Ants, Agents and Product Form

Nigel Thomas[1] Jeremy T. Bradley[2]
William J. Knottenbelt[2]

1. School of Computer Science, University of Newcastle,
Newcastle-upon-Tyne, NE1 7RU
Email: *nigel.thomas@ncl.ac.uk*
2. Department of Computing, Imperial College London
180 Queen's Gate, London SW7 2BZ, United Kingdom
Email: `{jb,wjk}@doc.ic.ac.uk`

Abstract

In this paper we consider a form of model interaction using pair wise cooperation accross a large number of small identical components. This style of model is derived from earlier approaches to modelling insect colonies. Such models have the feature of a potentially very large state space, despite thier specifications being concise. To address this problem we show that some models of this type can be solved using a product form.

1 Introduction

The analysis of very large systems that are composed of many interacting agents is a difficult and practically important problem in many diverse fields:

- **Biology**: the dynamics of insect colonies [10, 11] and spread of infection
- **Sociology**: population growth and crowd dynamics [8, 9]
- **Economics**: fluctuations of financial markets as a result of individual trader behaviour
- **Computer Science**: diffusion of computer worms and viruses [6, 7], and dynamic load modelling in computational GRIDs

All of these models share the property that their overall state space, or the number of possible behaviours, is huge, far greater than the feasible limit of direct analytical study. This means that such models have traditionally been tackled using simulation. However, for large models, even long-running simulations often suffer from low state-space coverage which in turn leads to problems with stability and accuracy of performance metrics.

One potential breakthrough in the area of massive concurrent and probabilistic system modelling is the observation that, in many cases, the large state space is due to the combination of a large number of simple components. That is, each entity in the system can be accurately modelled by an agent with a few distinct modes of behaviour. However, when a large number of these simple agents are combined, the overall effect is a very complex interconnected system with emergent behaviour.

It is a further feature of these systems that there are very few types of entity. Thus the models contain very few distinct types of agent, but many instances of each type. This simple structure allows models to be specified relatively easily. Using a compositional process algebra, a model of many millions of states may be specified in a few tens of lines or less. This succinctness of specification is very attractive, but cannot as yet be leveraged in current analytical solution methods.

The aim of this paper is to exploit the structure of this class of model specification to provide a means to analyse the entire model compositionally. That is, global properties can be found by studying individual components and applying those local results across the system. In order to do this it is essential that in studying the behaviour of individual components they are observed to behave in a way which is consistent with their being part of a large interconnected complex system. Clearly this is not simple.

The basic philosophy behind the argument presented here derives in concept from a form of analysis that has been present in computer performance evaluation for many years. A product form result is formed if the steady state probability of being in a given global state can be found by forming the product of the steady state probabilities found from the individual components in the system. A great many product form results exist, particularly in queueing theory, indeed their existence is one of the reasons for queueing theory's popularity. The fundamental difference in the proposal here is that individual queues tend to be very large, possibly unbounded (infinitely many states) and systems tend to consist of only a small number of queues; tens at most. However, without a product form result, a model consisting of only two or three queues could be intractable.

In recent years a number of product form results have been applied to stochastic process algebra [3], such as PEPA [2]. Amongst these is the well known Boucherie product form [1]. This result relies on two or more processes only interacting by competing for resources, possession of a resource by one process imposes restrictions on the behaviour of the other processes. The first published characterisation of this result in PEPA [5] relies on the competition between processes to regulated by an explicitly defined resource. In practice this is actually not required, as the competition should be over behaviours rather than entities. For example, if an agent A observes another agent B acting in a particular way, it may affect their subsequent behaviour in some way. This is the approach which has been taken subsequently and enables decomposition over very many instances of components subject to the behavioural constraints specified [13].

2 Biologically inspired modelling techniques

Agent-based modelling of massively parallel systems using a process algebra is exemplified by Sumpter et al [10, 11]. Sumpter has studied a series of models of insect colony behaviour which have added insight into how simple individual behaviours can combine to give the appearance of apparently very sophisticated communal behaviour. The models correctly predict emergent behaviours in large ant and bee colonies using composed agent models of individual insects which move probabilistically from one behavioural mode to another.

Consider the simplest example given in [10], specified in the discrete time probabilistic process algebra WSCCS [16].

$$\begin{aligned} \mathit{Active} &\equiv p : \surd.\mathit{Passive} + q : \surd.\mathit{Active} \\ \mathit{Passive} &\equiv 1 : \surd.\mathit{Passive} \end{aligned}$$

$$\mathit{Colony}_n(i) \equiv \underbrace{\mathit{Active} \times \cdots \times \mathit{Active}}_{i} \times \underbrace{\mathit{Passive} \times \cdots \times \mathit{Passive}}_{n-i}$$

At any time the colony will consist of i *Active* ants and $n - i$ *Passive* ants. In order to analyse such a model Sumpter [10] constructs random variables which count agents in meaningful states (in this instance, `active` and `passive`).

