Semi-Markov PEPA: Compositional Modelling and Analysis with Generally Distributed Actions

Jeremy T. Bradley

Department of Computing, Imperial College London,
Huxley Building, 180 Queen's Gate, London SW7 2BZ. Email: jb@doc.ic.ac.uk

Abstract. We present a conservative extension to the popular Markovian process algebra, PEPA, which incorporates generally distributed sojourn-times for action duration. Just as a PEPA model generates a Markov chain for analysis purposes, so semi-Markov PEPA produces a semi-Markov chain. We discuss how semi-Markov PEPA models are analysed through Knottenbelt's semi-Markov DNAmaca analyser and present a small example for passage time analysis.

1 Introduction

PEPA [1] is a Markovian process algebra that is easy to use and has a wide user community [2]. It is perhaps unique among stochastic process algebras for this. It is a common request from PEPA users that the type of distribution that PEPA uses could be extended [3]. PEPA is designed to be a Markovian process algebra which means that the duration of action events is defined by a sample from a random variable with an exponential distribution. What we have sought to do here is allow generally distributed actions in a PEPA framework but in a restricted (or semi-Markov) way that still allows for the production of analytic results.

The question might reasonably be asked: why not simply replace the exponential distribution in PEPA with a general distribution? The reason for not doing this is that it generates an underlying Generalised Semi-Markov Process (GSMP) which is analytically intractable and can only be studied through simulation. Stochastic process algebras which follow this approach include originally Strulo and Harrison's SPADES [4], followed by D'Argenio's ♤ [5, 6], Bravetti et al's GSMPA [7] and Bravetti's iGSMPA [8].

While some SPAs have become *feature rich* and have intricate syntaxes to support such features, PEPA has successfully maintained a very small set of base operators. Thus, any attempt to modify or extend PEPA should hold to the original design qualities, especially that of syntactic simplicity. A further requirement is that there should be an easy recovery of the existing body of work—it

would be easy to produce a PEPA-with-extra-distributions which could not take advantage of the many useful results that Markovian PEPA has given rise to.

So our key design factors for semi-Markov PEPA or SM-PEPA are that it should:

- have PEPA as a subset
- continue to have as simple a syntax as possible
- have an analytically tractable stochastic model

In section 2, we look at what modelling advantages and applications we get from using a semi-Markov model. Section 3 defines the syntax for the extended process algebra. Finally, in Section 4, we illustrate the analysis of a semi-Markov PEPA model with a simple voting system.

2 Applications of Semi-Markov Models

Semi-Markov processes [9] seem to be a low-level model searching for a high-level formalism. Markov models mapped particularly well onto concurrency-modelling paradigms, such as Petri nets and process algebras, largely due to the *memoryless* nature of the exponential distribution. This property allows for the construction of an *expansion law* for concurrent Markov processes through an interleaving semantics.

Semi-Markov models have no such obvious mapping. SMPs have no easy interleaving semantics[1] that would allow the easy expression of the residual distribution for an interrupted action.

In order to define large SMPs, you do have to abstract yourself away from the transition matrix level. To this end, we invented SM-SPN—or semi-Markov Stochastic Petri Nets [10], an instantiation of ESPN [11]. In an SMP, for the process to select one of its possible successor states, a probabilistic choice is made and an arbitrary random delay is observed. In the same way, in an SM-SPN, a probabilistic choice is made between concurrently enabled transitions (based on pre-assigned weights) and an arbitrary random firing delay is observed. There is no concurrent running of transition clocks—only a single transition clock runs at a time.

To the best of our knowledge there is no equivalent semi-Markov style of stochastic process algebra although initial ideas behind adding general distributions to SPAs have been discussed in, for instance [12]. Also an initial report on an early SM-PEPA design was produced as [13].

