RAIRO. RECHERCHE OPÉRATIONNELLE

JEAN-MARIE AUGER
SÉBASTIEN BERTEAUX

Implantation de la méthode des formes produits matricielles pour l'exploitation du n, p Fork-Join

RAIRO. Recherche opérationnelle, tome 23, n° 1 (1989), p. 17-42

<http://www.numdam.org/item?id=RO_1989__23_1_17_0>

Recherche opérationnelle/Operations Research
(vol. 23, n° 1, 1989, p. 17 à 42)

IMPLANTATION DE LA MÉTHODE DES FORMES PRODUITS MATRICIELLES POUR L'EXPLOITATION DU *n, p* FORK-JOIN (*)

par Jean-Marie AUGER (1) et Sébastien BERTEAUX (1)

Résumé. — *Cet article présente, tout d'abord, une modélisation du n, p Fork-Join (file multi-serveurs avec des arrivées groupées) à l'aide d'un réseau de Petri stochastique coloré. Celui-ci a une place non bornée. L'implantation et l'expérimentation d'une méthode d'exploitation d'un tel réseau est présentée dans la suite de l'article.*

Mots clés : Modélisation; Fork-Join; réseaux de Petri stochastiques; Markov.

Abstract. — *In the first section of this paper, we present a n, p Fork-Join (multiservor queue with bulk arrivals) modelisation with a coloured stochastic Petri net. This net has one unbounded place. The implantation and experimentation of a method of exploiting such a net is presented in the second section.*

Keywords : Modelisation; Fork-Join; stochastic Petri nets; Markov.

I. INTRODUCTION

L'étude des réseaux de Petri stochastiques non bornés peut apporter de nouveaux résultats concernant l'évaluation des performances, car ils généralisent la notion de réseaux ouverts de files d'attente.

En [FLO1], G. Florin et S. Natkin ont défini une classe de réseaux de Petri stochastiques markoviens appelés « files complexes ». Ces réseaux ont une seule place non bornée et représentent une file d'attente avec des arrivées et des départs groupés qui dépendent d'un processus de Markov fini.

(*) Reçu juillet 1988.
(1) I.I.E., Les Passages, 18, allée Jean-Rostand, B.P. n° 77, 91002 Évry Cedex.
Ce travail a été réalisé à l'I.N.T., Les Épinettes, 9, rue Charles-Fourrier, 91011 Évry Cedex.

Recherche opérationnelle/Operations Research, 0399-0559/89/01 17 26/$ 4.60

Notre travail a consisté à développer un logiciel qui calcule les probabilités en régime stationnaire des différents états d'une telle file. Ceci nous a permis d'étudier le modèle du n, p Fork-Join. Celui-ci correspond à une file multi-serveurs comportant p serveurs en parallèle, exponentiels, avec le même taux de service, et des arrivées groupées par groupe de n et priorité premier arrivé, premier servi.

La première partie de cet article concerne la construction d'un réseau de Petri stochastique markovien à une place non bornée modélisant le n, p Fork-Join. Nous rappelons dans la deuxième partie les principaux résultats théoriques concernant les « files complexes ». Nous nous intéressons dans la troisième partie à l'obtention sur le réseau de Petri des objets permettant la mise en œuvre de la méthode des formes produits matricielles établie par V. Wallace et M. F. Neuts (*cf.* [WAL] et [NEU]). Ces objets correspondent, en fait, aux différentes sous-matrices constituant le générateur du processus markovien associé au graphe des marquages. Pour écrire les algorithmes liés à ces traitements, nous avons principalement utilisé les résultats établis par G. Florin et S. Natkin (*cf.* [FLO1]). Il est à noter que, lors de cette troisième partie, nous avons écrit un algorithme de réduction des états instantanés et impossibles d'un graphe des marquages. L'ensemble de ces algorithmes a été implanté. La quatrième partie concerne l'implantation d'un programme mettant en œuvre la méthode des formes produits matricielles. Cette méthode permet de calculer l'ensemble des probabilités liées aux états du graphe des marquages par une formule de récurrence. Dans cette quatrième partie, nous rapportons également des résultats concernant l'expérimentation de ces différents programmes, notamment sur le modèle du n, p Fork-Join. Nous avons alors constaté que la taille des modèles exploitables était limitée. Un prolongement possible de nos travaux serait donc d'augmenter la taille des modèles résolvables automatiquement.

II. CONSTRUCTION D'UN RÉSEAU DE PÉTRI STOCHASTIQUE MODÉLISANT LE n, p FORK-JOIN

II.1. Présentation du n, p Fork-Join

Le n, p Fork-Join correspond à une file multi-serveurs comportant p serveurs en parallèle, exponentiels, avec le même taux de service μ et des arrivées groupées par groupe de n et priorité premier arrivé premier servi.

Dans le Fork-Join classique, il y a p tâches avec des arrivées groupées et chacune d'elles est affectée à un des p processeurs. Ici, il y a n tâches et elles

peuvent être servies par n'importe lequel des serveurs, quand il est libre. (Même si $n=p$, s'il se trouve que l'un des p serveurs est moins chargé qu'un autre, il est possible que deux tâches du même travail soient servies par le même serveur, l'une après l'autre.)

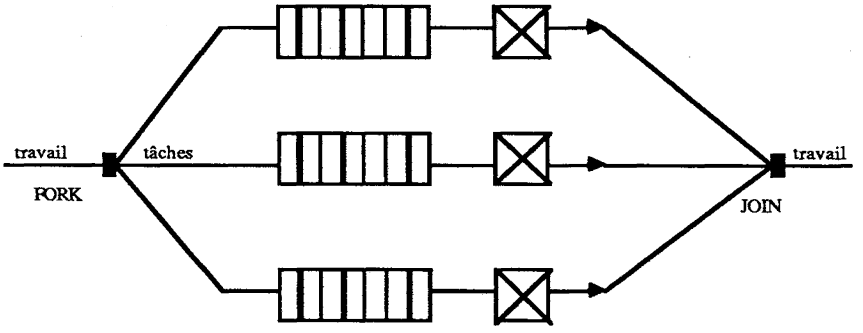


Figure 1

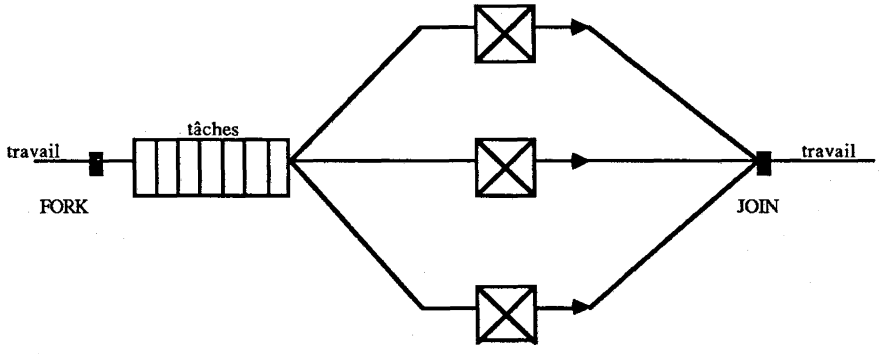


Figure 2

La figure 1 représente le problème classique du Fork-Join, la figure 2 représente le cas que nous traitons ici.

