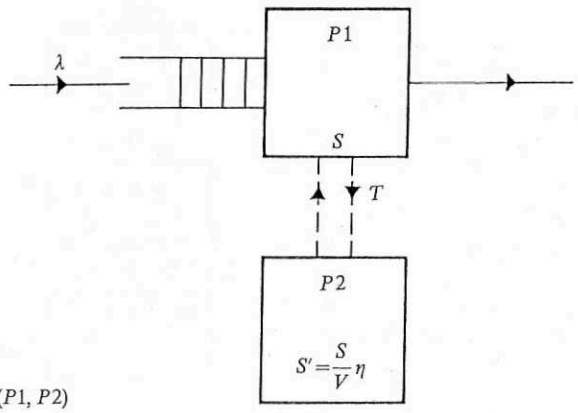


Fig. 2. Modèle de ($P1$, $P2$)

pour les *requêtes*) par:

$$S' = \frac{S\eta}{V}$$

où V est le rapport de puissance des deux processeurs. V constant et $V > 1$.
On suppose que η est connu à chaque instant par $P1$.

3. Définition du critère

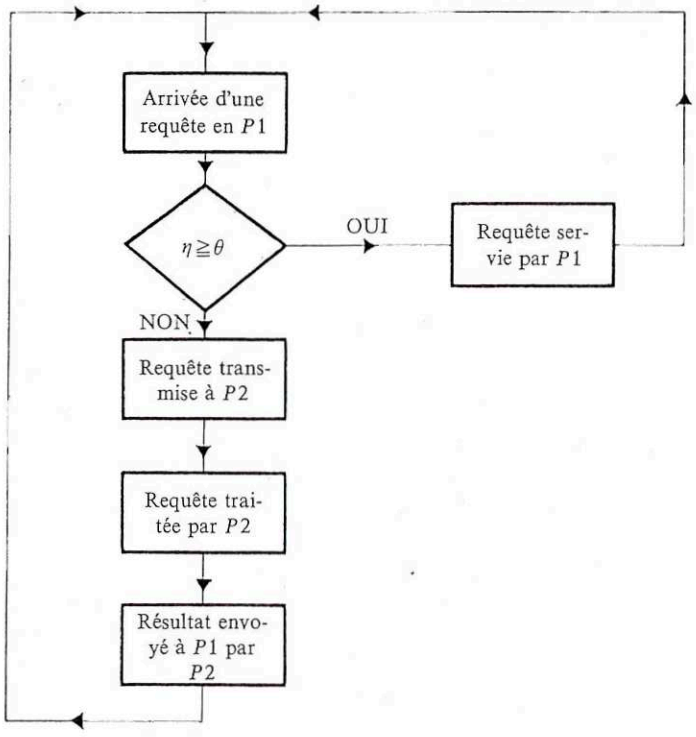
Le critère de décision de $P1$ en vue de la sous-traitance ou non de ses requêtes à $P2$, est défini de la façon suivante:

si $\eta \geqq \theta$ la requête est traitée par $P1$
si $\eta < \theta$ la requête est traitée par $P2$.

Ce critère est naturel d'après la définition de S', puisque si η est grand, le temps passé par la requête dans $P2$ est grand et il sera donc avantageux de faire traiter la requête par $P1$.

4. ($P1$, $P2$) ne travaillent pas en parallèle

Le schéma suivant montre les différentes situations possibles:



Notons $\hat{S}_\theta = \begin{cases} S & \text{si } \eta \geqq \theta, \\ S'+T & \text{si } \eta < \theta \end{cases}$ $F_{\hat{S}_\theta}$ sa fonction de répartition et $f_{\hat{S}_\theta}$ sa densité de probabilité.

4.1. Temps moyen minimum de service des requêtes

$$F_{\hat{S}_\theta}(x) = \int_1^\theta P(\hat{S}_\theta < x | \eta = y) f_\eta(y)\, dy + \int_\theta^\infty P(\hat{S}_\theta < x | \eta = y) f_\eta(y)\, dy$$

$$= F_S(x)(1 - F_\eta(\theta)) + \int_1^\theta P(S' + T < x | \eta = y) f_\eta(y)\, dy$$

d'où

$$F_{\hat{S}_\theta}(x) = F_S(x)(1 - F_\eta(\theta)) + \int_1^\theta P\left(\frac{S}{V} y + T < x\right) f_\eta(y)\, dy.$$

Par suite

$$f_{\hat{S}_\theta}(x) = f_S(x)(1 - F_\eta(\theta)) + \int_1^\theta \frac{d}{dx} P\left(\frac{S}{V} y + T < x\right) f_\eta(y)\, dy.$$

Si $E_\theta(\hat{S})$ et $E_\theta(\hat{S}^2)$ désignent les deux premiers moments de $\hat{S}_\theta$, il vient

$$E_\theta(\hat{S}) = \int_1^\theta \left(\frac{s\,y}{V} + t\right) f_\eta(y)\, dy + s(1 - F_\eta(\theta)), \tag{1}$$

$$E_\theta(\hat{S}^2) = \int_1^\theta \left(\frac{y^2}{V^2} E(S^2) + E(T^2) + \frac{2\,s\,t\,y}{V}\right) f_\eta(y)\, dy + E(S^2)(1 - F_\eta(\theta)) \tag{2}$$

en tenant compte de l'indépendance de S et T et où $E(S^2)$ (resp. $E(T^2)$) désigne le moment d'ordre deux de S (resp. T).