A possible argument for a lack of a semi-Markov SPA is: if only a single clock or distribution can be selected to run at a time in a semi-Markov model, what

[1] SMPs that run concurrently are GSMPs and stochastic process algebra semantics do exist for GSMPs but they are not interleaving.

application can it have for modelling concurrent processes. We suggest two applications:

1. with a concurrent design but a single-threaded architecture on which to run it, e.g. a Java Runtime Environment on any single processor machine. The machine must time-slice between the different threads of the program: it must select a thread to run and then complete a certain amount of work on that thread before getting the chance to select a new thread to execute. This is ideally suited to the transition-selection style of semi-Markov processes; the selecting of a new thread being the probabilistic selection of a particular distribution.
2. for modelling mutual exclusion, e.g. failure–recovery modes, where normal concurrent operation is suspended and a single recovery process is repairing the system. This recognises the fact that the modeller will still want to have proper Markovian concurrency at their disposal. Markov processes are a strict subset of SMPs and thus it is possible to let the modeller have true concurrency when there are no general distributions enabled.

3 Semi-Markov PEPA

Our modifications to the syntax of PEPA are necessarily subtle in order to keep to our design criteria of having PEPA as an included subset. We keep unchanged choice $+$, cooperation $\underset{L}{\bowtie}$, hiding $/$ and the constant definition A.

Although we will need to address changes in semantic meaning for these operators for semi-Markov PEPA, our focus in this section lies in the change to the prefix operator $(a.D).P$

The syntax for a semi-Markov PEPA is as follows:

$$P ::= (a, D).P \mid P + P \mid P \underset{L}{\bowtie} P \mid P/L \mid A \tag{1}$$

where:

$$D ::= \lambda \mid S \tag{2}$$

$$S ::= \top \mid \omega : L(s) \tag{3}$$

where λ is the standard PEPA exponential rate parameter, $\lambda \in \mathbb{R}^+ \cup \{n\top \mid n \in \mathbb{Q}, n > 0\}$.

Here D is an arbitrary delay parameter for the action a, which can either take on the usual exponentially distributed delay or a weight, ω, and a generally distributed delay, defined by the Laplace transform, $L(s)$.

We also annotate the standard a-action with an optional $*$, to indicate whether the action should be treated as semi-Markovian or not. If it is not semi-Markovian then it can only inherit the λ delay parameter.

We now prioritise the $*$ actions by saying that if, at any stage, a $*$ action can proceed then all Markovian actions are disabled by definition. If more than one $*$ action can proceed then the weight ω is used to form a probabilistic choice between them. Finally, if only Markovian transitions are enabled then standard PEPA semantic rules apply.

A formal semantics for a semi-Markov PEPA is presented at the end of the paper. Figure 4 preserves the standard Markovian PEPA behaviour, while Figure 5 adds those rules necessary to cope with the semi-Markov extensions to the formalism. The apparent rate function, $r_a(P)$, in Figure 4 is defined by:

$$r_a(P) = \sum_{P \xrightarrow{(a,\lambda_i)}} \lambda_i \tag{4}$$

One interesting design decision is that we have forbidden anything other than passive synchronisation on generally distributed transitions. Although perfectly possible to define a maximum distribution for two synchronising generally distributed actions (representing a last-to-finish style of synchronisation [12]), there is no closed from representation of the necessary Laplace transform for this (the product of cumulative distribution functions) so it would break our analytic tractability requirement to allow it.

4 SM-PEPA Model Analysis

Figure 1 describes a semi-Markov PEPA example of a simple voting model with three voters and two polers. Note that the semi-Markov aspect of the system is derived from the gamma distribution attached to the mutually exclusive recover_all* action. This is an example of the second type of semi-Markov system as defined in Section 2.

The voter model is expanded to its global state space using the expansion rules of Figures 4 and 5. This is translated into an underlying semi-Markov chain and from there passage time analysis can be performed. The expansion and analysis process is done mechanically by DNAmaca after ipc (the Imperial PEPA Compiler) [14] has compiled the semi-Markov PEPA description into a semi-Markov stochastic Petri net (the input language for DNAmaca).

Figures 2 and 3 show two passage results from DNAmaca for the voter model. They show the passage time from the first observed vote action to the first full system recover, or recover_all* action.

From Figure 3, we can get specific quantile information (useful for probabilistic quality-of-service guarantees) e.g. the probability that the passage from first vote action to the first full system recover has taken place by time, $t = 27.5$, is 60.8%.