Les travaux se succèdent. Par conséquent, dans la file Fork on peut trouver des tâches correspondant à des travaux différents. Pour ce qui est des tâches correspondant à des travaux différents, la priorité est premier arrivé premier servi. (Remarque : cette loi de priorité est couramment appelée FIFO, ceci serait en fait incorrect dans le cas présent. En effet, il y a une différence entre premier arrivé premier servi et premier arrivé premier sorti, lorsqu'il y a plusieurs serveurs.)

Les architectures faiblement couplées sont des exemples type d'un tel mode de fonctionnement. Nous retiendrons notamment les études déjà effectuées pour la modélisation :

– de LCAP, un réseau de supercalculateurs FPS reliés en étoile à un ordinateur IBM central (à Kingston, U.S.A.).

– d'une architecture parallèle pour le traitement d'images (*cf.* [HOU]).

Des travaux antérieurs ont déjà été réalisés sur ce modèle permettant le calcul exact du temps de réponse d'un groupe de tâches (*cf.* [HOU]). Pour obtenir des résultats plus complets sur l'état du système, différents types d'agrégation ont été utilisés. Pour notre part, nous avons mis en œuvre une méthode exacte de calcul de la distribution complète des probabilités liées aux états du système à partir d'une modélisation à l'aide d'un réseau de Petri stochastique.

II. 2. Construction d'un modèle

Dans le cas présent, nous utilisons un réseau coloré qui nous permet de distinguer les différents travaux en cours. Ce réseau est décrit sur la figure 3.

Lorsqu'un travail arrive dans le système, il se divise en n tâches qui sont matérialisées par n jetons supplémentaires dans la place FORK. Le franchissement de la transition S-DEBUT correspond à l'attribution d'un processeur à une tâche. L'utilisation maximale de p processeurs est garantie par la place CPT-PROC. Le franchissement de la transition S-FIN correspond à la libération d'un processeur par une tâche et celui de la transition SORTIE à la fin d'un travail. Pour que le modèle représente effectivement le fonctionnement du n, p Fork-Join, nous sommes obligés de distinguer les travaux entre eux quand ils sont traités simultanément. Pour cela, nous colorions les jetons au passage de la transition S-DEBUT. La politique de service étant premier arrivé premier servi, une nouvelle couleur est introduite quand n jetons ont franchi la transition S-DEBUT. Les n tâches d'un même travail sont donc représentées par n jetons de la même couleur. Il est donc facile de repérer la fin d'un travail (n jetons de la même couleur dans la place JOIN). Le nombre maximal de travaux traités simultanément étant p, p couleurs suffisent pour distinguer ces travaux. En effet, lorsqu'un travail est terminé, sa couleur peut être réutilisée. Il est à noter que la transition S-FIN a un taux dépendant du marquage de la place PROCESSEURS, et plus précisément du nombre de jetons de même couleur.

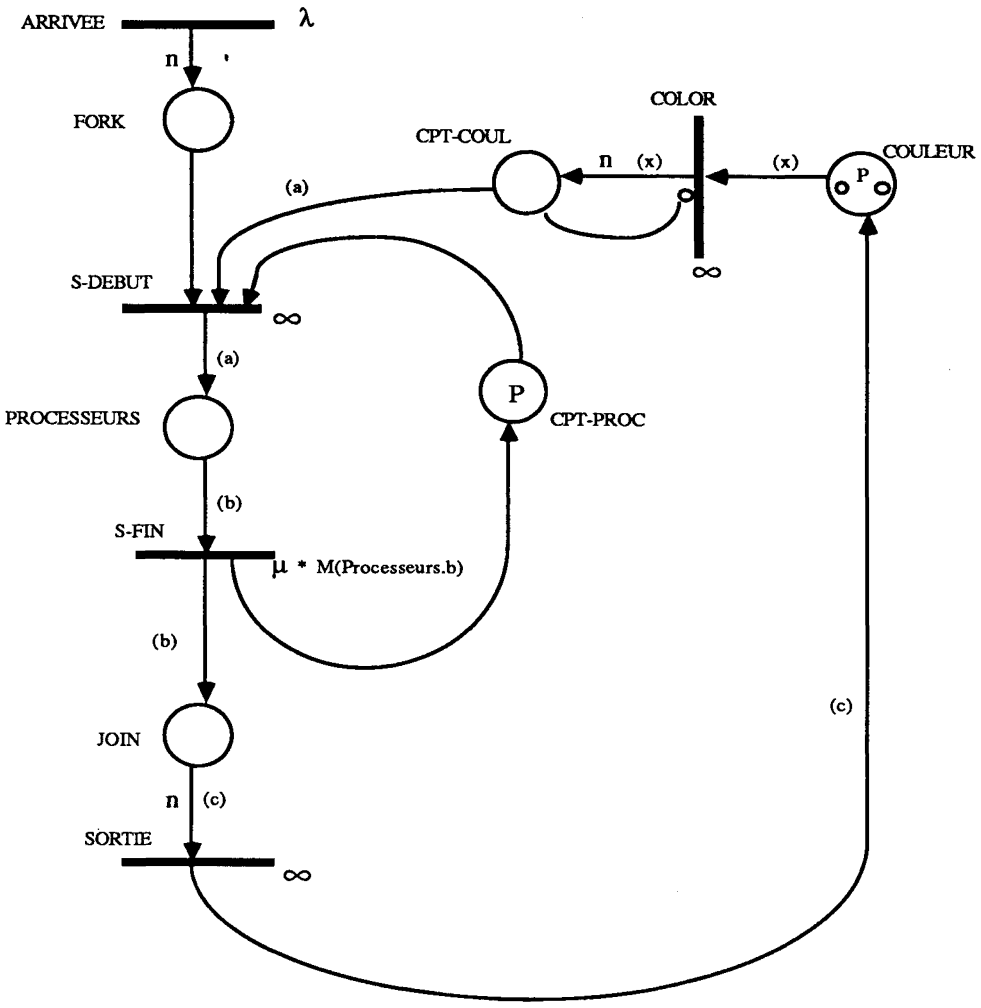


Figure 3

II. 3. Méthode d'exploitation du modèle

Après analyse du modèle, nous constatons que toutes les transitions sont vivantes et que toutes les places sont bornées sauf la place FORK. De plus, le graphe des marquages accessibles est fortement connexe. Ces propriétés font de notre modèle un exemple intéressant pour l'exploitation des résultats établis par G. Florin et S. Natkin sur les files complexes. Nous rappelons dans le paragraphe III ces résultats.

III. LES FILES COMPLEXES

III. 1. Analyse du graphe des marquages accessibles des files complexes

III. 1. 1. *Définition d'une file complexe*

Une file complexe est un réseau de Petri stochastique markovien tel que :

1. le réseau de Petri sous-jacent a exactement une place non bornée (la place p_1 dans la suite de l'article),

2. le graphe des marquages accessibles à partir du marquage initial M_0 est fortement connexe,

3. il n'y a pas d'arc inhibiteur issu de la place p_1,

4. le taux de chaque transition est indépendant du marquage.

