University of Bristol

DEPARTMENT OF COMPUTER SCIENCE

An Aggregation Technique for Analysing Some Generally Distributed Stochastic Processes

J Bradley   N Davies

Also issued as ACRC-99:CS-003

March 1999 | CSTR-99-003

An Aggregation Technique for Analysing Some Generally Distributed Stochastic Processes*

Jeremy Bradley and Neil Davies

Department of Computer Science, University of Bristol,
Woodland Road, Bristol BS8 1UB, UK
Email: {Jeremy.Bradley, Neil.Davies}@bristol.ac.uk,
Tel: +44 117 903-1112, Fax: +44 117 954-5208

23 February 1999

Abstract

A stochastic aggregation technique is developed and used to reduce generally distributed stochastic transition systems to a soluble example. By aggregating carefully, we can derive the entire stationary distribution for the original stochastic transition system.

This paper examines the possibility of constructing a solution technique for generally distributed process algebras. Further, we demonstrate that by creating a stochastic transition system, we can solve G/G/1/2 queues and, more generally, Markovian Process Algebras without the use of Markov Chains.

1 Introduction

While Markovian systems and solution techniques are well studied [16], there are few techniques for analysing systems with completely general distributions. Those that do exist are limited to stochastic processes of a particularly restricted structure, predominantly queueing systems [5, 17, 18].

It is the goal, and notable success, of process algebras [19, 14] to describe functionally any structure of communicating system. It is the goal of stochastic process algebras to inherit this descriptive power and additionally model the underlying stochastic process to derive performance and reliability information.

This has been largely achieved with Markovian Process Algebras [13, 9, 1, 4] for Markovian systems. However, although extensions to Markovian Process Algebras have been produced [3, 20, 2] to model more general distributions, a

*Submitted to PAPM'99.

completely general Stochastic Process Algebra is still elusive. This in turn is due to a distinct lack of tools and techniques for solving generally distributed stochastic systems. In this paper, we present an interesting technique for solving a particular class of such stochastic systems. We then relate some of the concepts and ideas from the stochastic process algebra paradigm, in the hope of aiding and furthering the development of a future generally distributed stochastic process algebra.

1.1 Summary of Approach

Our central theme involves the construction of a simple generally distributed stochastic transition system. We will show that this transition system is sufficiently simple that it can be solved for all cases, through a technique we call stochastic aggregation.

From the outset we have attempted to model generally distributed processes directly. We were keen to see how far we could take such an approach before issues like memory[1] completely blocked progress.

In other methodologies, such as stochastic petri nets, a popular and useful method of dealing with general distributions is to invoke the property of insensitivity [11]. This is a property in which it is shown that replacing a general distribution with an exponential distribution of the same mean has no effect on the stationary distribution. This clearly leads to a simpler model and is a useful result. Insensitivity is not called upon in this paper.

1.2 Stochastic Aggregation

Stochastic aggregation is a process of combining stochastic processes and structures into a single process. It is a well-known result (Trivedi 1982 [21]) that program structures such as if-then-else statements, while-loops and for-loops can be represented stochastically and then aggregated into a single equivalent process. This allows us to aggregate sequences and combinations of such structures into single processes.

What we are able to do is show that for cyclic systems expressed in our stochastic transition system, for each state, we can aggregate the system into a normal form. These normal forms are special precisely because they can be analytically solved to give stationary distributions.

The key to this aggregation technique is not to lose sight of the original system totally. By carefully aggregating around a selected state, the normal form is generated still containing a replica of that state, relative to the rest of the system. Solving this normal form for the stationary distribution now yields the correct stationary probability for the original state. This is then repeated for

[1]This issue occurs between generally distributed transitions, where concurrent evolution is being modelled. Markovian systems do not have this problem as their memoryless property allows them to be arbitrarily interleaved.

the remaining states, to obtain the entire stationary distribution for the whole system.

1.3 Generally Distributed Process Algebras

It is important to understand the distinction between a stochastic transition system and a stochastic process algebra (SPA). An SPA is a high level process description language which allows systems to be specified succinctly. A stochastic transition system would be used to represent the stochastic state space of the system. It should be possible to generate the stochastic transition system from an SPA, by a sequence of laws.

After describing the stochastic transition system in full, we will go on to discuss mappings between different classes of SPA and a stochastic transition system model. Our overall aim is to produce decomposition laws to the stochastic transition system level for as highly descriptive an SPA as possible. Thus we should be able to solve more general classes of generally distributed stochastic processes.

1.4 Summary of Results

It is worth taking some time to summarise the major results in this paper:

a. A class of generally distributed stochastic transition systems can be solved, through stochastic aggregation.

b. Using stochastic transition systems, an operational model for a restricted set of generally distributed stochastic process algebras can be constructed.

c. An analytic solution to the G/G/1/2 queueing system is presented, by constructing a mapping to stochastic transition systems.

d. It is discovered that a direct solution to G/G/1/K queues for $K \geq 3$ cannot be generated using this technique. However, an investigation into the reasons for this yields interesting insights into the problems that would be encountered in trying to develop an operational model for a fully-featured SPA.

e. Finally, we note that the problems that dog fully-featured generally distributed SPAs do not affect Markovian Process Algebras (MPA), and we construct a mapping between a generic MPA and stochastic transition systems. Thus, we find an operational model for MPAs which does not require the use of Markov Chains at all. Since the method works on a state by state basis, as required, we can let the reward structure inform our solution method; i.e. if only 3 stationary probabilities are required for a particular reward structure, then only 3 states need to be considered and manipulated; we only solve as much of the model as is required by the reward structure.

2 Stochastic Transition Systems

2.1 Introduction

In this section, we present a solution technique for a generally distributed stochastic transition system. A transition system is defined to describe individual evolutions from state to state. Time is represented along each transition by a continuous random variable and it is these random variables which can take any distribution.

In this instance, as with other stochastic models, our primary goal is to obtain a stationary distribution for the system. To this end, we describe a simple example model (the Cox-Miller Normal Form) which can be solved directly to give the stationary distribution. The remainder of the section is devoted to showing how general models can be simplified to similar normal forms, which can also be solved to give stationary distributions.

The process of simplification is called stochastic aggregation. It is carried out in such a way that from the stationary probabilities of the normal form, stationary values for the original system can be deduced.

2.2 Motivation

In designing a stochastic transition system, we are keen to avoid many of the complexities that make other generally distributed systems (queueing systems, for instance) very difficult to solve. In particular, by steering clear of any notion of concurrent processes, it is seen that the memoryless issue can be avoided at this stage. Although it is clear that we are potentially reducing the expressiveness of our language by doing so, a solution technique for some class of generally distributed system is still of interest to us. If successful, then further complexities can be considered.

While studying some results and methods from single-server finite queues, the question kept on arising: 'What *is* the actual distribution between this state and the next one?'. It became apparent that saying that a queue had a generally distributed service time was attributing a high level description to the queue, much as a process algebra might do. Precisely as a result of complexities like memory, the actual inter-state distributions were distinct, complex and certainly not obvious from the stated general distribution.

In Markovian queues or systems, this is not a distinction that is come across. In saying that a queue is Markovian, its stochastic transition system is immediately attributed exponential distributions to the transitions between the states. It turns out there are many such simplifications and finesses that can be performed by using Markovian processes, which, of course, is why they are so useful and so heavily studied. However this does have the drawback of hiding a lot of the underlying complexity.

By observing that some kind of underlying state transition system was behind the operation of such generally distributed systems, it is a natural step to wanting to describe and solve such a model. In this way the use of techniques such as embedded Markov Chains might be avoided. This is desirable because, though a clever application, the embedded Markov Chain in queues is very reliant on the simple Birth-Death structure of the queue. Such nice structural properties cannot be assumed for general process graphs, which are typically described by process algebras.

With this in mind, it seemed necessary to abstract concepts like action labels, competitive choice and parallel composition up to a process algebra level. It will seen in later sections how such higher level functionality can be reintroduced.

2.3 Definition

Define a stochastic transition system to be a directed labelled graph. For the purposes of stationary distribution analysis let the digraph be biconnected, by which we mean that any vertex (state) is reachable from any other vertex, via a directed path. This latter condition is the equivalent of the structural ergodic condition for PEPA [13] and ensures that a stationary distribution exists.

In operation, a process will start in a designated state and proceed to follow one and only one path round the graph. At no stage, therefore, does memory become an issue since the process is only executing one transition at a time, in sequence.