On peut facilement déterminer les extremums de $\theta \to E_\theta(\hat{S})$ qui doivent vérifier l'équation $\left(\frac{s\,\theta}{V} + t - s\right) f_\eta(\theta) = 0$. Par suite, il y a un seul extremum pour $\theta = \frac{V(s-t)}{s}$.

Si $\frac{V(s-t)}{s} \geqq 1$ alors $\theta_0 = \frac{V(s-t)}{s}$ minimise $E_\theta(\hat{S})$.

Si $\frac{V(s-t)}{s} < 1$ alors $E_\theta(\hat{S})$ est strictement croissante et $\theta_0 = 1$ la minimise.

θ	1	θ_0	$+\infty$
$\frac{d}{d\theta} E_\theta(\hat{S})$	−	0	+
$E_\theta(\hat{S})$	↘		↗

Fig. 3. Variations de $E_\theta(\hat{S})$

Le critère minimisant le temps moyen de service des requêtes est:

$$\text{si } s \geqq \frac{Vt}{V-1} \text{ alors } \begin{cases} \text{si } \eta \geqq \dfrac{V(s-t)}{s} \text{ la requête est traitée } P1 \\ \text{sinon la requête est traitée par } P2 \end{cases}$$

sinon la requête est traitée par $P1$.

Dans la suite nous supposerons $s \geqq \dfrac{Vt}{V-1}$ ce qui implique $s > t$. Nous supposerons en outre que $E(S^2) \geqq E(T^2)$.

4.2. Temps de réponse moyen minimum des requêtes

On suppose que les arrivées des requêtes sont poissonniennes de taux λ. On remplace le modèle $(P1, P2)$ défini précédemment par une file $M/G/1$ de taux des arrivées λ et de loi de service $\hat{S}_\theta$.

Utilisant la formule de Pollaczek-Khinchin puis celle de Little [4], nous avons le temps moyen T_θ passé dans le système par une requête:

$$T_\theta = E_\theta(\hat{S}) + \frac{\lambda E_\theta(\hat{S}^2)}{2(1-\lambda E_\theta(\hat{S}))}$$

où $E_\theta(\hat{S})$ et $E_\theta(\hat{S}^2)$ sont donnés par (1) et (2). On suppose $\lambda E_\theta(\hat{S}) < 1$ (condition d'ergodicité de la file $M/G/1$).

La complexité des expressions de $E_\theta(\hat{S})$ et $E_\theta(\hat{S}^2)$ ne permet pas de calculer analytiquement les extremums de T_θ. Nous allons montrer que T_θ possède un minimum et nous donnerons un encadrement de ce minimum, ce qui aura pour effet d'en faciliter le calcul numérique.

Cependant, il n'a pas été possible de montrer l'unicité de ce minimum, bien que l'existence de plusieurs minimums n'a aucun sens physique. Tous les calculs numériques effectués pour des lois données de S, T, η confirment l'unicité de ce minimum.

4.2.1. Extremum(s) de T_θ. T_θ possède un extremum en un point $\theta = \theta^*$ si et seulement si

$$\frac{d}{d\theta} T_{\theta_{|\theta=\theta^*}} = 0$$

où

$$\frac{d}{d\theta} T_\theta = \frac{\left(\dfrac{d}{d\theta} E_\theta(\hat{S})\right)(2(1-\lambda E_\theta(\hat{S}))^2 + \lambda^2 E_\theta(\hat{S}^2)) + \lambda \left(\dfrac{d}{d\theta} E_\theta(\hat{S}^2)\right)(1-\lambda E_\theta(\hat{S}))}{2(1-\lambda E_\theta(\hat{S}))^2}.$$

Etude sommaire de $\theta \to E_\theta(\hat{S}^2)$, $\forall \theta \geqq 1$

$$\frac{d}{d\theta} E_\theta(\hat{S}^2) = 0 \Leftrightarrow \frac{\theta^2}{V^2} E(S^2) + \frac{2st}{V} \theta + E(T^2) - E(S^2) = 0.$$

Soit $\Delta = \frac{s^2 t^2}{V^2} - \frac{E(S^2)}{V^2}(E(T^2) - E(S^2))$ le discriminant de ce trinome en θ.

On montre facilement que le tableau de variations de $E_\theta(\hat{S}^2)$ est le suivant:

θ	1		θ_1		$+\infty$
$\frac{d}{d\theta}E_\theta(\hat{S}^2)$		$-$	0	$+$	
$E_\theta(\hat{S}^2)$		↘		↗	

Fig. 4. Variations de $E_\theta(\hat{S}^2)$

où $\theta_1 = \max\left(1; \frac{-s\,t\,V + V^2\sqrt{\Delta}}{E(S^2)}\right)$.