La place non bornée peut être considérée comme une file d'attente, pour laquelle les arrivées et les départs dépendent du marquage des places bornées (cf. *fig.* 4).

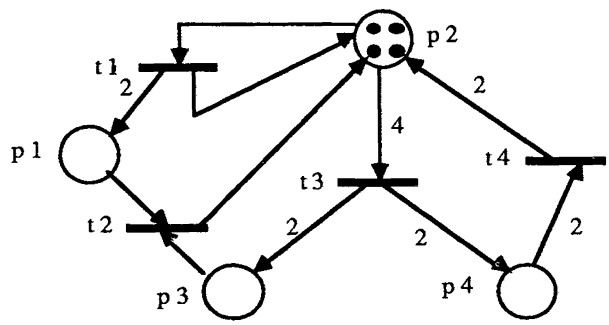


Figure 4

Remarque : La propriété 4 n'est pas utile pour les résultats énoncés ci-dessous. Il suffit que le taux de chaque transition soit indépendant du marquage de p_1.

III. 1. 2. *Propriété qualitative principale des files complexes*

Pour établir le théorème qualitatif principal pour les files complexes, il est nécessaire d'introduire la notion de sous-graphes d'accessibilité (*i. e.* sous-graphes du graphe des marquages accessibles) équivalents et trois propriétés des partitions des marquages accessibles.

III.1.2.1. *Sous-graphes d'accessibilité équivalents : définition*

Deux sous-graphes d'accessibilité d'un réseau de Petri sont dits équivalents s'ils sont isomorphes (au sens de la théorie des graphes) et si leurs arcs sont étiquetés par les mêmes noms de transitions.

III.1.2.2. *Théorème*

Pour toute file complexe, il existe une partition $V=\{V_0, V_1, \ldots, V_i, \ldots\}$ des marquages accessibles qui possède les trois propriétés suivantes :

1. Les sous-graphes générés par les marquages de toutes les classes V_i, $i>1$, sont équivalents.

2. Pour un entier donné $e>0$ et tout entier $i>0$, les sous-graphes générés par les marquages de $V_e \cup V_{e+i}$ sont équivalents.

3. Pour tous les entiers i et j tels que $|i-j|>1$, il n'y a pas d'arc entre les marquages de V_i et de V_j.

En d'autres termes, le graphe des marquages accessibles d'une file complexe peut être décomposé en un sous-graphe fini et une séquence infinie de sous-graphes finis équivalents. La figure 5 présente un exemple de file complexe avec son graphe des marquages. La démonstration du théorème se trouve en [FLO1].

III.2. Calcul des probabilités en régime permanent

III.2.1. *Introduction*

Si le processus de marquage d'une file complexe est récurrent positif (pour les problèmes d'ergodicité, se référer à [FLO1]), la distribution des probabilités en régime permanent est un vecteur positif infini Π, tel que :

$$\Pi . A = 0$$

$$\Sigma_i \Pi_i = 1$$

où A est le générateur du processus markovien associé à la file complexe.

Le calcul des probabilités en régime permanent pour une file complexe peut être effectué de différentes manières (*cf.* [FLO1]). Toutes les méthodes tiennent compte de la structure particulière du générateur markovien de la file complexe. Cette structure est complètement déterminée par les propriétés du graphe des marquages accessibles établies dans le paragraphe III.1.

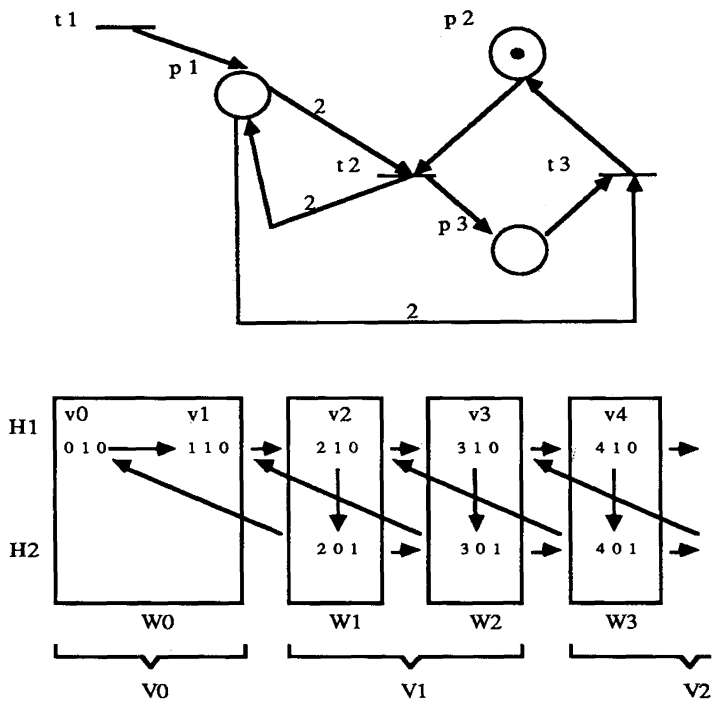


Figure 5

Comme le générateur markovien est infini mais a une structure régulière, il est possible d'avoir des équations de récurrence linéaires et finies. Les différentes partitions des marquages mènent à plusieurs sortes de récurrences. Dans tous les cas, deux approches peuvent être utilisées pour résoudre une récurrence :

- la méthode des formes produits matricielles
- la méthode des vecteurs propres généralisés.

La récurrence que nous utilisons est quadratique. D'après la partition V, le générateur est une matrice tridiagonale, également appelée matrice quasi-ment de naissance et de mort (*cf.* [WAL]). Dans ce cas, l'équation générique de la récurrence est donnée par :

$$X_{n-1}.B+X_n.A+X_{n+1}.C=0.$$

La figure 6 illustre cette décomposition.

La méthode que nous utilisons est celle des formes produits matricielles. Elle consiste à chercher un ensemble de matrices R_j de rayon spectral inférieur à 1 telles que :

$$\Sigma_{(i=1,q)} R_j^i . A_i = 0.$$

Dans ce cas, la solution recherchée est de la forme :

$$\Sigma_j X_q . R_j^{n-q}.$$

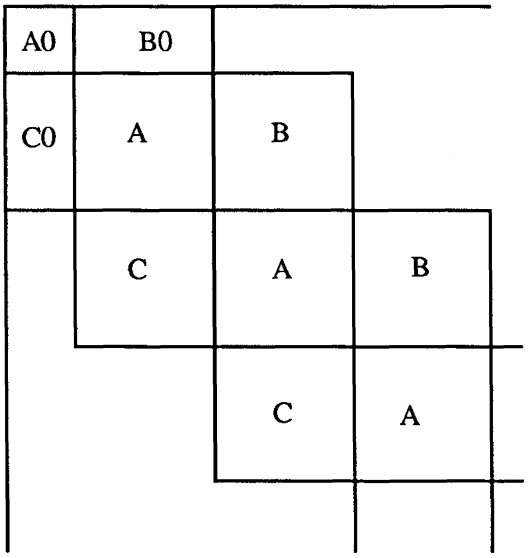


Figure 6

III. 2. 2. *Méthode des formes produits matricielles appliquée à l'étude de la récurrence quadratique*

Le traitement des processus aléatoires dont le générateur est quasiment de naissance et de mort a été abordé par V. Wallace et M. F. Neuts (*cf.* [WAL] et [NEU]). Si l'on considère la partition du générateur donné par la figure 5 et si l'on appelle P_n le vecteur des probabilités en régime permanent de la classe V_n, le système d'équations de Chapman-Kolmogorov est équivalent au

système d'équations récurrentes et linéaires suivant :

$$P_0 . A_0 + P_1 . C_0 = 0$$
$$P_0 . B_0 + P_1 . A + P_2 . C = 0$$
$$\cdots$$
$$P_{n-1} . B + P_n . A + P_{n+1} . C = 0$$
$$\cdots$$
$$\Sigma_i P_i = 1.$$