By using the underlying process algebra description, it is possible to derive a formula for the single-step evolution of the mean value of the count representing a single-step evolution in the model. The evolution of the mean-value as the model ages gives useful insight into the global observed behaviour of models of 10^{100} states and upwards. This would generally be well beyond the capability of traditional global state-space exploration techniques that would not be able to enumerate a state-space of that size let along perform any form of stochastic analysis.

3 Simple continuous time models of insect colonies

Consider the following very simple example.

$$\begin{aligned} \mathit{Active} &\overset{\mathit{def}}{=} (\mathit{sleep}, r_1).\mathit{Passive} + \\ &\phantom{\overset{\mathit{def}}{=}} (\overline{\mathit{sleep}}, \top).\mathit{Active} \\ \mathit{Passive} &\overset{\mathit{def}}{=} (\mathit{wakeup}, r_2).\mathit{Active} \end{aligned}$$

$$\prod^{n} \mathit{Active} \underset{\{sleep\}}{\bowtie}$$

Model 1: Active and Passive ants

This example uses a modified PEPA syntax, PEPA+, to allow pair wise cooperation (as in CCS) as defined in [4], although the exact notation used takes elements from CCS for clarity. In this model any agent may become passive, as long as at least one any remains active; a condition enforced by the cooperation of an input $\overline{\mathit{sleep}}$ action with an output *sleep* action. By the characterisation given in [13] this model has a product form over the components whereby the steady state probability of there being i *Active* agents and $n-i$ *Passive* agents is given by $\pi^{i}_{Active}.\pi^{n-i}_{Passive}/B$, where π_{Active} and $\pi_{Passive}$ are the steady state probabilities for a single component studied in isolation and B is a normalising constant ($B = 1 - \pi^{n}_{Passive}$). Thus, in this example, the complexity of the steady state solution is reduced from naively solving for $2^n - 1$ states, to merely 2 states.

The approach taken here is based on the notion of behavioural independence [14]. This is a simple yet powerful mechanism for identifying and relating possible decomposition approaches for a particular model. The inverse property, control, has also been used to form reduced solutions by substituting simpler versions of components into a model and solving iteratively to give a very good approximate results [15].

Clearly the example presented here is very simple, but it is evident that this model is structurally very similar to the WSCCS models studied by Sumpter [10], with the major difference that our model uses continuous time. Thus we have the potential to model much more complex behaviour and potentially identify decomposed solution in a similar way. In this analysis we never derive the entire state space but only need to consider the relatively small number of individual modes of behaviour.

It is important to emphasise that identifying a decomposed solution is not only a means of performing some numerical calculation on the model, but it is also an aid in behavioural analysis and system comprehension. Identifying where a component is influenced and in what way is an important means of understanding how global behaviours emerge. In addition it is our hypothesis that the existence of certain decomposed solutions may be good indicators of the accuracy of other analysis approaches, such as the mean difference analysis used by Sumpter [12].

A number of other possible bahavious do not affect the conditions for product form. For example, ants may forage away from the colony, either individually or in pairs.

$$\begin{array}{rcl}
Active & \overset{def}{=} & (goOut, r_3).Forage + (\overline{goOut}, \top).Active \\
 & & +(sleep, r_1).Passive + (\overline{sleep}, \top).Active \\
Forage & \overset{def}{=} & (goBack, r_4).Active \\
Passive & \overset{def}{=} & (wakeup, r_2).Active
\end{array}$$

$$\prod^{n} Active \underset{\{sleep, goOut\}}{\bowtie}$$

Model 2: Individually foraging ants

$$\begin{array}{rcl}
Active & \overset{def}{=} & (goOut, r_3).Forage + (\overline{goOut}, \top).Active \\
 & & +(goOut2, r_3).Forage + (\overline{goOut2}, \top).Forage \\
 & & +(sleep, r_1).Passive + (\overline{sleep}, \top).Active \\
Forage & \overset{def}{=} & (goBack, r_4).Active \\
Passive & \overset{def}{=} & (wakeup, r_2).Active
\end{array}$$

$$\prod^{n} Active \underset{\{sleep, goOut\}}{\bowtie}$$

Model 3: Ants foraging in pairs, or individually, returning individually

It might appear that in the case of ants foraging in pairs that the $goOut2$ action is not needed and these two terms could be replaced with $(\overline{goOut}, \top).Forage$. However, such a specification would not have a product form as there would then be two ways to reach the same state, i.e. when $N = 2$ there would be two transitions from the state where both are active to the state where both are foraging, thus the rate of moving from $Active|Active$ to $Forage|Forage$ would be twice that of going from $Active|Active$ to $Active|Forage$.