In algebraic terms we define a state, S to be:

$$S ::= \{X\}.S \;\big|\; [p]S + [1-p]S \;\big|\; A \tag{1}$$

where X represents a continuous positive random variable and p, $0 < p < 1$ is a probability and A is a constant label.

The first operator is analogous to the prefix operator in CCS [19] and represents a sequential process, which proceeds for duration X and then terminates and follows the path to the next transition.

The second operator represents branching. As can be seen from the definition, we have adopted probabilistic branching, mainly for reasons of simplicity given above. The selection of a path is instantaneous and samples from a Bernoulli trial (parameter, p) to decide which branch to take.

For the algebraic model to match the graph description, it remains to be demonstrated how n-way branching in the digraph is treated. For this, we simply need a structural associativity law to enable us to treat the n-way branch in a pairwise fashion.

Let:

$$S \stackrel{\text{def}}{=} [p]T + [1-p]\{Z\}.S_3 \tag{2}$$

$$T \stackrel{\text{def}}{=} [q]\{X\}.S_1 + [1-q]\{Y\}.S_2 \tag{3}$$

We require that:

$$S \stackrel{\text{def}}{=} [m]\{X\}.S_1 + [1-m]U \tag{4}$$

$$U \stackrel{\text{def}}{=} [n_1]\{Y\}.S_2 + [n_2]\{Z\}.S_3 \tag{5}$$

such that $n_1 + n_2 = 1$.

Now:

$$\mathbf{P}\{S_1 \text{ is reached from } S\} = pq \tag{6}$$

$$\mathbf{P}\{S_2 \text{ is reached from } S\} = p(1-q) \tag{7}$$

$$\mathbf{P}\{S_3 \text{ is reached from } S\} = 1-p \tag{8}$$

given that the branch probabilities are independent Bernoulli trials. Now we can see that, for the second model to be consistent with the first:

1. For state S_1:

$$m = pq \tag{9}$$

2. For states S_2, S_3:

$$n_1(1-pq) = p(1-q) \tag{10}$$

$$n_2(1-pq) = 1-p \tag{11}$$

Now:

$$n_1 + n_2 = \frac{p-pq}{1-pq} + \frac{1-p}{1-pq} = 1 \tag{12}$$

Hence result.

From this result, a general n-way probabilistic branch, $\sum_{i=1}^{n}[p_i]S_i$ where $\sum_{i=1}^{n} p_i = 1$, can be defined recursively, for $n \geq 2$:

$$\sum_{i=1}^{n}[p_i]S_i \stackrel{\text{def}}{=} [p_n - 1]\left(\sum_{i=1}^{n-1}\left[\frac{p_i}{p_n - 1}\right] S_i\right) + [p_n]S_n \tag{13}$$

$$\sum_{i=1}^{2}[p_i]S_i \stackrel{\text{def}}{=} [p_1]S_1 + [p_2]S_2 \tag{14}$$

2.4 Stochastic Aggregation

Here, we describe a technique called stochastic aggregation, which forms the basis of our solution technique for stochastic transition systems. The rules presented come from Trivedi 1982 [21] where they described program execution; they have been rewritten below to describe a stochastic transition system.

The associativity of probabilistic branching (section 2.3), means that it is sufficient to consider 2-way branching in equations (17) and (19).

Stochastic aggregation was born out of a need to try and reduce system complexity by combining (or aggregating) the edges and states of the transition graph. In doing so, it replaces a structure of the graph with a smaller equivalent structure, which takes the same amount of time to execute as the original.

Throughout, stochastic aggregation is defined in terms of Laplace transforms.

2.4.1 Convolution

Two sequential processes are aggregated into one:

$$A \stackrel{\text{def}}{=} \{X_1\}.\{X_2\}.B \longrightarrow A \stackrel{\text{def}}{=} \{Y\}.B \tag{15}$$

where:

$$L_Y(\omega) = L_{X_1}(\omega)L_{X_2}(\omega) \tag{16}$$

2.4.2 Branching

A branch point where both branches eventually recombine, can be aggregated into a single process:

$$A \stackrel{\text{def}}{=} [p]\{X_1\}.B + [1-p]\{X_2\}.B \longrightarrow A \stackrel{\text{def}}{=} \{Y\}.B \tag{17}$$

where:

$$L_Y(\omega) = pL_{X_1}(\omega) + (1-p)L_{X_2}(\omega) \tag{18}$$

2.4.3 Terminating Loop

A process that loops, with probability $(1-p)$, or takes an alternative path with probability p, can be aggregated into a single process:

$$A \stackrel{\text{def}}{=} [p]\{X_1\}.B + [1-p]\{X_2\}.A \longrightarrow A \stackrel{\text{def}}{=} \{Y\}.B \tag{19}$$

where:

$$L_Y(\omega) = \frac{pL_{X_1}(\omega)}{1-(1-p)L_{X_2}(\omega)} \tag{20}$$

2.5 Equilibrium Distributions

Before we launch into our first instance of solving a system for its equilibrium distribution, we must first discuss some properties of equilibrium distributions themselves. These are taken from Cox and Miller 1965 [6]:

Given a state transition system:

1. For all initial conditions the probability $p_i(t)$ that state i is occupied at time t tends to a distribution $\{\pi_i\}$ as $t \longrightarrow \infty$.

2. If the initial state has the probability distribution $\{\pi_i\}$ then $p_i(t) = \pi_i$ for all t.

3. After a long time t_0, the proportion of time spent in any state i converges to π_i as $t_0 \longrightarrow \infty$.

4. The equilibrium probabilities, π_i, if they exist, satisfy a system of ordinary simultaneous linear equations.

These properties lead to the following empirical interpretations of stationary distributions:

1. Given a large number of independent realisations (traces) of a system: at a large time t_0, the proportion of systems in state i tends to π_i.

2. Given a single long realisation, the proportion of time spent in state i tends to π_i.

It is the second of these properties which we use to define our stationary distribution in the Cox-Miller Normal Form example presented in the next section.

2.6 Cox-Miller Normal Form

Cox-Miller Normal Form is the simplest generally distributed cyclic process that can be solved directly to give an equilibrium distribution. It was described specifically in Cox and Miller [6, pp.172–173] for the Markovian case and is extended to general distributions here.

The system consists of two states, 0 and 1, and has two distinct generally distributed transitions S and T, as shown in figure 1. Two sequences of independent identically distributed random variables are constructed $X_1, X_2, \ldots$ and $Y_1, Y_2, \ldots$, where $F_{X_i}(t) = F_S(t)$ and $F_{Y_i}(t) = F_T(t)$ for all i. X_i represents the ith transition from state 0 to state 1 and Y_i represents the returning ith transition back from state 1 to state 0. Thus the sequence of transitions in any given realisation (or trace) of the system would be: $X_1, Y_1, X_2, Y_2, \ldots$ and so on.

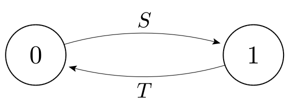

Figure 1: Cox-Miller Normal Form

Suppose for simplicity that the system starts in state 0 and undergoes $2n$ transitions. The total amount of time it will have spent in state 0 is $X_1 + \cdots + X_n$ and similarly the total amount of time it will have spent in state 1 is $Y_1 + \cdots + Y_n$. We now construct two random variables $\Pi_i^n, i = 0, 1$, representing the proportion of time spent in states 0 and 1, after $2n$ transitions.

$$
\begin{aligned}
\Pi_0^n &= \frac{X_1 + \cdots + X_n}{X_1 + Y_1 + X_2 + Y_2 + \cdots + X_n + Y_n} \\
&= \frac{\frac{1}{n}\sum_{i=1}^n X_i}{\frac{1}{n}\sum_{i=1}^n X_i + \frac{1}{n}\sum_{i=1}^n Y_i} \\
&= \frac{\overline{X}}{\overline{X} + \overline{Y}}
\end{aligned} \tag{21}
$$

Similarly:

$$
\Pi_1^n = \frac{\overline{Y}}{\overline{X} + \overline{Y}} \tag{22}
$$

where $\overline{X}$ and $\overline{Y}$ are sample means of the n independent random variables.

We now utilise the Strong Law of Large Numbers ([7, p.243]) to relate the sample means of X_i and Y_i to the distribution means of S and T.

