Modelling Markovian Queues and Similar Processes

Winfried Grassmann
Department of Computer Science
University of Saskatchewan

Queueing theory originated in 1909 when A. K. Erlang investigated queues as they arose in telephone network. His work was later incorporated into Operational Research. Unfortunately, the literature on queueing theory became increasingly mathematical, and this prevented its widespread use. However, this changed when people started applying queueing theory to computer performance evaluation. In these applications, it turned out that even relatively simple queueing models provided results that were in close agreement to actual observations. This led to a rapid development of queueing theory and its application in computer performance evaluation and communications. Indeed, queueing theory is now an integral part of computer performance evaluation[13]. Researchers in this area have pioneered many new methods, which were later applied successfully elsewhere, particularly in manufacturing situations[23]. There was also a resurgence of practical queueing theory[6] in more traditional areas of Operations Research, spearheaded by Peter Kolesar[14] and Richard Larson[15]. Due to these developments, queueing theory is very active today, and applications abound.

Besides providing a number of new theoretical results, the researchers of computer performance evaluation introduced a different approach to queueing theory: in particular, they emphasized the need for computer-based tools[19] that allow non-experts to use queueing theory effectively. There are, in particular, many applications programs in the area of so-called Generalized Stochastic Petri-nets (GSPNs) [1] and Stochastic Activity Nets (SANs) [22] that allow for graphics-based modelling.
GSPNs and SANs, like many other modelling tools in this area, are based on Markov chains, that is, all events, such as arrivals, service completions, changes of queues and so on, depend only on the present state of the system, and not on its past behaviour. This simplifies not only the mathematical treatment,

La modélisation des files d'attente markoviennes et processus analogues

Winfried Grassmann
Department of Computer Science
University of Saskatchewan

Les origines de la théorie des files d'attente remontent à 1909 à l'époque où A. K. Erlang en a posé les bases dans ses recherches sur le trafic téléphonique. Ses travaux ont par la suite été intégrés à la recherche opérationnelle. Malheureusement, les publications sur la théorie des files d'attente ont adopté un langage de plus en plus mathématique, ce qui a freiné l'utilisation de cette théorie. La situation a toutefois changé quand des gens ont commencé à appliquer la théorie des files d'attente à l'évaluation des performances. Pour ce type d'applications, il est apparu que même des modèles de files d'attente relativement simples fournissaient des résultats qui correspondaient de près aux observations réelles. On assista alors à une évolution rapide de la théorie des files d'attente qu'on appliqua alors à l'évaluation des performances des systèmes informatiques et aux communications. Aujourd'hui, la théorie des files d'attente fait partie intégrante de l'évaluation des performances des systèmes informatiques[13]. Les chercheurs oeuvrant dans cette branche d'activité ont élaboré plusieurs nouvelles méthodes qui ont ensuite été appliquées avec succès dans d'autres domaines, notamment dans le secteur de la fabrication[23]. On a aussi constaté une résurgence des applications pratiques[6]de la théorie des files d'attente dans des secteurs plus traditionnels de la recherche opérationnelle, un mouvement mené par Peter Kolesar[14] et Richard Larson[15]. Grâce à tous ces développements, la théorie des files d'attente est aujourd'hui largement utilisée et ses applications sont multiples.

En plus de fournir de nouveaux résultats théoriques, les chercheurs qui s'intéressent à l'évaluation des performances des systèmes informatiques ont proposé une approche différente de la théorie des files d'attente : ils ont notamment souligné le besoin de créer des outils informatiques[19] qui permettraient aux non-spécialistes d'utiliser la théorie des files d'attentes de manière efficace. Ainsi, on pense en particulier à de nombreux programmes d'application dans le domaine des réseaux de Petri stochastiques généralisés (GSPN) [1] et des réseaux d'activités stochastiques (SAN) [22] qui permettent une modélisation en mode graphique.