Soit R la solution positive minimale à l'équation matricielle :

$$R^2 . C + R . A + B = 0.$$

R est obtenu par la méthode itérative du point fixe :

$$R^{(0)} = 0$$
$$\cdots$$
$$R^{(n+1)} = -[R^{(n)}]^2 . C . (d(A))^{-1} - R^{(n)} . A . (d(A))^{-1} - B . (d(A))^{-1}$$

PROPRIÉTÉ 1 : Si le processus de Markov est récurrent positif, le rayon spectral de R est inférieur à 1 (*cf.* [NEU]).

PROPRIÉTÉ 2 : Pour toute file complexe dont le processus de Markov associé est récurrent positif, le système linéaire :

$$P_0 . A_0 + P_1 . C_0 = 0$$
$$P_0 . B_0 + P_1 . (A + R . C) = 0$$
$$P_1 . (I - R)^{-1} . e + P_0 . e = 1$$

a une solution positive unique : (P_0, P_1) (*cf.* [NEU]). C'est pourquoi le vecteur de probabilités en régime permanent pour chaque classe V_n est donné par :

$$P_n = P_1 . R^{n-1}.$$

III. 3. Conclusion

Le paragraphe III présente une méthode pour calculer la distribution des probabilités en régime permanant d'une file complexe. Il nous a paru intéressant dans un premier temps d'automatiser la construction du générateur à partir de la description d'une file complexe. Ceci constitue le sujet du paragraphe IV. D'autre part, il a fallu mettre en œuvre la méthode des formes produits matricielles, c'est à dire écrire un programme permettant le calcul des

probabilités en régime permanent pour un processus quasiment de naissance et de mort dont le générateur est connu. La réalisation d'algorithmes efficaces a posé différents problèmes. Tout ceci est présenté dans le paragraphe V.

IV. AUTOMATISATION DE LA CONSTRUCTION DU GÉNÉRATEUR

L'automatisation de la construction du générateur nécessite deux phases : la construction proprement dite et la réduction des états instantanés ou impossibles.

IV.1. Construction du générateur

IV.1.1. *Introduction*

Nous avons automatisé le repérage des classes de la partition V du graphe des marquages généré à partir du réseau de Petri. Pour cela nous avons utilisé plusieurs propriétés des sous-graphes caractéristiques d'une file complexe qui permettent de construire des algorithmes « efficaces » pour les générer.

En effet, les propriétés énoncées au paragraphe III ne sont pas suffisantes pour définir des algorithmes de génération efficaces. Elle prouvent que le graphe des marquages accessibles possède une structure répétitive de dimension 1. Il serait donc théoriquement possible de déterminer complétement les sous-graphes caractéristiques de la structure répétitive par une méthode de génération du graphe associée à une procédure de reconnaissance de forme. Mais cette approche est à éviter du fait de la difficulté de réalisation des algorithmes et de leur nature combinatoire.

IV.1.2. *Nature des problèmes*

Soit $W=\{W_0, W_1, \ldots, W_i, \ldots\}$ la partition correspondant aux plus petites classes répétitives (*cf.* exemple de la figure 5). Le problème consiste à déterminer la classe W_0, une classe W_i quelconque (W_1 par exemple) ainsi que les transitions de frontière entre W_0 et W_k d'une part et W_i et W_j d'autre part. Une classe V_i étant obtenue par concaténation de classes W_j adjacentes, les graphes caractéristiques de la partition V se déduisent des graphes caractéristiques de la partition W.

La construction d'un sous-graphe caractéristique d'une file complexe repose naturellement sur une procédure de type génération de graphe de marquages conséquents.

Le principal problème est donc celui du test d'arrêt de la procédure de génération. Pour obtenir un critère efficace, il est nécessaire de pouvoir caractériser la classe des marquages devant être retenus. Comme le graphe des marquages accessibles de la file complexe est infini, cette caractérisation peut être obtenue en bornant la variation de la marque de la place p_1 dans le sous-graphe considéré.

Les résultats que nous utilisons pour la construction d'une classe W_i quelconque, ont été établis par G. Florin et S. Natkin (*cf.* [FLO1]).

IV.1.3. *Notions nécessaires à l'automatisation de la construction du générateur*

IV.1.3.1. *Réseau saturé*

On appelle réseau saturé, le réseau de Petri stochastique déduit de la file complexe en supprimant la place non bornée p_1 et les arcs incidents à cette place.

Cette transformation équivaut à ne plus faire dépendre le tir des transitions de la marque de p_1 ou encore à considérer que la marque de p_1 est infinie.

IV.1.3.2. *Graphe limite*

On appelle graphe limite le graphe des marquages conséquents du réseau saturé.

IV.1.4. *Largeur d'une classe W_i*

IV.1.4.1. *Définition*

La largeur d'une classe W_i, notée l_w, est la variation minimale de la marque de la place p_1 entre deux marquages ayant même marque des places bornées.

$$l_w = \inf\{|M(p_1) - M'(p_1)|\}$$

où M et M' sont tels que :

$$M(p_1) \neq M'(p_1)$$
$$M(p_i) = M'(p_i), \qquad \forall i \geqq 2.$$

IV.1.4.2. *Théorème*

Soit $B = \{b_k\}$ une base de circuits du graphe limite et $\mathbf{B} = \{\mathbf{b}_k\}$ l'ensemble des vecteurs caractéristiques correspondants. La largeur d'une classe W_i est donnée par :

$$l_w = |\operatorname{pgcd}(C_1 . \mathbf{b}_k)| \qquad \text{pour} \quad \mathbf{b}_k \in \mathbf{B}$$

où C_1 représente la première ligne de la matrice d'incidence de la file complexe.

La démonstration de ce théorème est complexe et se trouve en [FLO1].

IV.1.5. *Algorithme de détermination de la largeur d'une classe W_i*

La construction d'une base de circuits du graphe limite est réalisée en phase de génération du graphe des marquages accessibles. Pour chaque nouveau sommet obtenu, on calcule selon la séquence s tirée depuis le marquage initial, la valeur $C_1 . \mathbf{s}$ correspondant à la variation relative de la marque (fictive) de la place p_1 (qui peut être positive ou négative).

