Reliable Performance Modelling with Approximate Synchronisations

Jeremy Bradley and Neil Davies

Department of Computer Science, University of Bristol, UK
Email: Jeremy.Bradley@bris.ac.uk, Neil.Davies@bris.ac.uk

Abstract. Markovian Process Algebras approximate their model of synchronisation events in order to preserve their Markovian nature. This paper investigates synchronisation models in a stochastic context and focuses on how the Markovian approximation of synchronisation affects the accuracy of the performance model. TIPP and PEPA are used as specific cases throughout, and their different methods of synchronisation are compared for effectiveness in performance modelling. The paper ends with a generally distributed example of real-world synchronisation, which we are able to solve analytically and then approximately with four Markovian Process Algebra models. From the results of this analysis, we show how such approximation affects the accuracy of the performance model for TIPP and PEPA.

1 Introduction

Ever since Stochastic Process Algebras were first mooted, the issue of synchronisation has been a difficult one. Pragmatic decisions have often been made about the modelling of synchronisation, which are then difficult to match to real-world phenomena.

With the exception of [1], very little detailed attention has been paid to the stochastic synchronisation event; yet the choice of synchronisation is one which fundamentally affects the quality of the performance prediction for any model.

This paper looks at two Markovian Process Algebras, TIPP and PEPA, and demonstrates how the adoption of *nice* synchronisation models affects the reliability of the resulting performance statistics.

Stochastic Process Algebras [2, 3] are derived from standard process algebra models of CCS [4] and CSP [5]:

$$P ::= \underbrace{(\alpha, f).P}_{\text{action}} \mid \underbrace{(P+P)}_{\text{choice}} \mid \underbrace{(P \parallel_s P)}_{\text{synchronisation}} \mid \underbrace{(P/L)}_{\text{hiding}} \mid \underbrace{(A)}_{\text{constant}} \tag{1}$$

The difference is that the standard non-deterministic action is augmented by a probability distribution function (f above), which describes the time evolution of that action before the system changes state. In a Markovian Process Algebra (MPA), all actions are assumed to have an exponential probability distribution function and thus the action is described using a label and an exponential rate parameter, (α, r). This allows for later analytic solution of the model through continuous-time Markov Chains [6, 7]. This specialisation to the Markovian case would seem an ideal simplification, if it were not for the fact that the various operations that are performed on the action distributions do not form a closed class of exponential distributions. Specifically, there are obvious models of synchronisation which cannot be represented within an MPA, because the concurrent combination of exponential distributions cannot itself be represented by a single exponential distribution. This Markovian approximation to synchronisation was recognised in [1]:

> ... there is an assumption that the interaction, or shared action, will have an exponentially distributed duration, in the same way as the individual actions of the agents. This assumption is a pragmatic one, although we will see... that it is not necessarily a justified one.

In section 2, we examine common types of synchronisation that occur in software engineering. Section 3 discusses the models used by TIPP and PEPA and presents an analysis on how they differ from the real-world synchronisation models. Section 4 discusses possible synchronisation models for MPAs which allow more accurate performance statistics to be generated. Finally, a worked example is presented in section 5 which serves to compare TIPP and PEPA with each other and with the suggested models of section 4.

2 Real-world Synchronisation Models

In this section, we distinguish between passive synchronisation, as described in section 2.1, and active synchronisations, where both or all the constituent processes make a contribution to the activity of the synchronisation. Active synchronisations are classified in sections 2.2–2.5.

Throughout this section, we use the following standard notation: f_X is the probability distribution function of X, F_X is the cumulative distribution function of X and $\overline{F}_X \equiv 1 - F_X$.

2.1 Client-Server

This is often referred to as passive synchronisation, where one of the processes (the client) idles while its request is dealt with by the server process. The client takes no active part in the activity of the synchronisation, so the duration of the combined event is completely dependent on the server process.

Most Markovian Process Algebras can deal with this type of model (certainly PEPA and TIPP have syntactic or numerical tricks for handling such synchronisations). It should be noted that the distinction between active and passive synchronisations is a physical one since in fact they represent limiting cases of the synchronisation models below.

2.2 First-to-Finish

The First-to-Finish (FTF) synchronisation of two processes X and Y states that the synchronisation can terminate on successful completion of either X or Y, whichever occurs first. Thus FTF synchronisation follows the minimum time duration of X and Y.

An example of this might be a distributed code-breaking effort between processors, where the first processor to find the correct key brings the entire effort to a close. No further processing is required as no further information is required.

In computer science terms, this is perceived as a fairly rare interaction. For a searching problem as above, it requires that the search-space be completely divisible between processes, without any subsequent inter-process communication. Furthermore, there must only be one solution, such that when found it is immediately known that the remaining processes are undertaking a fruitless search.

Stochastic Details For general distributions X and Y:

$$F_{X \overset{\|}{\mathrm{FTF}} Y}(t) = F_{\min(X,Y)}(t) = 1 - \overline{F}_X(t)\overline{F}_Y(t) \tag{2}$$

In the case of n synchronising processes $X_1, \ldots, X_n$:

$$F_{X_1 \overset{\|}{\mathrm{FTF}} \cdots \overset{\|}{\mathrm{FTF}} X_n}(t) = 1 - \prod_{i=1}^{n} \overline{F}_{X_i}(t) \tag{3}$$

If X and Y are exponentially distributed processes with rates λ and μ, then the well known result for First-to-Finish synchronisation of X and Y is generated:

$$f_{X \overset{\|}{\mathrm{FTF}} Y}(t) = f_{\min(X,Y)}(t) = (\lambda + \mu)e^{-(\lambda+\mu)t} \tag{4}$$

This is significant since it tells us that the FTF synchronisation of two exponential distributions is itself an exponential distribution. Also, if $\lambda \to 0$ in equation (4), equivalent to X having infinite mean duration, then we recover the passive client-server model, described in section 2.1.

The First-to-Finish synchronisation for exponential distributions has the following mean and variance properties:

$$\mathbf{E}[\min(X,Y)] = \frac{1}{\lambda + \mu} \qquad \mathbf{Var}[\min(X,Y)] = \frac{1}{(\lambda + \mu)^2} \tag{5}$$

2.3 Last-to-Finish

The Last-to-Finish (LTF) synchronisation of two processes X and Y only allows the synchronisation to terminate when both X and Y have completed. LTF synchronisation follows the maximum sojourn time of X and Y.