Les GSPN et les SAN, comme beaucoup d'autres outils de modélisation dans ce champ d'activité, se fondent sur les chaînes de Markov, c'est à dire que tous les

but also data collection because only averages are needed, such as average service times, average number of arrivals per time unit, and so on. In computer programs based on GSPNs and SANs, the user enters the system description in graphic form, and the program then automatically converts this description into a Markov chain. Traditional queueing theory is also based on Markov chains, but the transition matrices of these chains are usually given explicitly by means of mathematical symbols. The alternative, namely to describe the model by stating the different possible events, such as arrivals, changes of queues, departures, and so on, and mathematically describe their effect, is used only rarely. I think this is a mistake, because stating the events is much more straightforward, and the generation of the transition can then be left to a computer program. Designing such a program is not difficult. One merely enumerates all possible states, and applies the events to these different states[6].

The use of Markov chains is not as restrictive as it may seem. In fact, the progress of service times, interarrival times and the like can be described by introducing new discrete state variables, called phases. Another trick to make a non-Markovian system Markovian goes as follows: in some non-Markovian systems, there may be sequences of epochs that form a Markov chain. These epochs are called regeneration points, and the Markov chain formed by observing the system only at the regeneration points is called an imbedded Markov chain. For instance, if a system is Markovian, except for arrivals to the system, then the points immediately after an arrival form regeneration points, and the states of the system at the regeneration points form a discrete Markov process. As in all other Markovian systems, this imbedded Markov chain is determined by a transition matrix that must be generated before the system can be analyzed.

In many cases, the transition matrices of Markov chains are large, but sparse. The reason for this is that systems typically contain a number of different queues or other state variables, such as phases, customer types, priorities and similar things, and each

sur les chaînes de Markov, c'est-à-dire que tous les événements tels que les arrivées, les fins de service, les changements de files d'attente, etc. dépendent uniquement de l'état actuel du système et non de son comportement antérieur. Cela simplifie non seulement le traitement mathématique, mais aussi la collecte de données puisque seules les moyennes sont requises, par exemple une moyenne des délais de service, une moyenne des arrivées par unité de temps et ainsi de suite. Dans les programmes informatiques par GSPN et SAN, l'utilisateur saisit la description du système sous une forme graphique et le programme convertit alors automatiquement cette description en une chaîne de Markov. La théorie des files d'attente traditionnelle se fonde elle aussi sur les chaînes de Markov, mais les matrices de passage de ces chaînes sont généralement représentées de manière explicite par des symboles mathématiques. L'autre méthode, qui consiste à décrire le modèle en formulant les différents événements possibles, tels que les arrivées, les changements de files d'attente, les départs, etc., et en décrivant leur effet mathématiquement, est rarement utilisée. Je crois que cela constitue une erreur, puisque la formulation des événements se révèle une méthode beaucoup plus directe; il suffit ensuite d'avoir recours à un programme informatique pour la génération de la transition. La conception d'un tel programme n'est pas difficile. Il suffit d'énumérer tous les états possibles et d'appliquer les événements aux différents états[6].

L'utilisation des chaînes de Markov n'est pas aussi restrictive qu'il y paraît à première vue. En réalité, pour décrire l'évolution des délais de service, des intervalles entre les arrivées, etc., on peut utiliser de nouvelles variables d'état discrètes, appelées phases. Il existe une autre façon de faire d'un système non markovien un système markovien : dans certains systèmes non markoviens, il y a parfois des séquences de périodes qui forment une chaîne de Markov. Ces périodes portent le nom de points de regénération et la chaîne de Markov formée par l'observation du système aux points de régénération seulement est appelée une chaîne de Markov incluse. Par exemple, si un système est markovien, sauf en ce qui concerne les arrivées au système, alors les points se trouvant immédiatement après une arrivée constituent des points de régénération et les états du système aux points de régénération forment un processus de Markov discret. Comme dans tous les autres systèmes markoviens, cette chaîne de Markov incluse est déterminée par une matrice de passage qui doit être générée pour qu'on puisse analyser le système.

Dans bon nombre de cas, les matrices de passage des chaînes de Markov sont vastes, mais creuses. Cela est

combination of the state variables leads to a separate state. Hence, the number of states increases exponentially with the number of state variables. The number of events, on the other hand, tends to increase only with the square of the number of state variables, or even slower. This means that as the number of state variables increases, the transition matrices become sparser and sparser, and we conclude that systems with several state variables are huge, but sparse.

