Efficacite de la méthode des puissances uniformisées pour les chaînes de Markov raides

Haïscam Abdallah, Raymond Marie

INRIA

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P 105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

N° 1129

Programme 3
Réseaux et Systèmes Répartis

EFFICACITE DE LA METHODE DES PUISSANCES UNIFORMISEES POUR LES CHAINES DE MARKOV RAIDES

Haïscam ABDALLAH
Raymond MARIE

Décembre 1989

RR-1129

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CÉDEX
FRANCE
Téléphone : 99 36 20 00
Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Publication Interne n°500 - Octobre 1989 - 18 Pages

Efficacité de la méthode des Puissances Uniformisées pour les chaînes de Markov raides

Haïscam ABDALLAH et Raymond MARIE
I.R.I.S.A. av. du Général Leclerc
35042 Rennes Cedex France

Résumé

Très souvent, pour les systèmes informatiques tolérant les pannes et à haute disponibilité, les taux de pannes et de recouvrement sont très éloignés les uns des autres. Par conséquent, toute chaîne de Markov à temps continu homogène modélisant le comportement de tels systèmes est raide. L'évaluation quantitative des mesures de la sûreté de fonctionnement des systèmes considérés passe par l'étude du comportement de la chaîne raide en régime transitoire. Cette étude se heurte à des problèmes de précision et de complexité temporelle. Dans ce rapport, on compare la complexité temporelle de notre méthode des Puissances Uniformisées à celle de l'Uniformisation Directe.

Efficiency of the Uniformized Powers method for stiff Markov chains

Abstract

Very often, for fault-tolerant computer systems with high availability, the failure and recovery rates have very different orders of magnitude. Consequently, a continuous time Markov chain used for these systems is stiff. Computation of the dependability measures comes back to study the transient behavior of the stiff Markov chain. With such a study, we are confronted to precision and time complexity problems. In this report, we compare our Uniformized Powers method with the Uniformization one, from a time complexity point of view.

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L. A. 227)
UNIVERSITÉ DE RENNES 1 I. N. S. A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

1 Introduction

Pour certaines applications, la panne d'un système informatique a des conséquences désastreuses sur le plan humain ou sur le plan économique. Pour tenter d'éviter ces conséquences sont apparus des systèmes dits tolérants aux pannes dans le sens où, en présence de pannes, un système informatique peut continuer à fonctionner en mode dégradé et permettre ainsi la réalisation d'une mission. Bien sûr, cette conception d'architecture a sa limite car, souvent, pour réaliser sa mission, le système doit rester au dessus d'un certain niveau de dégradation. Notons que le système pourra satisfaire à cette contrainte d'autant plus facilement que la mission sera courte ou qu'il sera possible de réparer le système.

L'évaluation de tels systèmes informatiques passe normalement par l'étude transitoire de Chaînes de Markov à Temps Continu (CMTC); en effet, ces processus constituent actuellement le meilleur objet mathématique pour modéliser et analyser de tels systèmes.

On s'intéresse plus particulièrement aux systèmes à haute disponibilité dont les taux de pannes et de recouvrement sont très éloignés les uns des autres. Le générateur infinitésimal de la CMTC présente alors entre ses coefficients un rapport de grandeur très élevé (rapport pouvant dépasser 10^8). Une CMTC ayant un tel générateur est dite *raide*. Or, en menant l'étude transitoire d'une CMTC raide, nous sommes généralement confrontés à des problèmes de précision et de complexité temporelle.

Dans ce rapport, nous comparons la complexité temporelle de notre méthode des Puissances Uniformisées (PU) avec celle de l'Uniformisation Directe (UD) (i.e., la méthode standard de l'Uniformisation). Tout d'abord, nous rappelons les idées de base des deux méthodes et les problèmes de mise en œuvre qu'elles soulèvent.

Considérons dans ce but une CMTC $\mathbf{X} = \{X(t), t \in \mathbb{R}^+\}$ ayant un espace fini d'états $\mathbb{E}$={1,2,3,...,M}. Soit $A = (a_{ij})$ son générateur infinitésimal. Il est bien connu que l'étude de $\mathbf{X}$ en régime transitoire repose sur la résolution du système des M^2 équations différentielles de Chapman-Kolmogorov

$$P'(t) = P(t)A \ , \ t \in \mathbb{R}^+ \ , \ P(0) = I \tag{1}$$

où $P(t)$ désigne la matrice des probabilités de transition entre états à l'instant t et où I désigne la matrice unité; mais la résolution numérique directe de (1) est compliquée et coûteuse [1]. On sait aussi que sa solution unique s'écrit

$$P(t) = e^{At} = I + \sum_{n=1}^{\infty} A^n \frac{t^n}{n!} \tag{2}$$

Malheureusement, l'utilisation directe de la formule (2) donne des résultats imprécis.