Lorsqu'un marquage a pour marquage conséquent un marquage déjà atteint sur la même branche, on a donc obtenu là, l'un des circuits b_k d'une base de circuits et nous pouvons alors noter la valeur de $C_1 . \mathbf{b}_k$ (variation de la marque de la place p_1 selon ce circuit).

Lorsque tous les circuits ont ainsi été générés, il est alors possible de calculer le plus grand dénominateur commun des valeurs $C_1 . \mathbf{b}_k$ obtenues.

Pour l'exemple de file complexe présentée au paragraphe III (*fig.* 4) et dont le graphe limite est présenté figure 7, les circuits de la base de circuits engendrée sont au nombre de huit :

$s_1 = t_3 t_4 t_2 t_2$ à partir de 400
$s_2 = t_3 t_2 t_2 t_4$ à partir de 400
$s_3 = t_3 t_2 t_4 t_2$ à partir de 400
$s_4 = t_1$ à partir de 400
$s_5 = t_1$ à partir de 220
$s_6 = t_1$ à partir de 112
$s_7 = t_1$ à partir de 310
$s_8 = t_1$ à partir de 202

Les variations de $C_1 . \mathbf{s}_i$ sont respectivement -2 pour les trois premières séquences et $+2$ pour les cinq dernières. Le pgcd vaut donc 2 et par suite $l_w = 2$.

IV.1.6. *Construction du graphe interne et des graphes frontières d'une classe W_i*

Les marquages d'une classe W_i quelconque, en ce qui concerne les places bornées, sont les mêmes que ceux du graphe limite. Il suffit donc de déterminer la variation relative de la marque de la place p_1. Cette opération est réalisable en modifiant légèrement l'algorithme précédent. Pour cela il suffit de conserver au fur et à mesure de la construction de l'arbre, pour chaque marquage

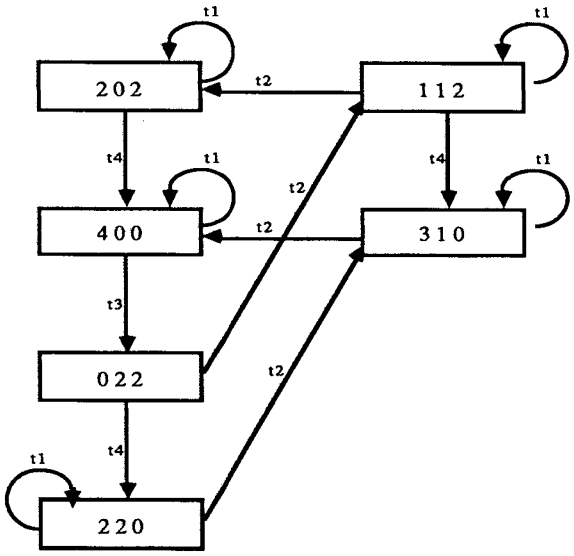


Figure 7

atteint, la variation relative de la marque de la place p_1, avec le marquage initial.

Soit la partition $H=\{H_0, H_1, \ldots, H_q, \ldots\}$ définie par l'ensemble des classes d'équivalence pour la relation $\mathfrak{R}_H$ entre marquages.

$$M_i \mathfrak{R}_H M_j \quad \Leftrightarrow \quad M_i(p_k)=M_j(p_k), \qquad \forall k \geqq 2.$$

Deux marquages sont équivalents s'ils ont même marque pour l'ensemble des places bornées (*cf.* exemple de la figure 5). Comme toutes les places différentes de p_1 sont bornées par hypothèse, le cardinal de H est fini. Chaque classe H_q est donc caractérisée par une configuration atteignable des marques des places bornées. Par abus de notation, H_q désignera également le vecteur ligne constitué par les marques des places bornées pour tous les marquages de la classe H_q.

Lorsque l'on a déterminé l_w, on considère un représentant de chacune des classes H_q et l'on calcule le reste de la division de la marque associée à la place p_1 par l_w. Soit x_q ce reste pour le représentant de la classe H_q. Ceci détermine pour les marquages internes d'une classe W_i, la variation relative de la marque de la place p_1 entre ces marquages internes.

La génération des graphes internes et des graphes frontières consiste à considérer tous les marquages d'une classe W_i de la forme :

$$M=[x_q,\ H_q]$$

Pour chaque marquage ainsi défini on considère toutes les transitions tirables en ce marquage. Trois cas sont possibles :

– les transitions t_j telles que :

$$0 \leqq x_q + C_1(j) < l_w$$

sont internes à la classe W_i. Elles permettent donc de construire la sous-matrice A du générateur.

– les transitions t_j telles que :

$$x_q + C_1(j) < 0$$

permettent de définir le graphe frontière entre la classe W_i et la classe W_{i-k}. L'indice k est défini par :

$$k = \mathrm{Ent}[|x_q + C_1(j)| \div l_w].$$

La transition t_j conduit d'un marquage initial (le marquage interne H_q) à un marquage H_p. Ce marquage H_p de la classe W_{i-k} est déterminé par la marque des places bornées. Ces transitions permettent de construire la sous-matrice C du générateur.

– les transitions t_j telles que :

$$x_q + C_1(j) \geqq l_w$$

permettent de définir le graphe frontière entre la classe W_i et W_{i+k}. L'indice k est défini par :

$$k = \mathrm{Ent}[(x_q + C_1(j)) \div l_w]$$

Ces transitions permettent de construire la sous-matrice B du générateur.

Exemple : Compte tenu de la variation relative de la place p_1, les marquages obtenus lors de la construction du graphe des marquages de l'exemple précédent sont :

$$0400$$
$$0022$$
$$0220$$
$$-1310$$
$$-1112$$
$$-2202$$

Si nous considérons la place de p_1 modulo l_w, nous obtenons :

$$0400$$
$$0022$$
$$0220$$
$$1310$$
$$1112$$
$$0202$$

En tirant toutes les transitions possibles à partir de ces marquages, nous obtenons le graphe interne et les graphes frontières représentés par la figure 8.

IV.1.7. *Construction de la classe* W_0

IV.1.7.1. *Problème posé*

Cet algorithme est comme précédemment un algorithme de génération de graphe des marquages accessibles d'un réseau de Petri, adapté à la construction d'un sous-graphe fini d'un graphe infini de marquages.

Pour obtenir un algorithme efficace nous supposons que le marquage initial se trouve dans la classe W_0.

Le problème est alors de trouver un majorant B_0 du nombre de saturation $r(M_0)$, défini comme la valeur de la place p_1 au-delà de laquelle le graphe des marquages présente une structure répétitive, tel que ce soit la plus petite valeur de la marque de la place p_1 à l'intérieur d'une classe W_i repérée par l'algorithme précédent.