The methods used for analyzing huge, but sparse matrices are quite different from the ones used in more traditional queueing models. For instance, to find transient solutions of Markov chains, the traditional queueing theorist may suggest similarity transforms, which would require the evaluation of eigenvectors and eigenvalues. If the number of states is in the thousands or even millions, this is clearly unfeasible. Instead, the randomization method[6] is the method of choice for finding transient solutions in large Markov chains. Randomization also preserves sparsity, which is essential for the huge, but sparse matrices we are dealing with. Traditionally, steady-state equations are solved by using elimination methods, such as Gaussian elimination. For huge matrices, this is not only inefficient, but due to rounding errors, the results of Gaussian elimination are also inaccurate. Instead of Gaussian elimination (a direct method), so called iterative methods are used[12,21]. These methods start with some approximation, which is improved step-by-step until one is close enough to the true solution. Iterative methods have several advantages over direct methods, such as Gaussian elimination. They typically preserve sparsity, which, as indicated earlier, is essential. They are also very resistant to rounding errors since they work with approximations, and making additional errors in these approximations may slow convergence, but it seldom affects the precision of the final result. Finally, the number of floating point operations per iteration is typically of the order ***s***, where ***s*** is the number of non-zero entries in the transition matrix. It follows that if the number of iterations does not increase significantly with the matrix size, iterative methods have a lower computational complexity than direct methods, where the number of floating point

attribuable au fait que les systèmes contiennent habituellement différentes files d'attente ou autres variables d'état, telles que les phases, les catégories de clients, les priorités etc. et que chaque combinaison de ces variables d'état engendre un état distinct. Ainsi, le nombre d'états augmente de façon exponentielle en fonction du nombre de variables d'état. Le nombre d'événements, d'autre part, a tendance à augmenter seulement en fonction du carré du nombre de variables d'état, voire même plus lentement. Cela signifie que plus le nombre de variables d'état augmente, plus les matrices de passage deviennent creuses et c'est pourquoi nous disons que les systèmes comportant plusieurs variables d'état sont vastes, mais creux.

Les méthodes employées pour analyser des matrices vastes, mais creuses, sont assez différentes de celles utilisées dans les modèles de files d'attente plus traditionnels. Par exemple, pour trouver des solutions transitoires pour des chaînes de Markov, un partisan de la théorie des files d'attente traditionnelle pourra suggérer des transformées de similitude, pour lesquelles il faudrait évaluer les vecteurs propres et les valeurs propres. Si le nombre d'états atteint plusieurs milliers, voire des millions, cette méthode est absolument irréalisable. La randomisation[6] sera plutôt la méthode à privilégier pour trouver des solutions transitoires pour de grandes chaînes de Markov. La randomisation préserve aussi le caractère fondamental qui fait de ces matrices de grande taille, des matrices creuses. Habituellement, pour résoudre les équations stationnaires, on utilise des méthodes d'élimination, telles que l'élimination gaussienne. Pour les matrices d'une grande amplitude, non seulement cette méthode est inefficace, mais en raison des erreurs d'arrondi, les résultats de l'élimination gaussienne sont également inexacts. Au lieu de l'élimination gaussienne (une méthode directe), on utilise des méthodes dites itératives[12,21]. Ces méthodes ont comme point de départ une approximation, qu'on améliore progressivement, jusqu'à ce qu'on soit suffisamment près d'une vraie solution. Les méthodes itératives présentent plusieurs avantages par rapport aux méthodes directes telles que l'élimination gaussienne. Elle préserve le caractère creux des matrices qui, comme nous l'avons mentionné précédemment, est fondamental. Elles sont aussi très robustes aux erreurs d'arrondi puisqu'elles fonctionnent par approximations et les erreurs supplémentaires que peuvent entraîner ces approximations peuvent ralentir la convergence, mais elles affectent rarement la précision du résultat final. Enfin, le nombre d'opérations en virgule flottante par itération est généralement de l'ordre de ***s***, où ***s*** correspond au nombre d'entrées non nulles dans la matrice de passage. Il s'ensuit que si le nombre

methods, where the number of floating point operations increases with the third power of the matrix size.