Cela est dû essentiellement au fait que les éléments diagonaux a_{ii} du générateur infinitésimal A sont négatifs ou nuls. De plus, la complexité temporelle est très élevée en raison du nombre important de produits matriciels à effectuer [2].

2 Méthodes de l'Uniformisation Directe et des Puissances Uniformisées

2.1 Méthode de l'Uniformisation Directe

La méthode de l'Uniformisation Directe (UD) [3] repose sur l'équivalence, au sens des distributions conjointes de probabilités, entre une CMTC et une chaîne de Markov à temps discret dont les transitions se produisent au rythme d'un processus de Poisson. La matrice $P(t)$ obtenue par cette méthode s'écrit

$$P(t) = e^{-vt}e^{vP^*t} = \sum_{n=0}^{\infty} e^{-vt}\frac{(vt)^n}{n!}P^{*n} \tag{3}$$

où le scalaire v et la matrice stochastique P^* sont donnés par

$$v = \max_i \mid a_{ii} \mid$$

$$P^* = I + \frac{A}{v}$$

Désignons par $\Pi(0)$ le vecteur de la distribution initiale de la CMTC $\mathbf{X}$. L'état de $\mathbf{X}$ à l'instant t peut être décrit par le vecteur $\Pi(t)$ des probabilités d'états à cet instant. Ce vecteur s'écrit alors

$$\Pi(t) = \Pi(0)P(t) = \sum_{n=0}^{\infty} e^{-vt}\frac{(vt)^n}{n!}\Pi(0)P^{*n} \tag{4}$$

Notons que la formule (4) permet d'obtenir directement le vecteur $\Pi(t)$ par des produits vecteur-matrice de la forme $\Pi^{(n)} = \Pi^{(n-1)}P^*$, avec $\Pi^{(0)} = \Pi(0)$; cela permet d'éviter les produits matriciels de la formule (3). Bien sûr, une troncature de la sommation infinie est nécessaire, elle est basée sur la recherche d'un entier

$$N_s = Min\{N/1 - \sum_{n=0}^{N} e^{-vt}\frac{(vt)^n}{n!} \leq \epsilon\} \tag{5}$$

où ϵ est une tolérance fournie a priori par l'utilisateur. Du point de vue de la précision, l'avantage de cette méthode, intéressante vis-à-vis des méthodes précédentes, est qu'elle

ne manipule que des coefficients non négatifs (ceux de P^* au lieu de ceux de A). En revanche, la troncature pose le problème du contrôle des erreurs pour un utilisateur (non spécialiste) cherchant à obtenir une bonne précision. Du point de vue de la complexité temporelle, notons que la méthode UD permet d'obtenir le vecteur $\Pi(t)$ pour un instant donné t. Par conséquent, une répétition des calculs est nécessaire pour obtenir la réponse transitoire sur un ensemble de valeurs du temps. Dans la méthode proposée ci-après, nous verrons qu'une bonne précision pourra être obtenue sans que le choix de la troncature soit à la charge de l'utilisateur. De plus, cette méthode permettra d'obtenir l'ensemble de la réponse transitoire sans avoir à fournir successivement plusieurs valeurs de t.

2.2 Méthode des Puissances Uniformisées

La démarche générale de la méthode des Puissances Uniformisées (PU) (cf [4] et [5]) peut être résumée selon la séquence suivante :

- Calcul d'un instant t_0 tel que vt_0 soit strictement compris entre 0 et 1,
- Calcul de la matrice $P(t_0)$ en utilisant la méthode UD et l'algorithme de décomposition descendante d'Horner,
- Retour à $P(t) = P(2^m t_0)$ à l'aide des équations de Chapman-Kolmogorov $P(t_k) = P^2(t_{k-1})$, $t_k = 2^k t_0$, $k = 1, 2, 3, ..., m$; où m est un entier fonction de vt et du maximum de termes non nuls dans les colonnes de la matrice P^*,
- Obtention du vecteur $\Pi(t)$ à partir de la matrice $P(t)$ par la relation connue

$$\Pi(t) = \Pi(0)P(t)$$

Sur le plan de la précision, cette méthode a l'avantage de permettre le contrôle des erreurs de précision. En effet, des bornes relatives aux différentes erreurs sont mises en évidence [4]. De plus, il est possible de calculer une valeur critique de t définie comme la valeur à partir de laquelle la précision des résultats avec un chiffre décimal significatif n'est plus garantie.