Theorem 1 (The Strong Law of Large Numbers) *Let X_k be a sequence of mutually independent random variables with a common distribution. If the expectation $\mu = \mathbf{E}\{X_k\}$ exists, then for every $\epsilon > 0$ as $n \longrightarrow \infty$:*

$$
\mathbf{P}\left\{\left|\frac{X_1 + \cdots + X_n}{n} - \mu\right| < \epsilon\right\} \longrightarrow 1 \tag{23}
$$

Now applying this to our sample means $\overline{X}$ and $\overline{Y}$, for all $\epsilon > 0$:

$$
\begin{aligned}
\lim_{n \longrightarrow \infty} \mathbf{P}\left\{|\overline{X} - \mathbf{E}\{S\}| < \epsilon\right\} &= 1 \\
\Rightarrow \mathbf{P}\left\{\lim_{n \longrightarrow \infty} \overline{X} = \mathbf{E}\{S\}\right\} &= 1
\end{aligned} \tag{24}
$$

Similarly:

$$
\mathbf{P}\left\{\lim_{n \longrightarrow \infty} \overline{Y} = \mathbf{E}\{T\}\right\} = 1 \tag{25}
$$

So when the limit is applied to Π_i^n, we get:

$$
\mathbf{P}\left\{\lim_{n \longrightarrow \infty} \Pi_0^n = \frac{\mathbf{E}\{S\}}{\mathbf{E}\{S\} + \mathbf{E}\{T\}}\right\} = 1 \tag{26}
$$

$$
\mathbf{P}\left\{\lim_{n \longrightarrow \infty} \Pi_1^n = \frac{\mathbf{E}\{T\}}{\mathbf{E}\{S\} + \mathbf{E}\{T\}}\right\} = 1 \tag{27}
$$

From our definition of an equilibrium distribution:

$$\lim_{n \longrightarrow \infty} \Pi_i^n = \pi_i \tag{28}$$

So we can now obtain π_0 and π_1:

$$\pi_0 = \frac{\mathbf{E}\{S\}}{\mathbf{E}\{S\} + \mathbf{E}\{T\}} \tag{29}$$

$$\pi_1 = \frac{\mathbf{E}\{T\}}{\mathbf{E}\{S\} + \mathbf{E}\{T\}} \tag{30}$$

2.7 Stochastic Normal Forms

2.7.1 Motivation

In this section, we show that general systems can be reduced to a normal form, using stochastic aggregation. The simplest of these normal forms is the Cox-Miller Normal Form, which we solved in the previous section. In the next section we will show how the other normal form cases can themselves be solved using the Cox-Miller result.

To understand the process of aggregation to the normal form, we should explain the stochastic motivation. This in turn will explain the structure of the normal forms.

Given any chosen state in a system of which we wish to find the steady-state probability, we have to maintain the stochastic characteristics of that state within the system. In a stochastic transition system, the system remains in a given state for a random amount of time; the exact distribution of this amount of time is defined by the distributions on the leaving transitions. Therefore, a state in such a system is defined by its out-edges and preserving the stochastic characteristic of the state means preserving its out-edges.

Under this restriction we would like to be able to simplify the remainder of the system as much as possible. The rest of this section is devoted to showing that for each out-edge from a state, the rest of the system can be stochastically aggregated to a single state with a single return edge. So we create a class of systems with the same number of 2-state cycles as there are out-edges from the state that we were originally carefully "aggregating around", shown in figure 2.

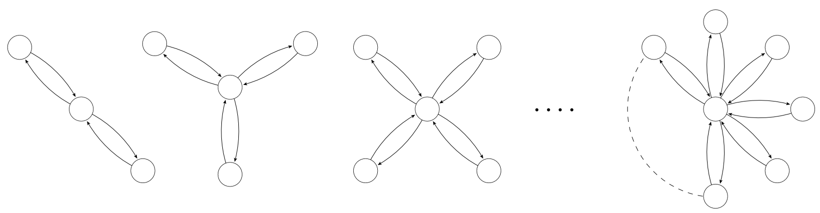

Figure 2: The set of normal forms for a state with finite branching degree.

In doing this, we have fulfilled our wish to preserve the stochastic nature of the state, since each out-edge still exists explicitly in the reduced system. We have also preserved the context of the state within the original system, since the "remainder of the system" behaviour is modelled in aggregated form by each of the return edges in the cycles. These edges effectively represent the aggregated system behaviour given that a particular out-edge was traversed.

2.7.2 Reduction Rules

Now we have a good idea of exactly what we want to reduce our system to, and why, we can abstract the problem to a completely graph-based domain for proof.

Five reduction rules are constructed and shown in figure 3.

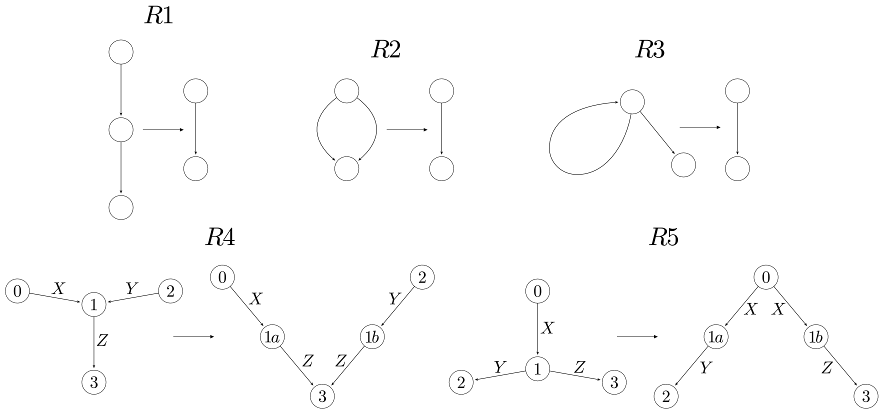

Figure 3: The digraph reduction rules.

R_1 represents the convolution rule of equation (15). R_2 represents the branching rule of equation (17). R_3 represents the terminating loop reduction of equation (19). R_4 and R_5 represent some simple laws of graph sharing which are true for a stochastic transition system.

We will quickly demonstrate R_4 and R_5 in stochastic terms:

R4	Before Reduction		After Reduction	
$0 \to 3$	$0 \to 1 \to 3$	$L_X(z)L_Z(z)$	$0 \to 1a \to 3$	$L_X(z)L_Z(z)$
$2 \to 3$	$2 \to 1 \to 3$	$L_Y(z)L_Z(z)$	$2 \to 1b \to 3$	$L_X(z)L_Z(z)$

This is just a reduction of graph sharing.

R5	Before Reduction		After Reduction	
$0 \to 2$	$0 \to 1 \to 2$	$L_X(z)pL_Y(z)$	$0 \to 1a \to 2$	$pL_X(z)L_Z(z)$
$0 \to 3$	$0 \to 1 \to 3$	$L_Y(z)qL_Z(z)$	$0 \to 1b \to 3$	$qL_X(z)L_Z(z)$

These extend easily to having many in-edges and out-edges into state 1, by appealing to the associativity of probabilistic branching.

2.7.3 Proof of Digraph Reduction to Normal Forms

In the argument below, a circuit is defined to be a cyclic path of a digraph with no repeated edges. There are a finite number of such circuits in a strongly connected digraph. We also define the *parents* of a vertex, v, to be the set of adjacent vertices which are connected to v via their out-edges. The *children* of v are the set of vertices which have v as a parent.

Let G be a strongly connected or biconnected digraph:

- It is possible to get to any vertex from any other vertex.
- For a chosen vertex v in G, v is a member of the set C of circuits which contain v and not a member of the set C' of circuits which do not contain v.
- $G = C \cup C'$
- We require to remove C' by aggregation and reduce C to one of the normal forms of figure 2, where the out-degree out-edges of v is preserved.

It can be seen that R2 and R3 reduce the number of circuits in a digraph, whereas R1, R4 and R5 preserve the number of circuits. By applying the reduction rules to a general digraph, we can only reduce or maintain the number of circuits.

We describe the following digraph reduction method, to remove C':

1. Select a chosen vertex v.
2. Remove all immediately reducible cycles with R2, R3.
3. Select the circuit $c \in C'$ of shortest length (so c does not contain v).
4. Select a vertex $x \in c$ and label as the root of the circuit. Let $u := x$:
   (a) Go to u', the child of u, in circuit c; if u' has more than one parent then split using R4 into a vertex with u as its only parent and a vertex with the remaining parents of u'.
   (b) Let $u :=$ child of vertex u, in c, whose parents we may just have increased. Repeat (4a) until all vertices in c, except x, have been processed (as in figure 4).
5. Essentially we now perform 4 in reverse using R5 (figure 5). As before, let $u := x$:
   (a) Go to u', the parent of u, in circuit c; if u' has more than one child then split using R5 into a vertex with u as its only child and a vertex with the remaining children of u'.