In direct methods, the effect of rounding errors accumulates exponentially with the problem size. Hence, whereas rounding errors may not cause any problem when dealing with small models, they may prove disastrous in large models unless proper precautions are taken. One way of reducing rounding errors is to avoid subtractions. In fact, the randomization method for finding transient solutions in Markov chains does not contain any subtractions, and this makes the method very stable. It is even possible to solve the equilibrium equations without any subtractions. This is accomplished by using the so-called GTH algorithm[11], which is a variant of Gaussian elimination. As discussed already, rounding errors tend not to be a problem in iterative methods, which is one of the reasons to use them.

Queues are often unrestricted in size, which means that the resulting Markov chains are infinite. Under some restrictive conditions, it is possible to find steady-state solutions of infinite-state Markov chains. In particular, it is possible to find equilibrium solutions when the transition matrix is block-structured and the rows of blocks repeat form a certain point onward, provided they are shifted such that the same element always occupies the diagonal. Such transition matrices arise in Markovian systems where there is only one unbounded state variable, called the level, and, except for some initial states, the level does not affect the remaining state variables. The methods for solving the equilibrium equations with repeating rows fall into two groups: matrix analytic methods and methods based on eigenvalues.

Matrix analytic methods were introduced by Neuts. In these methods, the performance measures of the system are expressed in terms of a certain key matrix, which is determined by solving a non-linear matrix equation. Neuts distinguishes between transition matrices of GI/M/1 type[17] and matrices of M/G/1 type[18]. In matrices of GI/M/1 type, no transition can increase the level by more than one, but the level can

d'itérations n'augmente pas de manière significative en fonction de la taille de la matrice, la complexité numérique des méthodes itératives est moins grande que celle des méthodes directes, pour lesquelles le nombre d'opérations en virgule flottante augmente à la puissance trois de la taille de la matrice.

Dans les méthodes directes, l'effet des erreurs d'arrondi augmente de façon exponentielle en fonction de la taille du problème. Ainsi, alors que les erreurs d'arrondi peuvent ne causer aucun problème lorsqu'on utilise des petits modèles, elles ont parfois un effet désastreux dans des modèles de grande taille, si les précautions appropriées n'ont pas été prises. L'un des moyens de réduire les erreurs d'arrondi consiste à éviter les soustractions. En réalité, la méthode de randomisation permettant de trouver des solutions transitoires pour les chaînes de Markov ne contient aucune soustraction et c'est pourquoi elle constitue une méthode très stable. Il est même possible de résoudre des équations d'équilibre sans soustraction. Pour ce faire, on utilise l'algorithme dit GTH[11], qui est une variante de l'élimination gaussienne. Comme nous l'avons déjà souligné, les erreurs d'arrondi ne constituent généralement pas un problème dans les méthodes itératives, ce qui est l'une des raisons pour lesquelles nous les utilisons.

Souvent, les files d'attente ne comportent aucune contrainte de taille, ce qui signifie que les chaînes de Markov qui en résultent sont infinies. Dans des conditions de contraintes, il est possible de trouver des solutions stationnaires pour des chaînes de Markov en état infini. Plus spécifiquement, il est possible de trouver des solutions stationnaires quand la matrice de passage est en structure de bloc et que les rangs de blocs se répètent à partir d'un certain point, à condition qu'ils soient placés de telle manière que le même élément occupe toujours la diagonale. On retrouve de telles matrices de passage dans des systèmes markoviens où il n'y a qu'une variable d'état non bornée, appelée le niveau, et sauf pour certains états initiaux, le niveau n'affecte pas les variables d'état restantes. Les méthodes de résolution des équations d'équilibre comportant des rangs qui se répètent se répartissent en deux groupes : les méthodes d'analyse matricielle et les méthodes fondées sur les valeurs propres.

C'est Neuts qui a mis de l'avant les méthodes d'analyse matricielle. Dans ces méthodes, les mesures de performance du système sont exprimées sous la forme d'une matrice clé, qu'on établit en résolvant une équation matricielle non linéaire. Neuts fait une distinction entre les matrices de passage de type

level by more than one, but the level can decrease by any amount. In matrices of M/G/1 type, however, the level can increase by any amount, but it can decrease only by one at most. Matrices of GI/M/1 type lead to so-called matrix-geometric solutions, that is, there exists a matrix $\boldsymbol{R}$, and the probabilities of being in the different states of level $\boldsymbol{i}$ are given by the probabilities at level zero, multiplied by $\boldsymbol{R}^{\boldsymbol{i}}$. Grassmann and Heyman[9,24] generalized the paradigms of Neuts by considering matrices that can go up or down by any amount, and they solved these matrices by considering a generalization of Gaussian elimination to infinite-state matrices. As it turns out[10], this construct allows one to give a new meaning to many of the results derived by Neuts.