A voter can cast a vote and then pause before becoming a voter again.

$$\mathrm{Voter} \stackrel{def}{=} (\mathrm{vote}, \lambda).(\mathrm{pause}, \mu).\mathrm{Voter}$$

A poler unit can receive a vote or break; if it breaks it can recover or wait for a global recover_all*.

$$\mathrm{Poler} \stackrel{def}{=} (\mathrm{vote}, \top).(\mathrm{register}, \gamma).\mathrm{Poler} + (\mathrm{break}, \nu).\mathrm{Poler_broken}$$
$$\mathrm{Poler_broken} \stackrel{def}{=} (\mathrm{recover}, \tau).\mathrm{Poler} + (\mathrm{recover_all}^*, \top).\mathrm{Poler}$$

The Poler_group is used to manage many parallel poler units. If they all break then the Poler_group_2 can initiate a hight priority semi-Markov recover_all* action that can repair all of the poler units simultaneously—essentially a system reset.

$$\mathrm{Poler_group_0} \stackrel{def}{=} (\mathrm{break}, \top).\mathrm{Poler_group_1}$$
$$\mathrm{Poler_group_1} \stackrel{def}{=} (\mathrm{break}, \top).\mathrm{Poler_group_2} + (\mathrm{recover}, \top).\mathrm{Poler_group_0}$$
$$\mathrm{Poler_group_2} \stackrel{def}{=} (\mathrm{recover_all}^*, 1 : \mathtt{gamma}(\delta, \mathtt{k}, \mathtt{s})).\mathrm{Poler_group_0}$$

The system definition combines 3 voters with 2 poling units.

$$\mathrm{System} \stackrel{def}{=} (\mathrm{Voter} \parallel \mathrm{Voter} \parallel \mathrm{Voter}) \underset{\{\mathrm{vote}\}}{\bowtie} ((\mathrm{Poler} \underset{L}{\bowtie} \mathrm{Poler}) \underset{L'}{\bowtie} \mathrm{Poler_group_0})$$
$$\text{where } L = \{\mathrm{recover_all}^*\},\ L' = \{\mathrm{recover}, \mathrm{break}, \mathrm{recover_all}^*\}$$

Fig. 1. Semi-Markov PEPA description of a simple voting model

5 Conclusion

We have presented a semi-Markov version of PEPA, so that greater distribution expressiveness can be used within the PEPA environment. This will allow PEPA users to make use of recent advances in stochastic tool support, in particular provided by semi-Markov DNAmaca.

This is the first stage of semi-Markov PEPA development; currently there are two layers of operation: semi-Markov actions, which have a higher priority, and Markov actions, which have a lower. We aim to extend SM-PEPA further to enable multiple priority layers, each with designated Markov or semi-Markov status. This will involve more detailed transition rules as well as resolution of questions of synchronisation between actions at different priority layers.

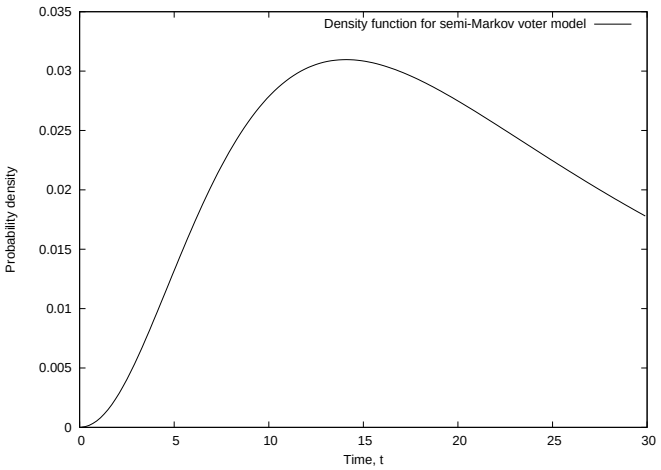

Fig. 2. Passage time density function for the time taken from the first cast vote to the first full failure recovery.