Cette méthode est mise en œuvre dans le logiciel PUMAR (Puissances Uniformisées pour les chaînes de MARkov) [4]. Dans [4], nous avons aussi développé un algorithme efficace de calcul précis par la méthode UD et nous avons comparé la précision des deux méthodes (UD et PU) faisant ici l'objet de cette étude comparative des complexités temporelles; cette comparaison de précision a montré que la précision de notre méthode (PU) était légèrement meilleure que celle de la méthode UD dans le cas des chaînes raides.

3 Cas des chaînes raides

Pour les CMTC raides, l'ordre de grandeur du rapport des taux est très élevé (10^8 par exemple). Le régime transitoire est donc "très long". Il est alors nécessaire de calculer $\Pi(t)$ pour des valeurs de t telles que $vt \approx 10^8$.

Lorsque vt est très grand, la troncature de la méthode UD se fait à la fois à gauche et à droite, c'est-à-dire que, pour un ϵ donné, on calcule la borne inférieure N_i et la borne supérieure N_s de la façon suivante

$$N_i = Max\{N; \sum_{n=0}^{N} e^{-vt}\frac{(vt)^n}{n!} \leq \frac{\epsilon}{2}\}$$

$$N_s = Min\{N; 1 - \sum_{n=N_i+1}^{N} e^{-vt}\frac{(vt)^n}{n!} \leq \frac{\epsilon}{2}\}$$

Remarquons que N_i se rapproche relativement de N_s lorsque vt augmente; ceci correspond au fait que le carré du coefficient de variation de la distribution de Poisson diminue lorsque vt augmente, bien que la variance continue à croître. Notons aussi que cet ϵ de troncature ne doit pas être confondu avec la précision des résultats de la méthode; en effet, lorsque vt augmente, le nombre d'opérations arithmétiques augmente et les erreurs d'arrondi peuvent entraîner une imprécision qui dépasse largement celle dûe à la troncature. Ainsi par exemple, si $\epsilon \approx 10^{-14}$, $M = 9$ et $vt = 3276800$, alors $N_i = 3263205$ et $N_s = 3288489$; dans ce cas, le maximum de la valeur absolue des erreurs commises est approximativement $5 * 10^{-11}$ (dans ce présent rapport, tous les calculs sont faits en double précision sur un VAX 8250). Le contrôle des erreurs par la méthode des PU est, au contraire de la méthode UD, efficace et les erreurs commises sur les résultats ne dépassent pas le majorant théorique. Les deux méthodes UD et des PU perdent en précision au niveau des résultats lorsque vt devient très grand. Rappelons que pour la méthode des PU, une limite d'utilisation est mathématiquement déterminée (valeur critique de vt). La complexité temporelle des deux méthodes est examinée dans la section suivante.

4 Étude comparative

Du point de vue de la complexité temporelle de la méthode UD, la croissance de N_s avec vt est rapide et N_s reste toujours supérieur à vt. Pour effectuer le calcul de $\Pi(t)$ pour une seule valeur de t, il faut effectuer N_s produits vecteur-matrice. Ce qui nécessite un temps de calcul qui devient prohibitif pour les très grandes valeurs de vt. De plus, s'il s'agit

d'obtenir la réponse transitoire, il faut répéter les calculs pour un ensemble de valeurs de t et le temps de calcul est alors encore plus prohibitif. Toutefois, comme le générateur infinitésimal A est en général une matrice creuse, on verra ci-après que l'utilisation d'un stockage compact permet une diminution relative de ce temps de calcul.