Localisation des extremums de T_θ. Considérons le cas particulier où $\frac{d}{d\theta}E_\theta(\hat{S})$ et $\frac{d}{d\theta}E_\theta(\hat{S}^2)$ sont nuls simultanément. Ceci est équivalent à $\theta_0 = \theta_1$. Dans ce cas, T_θ a un extremum unique atteint en $\theta^* = \theta_1 = \theta_0$.

En effet, d'après les tableaux de variations de $E_\theta(\hat{S})$ et $E_\theta(\hat{S}^2)$ on a que

$$\frac{d}{d\theta}E_\theta(\hat{S}) \cdot \frac{d}{d\theta}E_\theta(\hat{S}^2) > 0, \quad \forall \theta \neq \theta_0 \ (=\theta_1),\ \theta \in [1, +\infty[.$$

Par suite il est clair, en tenant compte du fait que $1 - \lambda E_\theta(\hat{S}) > 0$ que

$$\frac{d}{d\theta}T_\theta \neq 0 \quad \forall \theta \neq \theta_0 \ (=\theta_1),\ \theta \in [1, +\infty[.$$

Supposons maintenant que $\theta_0 \neq \theta_1$.

Une condition nécessaire pour que $\frac{d}{d\theta}T_\theta = 0$ est que $\frac{d}{d\theta}E_\theta(\hat{S}) \cdot \frac{d}{d\theta}E_\theta(\hat{S}^2) < 0$.

Une étude simple montre que ceci est vérifié si et seulement si

$$\theta \in]\min(\theta_0, \theta_1); \max(\theta_0, \theta_1)[.$$

On peut résumer les deux cas $\theta_0 = \theta_1$, $\theta_0 \neq \theta_1$ par:

$$\text{Si } \frac{d}{d\theta}T_{\theta|\theta=\theta^*} = 0 \text{ alors } \theta^* \in [\min(\theta_0, \theta_1); \max(\theta_0, \theta_1)] \triangleq I_{\theta_0, \theta_1}.$$

Existence d'au moins un minimum de T_θ. Montrons que T_θ possède au moins un minimum dans l'intervalle I_{θ_0, θ_1}.

A cet effet, dressons un tableau de variations de l'application $\theta \to T_\theta$, $\theta \geqq 1$.

θ	1	$\min(\theta_0, \theta_1)$	$\max(\theta_0, \theta_1)$
$1-\lambda E_\theta(\hat{S})$	+	+	+
$\frac{d}{d\theta} E_\theta(\hat{S})$	−		+
$\frac{d}{d\theta} E_\theta(\hat{S}^2)$	−		+
$\frac{d}{d\theta} T_\theta$	−		+
T_θ	↘		↗

Fig. 5. Variations de T_θ

L'application $\theta \to T_\theta$ est continue sur $[1; +\infty[$. Par suite la Fig. 5 montre clairement que T_θ possède au moins un minimum dans l'intervalle I_{θ_0,θ_1}.

Cas particulier. Supposons que la loi de probabilité de la variable aléatoire S (resp. T) soit exponentielle de paramètre $\frac{1}{s}$ $\left(\text{resp. } \frac{1}{t}\right)$.

Nous avons:

$$\theta_0 - \theta_1 = \frac{V(s-t)}{s} - \frac{t\,V + V\sqrt{4s^2-3t^2}}{2s}$$

ou encore

$$\theta_0 - \theta_1 = V - \frac{\alpha}{2} V - V(1-\tfrac{3}{4}\alpha^2)^{1/2} \quad \text{où } \alpha = \frac{t}{s}.$$

Supposons que le temps moyen de transmission t soit très petit devant le temps moyen de service s.

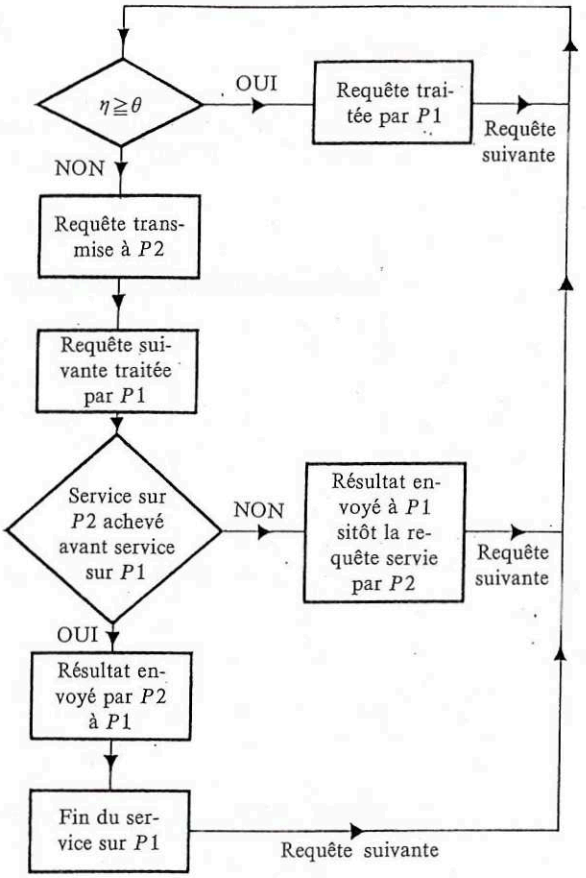
Utilisant un développement limité de $(1-\frac{3}{4}\alpha^2)^{1/2}$ il vient:

$$\theta_0 - \theta_1 = -\frac{\alpha}{2} V + o(\alpha^2)$$

ce qui montre que la longueur de I_{θ_0,θ_1} tend vers zéro lorsque $t \ll s$.