To use the searching analogy again: in a distributed search-effort, where there is an unknown number of solutions to be found, each process must search all of its allocated search-space to ensure that all solutions are identified. Thus the overall search time is governed by the longest running process.

In general, this type of synchronisation requires all participants to have completed before allowing the process to continue. It is widely used in parallel algorithms because it reduces algorithmic complexity, despite the fact that it can introduce considerable delay. The searching example given above required a LTF synchronisation for functional correctness, however most instances of LTF synchronisation do not require it for correct operation. In fact, most synchronisations use the LTF model to ensure that a parallel system is in a known state, for areas of program execution which require a safety condition.

Stochastic Details For general distributions X and Y:

$$F_{X \overset{\|}{\text{LTF}} Y}(t) = F_{\max(X,Y)}(t) = F_X(t) F_Y(t) \tag{6}$$

In the case of n synchronising processes $X_1, \ldots, X_n$:

$$F_{X_1 \overset{\|}{\text{LTF}} \cdots \overset{\|}{\text{LTF}} X_n}(t) = \prod_{i=1}^{n} F_{X_i}(t) \tag{7}$$

If X and Y are exponentially distributed processes with parameters λ and μ, then the similarly well known result for Last-to-Finish synchronisation of X and Y is:

$$f_{X \overset{\|}{\text{LTF}} Y}(t) = f_{\max(X,Y)}(t) = \left(\lambda e^{-\lambda t} + \mu e^{-\mu t}\right) - (\lambda + \mu) e^{-(\lambda+\mu)t} \tag{8}$$

This is the crux, for the reason that the LTF synchronisation of two exponential processes is not itself an exponential distribution. Frequently, Markovian Process Algebras have to model Last-to-Finish synchronisations, because it is such a common form of synchronisation. However they can only do so with a single exponential distribution as a *best guess*. It is exactly this forced approximation and its consequences that we set out to investigate in the course of this paper. Also note that by letting $\mu \to \infty$ in equation (8), equivalent to Y being instantaneous, we recover the passive client-server model, described in section 2.1.

The Last-to-Finish synchronisation for exponential distributions has the following mean and variance properties:

$$\mathbf{E}[\max(X,Y)] = \frac{1}{\lambda} + \frac{1}{\mu} - \frac{1}{\lambda + \mu} \qquad \mathbf{Var}[\max(X,Y)] = \frac{1}{\lambda^2} + \frac{1}{\mu^2} - \frac{3}{(\lambda+\mu)^2} \tag{9}$$

2.4 N-to-Finish

FTF and LTF synchronisations are not the only two possibilities. There are many possible synchronisations which lie between the two. These intermediate synchronisations can be roughly aggregated into an N-to-Finish class. Specifically, this is a system where N of M processes are required to finish before the synchronisation is complete. This class of cases clearly lies between the two FTF and LTF and will not be considered explicitly in this paper. Instead we use the FTF and LTF models as bounding limits to these more esoteric synchronisations.

Real instances of these synchronisations do exist. In safety-critical computing a critical function might be managed by 3 or more controlling programs running in parallel. When a change of state is required, a single arbiter program will count the *votes* of the controlling programs and go with the majority decision. Because of the complexity of implementation, these forms of synchronisation are rarely encountered.

There is no obvious 2-process version of this model, but in the event that a synchronisation with performance characteristics between that of FTF and LTF is required, we will not discard the notion. Similarly 2-process FTF, LTF can be used as bounding cases.

2.5 Other Models

We now argue intuitively that there are no other synchronisation models outside the bounds of FTF and LTF synchronisation. Consider a 2-process synchronisation which performs slower than a Last-to-Finish model. By modelling the overall sojourn time of the synchronisation as a third process and incorporating this, via another LTF synchronisation, into the original synchronisation, a Last-to-Finish model can be recovered.

Similarly, consider a 2-process synchronisation which performs faster than the First-to-Finish synchronisation. By representing the faster model as a third process and incorporating its performance into the synchronisation, a First-to-Finish synchronisation is recovered.

Summarising our approach to synchronisation models: synchronisation is at best First-to-Finish and at worst Last-to-Finish. Passive synchronisations can be encompassed within this framework too: by considering them as either a Last-to-Finish model where the client terminates instantly or a First-to-Finish model where the client never terminates.

3 Comparison with MPA Synchronisations

In this section, we compare the synchronisations of two Markovian Process Algebras (PEPA and TIPP) to the real-world LTF and FTF synchronisation, outlined in section 2.

3.1 PEPA

Introduction The PEPA synchronisation model selects the exponential distribution with the largest mean to represent the evolution of the overall synchronisation. Passive synchronisations have an alternative syntactic representation. In [1], the selection of the active synchronisation style is described as follows:

> The definition... is chosen to reflect the idea that the rate at which such a joint action can occur will be limited by the rate of the slowest participant.

Clearly, PEPA is emulating a Last-to-Finish strategy with its synchronisation model. However, given that it is restricted to using exponential distributions, this prevents it from using an exact LTF synchronisation, and so it has to settle for an approximation. The details are slightly more sophisticated than presented here, but in essence the synchronisation takes the minimum rate parameter of its constituent processes.

Stochastic Details If X is distributed with exponential parameter λ and Y is distributed with exponential parameter μ, then:

$$X \overset{\|}{\scriptstyle \mathrm{PEPA}} Y \sim \exp(\min(\lambda, \mu)) \tag{10}$$

This model has the following mean and variance properties for $0 < \lambda < \mu$:

$$\mathbf{E}[X \overset{\|}{\scriptstyle \mathrm{PEPA}} Y] = \frac{1}{\lambda} \quad \mathbf{Var}[X \overset{\|}{\scriptstyle \mathrm{PEPA}} Y] = \frac{1}{\lambda^2} \tag{11}$$

3.2 TIPP

Introduction The TIPP[1] synchronisation model generates an exponential distribution with parameter equal to the product of its component parameters. [7] makes this statement about this choice of synchronisation model:

> From a stochastic point of view this ensures that the resulting distribution remains proportional to its origins which allows [us] to model [the] scaling influence of certain components.