(b) Let $u :=$ parent of vertex u, in c, whose children we may just have increased. Repeat (5a) until all vertices in c, except x, have been processed.

6. Apply R1 to all the vertices of circuit c except x, until x is the only vertex left in c, with a single edge to itself.

7. Apply R3 to remove c from C'.

8. Repeat steps (3)–(7), until C' is empty.

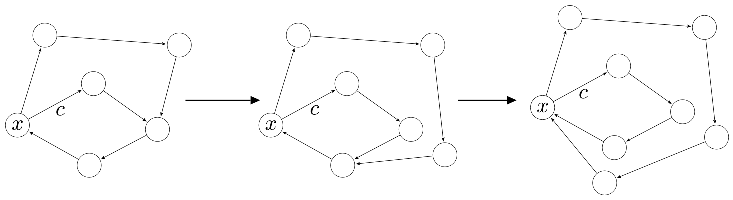

Figure 4: Rewriting the circuit c to have no vertices with multiple parents except x.

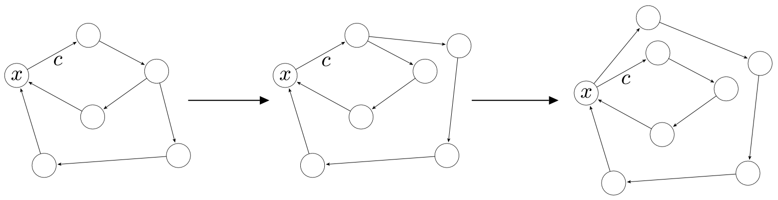

Figure 5: Rewriting the circuit c to have no vertices with multiple children except x.

Now we need to restructure G (which now consists entirely of C, i.e. circuits containing v), to look like one of the normal forms of figure 2. Our goal is to maintain the out-edges from v unaggregated.

1. Select the shortest circuit of length> 2 through v and reapply step (4)–(6) with $x := v$.

2. Apply R1 to all vertices of this circuit except v and child of v.

3. Apply R2 to remaining in-edges to v.

4. Repeat for each circuit through v until G is in normal form.

2.7.4 Solution of Normal Forms

In section 2.6, it was shown how to solve the simplest normal form, the Cox-Miller Normal Form. This normal form only occurs when aggregating around a state with a single out-transition.

We use this fact to solve the remaining normal forms. The normal forms of figure 2 each consist of a central vertex of out-degree $n \geq 2$. All the adjacent vertices to the central state, i.e. the next ones out along the out-edges, have out-degree 1.

This is the key. Using the stochastic aggregation technique of section 2.4, specifically the results from equations (19) and (15), we can take each of these out-degree 1 vertices in turn and aggregate to obtain a Cox-Miller Normal Form. It is a simple matter now to find the stationary probabilities for each of the degree 1 vertices and thus obtain the central vertex stationary probability by summing and subtracting from 1.

More specifically, label the central vertex with out-degree $n \geq 2$ as v. Label the n adjacent vertices to v, v_i for $1 \leq i \leq n$. For each v_i, aggregate to obtain a Cox-Miller Normal Form and solve for the stationary probability π_{v_i}. Now $\pi_v = 1 - \sum_{i=1}^{n} \pi_{v_i}$.

2.8 Worked Example

We now demonstrate the stochastic aggregation technique on the example state transition system shown in figure 6.

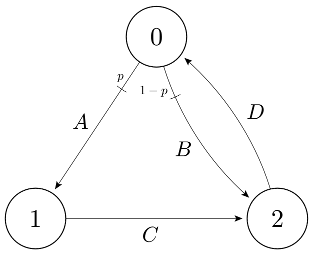

Figure 6: A generally distributed stochastic transition system.

$$
\begin{aligned}
A &\sim \Gamma(\alpha, \lambda) && (31)\\
B &\sim \text{Hyper}\left(\frac{2}{3}, \frac{1}{3}; \lambda, \mu\right) && (32)\\
C &\sim \text{U}\left[\frac{1}{\lambda}, \frac{1}{\mu}\right] && (33)\\
D &\sim \delta\left[\frac{1}{\mu}\right] && (34)
\end{aligned}
$$

where $0 < \mu < \lambda$ and $\alpha > 0$. A has a Gamma distribution; B has an hyperexponential distribution; C has a uniform distribution; and D is a deterministic process which occurs at $t = 1/\mu$ with probability 1. The random variables have the following transforms:

$$
\begin{aligned}
L_A(z) &= \left(\frac{\lambda}{\lambda+z}\right)^{\alpha} && (35)\\
L_B(z) &= \frac{2}{3}\frac{\lambda}{\lambda+z}+\frac{1}{3}\frac{\mu}{\mu+z} && (36)\\
L_C(z) &= \frac{\lambda\mu}{(\lambda-\mu)z}\left[e^{-z/\lambda}-e^{-z/\mu}\right] && (37)\\
L_D(z) &= e^{-z/\mu} && (38)
\end{aligned}
$$

First we select state 1, then aggregate around the state, to create the combined process ψ_1, as displayed in figure 7. Using the stochastic aggregation rules (15,19):

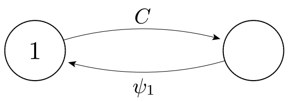

Figure 7: Aggregating states 0 and 2 to give a Cox-Miller Normal Form for state 1.

$$
L_{\psi_1}(z) = \frac{pL_A(z)L_D(z)}{1-(1-p)L_B(z)L_D(z)} \tag{39}
$$

The aggregated system for state 1 is now in Cox-Miller Normal Form. We now have to extract the expectations from the transforms in order to obtain the stationary probability for state 1:

$$
\mathbf{E}\{C\} = -\frac{d}{dz}L_C(z)\bigg|_{z=0} = \frac{1}{2}\left(\frac{1}{\mu}+\frac{1}{\lambda}\right) \tag{40}
$$

$$
\mathbf{E}\{\psi_1\} = -\frac{d}{dz}L_{\psi_1}(z)\bigg|_{z=0} = \frac{((3\alpha-2)\mu-\lambda)p+2(2\lambda+\mu)}{3\lambda\mu p} \tag{41}
$$

Now using equation (29), the stationary distribution of state 1 can be calculated:

$$
\pi_1 = \frac{\mathbf{E}\{C\}}{\mathbf{E}\{C\}+\mathbf{E}\{\psi_1\}} = \frac{3(\lambda+\mu)p}{(\lambda+6\alpha\mu-\mu)p+8\lambda+4\mu} \tag{42}
$$

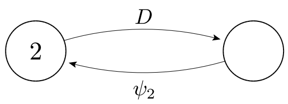

Figure 8: Aggregating states 0 and 1 to give a Cox-Miller Normal Form for state 2.

Undertaking a similar aggregation for state 2, shown in figure 8, we obtain:

$$L_{\psi_2}(z) = pL_A(z)L_C(z) + (1-p)L_B(z) \tag{43}$$

Thus:

$$\pi_2 = \frac{\mathbf{E}\{D\}}{\mathbf{E}\{D\} + \mathbf{E}\{\psi_2\}} = \frac{6\lambda}{(\lambda + 6\alpha\mu - \mu)p + 8\lambda + 4\mu} \tag{44}$$

At this stage, we can deduce that:

$$\pi_0 = 1 - (\pi_1 + \pi_2) = 2\frac{(3\alpha\mu - 2\mu - \lambda)p + \lambda + 2\mu}{(\lambda + 6\alpha\mu - \mu)p + 8\lambda + 4\mu} \tag{45}$$

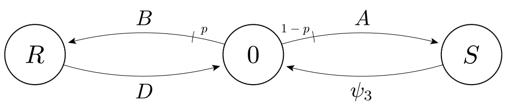

Figure 9: Aggregating around state 0 to get a more complicated normal form.

However the full calculation of π_0 will be shown in order to demonstrate how states which don't aggregate directly to Cox-Miller Normal Forms can be solved. In this case, it is only possible to aggregate to a 2-cycle normal form (figure 9), since state 0 has 2 out-edges that we wish to preserve. ψ_3 is given by:

$$L_{\psi_3}(z) = L_C(z)L_D(z) \tag{46}$$

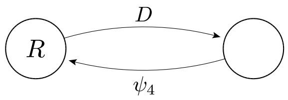

Figure 10: Aggregating around state R, achieving a CMNF structure.