5. ($P1$, $P2$) travaillent en parallèle

Le dessin ci-dessous illustre les situations possibles:



Ce schéma suppose implicitement qu'il y a toujours au moins deux requêtes en attente de service sur $P1$.

Nous supposerons cette condition toujours réalisée, puisqu'on veut maximiser le débit de traitement des requêtes.

5.1. *Transmission de messages – Période d'activité*

On suppose que les échanges d'information entre les processeurs se font suivant le *mode interruption*. Nous devons donc distinguer le temps d'interruption d'un processeur pour émettre un message et le temps de transmission de ce message (émission + réception). Nous définissons ainsi:

- t_I temps moyen total d'interruption de $P1$ lors de l'émission *et* de la réception d'un message.
- t_T temps moyen total de transmission d'un message entre $P1$ *et* $P2$ et retour $(t_I < t_T)$.

A titre d'exemple dans la section 4, $t = t_T$.

Nous pouvons donc rencontrer *trois types de périodes d'activité* de $P1$ et de $P2$ suivant que

- seul $P1$ traite une requête (période 1).
- $P1$ et $P2$ traite chacun une requête mais $P1$ (resp. $P2$) termine le service de sa requête après $P2$ ($P1$) notée période 2 (période 3) (Fig. 6).

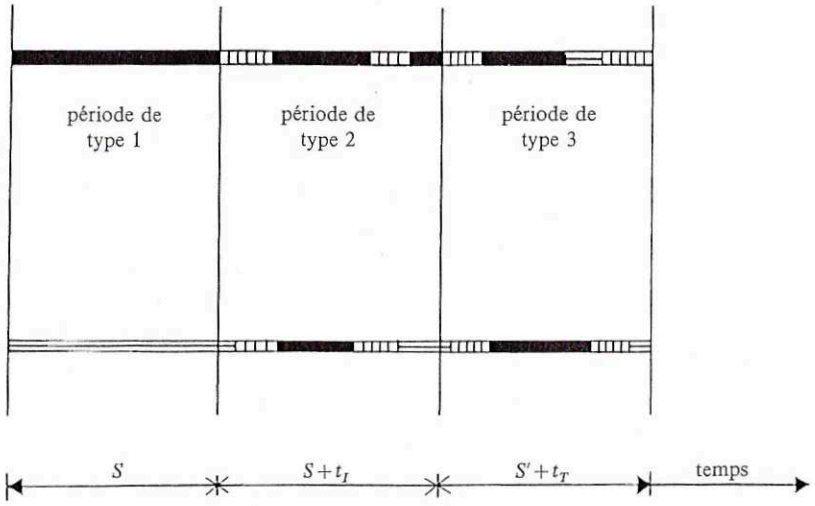


Fig. 6. Schéma des 3 périodes d'activité

5.2. *Débit de traitement maximum des requêtes*

On appellera *période* une période de type 1, 2 ou 3.

Durée moyenne d'une période.

$$\text{On note} \begin{cases} S'' \text{ la variable aléatoire de la durée d'une période} \\ E_\theta(S'') \text{ la durée moyenne d'une période} \end{cases}$$

$$E_\theta(S'') = s(1-F_\eta(\theta)) + \int_1^\theta E_\theta(S''/\eta = y)\, f_\eta(y)\, dy. \tag{5}$$

Lemme. *Soit* $y_0 = \dfrac{V}{s}(s+t_I-t_T)$.

$$Si\ 1 \leqq y_0 \leqq \theta, \quad E_\theta(S''/\eta = y) = \begin{cases} s + t_I & \forall y \in [1, y_0] \\ \dfrac{s\,y}{V} + t_T & \forall y \in [y_0, \theta]. \end{cases}$$

$$Si\ 1 \leqq \theta \leqq y_0, \quad E_\theta(S''/\eta = y) = s + t_I \qquad \forall y \in [1, \theta].$$

$$Si\ y_0 < 1\ alors\ E_\theta(S''/\eta = y) = \frac{s}{V} y + t_T \qquad \forall y \in [1, \theta]. \quad \square$$

Preuve. Soit $y \in [1, \theta]$. La période est forcément de type 2 ou 3 (critére (c)). La période sera de type 2 si

$$s + t_I \geqq \frac{s}{V} y + t_T$$

c'est à dire

$$y_0 \triangleq \frac{V}{s}(s + t_I - t_T) \geqq y. \qquad (*)$$

Par conséquent si $y_0 < 1$ l'inégalité ci-dessus n'est jamais vérifiée et la période est de type 3 d'où

$$E_\theta(S''/\eta = y) = \frac{s}{V} y + t_T.$$

Si $y_0 \geqq \theta \geqq 1$ alors la période est de type 2 d'après $(*)$ et $E_\theta(S''/\eta = y) = s + t_I$.