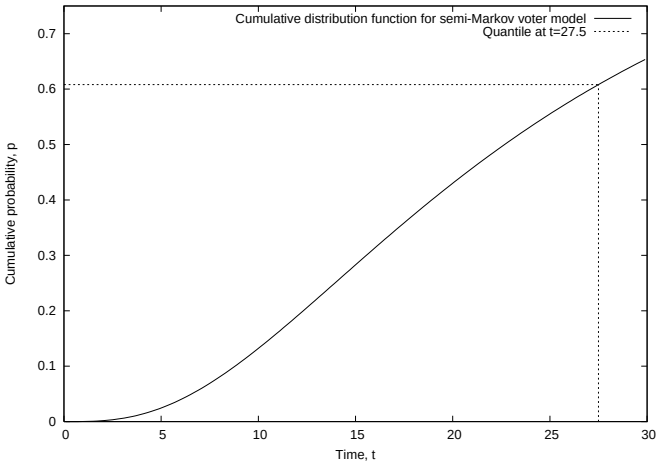

Fig. 3. Passage time cumulative density function and quantile for the time taken from the first cast vote to the first full failure recovery.

Acknowledgements

I would like to thank Nick Dingle for help in generating the results from DNAmaca and also Jane Hillston, Stephen Gilmore, Leïla Kloul and Tony Field for useful comments and suggestions made on an early draft presented at PASTA 2003. Jeremy Bradley is supported in part by the Nuffield Foundation under grant NAL/00805/G.

References

1. 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.
2. "PEPA resource site." `http://www.dcs.ed.ac.uk/pepa/`.
3. H. Bowman, J. W. Bryans, and J. Derrick, "Analysis of a multimedia stream using stochastic process algebras," *The Computer Journal*, vol. 44, no. 4, pp. 230–245, 2001.
4. P. G. Harrison and B. Strulo, "Process algebra for discrete event simulation," *Qualitative Methods in Parallel Systems*, 1995.
5. P. R. D'Argenio, J.-P. Katoen, and E. Brinksma, "A compositional approach to Generalised Semi-Markov Processes," in *WODES'98, Proceedings of the 4th International Workshop on Discrete Event Systems*, (Caligari), pp. 391–387, IEE, 1998.
6. P. R. D'Argenio, *Algebras and Automata for Timed and Stochastic Systems*. PhD thesis, Department of Computer Science, University of Twente, November 1999.
7. M. Bravetti, M. Bernardo, and R. Gorrieri, "Towards performance evaluation with general distributions in process algebras," in *CONCUR'98, Proceedings of the 9th International Conference on Concurrency Theory* (D. Sangiorgi and R. de Simone, eds.), vol. 1466 of *Lecture Notes in Computer Science*, pp. 405–422, Springer-Verlag, Nice, September 1998.
8. M. Bravetti and R. Gorrieri, "Interactive Generalized Semi-Markov Processes," in *Process Algebra and Performance Modelling Workshop* (J. Hillston and M. Silva, eds.), pp. 83–98, Centro Politécnico Superior de la Universidad de Zaragoza, Prensas Universitarias de Zaragoza, Zaragoza, September 1999.
9. R. Pyke, "Markov renewal processes: Definitions and preliminary properties," *Annals of Mathematical Statistics*, vol. 32, pp. 1231–1242, December 1961.
10. J. T. Bradley, N. J. Dingle, W. J. Knottenbelt, and P. G. Harrison, "Performance queries on semi-Markov stochastic Petri nets with an extended Continuous Stochastic Logic," in *PNPM'03, Proceedings of Petri Nets and Performance Models* (G. Ciardo and W. Sanders, eds.), (University of Illinois at Urbana-Champaign), pp. 62–71, IEEE Computer Society, September 2003.
11. J. B. Dugan, K. S. Trivedi, R. M. Geist, and V. F. Nicola, "Extended stochastic Petri nets: Applications and analysis," in *Performance'84, Proceedings of 10th Intl. Symp. on Models of Computer System Performance* (E. Gelenbe, ed.), pp. 507–519, Elsevier, Paris, December 1984.
12. J. T. Bradley, *Towards Reliable Modelling with Stochastic Process Algebras*. PhD thesis, Department of Computer Science, University of Bristol, Bristol BS8 1UB, UK, October 1999.
13. J. T. Bradley, "Semi-Markov PEPA: a contradiction in terms?," in *PASTA'03, Proceedings of 2nd Workshop on Process Algebras and Stochastically Timed Activities* (S. T. Gilmore, ed.), (Edinburgh), pp. 1–6, LFCS, June 2003.
14. J. T. Bradley, N. J. Dingle, S. T. Gilmore, and W. J. Knottenbelt, "Derivation of passage-time densities in PEPA models using ipc: the Imperial PEPA Compiler," in *MASCOTS'03, Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems* (G. Kotsis, ed.), (University of Central Florida), pp. 344–351, IEEE Computer Society Press, October 2003.