There are two approaches using eigenvalues to solve the equilibrium equations of Markov chains with repeating rows[5]: the generating functions approach and what might be called the quasi-difference equations approach. In the generating functions approach[4], a matrix-based generating function is inverted by using eigenvalues. The quasi-difference equations method[16] expresses the steady-state probabilities in terms of difference equations, except that the coefficients of these equations are matrices. The results obtained from either of these two methods are encouraging[4,16]. Moreover, methods based on eigenvalues have, in many cases, a computational complexity that is significantly lower than the one observed in matrix analytic methods[8]. In some cases of interest, only roots of characteristic functions are required, and these can be found efficiently, even if the polynomials involved are of a high degree[2,3].

So far, no good algorithms exist when more than one state variable has an infinite range. Of course, it is always possible to truncate all except one of these variables, a device used by Stanford and Grassmann[20]. This solution, however, is only partially satisfactory, and more research in this area is needed. This is one of the many problems in the area still waiting for a good solution.

GI/M/1[17] et les matrices de type M/G/1[18]. Dans les matrices de type GI/M/1, aucune transition ne peut augmenter le niveau de plus que un, mais la diminution du niveau peut prendre n'importe quelle valeur.. Dans les matrices de type M/G/1, cependant, l'augmentation du niveau peut prendre n'importe quelle valeur, mais le niveau ne peut descendre de plus de un. Les matrices de type GI/M/1 produisent des solutions dites matricielles-géométriques, c'est-à-dire qu'il existe une matrice $\boldsymbol{R}$, et les probabilités qu'elle soit dans les différents états du niveau $\boldsymbol{i}$ sont données par les probabilités au niveau zéro, multipliées par $\boldsymbol{R}^{\boldsymbol{i}}$. Grassmann et Heyman[9,24] ont généralisé les paradigmes de Neuts : ils ont considéré les matrices qui peuvent croître ou décroître de n'importe quelle valeur et ils ont résolu ces matrices en procédant à une généralisation de l'élimination gaussienne pour des matrices en état infini. Le concept obtenu[10] permet de donner une nouvelle signification à beaucoup des résultats présentés par Neuts.

Il existe deux approches qui utilisent les valeurs propres pour résoudre les équations d'équilibre des chaînes de Markov contenant des rangs répétés[5]: l'approche des fonctions génératrices et ce qu'on pourrait appeler l'approche des équations quasi-différentielles. Dans l'approche des fonctions génératrices[4], on inverse une fonction génératrice fondée sur une matrice en utilisant des valeurs propres. La méthode des équations quasi-différentielles[16] exprime les probabilités stationnaires sous la forme d'équations différentielles, sauf que les coefficients de ces équations sont des matrices. Les résultats obtenus avec l'une ou l'autre de ces méthodes sont encourageants[4,16]. En outre, la complexité arithmétique des méthodes fondées sur les valeurs propres est nettement inférieure à celle observée dans les méthodes d'analyse matricielle[8]. Dans certains cas, seules les racines des fonctions caractéristiques sont requises et on peut les trouver efficacement, même si les polynômes en présence sont d'un haut niveau[2,3].

Jusqu'à maintenant, il n'existe toujours pas de bon algorithme pour les cas où plus d'une variable d'état a un horizon infini. Bien sûr, il est possible de tronquer toutes ces variables à l'exception d'une seule, un mécanisme utilisé par Stanford et Grassmann[20]. Cette solution, cependant, n'est que partiellement satisfaisante et il faudra poursuivre les recherches dans ce secteur. Ce n'est là qu'un des nombreux problèmes pour lesquels il reste à trouver une bonne solution.