Si $\theta \geqq y_0 \geqq 1$ d'après $(*)$ on a que la période est de type 2, $\forall y \in [1, y_0]$ et donc de type 3, $\forall y \in [y_0, \theta]$. $\square$

Débit des requêtes.

On note: • D_θ = Débit des requêtes
• N_θ = Nombre moyen de clients servis par période.

Considérons n périodes successives:

Soient $N_1(n)$ (resp. $N_2(n)$) le nombre de périodes actives de $P1$ ($P2$) parmi ces n périodes.

Nous avons

$$N_\theta = \lim_{n \to \infty} \frac{N_1(n) + N_2(n)}{n} \qquad (n \in \mathbb{N}^*).$$

Mais par définition du système, $N_1(n) = n$ d'où

$$N_\theta = 1 + \lim_{n \to \infty} \frac{N_2(n)}{N_1(n)}.$$

D'autre part on a clairement

$$F_\eta(\theta) = \lim_{n \to \infty} \frac{N_2(n)}{N_1(n)}.$$

Par suite, le nombre moyen de requêtes servies par période est:

$$N_\theta = 1 + F_\eta(\theta).$$

Le débit du système vaut donc

$$D(\theta)=\frac{1+F_\eta(\theta)}{E_\theta(S'')}.$$

D'après (5) et le lemme, nous en déduisons que:

si $y_0 \geqq 1$

$$D(\theta)=\begin{cases}\dfrac{1+F_\eta(\theta)}{s+t_I F_\eta(\theta)} & 1\leqq\theta\leqq y_0\\[2ex] \dfrac{1+F_\eta(\theta)}{s+(s+t_I-t_T)\,F_\eta(y_0)+F_\eta(\theta)\,(t_T-s)+\dfrac{s}{V}\displaystyle\int_{y_0}^{\theta} y f_\eta(y)\,dy} & 1\leqq y_0\leqq\theta\end{cases}$$

si $y_0 \leqq 1$

$$D(\theta)=\frac{1+F_\eta(\theta)}{s+(t_T-s)\,F_\eta(\theta)+\dfrac{s}{V}\displaystyle\int_1^{\theta} y f_\eta(y)\,dy} \qquad \forall\,\theta\geqq 1.$$

5.3. Critère optimum

Supposons $s>\dfrac{Vt_T}{V-1}$ (sinon le critère est optimum pour le débit de traitement si $\theta=1$, cf. section 4).

La recherche des extremums de D_θ se fait en étudiant séparément les cas $y_0\leqq 1$ et $y_0\geqq 1$. Il est clair que dans les deux cas, l'application $\theta\to D(\theta)$ de $[1, +\infty[$ dans $\mathbb{R}^+$ est dérivable. Par suite une étude classique nous assure que dans chacun des cas cette application possède un et un seul maximum atteint en un point $\theta_{y_0}\in[1, +\infty[$ tel que:

si $y_0\geqq 1$, θ_{y_0} est l'unique solution de l'équation $h(\theta)=0$, $\forall\,\theta\geqq y_0$

si $y_0\leqq 1$, θ_{y_0} est l'unique solution de l'équation $k(\theta)=0$, $\forall\,\theta\geqq 1$

où

$$h(x)\triangleq F_\eta(y_0)\frac{s}{V}y_0+2s-t_T+\frac{s}{V}\left(\int_{y_0}^{x} y f_\eta(y)\,dy-(1+F_\eta(x))\,x\right) \qquad x\geqq y_0$$

$$k(x)\triangleq 2s-t_T+\frac{s}{V}\left[\int_1^{x} y f_\eta(y)\,dy-(1+F_\eta(x))\,x\right] \qquad \forall\,x\geqq 1.$$

Le critère optimisant le débit de traitement des requêtes est:

si $\eta\geqq\theta_{y_0}$ alors $P2$ non disponible pour traiter une requête

si $\eta<\theta_{y_0}$ $P2$ disponible (*P1 et P2 travaillent en parallèle*).

6. Le facteur de charge de $P2$

Nous allons distinguer deux modes de fonctionnement du processeur $P2$:

- les clients sont servis suivant la discipline «FIFO»
- les clients sont servis suivant la discipline «TIME-SHARING».

Notons que dans un processeur, il est en général possible de connaître à chaque instant le nombre de tâches en attente ou en cours de traitement. Il est par contre, difficile d'en connaître son temps d'occupation (temps nécessaire pour qu'il devienne inoccupé sachant qu'on empêche son accès aux nouveaux clients).

Dans le cas «FIFO», $S'=\frac{S}{V}+T0$ où $T0$ est le temps d'occupation de $P2$ à l'arrivée de la requête. Nous avons donc $\frac{S}{V}\eta=\frac{S}{V}+T0$ et par suite $\eta=(T0)\times\frac{V}{S}+1$.