Au niveau de la méthode des PU, la croissance avec vt n'a que peu d'incidence sur le nombre de produits vecteur-matrice; en effet, le calcul de la matrice $P(t_0)$ se fait, suivant la formule (3), à l'aide de N'_s produits matrice-matrice, N'_s étant toujours inférieur à 10 compte tenu du choix de t_0. Le retour à la matrice $P(t)$ se fait grâce à un nombre m de produits matriciels où m est compris entre 30 et 40 pour les chaînes les plus raides ($vt \approx 10^8$). Rappelons que les $N'_s + m$ produits matriciels fournissent l'ensemble de la réponse transitoire. Ceci est un avantage important de la méthode des PU vis-à-vis des méthodes qui ne fournissent la solution que pour une valeur de t à chaque fois. Bien sûr, les produits matriciels s'effectuent sur des matrices de moins en moins creuses et le stockage compact n'est pas utilisable pour la méthode des PU.

Pour introduire (sous une forme quelque peu simpliste) la comparaison des complexités temporelles, disons que la méthode UD nécessite de nombreux produits vecteur-matrice alors que la méthode des PU demande peu de produits matrice-matrice.

Comparons maintenant plus précisément la complexité temporelle des deux méthodes en fonction de la complexité spatiale (i.e., la cardinalité M de l'espace d'état de $\mathbf{X}$) et de vt. Deux cas sont envisageables suivant qu'un stockage compact est réalisé ou non pour la méthode UD. De plus, pour chacun des deux cas, on peut envisager, soit la recherche de la solution pour une seule valeur de t, soit la recherche de l'ensemble de la réponse transitoire. Si on considère la réponse transitoire, le temps CPU de calcul par la méthode UD pour les différentes valeurs fournies de t est la somme des temps CPU nécessaires au calcul des différents vecteurs $\Pi(t)$. En conséquence, on retient le choix défavorable pour la méthode PU vis-à-vis de la méthode UD en considérant uniquement l'obtention d'une seule valeur transitoire mais en comparant les deux méthodes suivant qu'un stockage compact est utilisé ou non. Dans un premier temps, on compare le nombre des opérations non linéaires intervenant dans les produits matriciels et dans les produits vecteur-matrice. Dans un deuxième temps, on donne quelques exemples de valeurs de temps CPU pour la méthode UD et pour la méthode des PU. Ces temps CPU sont obtenus par expérimentation lorsqu'ils sont modestes (quelques heures au plus); ils sont estimés pour les autres.

4.1 Stockage non compact du générateur A

Etudions d'abord la complexité temporelle de la méthode des PU. Pour simplifier, nous retenons encore la situation la plus pénalisante pour PU en considérant la valeur maximale de N_s' calculée par cette méthode. Cette valeur est égale à 9 et correspond à une valeur de vt_0 arrondie à 0.1. La matrice $P(t)$ s'obtient donc avec $(m+9)$ produits matrice-matrice, soit $(m+9)M^3$ Opérations Non Linéaires (ONL); de plus, M^2 ONL sont nécessaires pour effectuer le produit vecteur-matrice $\Pi(0)P(t)$ redonnant le vecteur $\Pi(t)$. Finalement, en désignant par $C_\Pi(PU, vt)$ le nombre d'ONL nécessaires, on a

$$C_\Pi(PU, vt) = (m+9)M^3 + M^2$$

Pour effectuer le même calcul par la méthode UD, nous devons faire N_s produits vecteur-matrice, soit un nombre d'ONL

$$C_\Pi(UD, vt) = N_s M^2$$

Dans ce cas où le stockage compact n'est pas utilisé, en désignant par R_{nc} le rapport des ONL (dans le sens UD sur PU), on obtient