It should be emphasised that the conditions for product form are fragile. It is very easy to construct models which appear to be similar but do not satisfy the same criteria for a product from solution to exist. Take for example, a recruiting ants model based on that of Sumpter, which requires one ant to wake up another.

$$
\begin{aligned}
\mathit{Active} &\overset{\mathit{def}}{=} (\mathit{sleep}, r_1).\mathit{Passive} + (\overline{\mathit{sleep}}, \top).\mathit{Active} + (\mathit{recruit}, r_3).\mathit{Recruit} \\
\mathit{Recruit} &\overset{\mathit{def}}{=} (\mathit{sleep}, r_1).\mathit{Passive} + (\overline{\mathit{sleep}}, \top).\mathit{Recruit} \\
&\quad + (\mathit{wakeup}, r_2).\mathit{Recruit} + (\mathit{return}, r_4).\mathit{Active} \\
\mathit{Passive} &\overset{\mathit{def}}{=} (\overline{\mathit{wakeup}}, \top).\mathit{Active}
\end{aligned}
$$

$$\prod^{n} \mathit{Active} \underset{\{sleep, wakeup\}}{\bowtie}$$

Model 4: Recruiting ants

Although this model does not have a product form it does share the symmetry common to all the PEPA models presented here. Hence the model may be solved more efficiently by folding the state space, although for large N the model may still be computationally intensive to solve.

4 Conclusions and fruther work

In this paper a number of models are presented, some of which have a product form solution. These models share a property we have termed *massively parallel*, in that they are specified by many instance of the same component in parallel. As such the state space of the underlying Markov chains is huge and so identifying product forms is of considerable benefit.

These models represent a subset of those we are studying. Further, more complex examples appear in crowd dynamics and computer virus modelling, which are the subject of continuing work.

References

[1] R. Boucherie,A Characterisation of Independence for Competing Markov Chains with Applications to Stochastic Petri nets,*IEEE Transactions on Software Engineering* **20**(7) (1994) 536–544.

[2] J. Hillston, *A Compositional Approach to Performance Modelling*, vol. 12 of *Distinguished Dissertations in Computer Science.* Cambridge University Press, 1996. ISBN 0 521 57189 8.

[3] J. Hillston, "Exploiting structure in solution: Decomposing composed models," in *Lectures on Formal Methods and Performance Analysis*, no. 2090 in LNCS, pp. 278–314, Springer-Verlag, 2001.

[4] J. Hillston, L. Recalde, M. Ribaudo, and M. Silva, “A comparison of the expressiveness of SPA and bounded SPN models,” in *Proceedings the 9th International Workshop on Petri Nets and Performance Models* (B. Haverkort and R. German, eds.), IEEE Computer Society Press, Aachen, September 2001.

[5] J. Hillston and N. Thomas, “Product form solution for a class of PEPA models,” *Performance Evaluation*, vol. 35, pp. 171–192, May 1999.

[6] M. Liljenstam, Y. Yuan, B. J. Premore, and D. Nicol, “A mixed abstraction level simulation model of large-scale internet worm infestations,” in *MASCOTS’02, Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, IEEE Computer Society Press, October 2002.

[7] D. Nicol, M. Liljenstam, and J. Liu, “Multiscale modeling and simulation of worm effects on the internet routing infrastructure,” in *TOOLS’03, Proceedings of Computer Performance Evaluation: Modelling Techniques and Tools*, Lecture Notes in Computer Science, (University of Illinois at Urbana-Champaign), Springer-Verlag, September 2003.

[8] R. Gomez and J. E. Rowe, “An agent-based simulation of crowd movement,” in *UKSIM’03, Conference on Computer Simulation 2003*, (Cambridge), UKSIM, April 2003.

[9] J. E. Rowe and R. Gomez, “El Botellón: Modelling the movement of crowds in a city,” *Complex Systems*, vol. 14, pp. 363–370, 2003.

[10] D. J. T. Sumpter, *From Bee to Society: An Agent-based Investigation of Honey Bee Colonies.* PhD thesis, Department of Mathematics, University of Manchester Institute of Science and Technology, March 2000.

[11] D. J. T. Sumpter and D. S. Broomhead, “Relating individual behaviour to population dynamics,” *Proceedings of the Royal Society: Series B*, vol. 268, pp. 925–932, 2001.

[12] D. J. T. Sumpter, “Relating individual behaviour to population dynamics,” *Performance Evaluation*, 2003.

[13] N. Thomas, “Product form over on-off components in PEPA,” in *ESM’03, Proceedings of 17th European Simulation Multiconference*, (Nottingham), pp. 245–250, ESM, June 2003.

[14] N. Thomas, “Behavioural independence and control in PEPA,” in *PASTA’02, Proceedings of Process Algebra and Stochastically Timed Activities* (S. Gilmore, ed.), (Edinburgh), June 2002.

[15] N. Thomas, J. T. Bradley, and D. J. Thornley, “An approximate solution of PEPA models using component substitution,” *IEE Journal of Computers and Digital Techniques*, vol. 150, pp. 67–74, March 2003.

[16] C. Tofts, “Processes with probabilities, priorities and time,” *Formal Aspects of Computing*, vol. 6, no. 5, pp. 536–564, 1994.