Now we treat the normal form of figure 9 just as any other system and aggregate to obtain Cox-Miller Normal Forms for the out-degree 1 states R and S. The R aggregate system in shown in figure 10, where:

$$L_{\psi_4}(z) = \frac{(1-p)L_B(z)}{1 - pL_A(z)L_{\psi_3}(z)} \tag{47}$$

Now:

$$\pi_R = \frac{\mathbf{E}\{D\}}{\mathbf{E}\{D\} + \mathbf{E}\{\psi_4\}} = \frac{6\lambda(1-p)}{(\lambda + 6\alpha\mu - \mu)p + 8\lambda + 4\mu} \tag{48}$$

Similarly for state S in figure 11 where:

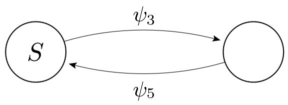

Figure 11: Aggregating around state S, achieving a CMNF structure.

$$L_{\psi_5}(z) = \frac{pL_A(z)}{1-(1-p)L_B(z)L_D(z)} \tag{49}$$

Thus:

$$\pi_S = \frac{\mathbf{E}\{\psi 3\}}{\mathbf{E}\{\psi_3\}+\mathbf{E}\{\psi_5\}} = \frac{3(3\lambda+\mu)p}{(\lambda+6\alpha\mu-\mu)p+8\lambda+4\mu} \tag{50}$$

and:

$$\pi_0 = 1-(\pi_R+\pi_S) = 2\frac{(3\alpha\mu-2\mu-\lambda)p+\lambda+2\mu}{(\lambda+6\alpha\mu-\mu)p+8\lambda+4\mu} \tag{51}$$

which agrees with our earlier answer of equation (45).

3 Applications

3.1 Generally Distributed Process Algebras

By finding a mapping between a generally distributed process algebra and a stochastic transition system, we would immediately achieve a solution technique for a generally distributed process algebra, such as:

$$S ::= \quad (x, F_X(t)).S \mid S+S \mid S||S \mid A \tag{52}$$

The problem of solving a generally distributed process algebra with the full complement of major operators is a difficult one and should not be in any way underestimated.

Given that the problem of solving a simple stochastic transition system with general distributions was reasonably straightforward, the difficulty in the process algebras problem must lie in the task of mapping such a process algebra onto its equivalent stochastic state space.

This does indeed turn out to be the case, and the mapping problem is demonstrated in the next section with a generally distributed queue example.

As we might have expected, this problem is a result of our side-stepping the memory problem that is encountered when modelling concurrent general distributions. This means that we will be able to model a subset of a process algebra, which does not contain the concurrent composition operator:

$$S ::= \quad (x, F_X(t)).S \mid S + S \mid A \tag{53}$$

This is unfortunate, given the power of the concurrent operator, but necessary to start with. As we will see, it is does not mean that concurrent processes cannot be modelled at all. In a restricted set of circumstances, concurrent processes can be dealt with. However, for the time being, concurrency in its most general form cannot be treated in the same direct manner as the other operators.

3.1.1 Constructing a Stochastic Transition System

The rules for mapping a restricted generally distributed process algebra (equation (53)) to its stochastic transition system are as follows:

$$\begin{aligned} (x, F_X(t)).P &\longrightarrow \{X\}.P && (54) \\ (x, F_X(t)).P + (y, F_Y(t)).Q &\longrightarrow [\mathbf{P}\{X < Y\}]\{X | X < Y\}.P \\ &\qquad + [\mathbf{P}\{Y < X\}]\{Y | Y < X\}.Q && (55) \end{aligned}$$

Equation (54) is fairly self-explanatory. In the process algebra version, it describes a transition labelled x which proceeds with random variable X (cumulative distribution function, $F_X(t)$), to a new state, P. In the stochastic transition system, the label is discarded and the process is represented by the evolution of the random variable X.

Equation (55) describes the mapping of competitive choice onto probabilistic branching and does deserve some discussion.

3.1.2 Competitive Choice

Process algebras have long used a mechanism called competitive choice for describing branching or choice points in a transition system. When a choice is to be made, two processes enter what is known as a *race condition*; whichever of the two finishes first is the successful branch. The losing or slower process is interrupted and discarded. Given that this is a popular representation of process choice it is as well to consider it, if only for reasons of compatibility. In the event that a process algebra were to use probabilistic choice then this would obviously have a much more straightforward conversion to the probabilistic branching model of the stochastic transition system.

Although this stochastic mapping has been used before in TIPP [9], it is worth proving a couple of properties to justify its use. The first is an observational property: that is, that the total time that a process spends in a competitive choice should be distributed as the minimum of the two competing processes.

The second is that of associativity and its implications for the general n-way choice.

Calculating the minimum distribution of two independent random variables, we use the fact that: $\min(X, Y) > t \Leftrightarrow X > t$ and $Y > t$:

$$\begin{aligned} F_{\min(X,Y)}(t) &= \mathbf{P}\{\min(X,Y) \le t\} \\ &= 1 - \mathbf{P}\{\min(X,Y) > t\} \\ &= 1 - \mathbf{P}\{X > t, Y > t\} \\ F_{\min(X,Y)}(t) &= 1 - \overline{F}_X(t)\overline{F}_Y(t) \end{aligned} \tag{56}$$

where:

$$\overline{F}_X(t) = \int_t^\infty f_X(x)\, dx = 1 - F_X(t) \tag{57}$$

Let us say that competitive choice between two random variables is represented by the notation:

$$C = X \oplus Y \tag{58}$$

$$\begin{aligned} F_C(t) &= \mathbf{P}\{X < Y\} F_{X|X<Y}(t) + \mathbf{P}\{Y < X\} F_{Y|Y<X}(t) \\ &= \mathbf{P}\{X \le t, X < Y\} + \mathbf{P}\{Y \le t, Y < X\} \\ &= \int_0^\infty \overline{F}_Y(u) f_X(u) + \overline{F}_X(u) f_Y(u)\, du \\ &= \int_0^t -\frac{d}{du}\left[\overline{F}_X(u)\overline{F}_Y(u)\right]\, du \end{aligned} \tag{59}$$

Since $\overline{F}_X(0)\overline{F}_Y(0) = 1$:

$$F_C(t) = 1 - \overline{F}_X(t)\overline{F}_Y(t) \tag{60}$$

From this we get the associativity result:

$$F_{X\oplus(Y\oplus Z)}(t) = 1 - \overline{F}_X(t)\overline{F}_Y(t)\overline{F}_Z(t) = F_{(X\oplus Y)\oplus Z}(t) \tag{61}$$

From this and [9], we state (it is easily shown) the general form of competitive choice. For $C = X_1 \oplus \cdots \oplus X_n$:

$$\begin{aligned} F_C(t) &= \mathbf{P}\{X_1 < X_2, \ldots, X_1 < X_n\} F_{X|X_1<X_2,\ldots,X_1<X_n}(t) + \cdots \\ &\quad + \mathbf{P}\{X_n < X_1, \ldots, X_n < X_{n-1}\} F_{X|X_n<X_1,\ldots,X_n<X_{n-1}}(t) \\ &= 1 - \Pi_{i=1}^n \overline{F}_{X_i}(t) \end{aligned} \tag{62}$$

Thus, competitive choice between n processes can be mapped pairwise onto the equivalent probabilistic branching in the stochastic transition system.

3.2 Queueing Systems

3.2.1 M/G/1/K and G/G/1/K Queues

The first attempt at using the stochastic aggregation technique was applied to queueing systems. Queueing theory is one of the few theories of stochastic processes where generally distributed transitions occur and can be solved using analytical methods. As such they were a useful and ultimately successful testing ground for the technique.

In this section we demonstrate a mapping onto a stochastic transition system for a G/G/1/2 queue. We provide a general analytic solution and then specialise it to a specific M/G/1/2 case. It is shown that the solution from the embedded Markov Chain technique is identical to that produced by stochastic aggregation.

We further show that this technique cannot be applied to larger G/G/1/K queues, since the equivalent stochastic state space is infinite in size.

3.2.2 An Analytic Solution of a G/G/1/2 Queue

In this section, we describe a 3-state single server queue with independent arrival process X and service process Y (figure 12). Converting this high level description to the stochastic transition system of figure 13 requires a little explanation.

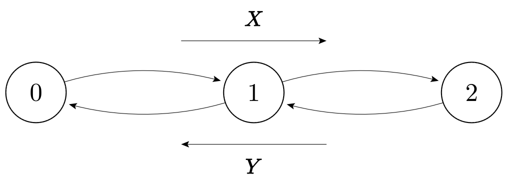

Figure 12: The state diagram for a buffer of a G/G/1/2 queue.