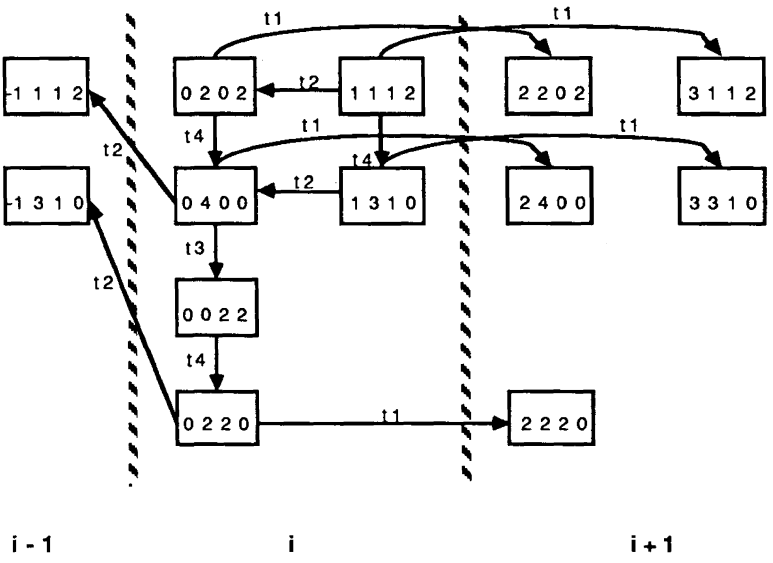


Figure 8

IV.1.7.2. *Théorèmes*

THÉORÈME 1 :

$$r(M_0) \leqq \mathrm{Sup}_{t_j} C_1(j) + \mathrm{Sup}_{t_j} C_1^-(j)$$

THÉORÈME 2 :

$$\forall M,$$

$$(\forall i \geqq 2,\ M(p_i) = M_0(p_i)) \Rightarrow (\exists k \in N / M(p_1) = M_0(p_1) + k \,.\, l_w)$$

IV.1.7.3. *Solution*

Pour s'assurer que B_0 est un majorant de $r(M_0)$ et qu'il est également la plus petite valeur de la marque de la place p_1 à l'intérieur d'une classe W_i repérée à l'étape précédente, il suffit de chercher le plus petit entier k tel que :

$$B_0 = M_0(p_1) + k \,.\, l_w > \mathrm{Sup}_{t_j} C_1(j) + \mathrm{Sup}_{t_j} C_1^-(j)$$

IV.1.7.4. *Principe de l'algorithme*

On commence à générer le graphe des marquages à partir du marquage initial. Les marquages dont la marque de p_1 est inférieure à B_0 constituent la classe W_0. Pour être certain de créer tous ces marquages ainsi que les arcs de W_0 et de la frontière $W_0 - V_1$, il est nécessaire d'évaluer tous les marquages dont la marque de p_1 est inférieure ou égale à :

$$B_0 + \mathrm{Sup}_{t_j}\{-C_1(j)\} - 1$$

En effet, de tels marquages peuvent être antécédents d'un marquage de W_0. On s'arrête quand tous ces marquages ont été évalués.

Pour les arcs étudiés entre les marquages M_1 et M_2, quatre cas sont possibles :

– si $M_1(p_1) < B_0$ et $M_2(p_1) < B_0$ il s'agit alors d'un arc interne à W_0.

– si $M_1(p_1) < B_0$ et $M_2(p_1) \geqq B_0$ il s'agit alors d'un arc entre W_0 et W_k. L'indice k est défini par :

$$k = \mathrm{Ent}\frac{[M_2(p_1) - B_0]}{l_w} + 1$$

– si $M_1(p_1) \geqq B_0$ et $M_2(p_1) < B_0$ il s'agit alors d'un arc entre W_k et W_0. L'indice k est défini par :

$$k = \mathrm{Ent}\frac{[M_1(p_1) - B_0]}{l_w} + 1$$

– si $M_1(p_1) \geqq B_0$ et $M_2(p_1) \geqq B_0$ il s'agit alors d'un arc entre une classe W_i et une classe W_j, qui ne nous intéresse donc pas.

IV.2. Réduction des états instantanés et impossibles

Les algorithmes décrits ci-dessus permettent de générer l'ensemble des marquages de la classe W_0 et d'une classe répétitive quelconque W_i, ainsi que les informations nécessaires à la construction des sous-matrices A_0, B_0, C_0, A, B et C du générateur. Nous n'avons pas pris en considération jusqu'à présent le taux propre à chaque transition. En fait, il faut valuer les arcs du graphe des marquages par ces taux. Néanmoins, il n'est utile d'en tenir compte qu'après la génération complète des marquages de W_0 et de W_i. Le fait de les introduire à ce niveau facilite d'ailleurs le changement éventuel de la valeur de ces taux.

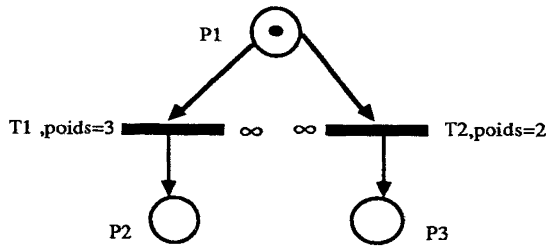
Le logiciel RdPS, que nous utilisons et qui traite les réseaux de Petri stochastiques, permet de saisir interactivement trois types de taux : les taux infinis, les taux constants et les taux dépendants par une fonction linéaire de la marque d'une place.

L'existence de taux infinis implique l'existence de marquages évanescents et éventuellement impossibles (cas d'un conflit entre une transition à taux infini et une à taux fini). Pour pouvoir calculer la distribution de probabilités associées aux marquages, il est obligatoire de supprimer ces états. Une partie de notre travail a donc consisté à mettre au point un algorithme général de réduction des états instantanés et à l'adapter au problème d'une file complexe.

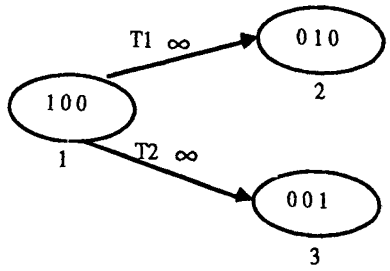
IV.2.1. *Fonctionnalité nouvelle de la chaîne RdPS*

Il est possible maintenant d'associer un poids à chaque transition à taux infini. Ceci permet, en fait, d'associer une probabilité de tir pour des transitions concurrentes à taux infini. Illustrons ceci par un exemple.

Soit le réseau de Petri représenté ci-dessous :



Son graphe des marquages est le suivant :



La distribution de probabilités est alors :

$$P(1)=0 \qquad \text{(état évanescent)}$$

$$P(2)=3/5$$

$$P(3)=2/5$$

Pour chacun des états évanescents du graphe des marquages, il s'agit de faire la somme des poids attachés aux transitions tirables depuis ce marquage, puis de diviser chacun de ces poids par cette somme.

IV.2.2. *Principes de base*

IV.2.2.1. *Réduction des états instantanés*

L'idée est de parcourir l'ensemble des couples de marquages (M_1, M_2) reliés par un arc, de voir si cet arc correspond à une transition à taux infini, et, si tel est le cas (M_1 est alors évanescent), de relier les prédécesseurs de M_1 à M_2. Le problème est alors d'associer aux arcs ainsi créés des taux adéquats.