In [1], this is described as a generalisation of the client-server model of interaction. So if either participating rate parameter is 1, then the standard passive synchronisation model is recovered. For other values of the rate parameters the picture is less clear: one value can be seen as a basic level of service which the server delivers, and the other can be taken as a *scaling factor* of that basic rate of service. One problem with this view, apart from not distinguishing syntactically between the client and server, is that it contains no concept of the maximum possible server performance. That is, the client could be modelled to request a service at a rate which the server could not supply.

This is only an interpretation of the model. The primary reason for choosing the model was almost certainly because of the useful algebraic properties of multiplication. The automatic inheritance of commutativity, associativity and

[1] Since we are analysing Markovian Process Algebras, the version of TIPP considered here is the Markovian TIPP, presented in [7]. We will refer to it throughout the paper just as TIPP.

distributivity across addition make it an ideal choice. These properties allow for the construction of a congruence relation across the algebra which in turn allows compositional features of the SPA to be exploited.

In order for PEPA to achieve the same properties ($\min(\cdot\,,\cdot)$ is not distributive across addition), the actual operator had to use *apparent rates* and renormalise the result ([6, 1, 8] for details).

Stochastic Details If X is distributed with exponential parameter λ and Y is distributed with exponential parameter μ, then:

$$X \underset{\text{TIPP}}{\|} Y \sim \exp(\lambda\mu) \tag{12}$$

This model has the following mean and variance properties:

$$\mathbf{E}[X \underset{\text{TIPP}}{\|} Y] = \frac{1}{\lambda\mu} \quad \mathbf{Var}[X \underset{\text{TIPP}}{\|} Y] = \frac{1}{(\lambda\mu)^2} \tag{13}$$

3.3 Comparing TIPP and PEPA Synchronisations

In the following sections, we summarise the stochastic properties of TIPP and PEPA in comparison with the standard FTF and LTF models (details in appendix A of [9]). It is useful to compare expected synchronisation times, in each case, as this gives us an indication of how the mean performance predictions of the MPA models will differ from the real-world behaviour.

PEPA For PEPA, the synchronisation model turns out to be static relative to the LTF and FTF models. From [9], for all $0 < \lambda < \mu$, we have:

$$\mathbf{E}[X \underset{\text{FTF}}{\|} Y] < \mathbf{E}[X \underset{\text{PEPA}}{\|} Y] < \mathbf{E}[X \underset{\text{LTF}}{\|} Y] \tag{14}$$

Clearly PEPA is modelling a valid synchronisation and one closer to LTF than FTF (as is obvious from its choice of $\min(\cdot\,,\cdot)$ operator). However, the fact that it underestimates the synchronisation mean will have a big effect on the steady-state solution of the CTMC. By underestimating the mean or overestimating the rate, PEPA will tend to overestimate the performance or capacity of a system. This is further borne out by the worked example in section 5.

TIPP For TIPP, the synchronisation is not at all static compared to the LTF and FTF models. Summarising detailed results in [9]:

$$\mathbf{E}[X \underset{\text{TIPP}}{\|} Y] < \mathbf{E}[X \underset{\text{FTF}}{\|} Y] \text{ iff, for } 0 < \lambda < \mu :$$

$$1 + \frac{\lambda}{\mu} < \lambda < \infty \tag{15}$$

$\mathbf{E}[X \overset{\|}{_{\text{FTF}}} Y] < \mathbf{E}[X \overset{\|}{_{\text{TIPP}}} Y] < \mathbf{E}[X \overset{\|}{_{\text{LTF}}} Y]$ iff, for $0 < \lambda < \mu$, either:

$$\begin{cases} 1 < \lambda < 2 \ \text{ and } \ \lambda - 1 < \frac{\lambda}{\mu} < 1 \\ \frac{2}{3} < \lambda < 1 \\ 0 < \lambda < \frac{2}{3} \ \text{ and } \ 0 < \frac{\lambda}{\mu} < \frac{1}{2}\left[-1 + \sqrt{\frac{1+3\lambda}{1-\lambda}}\right] \end{cases} \tag{16}$$

$\mathbf{E}[X \overset{\|}{_{\text{LTF}}} Y] < \mathbf{E}[X \overset{\|}{_{\text{TIPP}}} Y]$ iff, for $0 < \lambda < \mu$:

$$0 < \lambda < \frac{2}{3} \ \text{ and } \ \frac{1}{2}\left[-1 + \sqrt{\frac{1+3\lambda}{1-\lambda}}\right] < \frac{\lambda}{\mu} < 1 \tag{17}$$

Since TIPP's synchronisation model varies over the whole spectrum of other models, it is difficult to generalise its effect on performance modelling. However, it is clear that it would be possible to construct a system which either underestimated or overestimated the performance across a particular synchronisation, according to the choice of parameters. To maintain the criterion discussed in section 2.5, TIPP should not have to stray outside the bounds provided by FTF and LTF synchronisation. Given this requirement, the allowable parameter space for TIPP needs to be restricted by the ranges specified in equation (16). TIPP's relative performance characteristics are compared with the other models in the worked example in section 5.

4 Alternative Synchronisation Strategies

The aim of this section is to suggest methods which accurately approximate a Last-to-Finish model. In one case, this involves maintaining the exponential distribution and suggesting an alternative synchronisation rate, and in the other two instances we suggest methods which use and augment the TIPP and PEPA models.

4.1 Mean-preserving Synchronisation

The result of solving the CTMC for an MPA system is the steady-state distribution for that system, that is, a description of the long-run average behaviour.

One possible solution would be to design an exponential synchronisation with the same mean as the LTF synchronisation. In this way, the correct average behaviour for each synchronisation is propagated into the CTMC. So we need to find the exponential distribution with mean:

$$\frac{1}{\lambda} + \frac{1}{\mu} - \frac{1}{\lambda + \mu} \tag{18}$$

That is, we require $Z \sim \exp(\gamma)$ such that:

$$\frac{1}{\gamma} = \frac{1}{\lambda} + \frac{1}{\mu} - \frac{1}{\lambda + \mu} \tag{19}$$

Thus:

$$\gamma = \frac{\lambda\mu(\lambda + \mu)}{(\lambda + \mu)^2 - \lambda\mu} \tag{20}$$

In the notation of [1], we would write:

$$R = \lambda \otimes \mu = \frac{\lambda\mu(\lambda + \mu)}{(\lambda + \mu)^2 - \lambda\mu} \tag{21}$$

It should be noted that this operator is neither associative nor distributive over addition, so it is not possible to construct a congruence and make use of system decomposition in solving the model. Its relative performance characteristics are compared with the other models in the worked example in section 5.