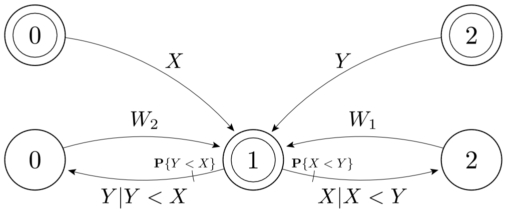

Figure 13: The stochastic transition system for a G/G/1/2 queue.

To start with, it is possible to start in any state in the system, however these start states (indicated by concentric circles in figure 13) are often different in stochastic terms to the same states during system operation. For instance, if the queue starts empty then the next transition is going to be an arrival sampled from X, however if the queue becomes empty during operation, the next arrival will be a different sample. On this occasion, the sample represents the remaining time left given that the X arrival process started while the system was in state 1 but was pre-empted by a service by the queue. This distribution W_2 and its counterpart W_1 will be defined precisely later.

The start states of 0 and 2 play no further part in any realisation (trace) since the system immediately moves to the recurrent states[2] on which the stationary distribution is defined.

With this particular queueing model we are able to do something we cannot do in general; that is, model concurrent processes. From state 1, X and Y always start afresh and spawn two processes each, according to whether $X < Y$ or $Y < X$. This is explained by a partitioning argument to construct the conditional probabilities and maps easily onto the probabilistic branching paradigm:

$$A_X = (A_X \cap B) \cup (A_X \cap \overline{B}) \tag{63}$$

where $A_X = \{\omega : X(\omega) \leq t\}$, $B = \{\omega : X(\omega) < Y(\omega)\}$ where ω is a sample point. Applying the probability measure to the set of events, A_X, we get:

$$\begin{aligned} F_X(t) &= \mathbf{P}\{X \leq t, X < Y\} + \mathbf{P}\{X \leq t, Y < X\} \\ &= \mathbf{P}\{X < Y\}\mathbf{P}\{X \leq t | X < Y\} + \mathbf{P}\{Y < X\}\mathbf{P}\{X \leq t | Y < X\} \\ &= \mathbf{P}\{X < Y\} F_{X|X<Y}(t) + \mathbf{P}\{Y < X\} F_{X|Y<X}(t) \end{aligned} \tag{64}$$

Similarly:

$$F_Y(t) = \mathbf{P}\{X < Y\} F_{Y|X<Y}(t) + \mathbf{P}\{Y < X\} F_{Y|Y<X}(t) \tag{65}$$

So in strict order we either get samples from $X|X < Y$ and then $Y|X < Y$ or $Y|Y < X$ and then $X|Y < X$. This is not quite the end of the story since we also have the memory issue to contend with now. In figure 13, in going from state 1 through 0 to 1 again, the whole event from leaving state 1 to returning to state 1 represents the larger sample from $X|Y < X$, and the transition from 1 to 0 represents the sample from $Y|Y < X$ which happened first. So the second transition in that path, labelled W_2, represents a type of "difference" between the two distributions; the nature of this difference operation is described below.

We define the operator $\ominus$ on a pair of independent positive random variables as follows:

$$C = A \ominus B \tag{66}$$

such that, in terms of Laplace transforms:

[2] In the language of Markov Chains; in our system, this is the biconnected subgraph.

$$L_C(z) = \frac{L_A(z)}{L_B(z)} \tag{67}$$

Thus C is defined as the positive random variable which, when convolved with B, will give A.

This now allows us to express W_1 and W_2 in terms of X and Y:

$$W_1 = (Y|X < Y) \ominus (X|X < Y) \tag{68}$$

$$W_2 = (X|Y < X) \ominus (Y|Y < X) \tag{69}$$

Now, using the reduction equation (19) to preserve the out-transition from state 0, we can generate the ψ_1 distribution of figure 14.

$$\begin{aligned} L_{\psi_1}(z) &= \frac{\mathbf{P}\{Y < X\} L_{Y|Y<X}(z)}{1 - \mathbf{P}\{X < Y\} L_{X|X<Y}(z) L_{W_1}(z)} \\ &= \frac{\mathbf{P}\{Y < X\} L_{Y|Y<X}(z)}{1 - \mathbf{P}\{X < Y\} L_{Y|X<Y}(z)} \end{aligned} \tag{70}$$

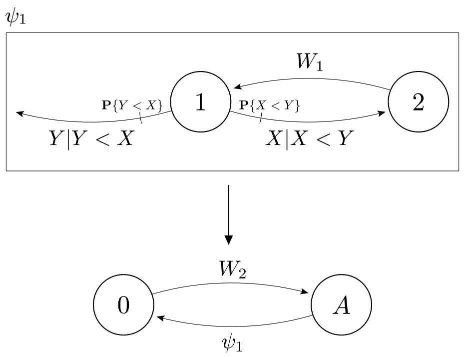

Figure 14: Aggregating the states 1 and 2 to produce a new state A and an aggregated transition ψ_1.

The expectation of a distribution is easily obtained from its Laplace transform:

$$\mathbf{E}\{A\} = -\frac{d}{dz} L_A(z)\Big|_{z=0} \tag{71}$$

This is all that is required to find out the stationary distribution of state 0, since the aggregated system of figure 14 is just the Cox-Miller Normal Form of section 2.6. Thus from equation (29), we obtain:

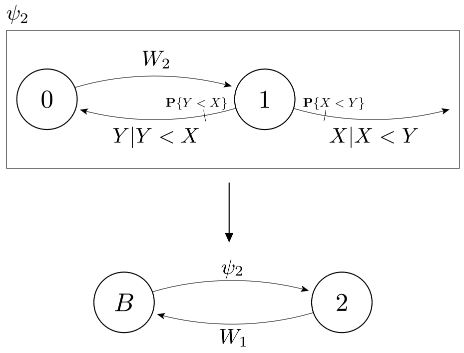

Figure 15: Aggregating the states 0 and 1 to produce a new state B and an aggregated transition ψ_2.

$$\pi_0 = \frac{\mathbf{E}\{W_2\}}{\mathbf{E}\{W_2\} + \mathbf{E}\{\psi_1\}} \tag{72}$$

Similarly, reducing the system to preserve state 2, we get the aggregated transition, ψ_2 in figure 15:

$$L_{\psi_2}(z) = \frac{\mathbf{P}\left\{X < Y\right\} L_{X|X<Y}(z)}{1 - \mathbf{P}\left\{Y < X\right\} L_{X|Y<X}(z)} \tag{73}$$

Now having obtained a second Cox-Miller Normal Form, π_2 follows:

$$\pi_2 = \frac{\mathbf{E}\{W_1\}}{\mathbf{E}\{W_1\} + \mathbf{E}\{\psi_2\}} \tag{74}$$

Finally:

$$\pi_1 = 1 - (\pi_0 + \pi_2) \tag{75}$$

3.2.3 M/G/1/2 Example

Specialising the G/G/1/2 result to use a Markovian arrival process, simplifies some of the expressions. Importantly the end result can be checked using standard M/G/1/K queueing techniques.

Let the arrival process X and service process Y be defined as follows:

$$f_X(t) = \lambda e^{-\lambda t} \tag{76}$$
$$f_Y(t) = \alpha e^{-\alpha t} + \beta e^{-\beta t} - (\alpha+\beta) e^{-(\alpha+\beta)t} \tag{77}$$

Now, we can specify the exact transition distributions of figure 13, and thus from equations (72, 75, 74), the stationary distribution of the queue:

$$\begin{aligned}
\pi_0 &= \alpha^2\beta^2(\alpha+\beta)(\alpha+\beta+2\lambda)\chi && (78)\\
\pi_1 &= \alpha\beta\lambda(\alpha+\beta)((\alpha+\beta+\lambda)^2-\alpha\beta)\chi && (79)\\
\pi_2 &= ((\alpha^2+\alpha\beta+\beta^2)\lambda^2+(\alpha+\beta)(2\alpha^2+\alpha\beta+2\beta^2)\lambda \\
&\quad +\alpha^4+2\alpha^3\beta+\alpha^2\beta^2+2\alpha\beta^3+\beta^4)\chi && (80)
\end{aligned}$$

where:

$$\begin{aligned}
\chi &= \big[\,(\alpha^2+\alpha\beta+\beta^2)\lambda^4+2(\alpha+\beta)(\alpha^2+\alpha\beta+\beta^2)\lambda^3 \\
&\quad +(\alpha^2+\alpha\beta+\beta^2)(\alpha^2+3\alpha\beta+\beta^2)\lambda^2 \\
&\quad +\alpha\beta(\alpha+\beta)(\alpha^2+3\alpha\beta+\beta^2)\lambda+\alpha^2\beta^2(\alpha+\beta)^2\,\big]^{-1} && (81)
\end{aligned}$$

These results are identical to those obtained using embedded Markov Chain techniques from traditional queueing theory [5, 15, 10].