References / Références

1. M. Ajmoni Marsan, G. Balbo, G. Conte, S. Donatelli and D. Franceschinis. 1995. Modelling with Generalized Stochastic Petri Nets. John Wiley & Sons, New York.
2. M. L. Chaudhry. 1992. QPACK Software Package. A&A Publications, 395 Carrie Crescent, Kingston, Ontario.
3. M. L. Chaudhry. 1992. QROOT Software Package. A&A Publications, 395 Carrie Crescent, Kingston, Ontario.
4. J. N. Daigle and D. M. Lucantoni, 1991. Queueing Systems Having Phase-Dependent Arrival and Service Rates. In: W. J. Stewart, editor. 1991. Numerical Solutions of Markov Chains, Marcel Dekker, New York, pages 179-215.
5. H. R. Gail, S. L. Hantler and B. A. Taylor, 1999. Use of Characteristic Roots for Solving Infinite State Markov Chains. In: W. K. Grassmann, editor. Computational Probability, Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, MA 02061, pages 205-254.
6. W. Grassmann, 1991. Finding Transient Solutions in Markovian Event Systems Through Randomization. In: W. J. Stewart, editor. Numerical Solutions of Markov Chains, Marcel Dekker, New York, pages 357-371.
7. W. K. Grassmann, 1988. Finding the Right Number of Servers in Real World Queuing Systems. Interfaces 18(2): 94-104.
8. W. K. Grassmann and S. Drekic, 1999. An Analytic Solution of a Tandem Queue with Blocking. Preprint.
9. W. K. Grassmann and D. P. Heyman, 1990. Equilibrium Distributions of Markov Chains with Repeating Rows, Journal of Applied Probability 27: 557-576.
10. W. K. Grassmann and D. A. Stanford, 1999. Matrix Analytic Models. In: W. K. Grassmann, editor. Computational Probability, Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, MA 02061, pages 153-203.
11. W. K. Grassmann, M. I. Taksar and D.P. Heyman, 1985. Regenerative Analysis and Steady State Distributions for Markov Chains, Operations Research 33(5): 1107-1116.
12. Greenbaum. 1997. Iterative Methods for Solving Linear Systems, SIAM, Philadelphia.
13. H. Kobayashi, 1978. Modelling and Analysis: An Introduction to System Performance Evaluation Methodology, The Systems Programming Series, Addison Wesley, Reading, MA.
14. P. Kolesar, 1979. A Quick and Dirty Response to a Quick and Dirty Crowd: Particularly to Jack Byrd's "The Value of Queueing Theory". Interfaces 9(2): 77-82.
15. R. Larson, 1987. Perspectives on queues: Social Justice and the Psychology of Queueing, Operations Research 35(6): 858-905.
16. I. Mitrani and R. Chakka, 1995. Spectral Expansion Solution for a class of Markov Models, Applications and Comparisons with Matrix-Geometric Methods, Performance Evaluation 23(3): 241-260.
17. M. F. Neuts. 1981. Matrix-Geometric Solutions in Stochastic Models, An Algorithmic Appraoch, Johns Hopkins University Press, Baltimore.
18. M. F. Neuts. 1989. Structured Stochastic Matrices of M/G/1 Type and Their Applications. Marcel Dekker, New York.
19. R. A. Sahner, K. S. Trivedi and A. Puliafito. 1995. Performance and Reliability Analysis of Computer Systems Using the SHARPE Software Package. Kluwer Academic Publishers, Dordrecht, The Netherlands.

20. D. A. Stanford and W. K. Grassmann, 1993. The Bilingual Server Model: A Queueing Model Featuring Fully and Partially Qualified Servers. INFOR 31(4): 261-277.
21. W. J. Stewart. 1994. An Introduction to the Numerical Solution of Markov Chains. Princeton University Press, New Jersey.
22. UltraSAN User's Manual. 1995. Center for Reliable High-Performance Computing, Coordinated Science Library, University of Illinois at Urbana-Champaign.
23. N. Viswanadhan and Y. Nahari, 1992. Performance Modelling of Automatic Manufacturing Systems. Prentice Hall, Englewood Cliffs.
24. Y. Q. Zhao, W. Li and W. Braun, 1998. Infinite Block-Structured Transition Matrices and Their Properties. Adv. Appl. Prob. 30: 365-384.