4.2 Using PEPA to Bound the LTF Synchronisation

Unlike TIPP, PEPA maintains the same mean characteristics in its synchronisation relative to the LTF and FTF models, for all parameter values. That is, from equation (14), for all $0 < \lambda < \mu$:

$$\mathbf{E}[X \parallel_{\text{FTF}} Y] < \mathbf{E}[X \parallel_{\text{PEPA}} Y] < \mathbf{E}[X \parallel_{\text{LTF}} Y]$$

A possible approach for system design would be to have a dual synchronisation model to that of PEPA whose mean is always greater than that of the LTF model. That is, for all $0 < \lambda < \mu$:

$$\mathbf{E}[X \parallel_{\text{FTF}} Y] < \mathbf{E}[X \parallel_{\text{PEPA}} Y] < \mathbf{E}[X \parallel_{\text{LTF}} Y] < \mathbf{E}[X \parallel_{\text{PEPA}*} Y] \tag{22}$$

If we define PEPA* to have synchronisation[2]:

$$X \parallel_{\text{PEPA}*} Y \sim \exp\left(\frac{\lambda\mu}{\lambda + \mu}\right) \tag{23}$$

Now:

$$\mathbf{E}[X \parallel_{\text{PEPA}*} Y] = \frac{1}{\lambda} + \frac{1}{\mu} \tag{24}$$

And since for all $\lambda, \mu > 0$:

$$\frac{1}{\lambda} + \frac{1}{\mu} - \frac{1}{\lambda + \mu} < \frac{1}{\lambda} + \frac{1}{\mu}$$

[2] The reason that we have assigned the new synchronisation the label PEPA*, is that, coincidentally, it used to be the old synchronisation model for PEPA [1].

$$\Rightarrow \mathbf{E}[X \underset{\text{LTF}}{\|} Y] < \mathbf{E}[X \underset{\text{PEPA*}}{\|} Y] \tag{25}$$

So PEPA* matches the criterion. Now, in order to obtain bounds for the LTF synchronisation we can derive solutions with the PEPA model and with the PEPA* model.

It should be noted that, as stated, the PEPA* operator is not distributive across addition (it is associative). It is possible that a similar finesse involving *apparent rates*, as on the PEPA operator, can be performed on this operator to make it distributive, however we are not aware of one.

4.3 Restricted TIPP Synchronisation

Unlike PEPA, TIPP does not have a *stable* synchronisation model; by which we mean for certain parameter values it can have a mean less than or greater than the means of the LTF or FTF synchronisation (section 3.3). If, therefore, we require TIPP to model the FTF or LTF synchronisation by forcing the mean of the TIPP synchronisation to equal that of the real-world model, what restrictions does this place on TIPP?

Theorem 1. $\mathbf{E}[X \underset{\text{TIPP}}{\|} Y] = \mathbf{E}[X \underset{\text{FTF}}{\|} Y]$ iff, for $0 < \lambda < \mu$, both:

$$1 < \lambda < 2 \text{ and } \mu = \frac{\lambda}{\lambda - 1} \text{ such that } \infty > \mu > 2 \tag{26}$$

Theorem 2. $\mathbf{E}[X \underset{\text{TIPP}}{\|} Y] = \mathbf{E}[X \underset{\text{LTF}}{\|} Y]$ iff, for $0 < \lambda < \mu$, both:

$$0 < \lambda < \frac{2}{3} \text{ and } \mu = \frac{1}{2}\left[(1-\lambda) + \sqrt{(1-\lambda)(1+3\lambda)}\right] \text{ such that } 1 > \mu > \frac{2}{3} \tag{27}$$

The proofs for these theorems appear in [9]. So now for these specific ranges of λ, the smaller rate, the value of μ is fixed and a synchronisation can take place for which the mean behaviour is identical to that of LTF or FTF synchronisation. This is not a general panacea however, since there is no guarantee that the synchronisations in a model will have the appropriate rates. Indeed, it is quite easy to construct an LTF synchronisation of two legitimate LTF synchronisations which cannot then be modelled in TIPP.

5 A Worked Example

In order to compare the different synchronisation models, we construct a test example which is designed to demonstrate relative performance of an LTF transition across a synchronisation.

$$P \stackrel{\text{def}}{=} Q \underset{\text{LTF}\{token\}}{\|} R, \quad Q \stackrel{\text{def}}{=} U \underset{\text{LTF}\{token\}}{\|} V \tag{28}$$

$$U \stackrel{\text{def}}{=} (token, \alpha).U, \quad V \stackrel{\text{def}}{=} (token, \beta).V \tag{29}$$

$$R \stackrel{\text{def}}{=} (token, \top).\underbrace{(ok, \gamma).R}_{R'} + (notok, \gamma).\underbrace{(token, \top).R}_{R''} \tag{30}$$

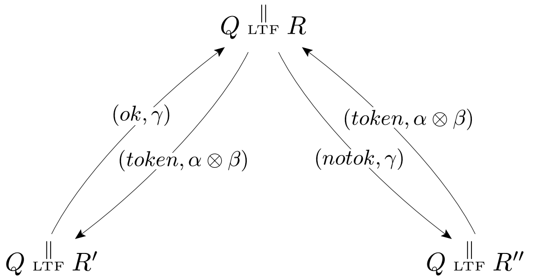

Fig. 1. The transition diagram for the Markovian Process Algebra system, of equations (28–30).