3.2.4 Larger G/G/1/K Queues

So, can the same techniques be applied to larger general queueing systems? The answer, probably unsurprisingly, is no; not directly at least. To see why, we need to look at how the processes can interleave. Figure 16 shows an interleaving for a G/G/1/3 queue. The synchronising states (states from which both arrival and service distributions start simultaneously) are marked by concentric circles.

This is not the stochastic state space of other diagrams, it instead represents how the processes interleave by showing how many services (or arrivals) have occurred while an arrival (or service) has still to occur. It is a distinct diagram from that of the normal queue buffer representation. For example, observe that there are distinct instances of the state with one item in the queue: one instance when both the arrival and service processes start simultaneously and one when the service process Y is starting again (having just occurred) while the arrival process X has been running for a single service time. Whether the arrival process happens next depends now on a race between a fresh service process and a one-service-time-old arrival process.

However, as already stated, this is not a stochastic transition system either. The reason is that although this represents a consistent interleaving, the distributions of the individual edges are not necessarily the same each time that edge is traversed. Specifically, for the cycle between $2a$ and $1b$, each time that cycle is traversed without first going through a synchronising state, a distinct distribution is sampled. This is where the memory of the system catches up with us. Until a synchronising state is passed through[3], the distribution of subsequent transitions reflects the entire trace history of the system. So having a cycle between two non-synchronising states, in effect generates an infinite stochastic state space.

[3]Synchronising states are marked with concentric circles. They represent states for which no history of past evolution has to be maintained.

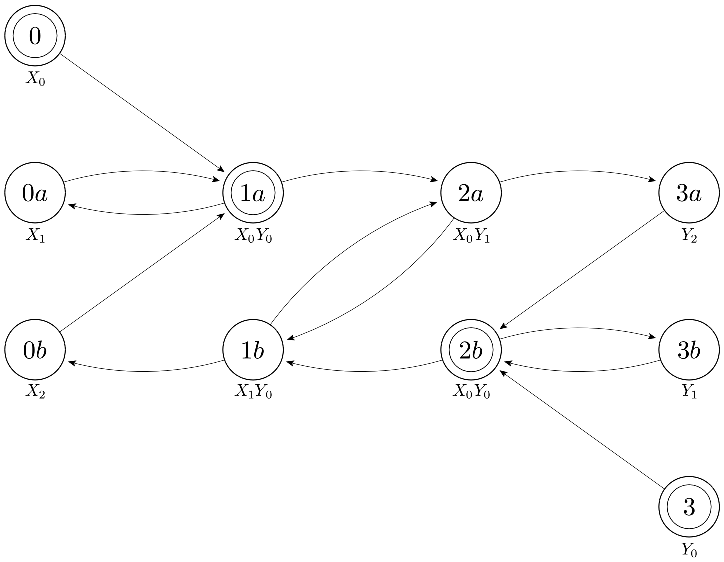

Figure 16: The process interleaving for a G/G/1/3 queue, taking into account how many evolutions of the opposing process have occurred.

Table 1: The distributions of successive transitions along the $2a$, $1b$ cycle.

n	Transition	Win	Service, $g_n(X,Y)$	Arrival, $h_n(X,Y)$
1	$2a \longrightarrow 1b$	S	$g_1(X,Y)$	$h_1(X,Y)$
2	$1b \longrightarrow 2a$	A	$Y\|k_1(X,Y) < Y$	$k_1(X,Y)\|k_1(X,Y) < Y$
3	$2a \longrightarrow 1b$	S	$j_2(X,Y)\|j_2(X,Y) < X$	$X\|j_2(X,Y) < X$
4	$1b \longrightarrow 2a$	A	$Y\|k_3(X,Y) < Y$	$k_3(X,Y)\|k_3(X,Y) < Y$
5	$2a \longrightarrow 1b$	S	$j_4(X,Y)\|j_4(X,Y) < X$	$X\|j_4(X,Y) < X$
			$\vdots$	

To see this, let us assume that at some stage during operation a large number of cycles occur between $2a$ and $1b$ states. For this to happen the system must have alternating arrivals and services. The system starts with a service from state $2a$ and for simplicity we assign $g_1(X,Y)$ and $h_1(X,Y)$ as the form of distributions of the service and arrival processes which come out of the competitive choice at $2a$ the first time. The first few evolutions of this cycle are presented in table 3.2.4 where $j_n(X,Y) = g_n(X,Y) \ominus h_n(X,Y)$, $k_n(X,Y) = h_n(X,Y) \ominus g_n(X,Y)$.

In general:

$$\begin{aligned}
g_{2n}(X,Y) &= Y|k_{2n-1}(X,Y) < Y \\
h_{2n}(X,Y) &= k_{2n-1}(X,Y)|k_{2n-1}(X,Y) < Y \\
g_{2n+1}(X,Y) &= j_{2n}(X,Y)|j_{2n}(X,Y) < X \\
h_{2n+1}(X,Y) &= X|j_{2n}(X,Y) < X
\end{aligned} \tag{82}$$

As a result of this dependence on the previous distribution, no successive transitions along the same edge will carry the same distribution and the stochastic state space effectively has to encode the history of every competitive choice outcome, until a synchronisation state is reached again. At this stage the memory is reset.

What made the G/G/1/2 case tractable was that it passed through a synchronising state (the middle 1 state in figure 13) every two transitions. Hence there was a finite amount of history to encode in the distributions, and hence a finite stochastic state space.

3.3 Markovian Process Algebras

Markovian Process Algebras model only Markov processes. This allows them to take advantage of many useful simplifications, which effectively make the general systems solvable in the general case.

It has already been shown that competitive choice can in the generally distributed case can be mapped onto the probabilistic branching of a stochastic transition system. Obviously competitive choice of Markov processes can be similarly mapped. The important difference to note from that of generally distributed SPAs and generally distributed queueing systems, is that an expansion law exists for parallel composition.

For PEPA an expansion law is found in [13]; for MTIPP, one can be found in [12].

The expansion law for PEPA, for instance, maps the different components of the composition onto a finite Markovian state space consisting of competitive choices, in a similar fashion to CCS. We are in the luxurious position of saying that the hard work has therefore been done for us. This mapping represents a proof that the stochastic aggregation technique can be used directly to solve any Markovian process algebra.

So long as the parallel composition has been removed using the expansion law mentioned, we are able to map the components of a typical MPA onto its stochastic transition system loosely as follows:

$$(x,\lambda).P \longrightarrow \{X\}.P \tag{83}$$

$$(x,\lambda).P + (y,\mu).Q \longrightarrow \left[\frac{\lambda}{\lambda+\mu}\right]\{Z\}.P + \left[\frac{\mu}{\lambda+\mu}\right]\{Z\}.Q \tag{84}$$

where $X \sim \exp(\lambda)$, $Y \sim \exp(\mu)$ and $Z \sim \exp(\lambda+\mu)$.

The immediate corollary of this is that an MPA no longer has to proceed via a Markov Chain solution technique in order to obtain its stationary distribution. Since it can be mapped onto a finite stochastic transition system, the stochastic aggregation technique can be applied. An example of this is shown below.

3.3.1 Reward Structures

What motivation might there be to use stochastic aggregation over the standard matrix methods of Markov Chains? Assuming, at this stage, that the complexities of the two approaches are similar, the major benefit is that the stationary distribution can be generated in a state-wise fashion.

This has a nice effect for calculating reward structures and performance measures, for MPAs. If a reward structure is sparse, i.e. it only rewards a few states, then the aggregation technique can be used to find only the stationary probabilities of the required states. Using traditional techniques, the entire model would have to be solved in order to calculate maybe only a few stationary values.

A second property of the method is that it is a progressive technique. It will generate the distribution, a result at a time, rather than the entire solution in one go. This may well make it amenable to lazy implementation.