$$R_{nc} = \frac{N_s M^2}{(m+9)M^3 + M^2} = \frac{N_s}{(m+9)M + 1}$$

Ce rapport constitue un critère de jugement de la supériorité de l'une des deux méthodes (vis-à-vis de la rapidité de calcul) *si les conséquences sur le temps de calcul des autres opérations sont négligeables devant le nombre des ONL pour chacune des deux méthodes.* Pour la méthode UD, cette dernière condition n'est pas toujours satisfaite car, lorsque vt est très grand, le temps de calcul de N_s (et de N_i) devient élevé. Comme il s'agit ici de prouver que la nouvelle méthode PU peut être meilleure que la précédente, nous allons une fois encore retenir la situation pénalisante pour PU et effectuer l'étude comparative en considérant cette condition satisfaite. En fait, ces choix successifs entraînent un rapport souvent très optimiste pour la méthode UD. En ce qui concerne la méthode des PU, le rapport considéré est tout à fait pessimiste; en effet, sachant N_s', l'entier m est fonction croissante (discontinue) du nombre maximal $M_{nz}(P^*)$ de termes non nuls dans les colonnes de la matrice P^*. Pour les systèmes informatiques, $M_{nz}(P^*)$ est généralement inférieur à M. Ici, on suppose aussi que $M_{nz}(P^*) = M$ (majoration). Bien entendu, le temps de calcul de N_s' est négligeable devant $C_\Pi(PU, vt)$.

D'après ce qui précède, le rapport R_{nc} des ONL est plus petit que le rapport réel des temps de calcul par les deux méthodes. Dans les tableaux 1.1 et 1.2 sont représentés

les valeurs du rapport R_{nc} pour $M = 2, ..., 2^{10}$ et $vt = 200, ..., 2^{19} \times 200 = 104857600$. Notons que pour $vt \geq 6553600$, N_s est supposé égal à vt. Sur ces tableaux, les limites des variations de R_{nc} autour de l'unité sont marquées par un trait gras.

104857600	1327311	651289	318716	155806	76149
52428800	680893	333941	163330	79800	38980
26214400	349525	171336	83752	40896	18490
13107200	179551	87968	42974	20972	10232
6553600	92304	45197	22066	10761	5247
3276800	47650	23323	11379	5546	2702
1638400	24679.4	12069.5	5884.4	2865.7	1395.4
819200	12768	6240	3040	1479.4	719.8
409600	6622.4	3234.2	1574.4	765.5	372.2
204800	3446	1681.6	817.9	397.4	193
102400	1800.6	878	426.5	207	100.5
51200	926.6	451.4	219.15	106.26	51.5
25600	487.5	237.25	115	55.7	27
12800	257.6	125.3	60.68	29.36	14.2
6400	137.7	68.87	32.35	15.64	7.56
3200	74.04	36	17.35	8.7	4.2
1600	40.76	19.75	9.53	4.6	2.3
800	23.2	11.22	5.4	2.71	1.3
400	13.3	6.41	3.08	1.54	0.74
200	8	4	1.85	0.92	0.44
vt	M=2	M=4	M=8	M=16	M=32

Tableau 1.1 : R_{nc} en fonction de M, $M = 2, ..., 2^5$ et de vt, $vt = 200, ..., 2^{19} \times 200$

104857600	37223	18201	8903.6	4357.3	2133.3
52428800	19044	9307	4551	2226	1089
26214400	9522	4654	2275	1113	545
13107200	4993	2497	1219	595	291
6553600	2559	1219	595.3	291	142.2
3276800	1317	642	313	153	75
1638400	679.6	331.1	161.5	78.8	38.4
819200	350.3	170.6	83.1	40.5	19.8
409600	181	88.07	42.9	20.9	10.2
204800	93.8	45.6	22.2	10.8	5.26
102400	48.8	23.7	11.5	5.6	2.7
51200	25	12.13	5.9	2.86	1.4
25600	13.08	6.3	3.07	1.5	0.72
12800	6.8	3.3	1.6	0.78	0.38
6400	3.56	1.76	0.85	0.4	0.2
3200	2.02	0.98	0.47	0.23	0.11
1600	1.1	0.5	0.26	0.12	0.06
800	0.62	0.3	0.15	0.07	0.034
400	0.35	0.17	0.082	0.04	0.02
200	0.21	0.1	0.05	0.023	0.011
vt	M=64	M=128	M=256	M=512	M=1024

Tableau 1.2 : R_{nc} en fonction de M, $M = 2^6, ..., 2^{10}$ et de vt, $vt = 200, ..., 2^{19} \times 200$

A titre d'exemple, ajoutons que, pour $M = 9$, $M_{nz}(P^*) = 3$ et $vt = 3276800$ ($v = 200$ et $t = 16384$), la méthode UD nécessite environ 3 heures CPU pour calculer le vecteur $\Pi(t)$ alors qu'on a besoin d'environ 0.6 secondes par les PU pour calculer ce même vecteur (en fait pour calculer l'ensemble de la réponse transitoire jusqu'à l'instant t).