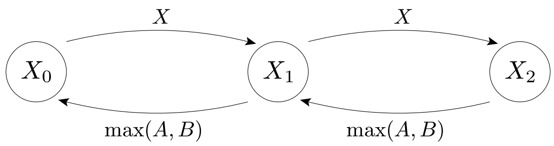

Fig. 2. Transition diagram for the generally distributed synchronisation, $X \sim \exp(\gamma)$, $A \sim \exp(\alpha)$ and $B \sim \exp(\beta)$.This model provides the correct stationary distribution of equations (39–41).

5.1 Markovian Model

In this model, we compete the evolution of the *token* with the evolution of the *notok* action (figure 1). If the system generates the *token* first then it also generates an *ok* action, before returning to its initial position. If it does not produce a *token*, it means that the *notok* has won the race condition; the token is subsequently executed to return the system to its initial position.

The design motivation behind this example is that, for each case, we can then compare the ratio of normal operation (a *token* followed by an *ok*) to error-handling operation (a *notok* then a *token*) in the steady-state. For each of the Markovian cases, where we assume an exponential synchronisation (figure 1), we solve the resulting Markov Chain and obtain the relevant stationary distribution for that system. We do this for each of: PEPA, TIPP, Mean-preserving synchronisation and PEPA* in section 5.3.

5.2 General Model

Now we have to construct the real-life model which takes into account the non-Markovian Last-to-Finish synchronised transition. This has to be done with a deal of care since there are a number of implicit features of the Markovian model which have to be made explicit in the general case.

The stochastic state space is represented in figure 2. To identify the states of the model from now on, we use the following state mapping:

$$X_0 \longleftrightarrow Q \underset{\text{LTF}}{\parallel} R', \quad X_1 \longleftrightarrow P = Q \underset{\text{LTF}}{\parallel} R, \quad X_2 \longleftrightarrow Q \underset{\text{LTF}}{\parallel} R'' \tag{31}$$

Starting from state X_1, the first event is a race between the X and $\max(A, B)$ events; this represents the initial race between the *notok* and *token* actions in the process algebra.

The distribution of the transition after a successful *notok* action from X_1 is key to the type of general model used (this represents the sojourn of the *token* action after *notok*). If, after the choice at X_1, there is a preemptive restart of *token* then we have a complete $\max(A, B)$ evolution after the X sample. Alternatively, if the system employs a preemptive resume strategy (as would be the case in a queueing system), then the *token* event has been allowed to run concurrently during the evolution of the X process. Thus only the *remaining* sampled time of the $\max(A, B)$ process is taken into account when the system moves from state X_2 back to X_1.

Stochastic process algebras have in the past used preemptive restart with re-sampling. However, in the case of Markovian algebras the distinction is a fine one since the memoryless nature of the evolutions in MPAs means that the strategies are identical stochastically. It is only when working with general distributions that the choice of preemptive strategy becomes important.

In either case, on returning to X_1 both processes start afresh at the same instant, since they are both enabled simultaneously by the algebra. Since MPAs such as PEPA state that they are modelling a preemptive restart strategy [6], this is the one we select for our generally distributed model (although it turns out that it makes very little difference to the final conclusions, whichever one is used).

Now for each of the above approximate Markovian models, we have a proper yardstick by which to judge their accuracy in Last-to-Finish synchronisation. The general model is solved in section 5.4 and the comparative results are presented in section 5.5.

5.3 Markovian Solution

We can solve this system generally for all MPAs of the TIPP and PEPA variety. Figure 1 shows the transition diagram for the general Markovian model. By setting $\delta = \alpha \otimes \beta$, the synchronisation operator, we obtain the Markov Chain generator matrix, G:

$$G = \begin{pmatrix} -\gamma & \gamma & 0 \\ \delta & -(\delta + \gamma) & \gamma \\ 0 & \delta & -\delta \end{pmatrix} \tag{32}$$

Using Gaussian Elimination, we find the stationary probability vector $\Pi(X_i)$ such that $\Pi G = 0$ and $\sum_i \Pi(X_i) = 1$. We obtain:

$$\Pi(X_0) = \frac{\delta^2}{\gamma^2 + \delta\gamma + \delta^2}, \quad \Pi(X_1) = \frac{\delta\gamma}{\gamma^2 + \delta\gamma + \delta^2}, \quad \Pi(X_2) = \frac{\gamma^2}{\gamma^2 + \delta\gamma + \delta^2} \tag{33}$$

Now we can define ψ_s (for a model S) to represent the ratio of normal operation to error-handling operation as the ratio of time spent in state X_0 to time spent in state X_2 (since every time the system generates an *ok* it must have gone through X_0 to do so and similarly when the system produces a *notok* it must then go through X_2):

$$\psi_s = \frac{\Pi(X_0)}{\Pi(X_2)} = \left(\frac{\delta}{\gamma}\right)^2 \tag{34}$$

PEPA For $\delta = \min(\alpha, \beta) = \alpha$ for $0 < \alpha < \beta$ WLOG:

$$\psi_{\text{PEPA}} = \left(\frac{\alpha}{\gamma}\right)^2 \tag{35}$$

TIPP For $\delta = \alpha\beta$:

$$\psi_{\text{TIPP}} = \left(\frac{\alpha\beta}{\gamma}\right)^2 \tag{36}$$

Mean-preserving Synchronisation

$$\delta = \frac{\alpha\beta(\alpha+\beta)}{(\alpha+\beta)^2 - \alpha\beta}, \quad \psi_{\text{MEAN}} = \left(\frac{\alpha\beta(\alpha+\beta)}{\gamma(\alpha^2 + \alpha\beta + \beta^2)}\right)^2 \tag{37}$$

PEPA*

$$\delta = \frac{\alpha\beta}{\alpha+\beta}, \quad \psi_{\text{PEPA*}} = \left(\frac{\alpha\beta}{\gamma(\alpha+\beta)}\right)^2 \tag{38}$$