3.3.2 Example

As an example of solving a Markovian Process Algebra system, we examine the PEPA resource usage model of [13]:

$$\begin{aligned}
Process &\stackrel{\text{def}}{=} (use, r_1).(task, r_2).Process \\
Resource &\stackrel{\text{def}}{=} (use, r_3).(update, r_4).Resource \\
System &\stackrel{\text{def}}{=} Process \bowtie_{\{use\}} Resource
\end{aligned} \tag{85}$$

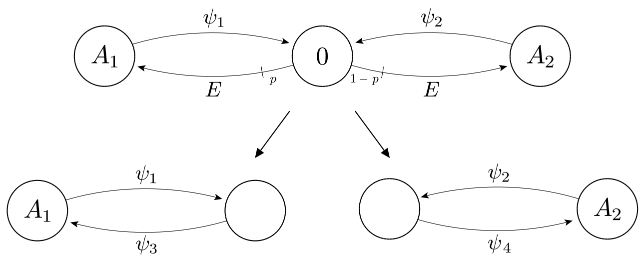

Figure 17: Using stochastic aggregation on a PEPA model with cooperation.

The cooperation is removed using the expansion law of [13], giving the stochastic transition system of the model in figure 17. The first stochastic aggregation manœuvre reduces the transition system to the normal form for state 0, where:

$$L_E(z) = \frac{r_2 r_4}{(r_2+z)(r_4+z)} \tag{86}$$

$$L_{\psi_1}(z) = \frac{r_4 r_{13}}{(r_4+z)(r_{13}+z)} \tag{87}$$

$$L_{\psi_2}(z) = \frac{r_2 r_{13}}{(r_2+z)(r_{13}+z)} \tag{88}$$

$$p = \frac{r_2}{r_2+r_4} \tag{89}$$

Note that $r_{13} = \min(r_1, r_3)$. Now we aggregate around states A_1 and A_2 to give:

$$L_{\psi_3}(z) = \frac{\frac{r_2}{r_2+r_4+z}}{1 - \frac{r_2 r_4 r_{13}}{(r_2+z)(r_{13}+z)(r_2+r_4+z)}} \tag{90}$$

$$L_{\psi_4}(z) = \frac{\frac{r_4}{r_2+r_4+z}}{1 - \frac{r_2 r_4 r_{13}}{(r_4+z)(r_{13}+z)(r_2+r_4+z)}} \tag{91}$$

A_1 and A_2 are now in Cox-Miller Normal Form and so we can get π_{A_1} and π_{A_2} from equation (29):

$$\pi_{A_1} = \frac{r_2^2(r_{13}+r_4)}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{92}$$

$$\pi_{A_2} = \frac{r_4^2(r_{13}+r_2)}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{93}$$

And now we can retrieve $\pi_0 = 1 - (\pi_{A_1} + \pi_{A_2})$:

$$\pi_0 = \frac{r_2 r_4 r_{13}}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{94}$$

This agrees with the result in Hillston 1994 [13, p.59] for the equivalent state X_1 which was obtained using Gaussian Elimination. The other stationary probabilities are similarly obtained using stochastic aggregation:

$$\pi_1 = \frac{r_2^2 r_{13}}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{95}$$

$$\pi_2 = \frac{r_4^2 r_{13}}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{96}$$

$$\pi_3 = \frac{r_2 r_4(r_2+r_4)}{r_2 r_4(r_2+r_4) + r_{13}(r_2^2 + r_2 r_4 + r_4^2)} \tag{97}$$

4 Conclusion

In this paper, we have shown that a class of generally distributed systems can be solved. By restating a well-known aggregation technique in the language of stochastic process algebras, we can reduce any stochastic transition system to a normal form for a given state. These normal forms, in turn, can be solved by using the Cox-Miller Normal Form solution. For each state of the original system the stationary distribution can be recovered, in a progressive and as-required fashion.

The generally distributed stochastic transition systems that are solved by this work purposefully have no concept of concurrency. This was a fundamental simplification that removed the major problems of distribution memory. However, it was shown that this does not stop stochastic transition systems from being able to model concurrent systems from other paradigms.

We were able to present an analytical solution to a G/G/1/2 queueing system. Further, we demonstrated that the reason that larger queues were not directly soluble using this technique was that no finite representation existed as a stochastic transition system. Since Markovian Process Algebras have an expansion law for their concurrent operator, however, it is possible to write down a finite stochastic transition system for a general MPA system. This fact means that we are able to solve MPA systems without having to resort to Markov Chains. This is an advantage since it means that, in order to gain a few stationary probabilities, we no longer have to solve the entire system.

The use of an aggregation technique means that we are no longer restricted to dealing in stationary distributions. The use of Laplace transforms has the property that we can easily derive the variance of the aggregated states as well as the mean. This will be useful for deriving reliability information.

The aggregation technique based on reducing the circuits in the graph is used here as an outline proof-method. Further work is required to develop an algorithm which could be embedded inside a tool.

Further research is being carried out into what other types of system have finite stochastic transition systems and can then be solved by this technique. Preliminary results suggest that M/G-type queueing networks will have a finite expansion. Finally, in an effort to create a more general solution technique, we are also trying to develop methods for dealing with concurrent processes within this framework.

Acknowledgements

The authors would like to thank: Helen Wilson and Ian Holyer for their valuable contribution to the graph theoretic aspects, and Judy Holyer for being a sounding-board for the mathematical concepts involved. This work is funded by NATS, a division of the Civil Aviation Authority.

References

[1] Bernardo, M., and Gorrieri, R. Extended markovian process algebra. In *CONCUR'96, Proceedings of the 7th International Conference on Concurrency Theory* (1996), no. 1119 in LNCS, pp. 315–330.

[2] Bravetti, M., Bernardo, M., and Gorrieri, R. Generalized semi-Markovian process algebra. UBLCS Report UBLCS–97–9, Department of Computer Science, University of Bologna, 40127 Bologna, Italy, October 1998.

[3] Brinksma, E., Katoen, J.-P., Langerak, R., and Latella, D. A stochastic causality-based process algebra. In *PAPM'95, Proceedings of the 3rd Workshop of Process Algebras and Performance Modelling* (1995), pp. 553–565.

[4] Buchholz, P. Markovian process algebra: Composition and equivalence. In *PAPM'96, Proceedings of the 4th Workshop of Process Algebras and Performance Modelling* (1996), pp. 11–30.

[5] Cohen, J. W. *The Single Server Queue*, vol. 8 of *Applied Mathematics and Mechanics*. North-Holland, 1969.

[6] Cox, D. R., and Miller, H. D. *The Theory of Stochastic Processes*. Methuen, 1965.

[7] Feller, W. *An Introduction to Probability Theory and its Applications*, 3rd ed. ed., vol. 1. Wiley International, 1968.

[8] Gibbons, A. *Algorithmic Graph Theory*. Cambridge University Press, 1985.

[9] Götz, N., Herzog, U., and Rettelbach, M. TIPP—Introduction and application to protocol performance analysis. In *Formale Beschreibungstechniken für verteilte Systeme* (Munich 1993), H. König, Ed., FOKUS, Saur-Verlag.

[10] Gross, D., and Harris, C. M. *Fundamentals of Queueing Theory*, 3rd ed. John Wiley & Sons, 1998.

[11] Henderson, W., and Lucic, D. Aggregation and disaggregation through insensitivity in stochastic petri nets. *Performance Evaluation 17* (1993), 91–114.

[12] Hermanns, H., and Rettelbach, M. Syntax, semantics, equivalences, and axioms for MTIPP. In *PAPM'94, Proceedings of the 2nd International Workshop on Process Algebras and Performance Modelling* (Regensberg, July 1994), U. Herzog and M. Rettelbach, Eds., Universtät Erlangen-Nürnberg.

[13] Hillston, J. *A Compositional Approach to Performance Modelling*. PhD thesis, Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, UK, 1994.

[14] Hoare, C. A. R. *Communicating Sequential Processes.* PHI Series in Computer Science. Prentice Hall, 1985.

[15] Keilson, J. The ergodic queue length distribution for queueing systems with finite capacity. *Journal of the Royal Statistical Society B*, 28 (1966), 190–201.

[16] Kemeny, J. G., and Snell, J. L. *Finite Markov Chains.* Van Nostrand, 1960.

[17] Kleinrock, L. *Queueing Systems, Volume I: Theory.* John Wiley & Sons, 1975.

[18] Kleinrock, L. *Queueing Systems, Volume II: Computer Applications.* John Wiley & Sons, 1976.

[19] Milner, R. *Communication and Concurrency.* PHI Series in Computer Science. Prentice Hall, 1989.

[20] Priami, C. A stochastic π-calculus with general distributions. In *PAPM'96, Proceedings of the 4th Workshop of Process Algebras and Performance Modelling* (1996), pp. 41–57.

[21] Trivedi, K. S. *Probability and Statistics with Reliability, Queuing and Computer Science Applications.* Prentice-Hall, 1982.