4.2 Stockage compact de A

Le stockage compact de la matrice A permet de diminuer le nombre d'ONL nécessaires pour réaliser les produits vecteur-matrice de la méthode UD. Cela revient à effectuer $N_s\eta$ opérations où η est le nombre de termes non nuls dans le générateur A. Dans ce cas, en désignant par R_c le rapport des nombres d'ONL (de la méthode UD sur la méthode PU), on obtient

$$R_c = \frac{N_s \eta}{(m+9)M^3 + M^2}$$

Bien sûr, le nombre η dépend du système considéré. Parmi tous les systèmes envisageables, le cas le plus favorable pour la méthode UD semble être $\eta = 2M - 1$; c'est notamment le cas d'un processus de mort pur. On va donc calculer le rapport des complexités respectives en prenant ce cas limite. Rappelons que ce choix (optimiste pour la méthode UD) s'ajoute au fait que le temps de calcul de la troncature (dans la méthode UD) n'est pas pris en compte et au fait que c'est la valeur maximale de m qui est retenue pour la méthode PU. Le rapport R_c devient dans ce cas

$$R_c = \frac{(2M-1)N_s}{(m+9)M^3 + M^2} = \frac{2M-1}{M^2} R_{nc}$$

On trouve dans les tableaux 2.1 et 2.2 les valeurs du rapport R_c pour $M = 2, 2^2, ..., 2^{10}$ et $vt = 200, ..., 2^{19} \times 200$. On suppose toujours que pour $vt \geq 6553600$, vt est égal à N_s. Une telle valeur de N_s correspond dans la méthode UD à un temps de calcul de la troncature très élevé. Une comparaison rapide des tableaux 1.1 et 1.2 avec 2.1 et 2.2 nous permet de constater un diminution relative de la complexité temporelle de la méthode UD lorsque le stockage compact du générateur A est utilisé.

Malgré l'approximation pessimiste du rapport R_c pour la méthode PU et malgré l'utilisation du stockage compact de A, on observe que pour les CMTC raides, la méthode des PU reste plus rapide que la méthode UD. En effet, si la CMTC est raide, on cherchera à obtenir la solution pour une grande valeur de vt, ce qui correspond aux lignes supérieures des tableaux 2.1 et 2.2. Si l'on considère l'ensemble de la réponse transitoire, le rapport des temps de calcul est la somme des valeurs figurant dans une colonne donnée pour les valeurs de vt calculées. Par exemple pour $M = 1024$, si l'on cherche la réponse pour les valeurs de vt égales successivement à 200×2^{12}, 200×2^{13}, ..., 200×2^{18}, 200×2^{19}, le rapport des temps de calcul est égal à la somme $(0.04 + 0.07 + ... + 2.13 + 4.16) = 8.45$. Rappelons que pour calculer les rapports R_{nc} et R_c, on s'est placé dans le cadre de la recherche d'une valeur unique de t.

104857600	995483.25	284938.9	74699.06	18867.13	4684.95
52428800	510669.75	146099.3	38280.47	9663.28	2398.2
26214400	262143.75	74959.5	19629.4	4952.25	1137.57
13107200	134663.25	38486	10072.03	2539.57	629.5
6553600	69228	19737.7	5171.76	1303.08	322.8
3276800	35737.5	10203.8	2666.95	671.58	166.23
1638400	18509.5	5280.4	1379.15	347.01	85.85
819200	9576	2730	712.5	179.14	44.3
409600	4966.8	1414.96	369	92.7	22.9
204800	2584.5	735.7	191.7	48.12	11.87
102400	1350.45	384.12	99.96	25.06	6.18
51200	624.95	197.5	51.36	12.86	3.16
25600	365.6	103.8	26.95	6.74	1.66
12800	193.2	54.8	14.22	3.55	0.87
6400	103.27	30.13	7.58	1.91	0.46
3200	55.53	15.75	4.05	1.05	0.26
1600	30.57	8.64	2.23	0.55	0.14
800	17.4	4.9	1.26	0.33	0.08
400	9.97	2.8	0.72	0.19	0.045
200	6	1.75	0.43	0.11	0.027
vt	M=2	M=4	M=8	M=16	M=32

Tableau 2.1 : R_c en fonction de M ($\eta = 2M - 1$), $M = 2, ..., 2^5$ et de vt, $vt = 200, ..., 2^{19} \times 200$