Prefix

$$\frac{}{(a,\lambda).P \xrightarrow{(a,D)} P}$$

Competitive Choice

$$\frac{P \xrightarrow{(a,\lambda)} P'}{P + Q \xrightarrow{(a,\lambda)} P'} \quad \text{if } Q \not\xrightarrow{(b^*,D)}$$

$$\frac{Q \xrightarrow{(a,\mu)} Q'}{P + Q \xrightarrow{(a,\mu)} Q'} \quad \text{if } P \not\xrightarrow{(b^*,D)}$$

Cooperation

$$\frac{P \xrightarrow{(a,\lambda)} P'}{P \underset{S}{\bowtie} Q \xrightarrow{(a,\lambda)} P' \underset{S}{\bowtie} Q} \quad \text{if } a \notin S \text{ and } Q \not\xrightarrow{(b^*,D)}$$

$$\frac{Q \xrightarrow{(a,\mu)} Q'}{P \underset{S}{\bowtie} Q \xrightarrow{(a,\mu)} P \underset{S}{\bowtie} Q'} \quad \text{if } a \notin S \text{ and } P \not\xrightarrow{(b^*,D)}$$

$$\frac{P \xrightarrow{(a,\lambda)} P' \quad Q \xrightarrow{(a,\mu)} Q'}{P \underset{S}{\bowtie} Q \xrightarrow{(a,R)} P' \underset{S}{\bowtie} Q'} \quad \text{if } a \in S$$

$$\text{where } R = \frac{\lambda}{r_a(P)} \frac{\mu}{r_a(Q)} \min(r_a(P), r_a(Q))$$

Hiding

$$\frac{P \xrightarrow{(a,D)} P'}{P\backslash L \xrightarrow{(a,D)} P'\backslash L} \quad \text{if } a \notin L \qquad \frac{P \xrightarrow{(a,D)} P'}{P\backslash L \xrightarrow{(\tau,D)} P'\backslash L} \quad \text{if } a \in L$$

Constant

$$\frac{P \xrightarrow{(a,D)} P'}{A \xrightarrow{(a,D)} P'} \quad A \stackrel{def}{=} P$$

Fig. 4. Operational semantics for SM-PEPA: normal Markovian semantics are preserved if only Markovian actions are enabled.

Competitive Choice

$$\frac{P \xrightarrow{(a^*, D)} P'}{P + Q \xrightarrow{(a^*, D)} P'} \qquad \frac{Q \xrightarrow{(a^*, D)} Q'}{P + Q \xrightarrow{(a^*, D)} Q'}$$

Cooperation

$$\frac{P \xrightarrow{(a^*, D)} P'}{P \underset{S}{\bowtie} Q \xrightarrow{(a^*, D)} P' \underset{S}{\bowtie} Q} \quad \text{if } a^* \notin S$$

$$\frac{Q \xrightarrow{(a^*, D)} Q'}{P \underset{S}{\bowtie} Q \xrightarrow{(a^*, D)} P \underset{S}{\bowtie} Q'} \quad \text{if } a^* \notin S$$

$$\frac{P \xrightarrow{(a^*, D)} P' \quad Q \xrightarrow{(a^*, \top)} Q'}{P \underset{S}{\bowtie} Q \xrightarrow{(a^*, D)} P' \underset{S}{\bowtie} Q'} \quad \text{if } a^* \in S$$

$$\frac{P \xrightarrow{(a^*, \top)} P' \quad Q \xrightarrow{(a^*, D)} Q'}{P \underset{S}{\bowtie} Q \xrightarrow{(a^*, D)} P' \underset{S}{\bowtie} Q'} \quad \text{if } a^* \in S$$

Fig. 5. Operational semantics for SM-PEPA: semi-Markov extensions