5.4 Analytic Solution

By observing that our example system has a simple underlying stochastic transition system (figure 2) and using pdf aggregation techniques from [12] and [14], we can obtain the precise analytical solution for the steady-state distribution using the *Strong Law of Large Numbers* [15]—further details in [14]. We obtain:

$$\Pi(X_0) = \frac{\alpha^2\beta^2(\alpha+\beta)(\alpha+\beta+2\gamma)}{\chi} \tag{39}$$

$$\Pi(X_1) = \frac{\alpha\beta\gamma(\alpha+\beta)\rho}{\chi} \tag{40}$$

$$\Pi(X_2) = \frac{\gamma^2(\alpha^2 + \alpha\beta + \beta^2)\rho}{\chi} \tag{41}$$

where $\rho = (\alpha + \beta + \gamma)^2 - \alpha\beta$ and:

$$\begin{aligned}\chi = &(\alpha^2 + \alpha\beta + \beta^2)\gamma^4 + (\alpha + \beta)(2\alpha^2 + 3\alpha\beta + 2\beta^2)\gamma^3 \\ &+ (\alpha^4 + 4\alpha^3\beta + 7\alpha^2\beta^2 + 4\alpha\beta^3 + \beta^4)\gamma^2 \\ &+ \alpha\beta(\alpha + \beta)(\alpha^2 + 3\alpha\beta + \beta^2)\gamma + \alpha^2\beta^2(\alpha + \beta)^2\end{aligned} \tag{42}$$

Now, as before we can calculate ψ_s using equation (34)[3]:

$$\psi_{\text{QUEUE}} = \frac{\Pi(X_0)}{\Pi(X_2)} = \frac{\alpha^2\beta^2(\alpha + \beta)(\alpha + \beta + 2\gamma)}{\gamma^2(\alpha^2 + \alpha\beta + \beta^2)\rho} \tag{43}$$

5.5 Model Comparisons

General Results First, we look at general statements that can be made about ψ_s in comparison to ψ_{QUEUE}, the correct ratio. It can be seen that $\psi_{\text{PEPA}} > \psi_{\text{QUEUE}}$ and $\psi_{\text{MEAN}} > \psi_{\text{QUEUE}}$, for all values of α, β, γ. The PEPA result is consistent with predictions from the expectation results of section 3.3.

Also, as we hoped, our Mean-preserving synchronisation lies between PEPA and PEPA*, that is $\psi_{\text{PEPA*}} < \psi_{\text{MEAN}} < \psi_{\text{PEPA}}$ for all α, β, γ.

However, unfortunately, it is not true to say that our suggested bounding model PEPA* always underestimates performance, since $\psi_{\text{PEPA*}} \not< \psi_{\text{QUEUE}}$. In fact, it turns out that no Markovian synchronisation model can always underestimate ψ_{QUEUE}. This statement has more general consequences and leads to:

Theorem 3. There does not exist any Markovian synchronisation which can consistently underestimate (or equal) the correct performance model of an LTF synchronisation for all possible systems.

Proof. We use the example defined in this section as a counterexample to the hypothesis: "There exists a Markovian synchronisation which can consistently underestimate (or equal) the correct performance model of an LTF synchronisation for all possible systems".

For any MPA, equation (34) gives:

$$\psi_s = \left(\frac{\delta_s(\alpha, \beta)}{\gamma}\right)^2 \tag{44}$$

From equation (43), we can derive:

$$\psi_{\text{QUEUE}} = \left(\frac{\alpha\beta}{\gamma}\right)^2 \frac{(\alpha + \beta)(\alpha + \beta + 2\gamma)}{R(\alpha, \beta, \gamma)} \tag{45}$$

where:

$$R(\alpha, \beta, \gamma) = (\alpha^2 + \alpha\beta + \beta^2)\left[\gamma^2 + 2(\alpha + \beta)\gamma + (\alpha^2 + \alpha\beta + \beta^2)\right] \tag{46}$$

[3] The value of ψ_s for the analytical solution is called ψ_{QUEUE}, since the general model is close, but not precisely identical, to that of a queue (see discussion of preemptive strategy in section 5.2).

Table 1. Some example values of ψ_s for various parameters and models.

α	β	γ	ψ_{QUEUE}	$\psi_{\text{PEPA*}}$	ψ_{MEAN}	ψ_{PEPA}	ψ_{TIPP}
0.5	1.0	0.7	0.292	0.227	0.375	0.510	0.510
1.1	7.2	0.7	2.313	1.858	2.372	2.469	128.013
5.0	92.0	0.7	50.696	45.896	50.736	51.020	431,836.735
0.5	2.0	5.1	0.00410	0.00615	0.00872	0.00961	0.0384

We require $\psi_s < \psi_{\text{QUEUE}}$ for all α, β, γ. So WLOG by setting $\delta_s(\alpha, \beta) = \Delta_s^{-1}(\alpha, \beta)$, we get equation (47).

$$\underbrace{(\alpha\beta)^2(\alpha+\beta)\Delta_s^2(\alpha,\beta)(\alpha+\beta+2\gamma)}_{(*)} - R(\alpha,\beta,\gamma) > 0 \tag{47}$$

However, $(*)$ can only be $O(\gamma)$, independent of the choice of $\Delta_s(\alpha, \beta)$, whereas $R(\alpha, \beta, \gamma)$ is $O(\gamma^2)$. Thus LHS of equation (47) < 0 for some α, β, γ. □

So our worked example acts as a counterexample and we have shown quite a powerful result. This result is almost certainly a consequence of the original Markovian assumption of all MPAs.

However, all is not lost for PEPA*. Examination of the algebra indicates that, for many parameter values, PEPA* will still underestimate the correct performance, as intended and as suggested by the mean properties of the PEPA* synchronisation. In the next section, the numerical results indicate that this is indeed the case.

Numerical Evidence For specific parameter values, we can see how the performance measures vary in the stationary case (table 1). In the graphs presented, we concentrate on varying the smaller parameter, α, of the synchronisation relative to γ, since the exponential distribution with the smaller parameter will tend to dictate the behaviour of the LTF synchronisation.

Figure 3 shows that, probably for large relative values of α and β to γ, PEPA and PEPA* still provide good bounding models for the analytic result. Also, satisfyingly, the Mean-preserving Markovian model comes closest to the actual solution.

Figure 4 demonstrates where PEPA* ceases to become a lower bound for ψ_{QUEUE}. Here, all the Markovian models overestimate the performance of the system, but still shadow it closely with the result that the absolute error is small.