104857600	1154.13	283.28	69.44	17	4.16
52428800	590.47	144.85	35.48	8.68	2.13
26214400	295.23	72.43	17.73	4.34	1.06
13107200	154.8	38.86	9.5	2.32	0.57
6553600	79.34	18.97	4.64	1.13	0.27
3276800	40.83	10	2.44	0.6	0.15
1638400	21.07	5.15	1.26	0.3	0.07
819200	10.86	2.65	0.64	0.16	0.04
409600	5.6	1.37	0.33	0.08	0.02
204800	2.9	0.7	0.17	0.042	10^{-2}
102400	1.51	0.37	0.09	0.02	5.3×10^{-3}
51200	0.77	0.18	0.046	0.04	2.7×10^{-3}
25600	0.4	0.01	0.024	5.8×10^{-3}	1.4×10^{-3}
12800	0.21	0.05	0.012	3.04×10^{-3}	7.4×10^{-4}
6400	0.11	0.027	0.066	1.5×10^{-3}	3.9×10^{-4}
3200	0.063	0.015	3.6×10^{-3}	8.9×10^{-4}	2.1×10^{-4}
1600	0.034	7.7×10^{-3}	2×10^{-3}	4.7×10^{-4}	1.2×10^{-4}
800	0.02	4.6×10^{-3}	1.1×10^{-3}	2.7×10^{-4}	6.6×10^{-5}
400	0.011	2.6×10^{-3}	6.4×10^{-4}	1.5×10^{-4}	3.9×10^{-5}
200	6.5×10^{-3}	1.5×10^{-3}	3.8×10^{-4}	8.9×10^{-5}	2.15×10^{-5}
vt	M=64	M=128	M=256	M=512	M=1024

Tableau 2.2 : R_c en fonction de M $(\eta = 2M - 1)$,$M = 2^6, ..., 2^{10}$ et de vt, $vt = 200, ..., 2^{19} \times 200$

4.3 Temps de calcul

Afin d'approcher le biais introduit dans les rapports calculés, quelques expérimentations ont été menées pour obtenir quelques valeurs réelles de rapports de temps CPU dans le cas d'un stockage compact de la matrice A; rappelons qu'elles sont faites en double précision sur VAX 8250.

Au niveau de la méthode UD, certains temps de calcul sont trop importants pour exécuter réellement le programme. Il s'agit donc d'une estimation fine réalisée grâce à la connaissance des complexités des différentes phases de la méthode et des valeurs des paramètres de l'exemple considéré.

L'exemple servant de base aux expériences est un système informatique constitué de K éléments actifs placés en parallèle et de K unités en réserve froide, il est présenté dans [4]. L'espace d'état de la CMTC modélisant ce système comporte $M = (K+1)^2$ éléments et $M_{nz}(P^*)$ est égal à 3. Dans le tableau 3, les valeurs de K considérées sont successivement 2, 8 et 16; M étant respectivement égal à 9, 81, 289.

Temps CPU

M	vt	PU	UD	$\frac{T(UD)}{T(PU)}$	R_c
9	13107200	0.59 s (c)	3 h (c)	18305.08	8720.7
9	104857600	0.7 s (c)	4 jours (e)	493714.3	64547.6
81	104857600	4.56 min (c)	12 jours (e)	3789.47	1065.13
289	104857600	3.5 h (c)	5 semaines (e)	240	88.61

Tableau 3 : Quelques valeurs calculées (c) ou estimées (e) du temps CPU

(c) : calculé

(e) : estimé

Le calcul du rapport $\frac{T(UD)}{T(PU)}$ des temps CPU (voir tableau 3) et du rapport R_c (pour l'exemple considéré) montre la supériorité du premier par rapport au second. Ceci signifie que la troncature dans la méthode UD nécessite un temps CPU non négligeable devant le nombre des ONL dans les produits vecteur-matrice et ce, malgré le stockage compact de la matrice A. En conséquence, le domaine de supériorité des PU montré sur les tableaux 1.1, 1.2 et 2.1, 2.2 est généralement étendu. Ce domaine sera encore élargi si l'on cherche à calculer la réponse transitoire pour plusieurs valeurs de t ($t = 2^k t_1$, $k = 0, 1, ..., k_{max}$ par exemple) puisque dans ce cas, le temps CPU mis par la méthode UD est la somme des temps CPU nécessaires aux calculs des différents vecteurs $\Pi(t)$. Pour la méthode des PU, il suffit d'exécuter le logiciel PUMAR une **seule** fois pour la valeur de t correspondant à k_{max} afin d'obtenir l'ensemble de la réponse transitoire.