IV.2.2.2. *Détection des états impossibles*

Il s'agit de supprimer du graphe des marquages les états impossibles à atteindre. De tels états apparaissent lorsque d'un état évanescent est issu un arc valué par un taux fini; l'état auquel on aboutit est alors impossible à atteindre par ce chemin. L'algorithme consiste donc à parcourir, à partir du marquage initial, le graphe des marquages auquel on a supprimé les états instantanés. On repère ainsi les états effectivement atteints.

IV.2.3. *Adaptation de l'algorithme au problème d'une file complexe*

Pour supprimer les états impossibles ou instantanés du graphe des marquages d'une file complexe, il suffit de travailler sur la concaténation du graphe correspondant à la classe W_0 et du graphe limite.

Nous définissons ainsi : l'indice d'un arc si le marquage initial appartient à la classe W_i et le marquage final à la classe W_j, alors l'indice vaut $(j-i)$.

Le problème est d'associer aux arcs l'indice leur correspondant. Il est à noter que plusieurs arcs peuvent exister entre deux marquages si au moins l'un d'entre eux appartient au graphe limite. La différence entre ces arcs porte sur la valeur de leurs indices. Le figure 9 illustre ce propos.

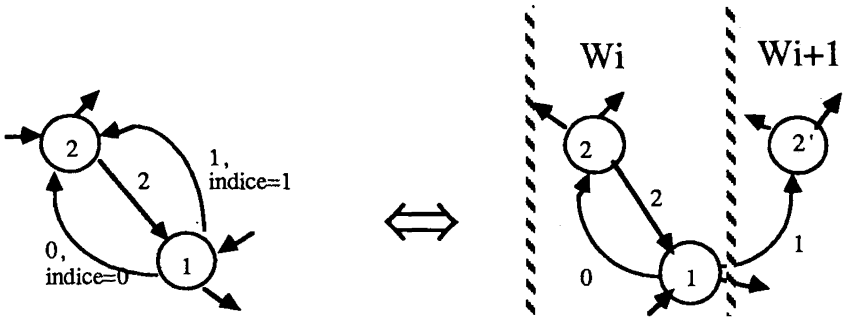


Figure 9

IV.3. Conclusion

A cette étape, il est possible de construire le générateur automatiquement en ne tenant compte que des états tangibles. Il reste donc à résoudre le système associé à ce générateur pour calculer la distribution complète des probabilités liées aux marquages.

V. IMPLANTATION DE LA MÉTHODE DES FORMES PRODUITS MATRICIELLES

Le traitement de la méthode des formes produits matricielles, même dans les cas simples, nécessite l'utilisation d'un programme. Il a été développé en Fortran l'ensemble des traitements correspondant au calcul des probabilités en régime permanent pour un processus quasiment de naissance et de mort.

V.1. Présentation du programme

Les différentes étapes de ce programme conduisent aux résultats intermédiaires suivants.

V.1.1. *Calcul de la solution matricielle non négative minimale de l'équation caractéristique*

Ce calcul est effectué par une méthode itérative de point fixe (*cf.* [NEU]) en utilisant le fait que la diagonale de la matrice A est complète. Les opérations effectuées [si l'on note $d(A)$ la matrice inversible diagonale de A] sont les suivantes :

$$R^{(0)}=0$$

$$R^{(n+1)}=-R^{2(n)}.C.d(A)^{-1}-R^{(n)}.A.d(A)^{-1}-B.d(A)^{-1}$$

Les itérations sont terminées par un test d'arrêt sur la norme de l'écart relatif :

$$\| R^{(n+1)} - R^{(n)} \| / \| R^{(n)} \| < 10^{-5}.$$

V.1.2. *Résolution des équations de partie commençante*

Les équations de partie commençante sont :

$$P_0 . A_0 + P_1 . C_0 = 0$$
$$P_0 . B_0 + P_1 (A + R . C) = 0$$

Le calcul de la solution initiale (P_0, P_1) nécessite donc le calcul de la matrice $H(R)$:

$$H(R) = \begin{matrix} A_0 & B_0 \\ C_0 & A+R.C \end{matrix}$$

puisque l'on a par définition $(P_0, P_1) . H(R) = 0$.

V.1.3. *Équation de normalisation*

Cette équation nécessite le calcul de l'inverse de $I-R$ et la sommation des éléments en ligne de cette matrice inverse puisque :

$$P_1 . (I-R)^{-1} . e + P_0 . e = 1$$

Le calcul de l'inverse est résolu par une méthode d'élimination de Gauss.

V.1.4. *Calcul de la solution initiale*

Par remplacement de l'une des colonnes de la matrice $H(R)$ par les coefficients de l'équation de normalisation, on obtient le système linéaire déterminant la solution initiale (P_0, P_1). Ce système linéaire est résolu par une méthode d'élimination de Gauss avec pivotage. La solution finale du calcul des probabilités en régime permanent par la méthode des formes produits matricielles est donc :

$$P_n = P_1 . R^{n-1}.$$

V.2. Évaluation du temps de résolution et analyse des résultats

Pour évaluer les capacités du programme, nous avons réalisé une série de tests sur le n, 2 Fork-Join. Nous voulions notamment savoir quelle était la taille maximale des problèmes qu'il était possible de traiter dans un temps

raisonnable, et comment le temps d'exécution se partageait entre les différentes parties du programme.

Il apparaît que le calcul de la matrice R est la partie du programme qui consomme le plus de temps CPU. Nous avons également observé que ce temps est supérieur à celui nécessaire à la construction automatique du générateur. Le calcul de la matrice R devait donc être amélioré en priorité pour obtenir un meilleur temps d'exécution du programme.