Figure 5 and some of the more eccentric results in table 1 show that TIPP can rather overestimate the correct performance value. However, if we restrict the parameters to the ranges specified in equation (27), TIPP's predictive accuracy can be greatly enhanced (as demonstrated in figure 6). Not unexpectedly, in the same figure, plotting the mean-preserving Markovian case completely overlaid

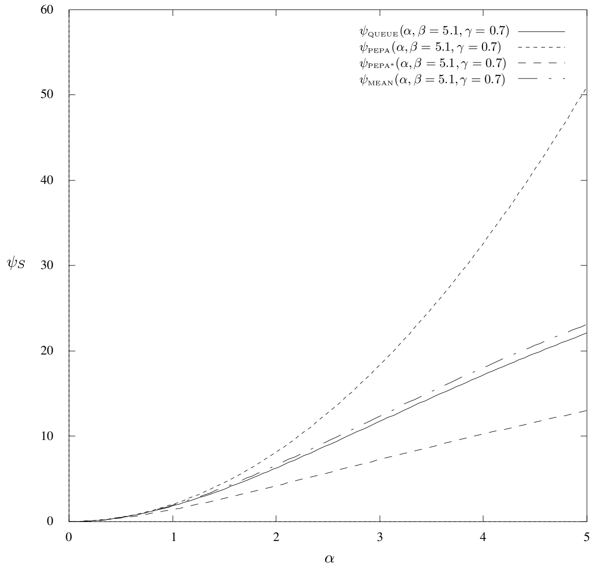

Fig. 3. PEPA and PEPA* bound the analytic solution with the Mean-preserving model lying in between.

the TIPP model; this confirms that TIPP has indeed taken on the mean of the LTF synchronisation for this set of restricted parameter values.

6 Conclusion

For a long time, it has been understood that popular Markovian Process Algebras approximated their model of synchronisation in order to maintain their Markovian property. Until now, it has not been understood how this approximation affected the accuracy of their performance model. We have been able to map the effect of this approximation through to the final performance measures for a number of different Markovian Process Algebras. In doing so, we have been able to demonstrate how these different MPAs can be used to gain a useful and accurate performance model for a system.

Summary In this paper, we have developed real-world synchronisation models and shown how they can be described in the field of Stochastic Process Algebras. We find two forms of general synchronisation model which usefully can act as bounds to other synchronisation models, namely Last-to-Finish (LTF) and First-to-Finish (FTF) synchronisation.

Since most of the current work on SPAs involves specialising to the Markovian case, we have examined in detail how these general models of synchronisation compare to the Markovian models used in TIPP and PEPA. It is interesting to

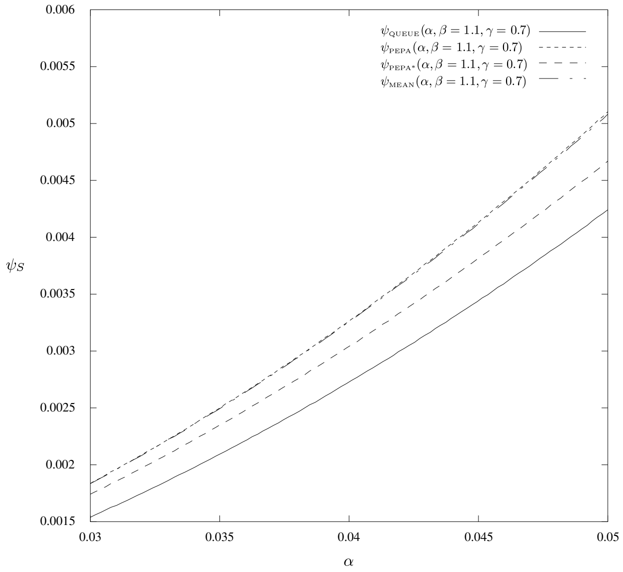

Fig. 4. All consistent MPAs overestimate the analytic solution for these low value parameters, but the absolute error is still small.

note that no MPA has implemented a form of FTF synchronisation, despite the fact that in the Markovian case, it can be represented precisely.

We have investigated three ways of improving the accuracy of predicted performance in Markovian Process Algebras:

Mean-preserving Synchronisation By attempting to replicate the mean of an individual general LTF synchronisation as an exponential distribution, we hope to improve the accuracy of the long term performance model. The numerical evidence suggests that this is the best of all the MPA models at consistently coming closest to an accurate solution.

Bounded PEPA Modelling We make use of the fact that PEPA consistently overestimates model performance and attempt to define a dual model, PEPA*, which similarly underestimates the LTF model. This was very successful for frequent occurrences of a synchronised event (large values of ψ_s), where PEPA and PEPA* bounded the correct solution almost equally. However for rare synchronised events (ψ_s small), both PEPA and PEPA* overestimated the performance measure, but in both cases only by very small absolute values.

Restricted TIPP Synchronisation By restricting TIPP's *scaling factor* concept to specific ranges of parameters we can emulate LTF or FTF synchronisation very effectively. In the case of LTF synchronisation, we obtain as good a set of results with Restricted TIPP as we do with the Mean-preserving model.

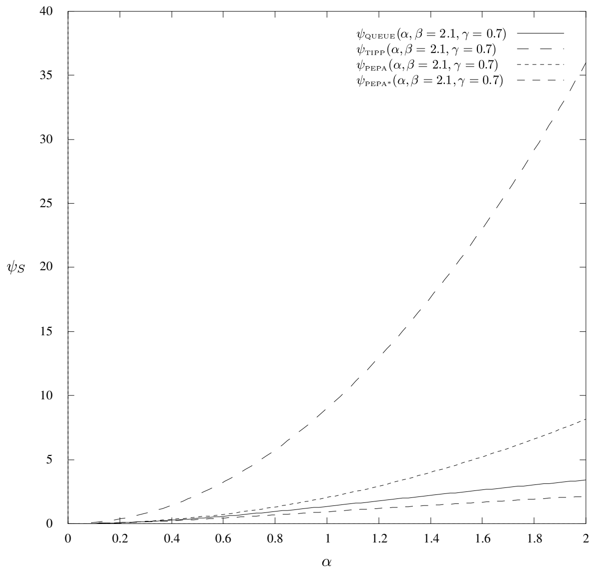

Fig. 5. TIPP can overestimate the analytic solution considerably in some circumstances.