Il suffit donc «d'estimer $T0$» en connaissant le nombre de clients présents dans $P2$ à l'arrivée de la requête. Une méthode d'estimation est d'observer statistiquement les temps de service que nécessitent j clients en $P2$, $\forall j=1, 2, \ldots$, et d'en déduire la fonction de répartition empirique de $T0$.

Le cas «TIME-SHARING» est plus complexe. En effet, le facteur de charge de $P2$ n'est pas uniquement fonction de $(T0, S, V)$.

Supposons par exemple qu'il y ait un client en $P2$ demandant 200 secondes de service à l'instant où une requête nécessitant 1 seconde ($\triangleq$1 quantum) arrive en $P2$, transmise par $P1$.

Si on suppose que $P2$ travaille suivant le schéma «TIME-SHARING» [5], le temps de réponse S' de la requête sera de 2 secondes soit $\eta=\frac{V}{S}\times(2\text{ secondes})$. Si maintenant à l'instant d'arrivée de cette requête en $P2$, il y a 100 tâches nécessitant chacune 1 seconde de service, alors S' sera de 101 secondes soit $\eta=\frac{V}{S}\times(101\text{ secondes})$. (On suppose qu'aucun client n'arrive en $P2$ tant que la requête n'est pas servie.) En prenant $T=0$ et $S=10$ secondes (donc $V=10^2$) or ($V=10^{(2)}$) nous avons, dans le premier cas $\eta=2$ et dans le deuxième cas $\eta=101$.

Cependant, dans ces deux cas, le temps d'occupation était respectivement $T0=200$ secondes et $T0=100$ secondes. Ceci montre que le facteur de charge η est une fonction de $(NC, T0, S, V)$ où NC est le nombre de clients dans $P2$ à l'arrivée de la requête.

On trouvera dans [3] un algorithme déterminant une estimation du temps de réponse d'un nouveau client dans un processeur travaillant suivant le schéma «TIME-SHARING».

Si on note TR ce temps de réponse estimé, nous avons alors

$$\eta=(TR)\times\frac{V}{S}.$$

---

[2] V est défini comme le rapport des temps d'exécution de la requête sur $P1$ et sur $P2$

7. Étude expérimentale

Nous utilisons dans cette étude expérimentale, un micro-ordinateur PDP 11/03 ($P1$) et un mini-ordinateur PDP 11/34 ($P2$) possédant les mêmes ressources et interconnectés (cf. Fig. 1).

$P2$ est plus puissant que $P1$, c'est-à-dire qu'une même tâche exécutée sur $P1$ et sur $P2$, dans le cas ou $P1$ et $P2$ sont inoccupés, aura un temps d'exécution plus faible sur $P2$ que sur $P1$.

La vitesse de transmission entre ces deux processeurs est de 9600 Bauds. Deux logiciels, jouant le rôle de moniteurs, ont été implantés sur $P1$ et sur $P2$. Ils ont pour fonctions de simuler les usagers de $P1$ et de $P2$ et de réaliser effectivement les transferts d'information et les exécutions des tâches. Par conséquent, cette expérience allie à la fois les techniques de *simulation* et de *temps réel.*

Les tâches demandées par les usagers de $P1$ et de $P2$ sont des *accès aux données,* c'est-à-dire essentiellement des accès disque. Tous les transferts d'information ont lieu en *mode interruption.*

Description de l'expérience et hypothèses

De façon pratique, au début de l'expérience, on active les deux programmes moniteurs sur $P1$ et $P2$ simulant des tâches demandées par les usagers.

L'occurrence d'activation des tâches sur $P2$ (c'est-à-dire *tâches non transmises par* $P1$) est un paramètre du logiciel de $P2$ sur lequel on peut intervenir pour varier la charge de $P2$.

Nous faisons les hypothèses expérimentales suivantes:

H_0: il y a toujours une requête en attente d'exécution sur $P1$. Cela signifie que le programme moniteur de $P1$ active une requête sitôt $P1$ et $P2$ libres de requêtes

H_1: la discipline de service sur $P2$ est «FIFO»

H_2: $P1$ et $P2$ ne travaillent pas en parallèle.

Nous définissons une *tâche élémentaire* comme une lecture d'un bloc de 256 mots sur le disque de $P1$ ou de $P2$.

H_3: les tâches de $P1$ ou de $P2$ sont des multiples entiers d'une tâche élémentaire, *de moyenne 5 tâches élémentaires*

H_4: le facteur de charge η_E de $P2$ à l'instant t est le nombre de tâches élémentaires sur $P2$ en attente de traitement.

Le protocole suivi pour la transmission du facteur de charge η_E de $P2$ à $P1$ est le suivant:

– $P2$ transmet un message à $P1$ *(codé sur 4 octets)* dès que le facteur de charge η_E franchit par valeur inférieure ou supérieure le seuil θ fixé au préalable. Ainsi l'information utile à $P1$ pour décider de sous-traiter ou non ses requêtes à $P2$ (cf. le critère), est connue à chaque instant par $P1$, à

l'exception des périodes critiques où $P2$ envoie un message sur l'état de sa charge à $P1$. La durée de ces périodes critiques est cependant infime étant donné la petite longueur de ce message et le débit de la ligne liant $P1$ à $P2$.

Les hypothèses H_0 et H_2 nous indiquent que nous sommes ramenés à la section 3.

Mesures expérimentales

Comme nous l'avons vu, le moniteur de $P2$ génère l'arrivée des usagers de $P2$; il contient donc une macro (ARRIV[p]; $p=0,\ldots,9$) qui, en fonction de la valeur de son paramètre d'entrée p, génère des tâches sur $P2$ plus ou moins souvent.

Nous avons supposé que l'occurrence d'arrivée des tâches de $P2$ est inversement proportionnelle à la grandeur de p.

Exemple. Avec ARRIV [3] la durée moyenne entre deux activations successives de tâches est plus courte qu'avec ARRIV [5].

D'autre part, si $p=0$, le temps d'occupation de $P2$ à *chaque instant* est *toujours* plus grand que le temps d'exécution d'une requête sur $P1{:}P2$ est dit saturé pour les requêtes.

De même, si $p=9$, le moniteur de $P2$ n'active pas de tâches: $P2$ est inoccupé et le débit de traitement des «usagers» de $P1$ est maximum. ($P1$ sous-traite alors tous ses clients à $P2$) (cf. Fig. 7).

Soit p *fixé*$\in]0,\ldots,9[$. On se donne donc un processus aléatoire de la charge de $P2$.

Les mesures expérimentales sont organisées comme suit: pour p fixé, des expériences successives sont effectuées en modifiant pour chacune d'elles la valeur θ du seuil de charge de $P2$.

Exemple. Supposons le seuil fixé à n ($\theta=n$, $n\in\mathbb{N}$).

D'après l'algorithme développé, si la charge de $P2$ (cf. $H4$) est supérieure ou égale à n tâches élémentaires en attente d'exécution sur $P2$, une requête arrivant à $P1$ sera servie par $P1$; sinon elle sera transmise à $P2$ par le biais de $P1$, exécutée par $P2$ et le résultat envoyé par $P2$ à $P1$.

Chaque expérience (p et θ fixés) s'achève lorsque 300 usagers de $P1$ ont été servis (par $P1$ ou $P2$). Ces usagers demandent donc en moyenne 1500 lectures de blocs de 256 mots.

L'horloge interne du PDP 11/03 (1 unité de temps = 1/50 seconde) est utilisée pour mesurer la durée de l'expérience - (si $p=0$ la durée est en moyenne de 55s et si $p=9$ de 75s).

Le débit de traitement des requêtes est donc obtenu en divisant 300 par la durée de l'expérience.

La Fig. 7 montre le débit de traitement des requêtes en fonction de la valeur du seuil de charge θ et pour 4 processus des arrivées des tâches de $P2$ ($p=0, 3, 5, 9$).

Le débit optimal est obtenu, si $p=3$ et $p=5$, pour $\theta=8$.

Lien avec le modèle théorique

Nous sommes dans le cas du modèle de la section 3. Nous définissons naturellement V comme

$$V=\frac{s_1}{s_2} \quad \text{où } s_1 \text{ (resp. } s_2)$$

est la durée moyenne d'exécution d'une tâche élémentaire sur $P1$ ($P2$) si $P1$ et $P2$ sont inoccupés.

s_1 et s_2 mesurés expérimentalement sur un échantillon de 200 mesures nous donnent V ($V=2.59\pm0.1$).

D'après la section 3, le seuil optimal minimisant le temps moyen de service des requêtes est

$$\theta_0=\frac{V(s-t)}{s}.$$

Les hypothèses H_0 et H_2 nous montrent que le débit D_θ des requêtes est

$$D_\theta=\frac{1}{E_\theta(\hat{S})}$$

où $E_\theta(\hat{S})$ est donné par (1).

Par suite le θ maximisant le débit des requêtes est $\theta=\theta_0$. Nous avons l'équivalence avec les définitions expérimentales de s et de η en faisant les transformations suivantes:

$$s\leftarrow 5\cdot s_1 \qquad \text{(hypothèse } H_3\text{)},$$

$$\eta\leftarrow 1+\frac{\eta_E}{5}\in[1;+\infty[\qquad \text{(hypothèse } H_1\text{)}.$$

En effet, d'après H_1 et la section 6, nous avons

$$\eta=T0\cdot\frac{V}{S}+1$$

mais

$$T0=\eta_E\cdot s_2, \quad V=\frac{s_1}{s_2} \quad \text{et } s=5s_1.$$

Par suite $\eta=1+\frac{\eta_E}{5}$.

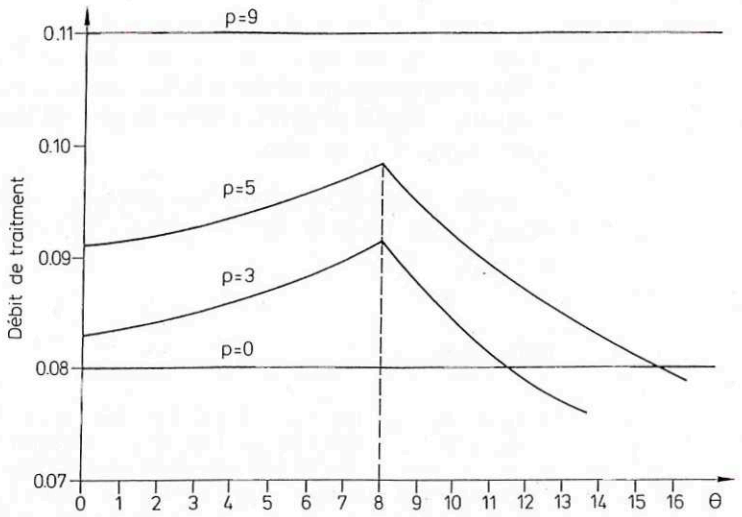
D'où

$$\theta_0=\frac{V}{s_1}(5s_1-t)-5.$$

D'après les valeurs mesurées de s_1 et t où t est établi sur un échantillon de 200 mesures, nous avons

$$\theta_0=7.95\pm0.1$$

qui est le seuil trouvé *expérimentalement.*



- θ = valeur du seuil de charge de $P2$
- Le débit des requêtes est mesuré en nombre de requêtes/(1/50 seconde)

Fig. 7. Débit de traitement des requêtes

8. Conclusion

Nous avons proposé un nouvel algorithme d'allocation de tâches *dynamique* entre deux processeurs interconnectés. Cet algorithme repose sur une modélisation du système ($P1, P2$) et a été optimisé de façon à minimiser les temps de service et de réponse des requêtes de $P1$, si ($P1, P2$) ne travaillent pas en parallèle, et à maximiser le débit de traitement des requêtes, si ($P1, P2$) travaillent en parallèle. Une étude expérimentale a permis de valider le modèle dans le cas de non parallélisme et a montré l'efficacité de l'algorithme pour le débit de traitement des requêtes de $P1$.

Une modification souhaitable du modèle proposé, serait de tenir compte, lors de la détermination du seuil optimal du critère de décision de $P1$, de la modification du facteur de charge de $P2$ pouvant survenir pendant la transmission d'une requête de $P1$ à $P2$.

L'étude expérimentale montre en effet des variations importantes de la charge de $P2$, pendant la transmission de *longs* messages entre $P1$ et $P2$. Par suite, une requête peut arriver en $P2$ alors que $n > \theta$. (Si cela se produisait dans l'expérience, nous abandonnions la requête.)

Le contexte bureautique, où un usager accède à un micro-ordinateur ($P1$), lui-même relié à un ordinateur central ($P2$) (ex: banques, ...) se prête très bien à l'implémentation de cet algorithme, pour trois raisons:

– les requêtes étant des accès à une base de données, on peut ainsi s'attendre à un fonctionnement homogène de $P2$ et par suite, à une bonne précision des estimations de son facteur de charge.

– les messages entre $P1$ et $P2$ (dans le sens $P1 \rightarrow P2$) sont courts. Ceci est dû au codage des messages d'accès à une base de données.
– les processeurs ne travaillent pas en parallèle. En effet, un usager interrogeant une base de données, attend en général la réponse à sa question avant de poser la question suivante.

Ainsi, dans cette application particulière, nous nous affranchissons des situations les moins favorables relevées dans cette étude, pour l'implémentation de cet algorithme.

Références

1. Agrawala, A.K., Tripathi, S., Ricart, G.: Adaptive routing using a virtual waiting time technique. Technical report TR-817, University of Maryland, Computer Science Department, 1979
2. Balachandran, V.: An integer generalized transportation model for optimal job assignment in computer networks. Operations Res. **24**, 4, 742–759 (1976)
3. Bryant, R.M., Finkel, R.A.: A stable distributed scheduling algorithm. Proceedings of the 2nd International Conference on Distributed Computing Systems, Paris, 314–323 (1981)
4. Kleinrock, L.: Queueing systems, vol. I, John Wiley and Sons, 1975
5. Kleinrock, L.: Queueing systems, vol. II, John Wiley and Sons, 1976
6. Morgan, H.L., Levin, K.D.: Optimal program and data locations in computer networks. Comm. ACM **20**, 5, 315–321 (1977)
7. Stone, H.S.: Multiprocessor scheduling with the aid of network flow analysis. IEEE Trans. Software Engrg. **SE-3**, 5, 315–321 (1977)
8. Stone, H.S.: Control of distributed processes. IEEE Computer 97–106 (1978)
9. Van Tilborg, A.M., Wittie, L.D.: Wave scheduling: distributed allocation of task forces in network computers. Proceedings of the 2nd Conference on Distributed Computing Systems, Paris, 337–347 (1981)
10. Yuon-Chieh Cho, Kohler, K.H.: Models for dynamic load balancing in a heterogeneous multiple processor system. IEEE Trans. Computers **C-28**, 5, 354–361 (1979)

Received July 9, 1981/August 10, 1982