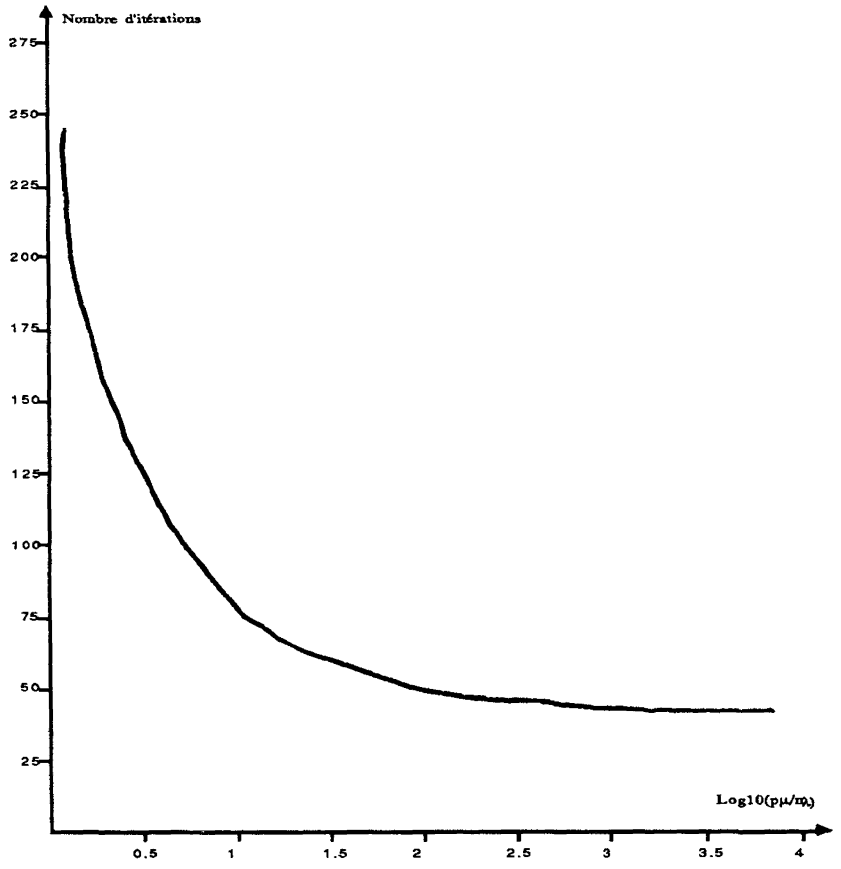


Figure 10

Pour cette série de tests, nous avons opéré à $p\mu/n\lambda$ constant ($p\mu/n\lambda = 2$). Cette grandeur, caractéristique de l'ergodicité du n, p Fork-Join (il faut que $p\mu/n\lambda$ soit supérieur à 1 pour que le processus de marquage associé soit récurrent positif), influe sur la vitesse de convergence de la matrice R par la

méthode itérative du point fixe. La courbe de la figure 10 montre l'évolution du nombre d'itérations en fonction de la variable $p\mu/n\lambda$ (dans le cas où $n=25$).

V.3. Amélioration du temps de résolution

La principale amélioration apportée à l'algorithme initial, concerne la prise en compte de la structure creuse des matrices A, B et C lors du calcul itératif de R par la méthode du point fixe. Chacune de ces trois matrices est stockée sous forme d'un tableau de listes. Chaque liste contient les éléments non nuls d'une colonne. Il a alors été nécessaire de définir le produit d'une matrice pleine par une matrice creuse ($R.C$) et l'addition d'une matrice creuse et d'une matrice pleine ($A+R.C$).

La figure 11 présente le gain principalement apporté par cette modification. Les deux courbes représentent l'évolution du temps d'exécution en fonction de la taille du problème. La courbe en pointillés concerne le programme initial, la courbe en trait plein le programme après modifications.

VI. CONCLUSION

Nos travaux ont donc mené à différents résultats intéressants : la modélisation du n, p Fork-Join sous forme de réseaux de Petri stochastiques, l'automatisation de la construction du générateur lié à une file complexe et la mise en œuvre de la méthode des formes produits matricielles. Pour ces deux derniers traitements, des programmes ont été développés. Ils constituent des nouveaux modules du logiciel RdPS.

Néanmoins, en ce qui concerne le programme lié à la méthode des formes produits matricielles, nous avons constaté qu'il subsistait des problèmes liés au nombre d'états appartenant à une classe répétitive. Ces problèmes se posent lors de la recherche d'une solution de l'équation $R^2.C+R.A+B=0$, car, dans ce cas, les matrices A, B, C et R sont de taille importante. Il semblerait qu'ils soient dus à l'utilisation d'une méthode lente de type Jacobi. Une possibilité pour les résoudre serait donc l'utilisation d'une méthode de type Newton dont la convergence vers la solution est plus rapide, mais les calculs plus complexes. D'autre part, des méthodes approchées d'agrégation-désagrégation pourraient également aider à trouver une solution à ces problèmes.

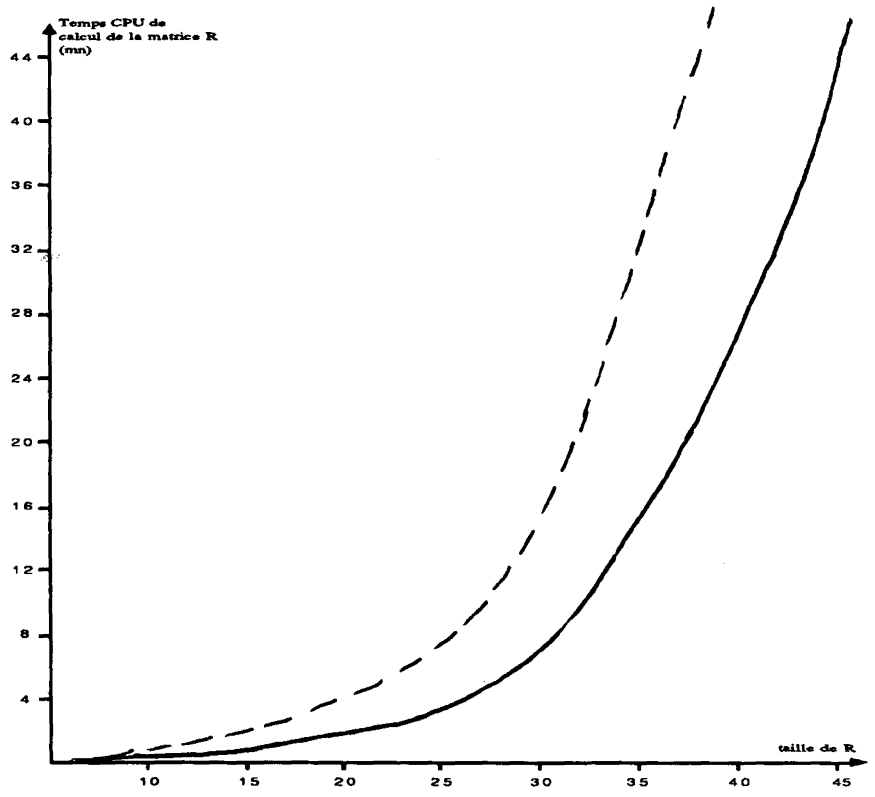


Figure 11

REMERCIEMENTS

Nous tenons à remercier M. Becker (I.N.T.), G. Florin (C.N.A.M.) et S. Natkin (C.N.A.M.) qui nous ont beaucoup aidés lors de la réalisation de ce travail.

BIBLIOGRAPHIE

[FLO1] G. FLORIN et S. NATKIN, *Les réseaux de Petri, théorie, techniques de calcul et applications*, Thèse de doctorat d'état, Université de Paris-VI, Juin 1985.

[FLO2] G. FLORIN et S. NATKIN, *Les réseaux de Petri stochastiques*, TSI, vol. 4, n° 1, 1985, p. 143-160.

[FLO3] G. FLORIN et S. NATKIN, *One-Place Unbounded Stochastic Petri Nets: Ergodic Criteria and Steady-State Solutions*, The journal of systems and software, vol. 6, n° 1 et 2, mai 103-115.

[HOU] P. HOUEIX et M. BECKER, *Modélisation d'une architecture parallèle pour le traitement d'images*, RAIRO Recherche opérationnelle/Operations Research, vol. 22, n° 1, janvier 1988, p. 45-65.

[NEU] M. F. NEUTS, *Matrix Geometric Solution in Stochastic Models*, Johns Hopkins University Press, Baltimore, 1981.

[WAL] V. WALLACE, *The Solution of Quasi-Birth and Death Processes Arising from Multiple Access Computer Systems*, Ph. D. dissertation, University of Michigan, Technical report, n° 07742-6-t, 1969.