We have also been able to show that it is impossible to design a Markovian synchronisation which consistently underestimates or equals the performance measure of the correct LTF synchronisation for all systems. However, despite this, we have defined PEPA*, which underestimates for a large proportion of the parameter space and this may have useful implications.

PEPA A good engineering solution to the performance modelling problem could well be to use a combination of PEPA, PEPA* and the Mean-preserving model: PEPA and PEPA* to bound the correct model (in the right circumstances) and the Mean-preserving model to compare to the average of the PEPA and PEPA* predictions. For the first time, it will also be possible to define a variance for the accuracy of the performance measures using these three values. This measure of model reliability could be very useful for designers of real-time systems or safety-critical systems who need to know how well they can trust their performance modelling paradigm.

TIPP It is clear from our analysis and results that TIPP was not really designed to model LTF synchronisation, and if not used carefully can produce startling performance results. The Restricted TIPP model gives a very good approximation but may well turn out to be too restrictive to use in practice.

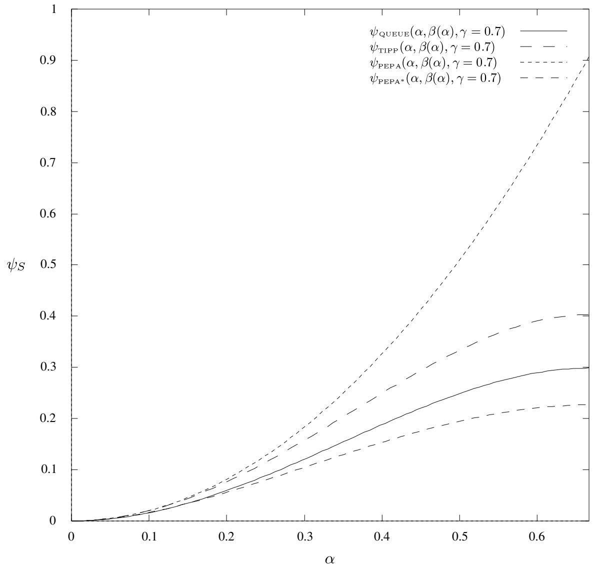

Fig. 6. If the parameter values are restricted to the ranges in equation (27), then TIPP can simulate LTF synchronisation.

General Remarks and Future Work As far as we know, this is the first time that a paper has been able to compare an analytical solution of a system with a general pdf, to the approximation of the same system by various MPA models. As such and given that our example system was designed to be a good measure for general synchronisations, a more complete exploration of the parameter space is currently being carried out for this example, to see what other qualitative features the different MPA systems possess.

Markovian Process Algebras are really only modelling approximations to reality. It has been assumed in this paper, as with the MPAs themselves, that the atomic actions are in fact Markovian. This is almost certainly a rather generous assumption and suggests that what is truly required is a framework for properly describing evolutions with general distributions. Such a system could then precisely describe all the real-world synchronisations, without approximation. Such models are still the subject of future work [16].

For MPAs the future is still fairly bright: they are easy to manipulate analytically and there seems to be plenty of room to introduce further approximate or precise synchronisation models as required. Further work needs to be done to investigate how multiple synchronisation models can coexist in the same MPA (for example, FTF and an approximate LTF) and how this affects their algebraic structure under bisimulations.

Acknowledgements The authors would like to thank Peter Thompson for his contribution on the subject of synchronisations in parallel systems. This work is funded by NATS, a division of the Civil Aviation Authority.

References

1. Jane Hillston. The nature of synchronisation. In Ulrich Herzog and Michael Rettelbach, editors, *PAPM'94, Proceedings of the 2nd International Workshop on Process Algebras and Performance Modelling*. Universtät Erlangen-Nürnberg, Regensberg, July 1994.
2. Holger Hermanns, Ulrich Herzog, and Jane Hillston. Stochastic process algebras—A formal approach to performance modelling. Tutorial, Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, UK, 1996.
3. Norbert Götz, Ulrich Herzog, and Michael Rettelbach. TIPP—Introduction and application to protocol performance analysis. In H König, editor, *Formale Beschreibungstechniken für verteilte Systeme*, FOKUS. Saur-Verlag, Munich 1993.
4. Robin Milner. *Communication and Concurrency*. PHI Series in Computer Science. Prentice Hall, 1989.
5. C A R Hoare. *Communicating Sequential Processes*. PHI Series in Computer Science. Prentice Hall, 1985.
6. Jane Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, UK, 1994.
7. Holger Hermanns and Michael Rettelbach. Syntax, semantics, equivalences, and axioms for MTIPP. In Ulrich Herzog and Michael Rettelbach, editors, *PAPM'94, Proceedings of the 2nd International Workshop on Process Algebras and Performance Modelling*. Universtät Erlangen-Nürnberg, Regensberg, July 1994.
8. Jane Hillston. Compositional Markovian modelling using a process algebra. In *Proceedings of the 2nd International Workshop on Numerical Solution of Markov Chains*. Kluwer Academic Press, Raleigh, January 1995.
9. J T Bradley and N J Davies. Reliable performance modelling with approximate synchronisations. Technical Report CSTR–99–002, Department of Computer Science, University of Bristol, February 1999.
10. Arnold A Allen. *Probability, Statistics and Queueing Theory with Computer Science Applications*. Academic Press, second edition, 1990.
11. D R Cox and H D Miller. *The Theory of Stochastic Processes*. Methuen, 1965.
12. Kishor Shridharbhai Trivedi. *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. Prentice-Hall, 1982.
13. Jacob Willem Cohen. *The Single Server Queue*, volume 8 of *Applied Mathematics and Mechanics*. North-Holland, 1969.
14. J T Bradley and N J Davies. An aggregation technique for analysing some generally distributed stochastic processes. Technical Report CSTR–99–003, Department of Computer Science, University of Bristol, March 1999.
15. William Feller. *An Introduction to Probability Theory and its Applications*, volume 1. John Wiley, third edition, 1968.
16. Mario Bravetti, Marco Bernardo, and Roberto Gorrieri. Towards performance evaluation with general distributions in process algebras. In *CONCUR'98, Proceedings of the 9th International Conference on Concurrency Theory*, volume 1466 of *Lecture Notes in Computer Science*. Springer-Verlag, Nice, September 1998.