5 Conclusion

Nous avons comparé dans ce rapport la complexité temporelle de notre méthode des Puissances Uniformisées à celle de l'Uniformisation Directe. En se plaçant dans le cadre de l'obtention d'une valeur transitoire unique, nous avons conclu que pour les CMTC raides ayant un espace d'état raisonnable, la méthode des PU est plus rapide que la

méthode UD et ce, malgré le stockage compact du générateur infinitésimal. L'utilisation de la méthode des PU devient impérative lorsqu'il s'agit de l'obtention de la réponse transitoire pour un ensemble de valeurs de t. Ainsi, pour un espace contenant 300 états, la méthode UD demanderait plus d'un mois de calcul pour obtenir l'ensemble de la réponse transitoire alors que la méthode des PU fournit ces résultats en quelques heures.

Bibliographie

[1] D. Gross and R. D. Miller. The Randomisation Technique as a Modeling Tool and Solution Procedure for Transient Markov Processes. *Operations Researchs*, Vol. 32(Num. 2):pp 344–361, 1984.

[2] C. Moler and C. Van Loan. Ninteen dubious ways to compute the exponential of a matrix. *SIAM*, Vol. 20(Num. 4):pp 801–836, 1978.

[3] W. K. Grassmann. Transient Solutions in Markovian Queueing Systems. *Computer and Operations Researchs*, (Num. 4):pp 47–56, 1977.

[4] H. Abdallah. *Construction d'un logiciel de calcul des éléments transitoires de chaînes de Markov à temps continu.* Thèse de doctorat de l'Université de Rennes I, juin 1989.

[5] H. Abdallah et R. Marie. Etude transitoire des modèles markoviens raides. $XXI^{èmes}$ *Journées de statistiques, Rennes*, Vol. I:pp 1–5, Mai 1989.

LISTE DES DERNIERES PUBLICATIONS INTERNES IRISA

PI 492 **SPARSE MATRIX MULTIPLICATION ON VECTOR COMPUTERS**
Jocelyne ERHEL
20 Pages, Septembre 1989.

PI 493 **THE SUPERIMPOSITION OF ESTELLE PROGRAMS : A TOOL FOR THE IMPLEMENTATION OF OBSERVATION AND CONTROL ALGORITHMS**
Benoît CAILLAUD
30 Pages, Septembre 1989.

PI 494 **EMPLOI DU TEMPS : PROBLEME MATHEMATIQUE OU PROBLEME POUR LA PROGRAMMATION EN LOGIQUE AVEC CONTRAINTES**
Xavier COUSIN
64 Pages, Septembre 1989.

PI 495 **NUMERICAL METHODS IN MARKOV CHAIN MODELING**
Bernard PHILIPPE, Youcef SAAD, William J. STEWART
46 Pages, Septembre 1989.

PI 496 **PLANIFICATION EN UNIVERS MONO ET MULTI-AGENTS (Définitions, concepts, objectifs, présentation d'un planificateur)**
Philippe PORTEJOIE
92 Pages, Octobre 1989.

PI 497 **LE TRAITEMENT D'EXCEPTIONS - ASPECTS THEORIQUES ET PRATIQUES**
Valérie ISSARNY
80 Pages, Octobre 1989.

PI 498 **IMPLEMENTATION D'UN LANGAGE DE PROGRAMMATION LOGIQUE D'ORDRE SUPERIEUR AVEC MALI**
Pascal BRISSET
28 Pages, Octobre 1989.

PI 499 **QUANTIFICATION IMAGE PAR LA METHODE DE LA VRAISEMBLANCE DU LIEN (A.V.L.) AVEC UN CODAGE PREORDONNANCE**
Israël-César LERMAN, Nadia GHAZZALI
60 Pages, Octobre 1989.

PI 500 **EFFICACITE DE LA METHODE DES PUISSANCES UNIFORMISEES POUR LES CHAINES DE MARKOV RAIDES**
Haîscam ABDALLAH, Raymond MARIE
18 Pages, Octobre 1989.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique