Turning Back Time - What Impact on Performance?

Peter G. Harrison
Department of Computing
Imperial College London

Abstract

Consistent with the divide-and-conquer approach to problem solving, a recursive result is presented in the domain of stochastic modelling that derives product-form solutions for the steady state probabilities of certain networks composed from interacting Markov chains. Practical applications include multi-tasking operating systems, communication channels and multi-tiered storage systems. The approach is also applied to the computation of response time quantiles, which are vital in transaction processing, computer communication service level agreements and other operational systems. The joint probability distribution of the sojourn times of a tagged task at each node in a network is determined by noting that this is the same in both the forward and reversed processes. In this way, existing results for response time probability densities in tandem, tree-like, and overtake-free Markovian queueing networks are quickly and systematically obtained. We further show how to apply the method in more general networks.

1 Introduction

The divide-and-conquer approach to problem solving is standard in software design and other engineering disciplines. Immense improvements in efficiency can result in the computation of numerical quantities since the complexity (e.g. number of states) of a complete system is typically of the order of the product of the complexities of its subsystems. In stochastic modelling, subsystems can often be solved either by direct methods or in a simple closed form – for example in many kinds of queueing networks. Consequently there has been considerable effort devoted to finding so-called *product-form* solutions for the steady state probabilities of systems composed of a set of interacting subsystems – for example multi-tasking operating systems, multi-processors, communication channels and multi-tiered storage systems. Indeed, it may be said that all such complex systems have product-forms, albeit approximate, since there is no other way to solve them in practice; the modelling process identifies reasonable approximations to render them such.

A stochastic process becomes greatly simplified when it possesses the *Markov property*, which essentially states that, at certain (maybe all) time points, the future evolution of the process depends only on its current state, not on its

past history. This is often an intuitively reasonable assumption that might be made when there is no information available about the past, for example. Moreover, in such a Markov process (or *Markov chain*) it may be possible to compute the state probabilities of a modelled system, at specified times or at equilibrium, by a numerically tractable algorithm; queueing models fall into this category [15, 2]. Product-form solutions for the equilibrium state probabilities of Markov chains that represent a set of such interacting stochastic processes lead to efficient algorithms for computing many performance measures of interest [19, 1, 4, 20, 15, 6, 8]. Although the class of networks that possess such product-forms – often called 'product-form networks' – is highly restricted, large systems usually cannot be solved unless approximating assumptions are made that yield separable solutions, such as product-forms. What has for long been needed, and what this paper addresses, is a unifying methodology for constructing established product-forms that also has the potential to derive new classes of separable solutions, or at least new particular instances within the classes already known. This is accomplished using a theorem, called "RCAT", that synthesises reversed processes, from which product-form equilibrium solutions follow. Recent research, which has provided a method to unify and automate the construction of product-forms in this way, is reviewed in this paper.

A related issue concerns sojourn times, i.e. the times spent by a task undergoing service at a resource or a sequence of resources. A perhaps more common, if not necessarily more apt, term for end-to-end delay is *response time*, which is the sum of a sequence of sojourn times. Response time quantile targets are an important performance criterion for almost all transaction processing, computer-communication and other operational systems – mean values are often not enough. For example in the United Kingdom, ambulances must arrive at the scene of a life-threatening emergency within 8 minutes at least 75% of the time. Expressions have been obtained for the Laplace transform of the probability density function of sojourn times in many queueing models, including some complex single queues and networks of simple queues. However, there is currently no uniform approach to finding sojourn time distributions in stochastic networks. They are usually developed in terms of sample path analyses beginning in an equilibrium state. We consider the joint probability distribution of the sojourn times of a tagged task at each node in a network and observe that this is the same in both the forward and reversed processes. Therefore if the reversed process is known, each node-sojourn time can be taken from either process. In particular, the reversed process can be used for the first node in a path and the forward process for the other nodes, in a recursive analysis. This approach derives, quickly and systematically, existing results for response time probability densities in tandem, open and closed tree-like, and overtake-free Markovian networks of queues. We also show how to apply the method in more general stochastic networks, illustrating with a pair of nodes, one of which is quite complex.

Section 2 introduces the concepts underlying the methodology we present, the supporting technical results and notation being available in other publications such as [12, 13, 14]. In section 3, we explain how to apply RCAT in practice, illustrating with the quite complex example of a G-network [7, 10].

Section 4 deals with separable response time densities in Markovian networks, based on the corresponding joint sojourn time probability distributions. The paper closes in section 5.

2 Stochastic process algebra and RCAT

Stochastic process algebra (SPA) is an extension of classical process algebra that includes time delays and probabilities, aimed at providing performance descriptions of concurrent systems. The inherent compositional structure separates the model of a system into successively more fundamental components and, through the interactions among the components, performance characteristics of complex systems can be assessed. The first such SPAs used for performance modelling were "Timed Processes and Performance Evaluation" (TIPP) and "Performance Evaluation Process Algebra" (PEPA), both of which are Markovian process algebras (MPAs) [24, 18]. PEPA, our choice, is the simplest language, having the fewest combinators. It deals with syntactic entities, *processes* or *agents*, that denote the *states* of an underlying continuous time Markov chain (CTMC) in a semantic model. A state transition in the CTMC is denoted syntactically by an *action* (or *activity*), which has a name (called a *type*) and a *rate*, which is the rate of the CTMC transition it represents. We use just two of PEPA's (syntactic) combinators:

- *prefix*, written as a 'dot' between an action and a process, e.g. $(a, \lambda).P$, which denotes a transition (labeled by the name a) in a Markov state transition graph with rate λ;
- *cooperation* between two *components* (themselves processes), e.g. $P \underset{L}{\bowtie} Q$. Here L is a set of actions that may occur in P and Q. Any action having a name in L can occur in P when and only when it occurs in Q simultaneously.

Choice between two processes, denoted '+' in PEPA, denotes alternative transitions from a state to more than one successor state in the state transition graph. However, we use multiple definitions to denote this, e.g. $P = Q; P = R$ rather than $P = Q + R$.

Existing product-forms have typically been derived in a rather ad-hoc way, by guessing that such a solution exists, then verifying that the Kolmogorov equations of the defining Markov chain are satisfied and appealing to uniqueness. We approach the problem in a constructive, hierarchical way, essentially by seeking the *reversed process* of a continuous time Markov chain (CTMC) in terms of the reversed processes of sub-chains that synchronise to form it [20]. From a reversed process, a separable solution for the equilibrium state probabilities follows immediately and, in fact, the product-forms can be constructed directly – without explicitly obtaining the reversed processes, nor even knowing of their existence. The formalism we use for this hierarchical analysis is a variant of PEPA, although the methodology requires no particular syntax nor indeed a process algebraic basis at all. However, the compositional approach inherent in MPA leads naturally to a mechanisable derivation.

The determination of the reversed process of a certain type of cooperation between two Markov processes at equilibrium is based on the Reversed Compound Agent Theorem (RCAT) of [12]. It provides an alternative methodology, with syntactically checkable conditions, that unifies many product-forms, far beyond those for queueing networks. The original study of G-networks – queueing networks with negative customers that remove ordinary customers – was considered at the time to be a major departure from previous product-form analyses since the property of so-called "local balance' [1] did not hold and the traffic equations were non-linear. In contrast, the RCAT-based approach goes through unchanged – the only difference is that, in a G-network, there are cooperations between two types of departure transitions at different queues, as well as between departure transitions and arrival transitions, as in conventional queueing networks [13]. Prior to the advent of G-networks, many believed that partial balance was a necessary condition for a product-form, but G-networks are no different in the RCAT approach: negative customers satisfy the same conditions (with respect to different action types) as do positive ones.

In its most general form, the theorem applies to multi-agent cooperations, in which collections of any number of processes may cooperate, provided all synchronisations are between two of the component-processes at a time. Under quite mild conditions, its rate equations, which reduce to the traffic equations in queueing networks [1, 15], have a unique solution [14]. In this paper we focus on the practical application of RCAT, rather than elaborate on, or extend, the theoretical foundations, which are amply covered elsewhere.

3 Practical application of the RCAT method

Although an application of RCAT does not require whole reversed processes to be determined in general, it does require the specific reversed rates of the synchronising active actions. These can be computed simply if the equilibrium state probabilities of each component process are known from the standard result that probability flux between two states in a stationary CTMC is equal to the flux in the opposite direction between the same two states in the reversed process. Thus, if the reversed process of a stationary Markov process with generator matrix Q and equilibrium probability vector $\vec{\pi}$ has generator matrix Q', then its instantaneous transition rates are defined by

$$q'_{ij} = \frac{\pi_j q_{ji}}{\pi_i}$$

This property features strongly in the following algorithm.

3.1 Generic algorithm

For simplicity, we consider the cooperation $P_1 \underset{L}{\bowtie} P_2$. The treatment is similar for n-way cooperations in MARCAT applications.

1. From P_k construct R_k by setting the rate of every instance of action $a \in L$ that is passive in P_k to x_a, for $k = 1, 2$ (note that each a will be passive for only one k);

2. For each active action type a in $R_k, k = 1, 2$, check that its reversed rate is the same for all of its instances, i.e. for all transitions $i \to j$ it denotes in the state transition graph of R_k. Compute and denote this reversed rate (in the reversed process $\overline{R_k}$) by

$$\overline{r_a} = \pi_k(i) r_a^i / \pi_k(j)$$

where r_a^i is the specified forward rate of the (any, if more then one) instance of action type a going out of state i;[1]

3. Noting that the symbolic reversed rate $\overline{r_a}$ will in general be a function of the x_b ($b \in L$), solve the equations $x_a = \overline{r_a}$ for each $a \in L$ and substitute the solutions for the variables x_a in each R_k;

4. Check the enabling conditions (detailed in [14]) for each co-operating action in each process P_k. For queueing networks, these are as in the original RCAT, namely that all passive actions be enabled in all states and that all states also have an incoming instance of every active action;

5. The required product-form for state $\underline{s} = (s_1, s_2)$ is now $\pi(\underline{s}) \propto \pi_1(s_1)\pi_2(s_2)$ where $\pi_k(s_k)$ is the equilibrium probability (which may be unnormalised) of state s_k in R_k.

3.2 Illustrative example: G-networks

G-queues and G-networks were introduced by Gelenbe, initially as a model for neural networks [6]. A G-queue is, in a sense, an M/M/1 queue with two kinds of customers: *positive* ones, which behave as standard customers in an M/M/1 queue, and *negative* ones, which are Poisson arrivals that remove, or kill, (positive) customers in the queue when it is not empty. Negative customers have no effect on an empty queue. The theory of G-queues and G-networks is quite elaborate [5, 8, 7] and G-queues have been extended with triggers – negative customers that transfer customers waiting in queues to other nodes in a network [9]. In this paper we only deal with the original G-networks, to illustrate how our methodologies can be applied in a more general context than conventional Markovian queueing networks. However, the product-forms of this section generalise directly to the most general case of a G-network, with heavier notation [13].

Consider, then, an M-node G-network with respective positive/negative external arrival rates $\lambda_1, \ldots, \lambda_M$ / $\Lambda_1, \ldots, \Lambda_M$, service rates $\mu_1, \ldots, \mu_M$, and positive/negative routing probabilities p_{ij} / n_{ij} from node i to node j ($1 \le i \ne j \le M$), where $p_{ii} = n_{ii} = 0, \sum_{j=1}^{M} p_{ij} + \sum_{j=1}^{M} n_{ij} \le 1$. Tasks leave the network from node i with probability $p_{i0} = 1 - \sum_{j=1}^{M}(p_{ij} + n_{ij})$. We do not consider departures from a node back to itself as this is considered part of the definition of the component process for that node. Such departures can be included easily with more complex components. This network can be described by the extended

[1] In fact, if the reversed process of the cooperation is required, the full reversed processes $\overline{R_k}$ must be computed.

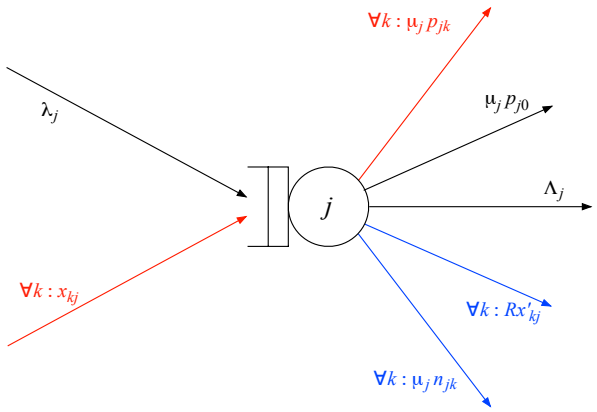

Figure 1: G-queue node

PEPA-like expression $\underset{k=1}{\overset{M}{\bowtie}} P_{k,0}$ (starting with an empty network), where, for $1 \leq k \leq M$:

$$\begin{array}{rcll}
P_{k,n} & = & (e_k, \lambda_k).P_{k,n+1} & n \geq 0 \\
P_{k,n} & = & (f_k, \Lambda_k).P_{k,n-1} & n > 0 \\
P_{k,n} & = & (a_{jk}, \top_{jk}).P_{k,n+1} & n \geq 0, 1 \leq j \neq k \leq M \\
P_{k,n} & = & (b_{jk}, \top_{jk}).P_{k,n-1} & n > 0, 1 \leq j \neq k \leq M \\
P_{k,n} & = & (d_k, p_{k0}\mu_k).P_{k,n-1} & n > 0 \\
P_{k,n} & = & (a_{kj}, p_{kj}\mu_k).P_{k,n-1} & n > 0, 1 \leq j \neq k \leq M \\
P_{k,n} & = & (b_{kj}, n_{kj}\mu_k).P_{k,n-1} & n > 0, 1 \leq j \neq k \leq M
\end{array}$$

with $L_k = \{a_{jk}, b_{jk} \mid j \neq k\}$. The action types e_k, f_k represent external arrivals (positive and negative, respectively), and a_{jk}, b_{jk} represent customers passing from node j to node k after a service completion at node j (positive and negative, respectively). In the sequel, we use the abbreviations $\top_{ij}$ for $\top_{a_{ij}}$, $\top'_{ij}$ for $\top_{b_{ij}}$, x_{ij} for $x_{a_{ij}}$ and x'_{ij} for $x_{b_{ij}}$, $1 \leq i \neq j \leq M$.[2]

Applying the algorithm of the preceding subsection, the process R_i represents node i in isolation, which, as far as the equilibrium state probability distribution is concerned, is equivalent to an M/M/1 queue with arrival rate $\lambda_i + \sum_k x_{ki}$ and service rate $\mu_i + \Lambda_i + \sum_k x'_{ki}$. The equilibrium probability of local state q is therefore well known to be $\pi_i(q) = (1 - \rho_i)\rho_i^q$ for all integers $q \geq 0$, where $\rho_i = \frac{\lambda_i + \sum_k x_{ki}}{\mu_i + \Lambda_i + \sum_k x'_{ki}}$. Every active action type a_{ij} or b_{ij} represents a service completion at node i, that is, a local state transition $q + 1 \rightarrow q$ for some integer $q \geq 0$. Thus, referring to step (2), $\frac{\pi_i(q+1)}{\pi_i(q)} = \rho_i$ which is constant for all q since $\lambda_i, \Lambda_i, \mu_i$ are state-independent. Hence we obtain the following

[2]It would have been just as easy in principle to define a much more complex G-network with resets, triggers and batches, as considered for two-node networks in [13] for example, but the PEPA definition would have been much longer and, perhaps, obscure. The correct traffic equations would emerge via the rate equations in exactly the same way, however.

expressions for the reversed rates of the active actions:

$$\overline{r_{a_{ij}}} = \rho_i p_{ij} \mu_i \qquad \overline{r_{b_{ij}}} = \rho_i n_{ij} \mu_i$$

at all instances of a_{ij}, b_{ij} respectively – see Figures 1 and 2. The equations of

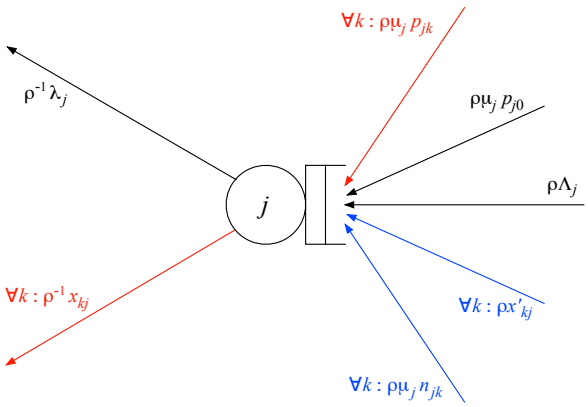

Figure 2: The reversed G-queue

step (3), in the variables x_{ij}, x'_{ij}, are now:

$$\begin{aligned} x_{ij} &= \frac{p_{ij}\mu_i\left(\lambda_i + \sum_k x_{ki}\right)}{\mu_i + \Lambda_i + \sum_k x'_{ki}} \\ x'_{ij} &= \frac{n_{ij}\mu_i\left(\lambda_i + \sum_k x_{ki}\right)}{\mu_i + \Lambda_i + \sum_k x'_{ki}} \end{aligned}$$

A unique solution exists for these by the result in [14].

Finally we have to check the enabling conditions of step (4) of the preceding algorithm. In this network, we add 'invisible transitions' to ensure negative arrivals have no effect on an empty queue. These are given by $P_{k,0} = (b_{jk}, \top_{jk}).P_{k,0}$ for $1 \leq j \neq k \leq M$ and have no effect on the semantics of the queue. Thus, all passive actions are now always enabled, sufficient for the original RCAT of [12]. The product-form is therefore given by step (5).

To show compatibility with previous results for this same network, let $v_i = \lambda_i + \sum_{k=1}^{M} x_{ki}$ and $v'_i = \Lambda_i + \sum_{k=1}^{M} x'_{ki}$ for $1 \leq i \leq M$. Then these equations reduce to

$$\begin{aligned} x_{ij} &= \frac{p_{ij}\mu_i v_i}{\mu_i + v'_i} \\ x'_{ij} &= \frac{n_{ij}\mu_i v_i}{\mu_i + v'_i} \end{aligned}$$

i.e. after summing over i

$$\begin{aligned} v_j - \lambda_j &= \sum_i \frac{p_{ij}\mu_i v_i}{\mu_i + v'_i} \\ v'_j - \Lambda_j &= \sum_i \frac{n_{ij}\mu_i v_i}{\mu_i + v'_i} \end{aligned}$$

These are precisely the non-linear traffic equations obtained for G-networks in [7]. Gelenbe's theorem now follows directly, i.e. the equilibrium probability for state $\underline{i}$ in the network is proportional to

$$\prod_{k=1}^{M} \left(\frac{v_k}{\mu_k + v'_k} \right)^{i_k}$$

Even in such complex networks with non-linear rate equations, we note how easy it is to check that a solution exists and to derive it by this approach, in contrast to using a customised analysis.

4 Response times in networks

The response time of a particular, 'tagged' task along a path in a network of nodes of some kind may be defined as the sum of the sojourn times of the task (i.e. its delays) at those nodes that constitute the path. More generally, the response time probability distribution follows directly from the joint probability distribution of the node-sojourn times. For a path comprising the sequence of nodes $(1, 2, \ldots, m)$, let the response time $R = T_1 + T_2 + \ldots + T_m$, where T_i is the sojourn time at node $i, (1 \leq i \leq m)$, with probability distribution function $T_i(t)$. Then the joint sojourn time distribution is $J(t_1, \ldots, t_m) = \mathbb{P}(T_1 \leq t_1, \ldots, T_m \leq t_m)$ and, denoting Laplace-Stieltjes transforms (LSTs) by asterisks, the m-dimensional LST of the joint sojourn time distribution is

$$J^*(\theta_1, \ldots, \theta_m) = \int_0^\infty \ldots \int_0^\infty e^{-(\theta_1 t_1 + \ldots + \theta_m t_m)} \mathbf{d}J(t_1, \ldots, t_m)$$

The response time distribution then has LST $R^*(\theta) = J^*(\theta, \ldots, \theta)$.

If the sojourn time at each node i depends solely on the state, N_i say, existing at the node *immediately prior to the arrival of the tagged task*, the conditional joint sojourn time LST is $J^*(\theta_1, \ldots, \theta_m \mid \mathbf{n}) = \Pi_{i=1}^m T_i^*(\theta_i \mid n_i)$ where $T_i^*(\theta_i \mid n_i) = \int_0^\infty e^{-\theta_i t} \mathbf{d}\mathbb{P}(T_i \leq t \mid N_i = n_i)$[3]. In such networks, response time distributions can be computed iteratively through their LSTs using the result that:

$$J^*(\theta_1, \ldots, \theta_m \mid \mathbf{l}) = \Pi_{i=1}^m T_i^*(\theta_i \mid n_i) \mathbb{P}(\mathbf{N} = \mathbf{n} \mid \mathbf{L}(0) = \mathbf{l})$$

where bold type indicates vectors and the random variable $L_i(t)$ is the state of node i at time t, so that the initial state is $\mathbf{L}(0)$ and $N_i = L_i(T_i^-)$ when the tagged task arrives at node i at time T_i. In queueing networks it is often the case that the node sojourn times depend only on the queue length at the arrival instant, for example in the overtake-free networks of [21], but the computation of the transient probabilities $\mathbb{P}(\mathbf{N} = \mathbf{n} \mid \mathbf{L}(0) = \mathbf{l})$ is problematic; see [15] for example.

We apply a completely different approach to the computation of the LSTs of response times in Markovian networks at equilibrium, via joint sojourn time

[3]For example, when $m = 2, J^*(\theta_1, \theta_2 \mid \mathbf{N} = \mathbf{n}) = \mathbb{E}\left[\mathbb{E}\left[e^{-(\theta_1 T_1 + \theta_2 T_2)} \mid T_1, \mathbf{N} = \mathbf{n}\right] \mid \mathbf{N} = \mathbf{n}\right] = \mathbb{E}[e^{-\theta_1 T_1} \mathbb{E}[e^{-\theta_2 T_2} \mid T_1, \mathbf{N} = \mathbf{n}] \mid \mathbf{N} = \mathbf{n}] = \mathbb{E}[e^{-\theta_1 T_1} \mathbb{E}[e^{-\theta_2 T_2} \mid N_2 = n_2] \mid N_1 = n_1]$.

distributions and using reversed processes. The key idea of our method is based on the observation that sojourn time distributions are the same whether one considers the forward process or its reversed process. When sojourn times depend only on the state existing at a node at the arrival instant and the reversed process is separable, i.e. a pairwise synchronising network of m reversed nodes, we can use the forward sojourn time at the nodes $2, \ldots, m$ in the 'tail' of a path and the reversed sojourn time at the first node 1, the 'head' of the path; a recursive analysis allows us to consider only the case $m = 2$, the tail-nodes $2, \ldots, m$ constituting a single aggregate 'super-node' in the recursion.

4.1 Node-sojourn times and reversed processes

First, consider the sojourn times spent by a task in a pair of nodes that are connected in the sense that the task first sojourns in node 1, for time T_1, after which it proceeds to node 2 and sojourns there, for time T_2, before departing from the system. We define the *middle state* $\mathbf{s}_0$ of the network to be that which excludes the tagged task itself at the instant when it passes from node 1 to node 2. The first component of the middle state is therefore the queue length at node 1 existing just after the instant of departure there, and the second component is the queue length existing just before the arrival instant at node 2. In many cases, e.g. a pair of tandem queues, the state $\mathbf{s}$ is a pair, $\mathbf{s} = (s_1, s_2)$, where s_i describes the state of node i only, $i = 1, 2$. We call such a state *separable*.

The sojourn time at node 1, T_1 say, can be calculated as the first passage time from the *initial state*, existing at the task's arrival instant, to exit from the state in which the task departs node 1. In general, this can involve arbitrary transitions in the whole system, i.e. be influenced by the evolution of node 2 as well as node 1. However, often, T_1 is determined solely by the initial state and the evolution of node 1, as in the case of constant rate queues, for example. In this case, the conventional approach to sojourn time analysis is to consider the state of the system at the instant of the task's departure from node 1 and use this as the initial state for the sojourn at node 2; this may also depend solely on the evolution of node 2.

The properties we need in order to use this (the traditional) technique are therefore:

- The state of the system is separable, i.e. $\mathbf{s} = (s_1, s_2)$, where s_i describes the state of node i only, $i = 1, 2$;
- The sojourn time of the tagged task at each node depends *solely* on the node's state at its arrival instant – implying that the node has the 'overtake-free' property of [21] which requires that the passage of the tagged task through the node is not influenced by tasks at any other node;
- The sojourn time at each node can be characterised as a first passage time in a Markov chain describing the node's behaviour during that sojourn *insofar as it affects the tagged task*.

Notice that the last point does not necessarily require the Markov chain describing the whole system or even the node: for example a transient chain representing a queue with no arrivals is sufficient if the first property holds. This is a traditional approach that was used to obtain the Laplace transform of response time distributions in cyclic, tree-like and overtake-free networks in the 1980s [21, 15].

An alternative approach uses the observation that sojourn times are the same whether one considers the forward process or its reversed process. For example, given initial state $\mathbf{i}_0 = (i_{0;1}, i_{0;2})$ in a two-node network, we might take the sojourn time at the first node in the forward process (conditioned on $i_{0;1}$) and the *reversed sojourn time* at the second node in the reversed process, conditioned on the state existing at the *end* of the two sojourns (in the forwards process). Notice that the reversed sojourn time is not necessarily dependent on only the initial state pertaining to the second node (final state in the forwards process). Indeed, the reversed process itself may depend on the joint state of the whole system, even if the forward node was overtake-free. In fact, this approach turns out to be no easier than the naive, purely 'forwards' one and a better method is as follows.

Theorem 1 *Suppose that a two-node Markovian network at equilibrium satisfies the following conditions:*

1. *The state of the system is separable, with middle states $(s_1, s_2) \in \mathcal{S}$ having probabilities $p_{s_1 s_2}$;*

2. *The reversed sojourn time at node 1 depends* solely *on the state existing at node 1 just after a particular, tagged task completes service at node 1 in the forwards process, i.e. on the first component of the middle state;*

3. *The forward sojourn time at node 2 depends* solely *on the state existing at node 2 just before the arrival of the tagged task there, i.e. on the second component of the middle state.*

Then the joint sojourn time probability distribution has LST

$$J^*(\theta_1, \theta_2) = \sum_{(s_1,s_2)\in\mathcal{S}} p_{s_1 s_2} \tilde{T}_1^*(\theta_1 \mid S_1(T_1^+) = s_1) T_2^*(\theta_2 \mid S_2(T_1^-) = s_2)$$

where $\tilde{T}_1^*(\theta_1 \mid T)$ denotes the conditional expectation $\mathbb{E}[e^{-\theta_1 \tilde{T}_1} \mid T]$ and similarly for $T_2^*(\theta_2 \mid T)$.

4.2 Queueing networks

The result of the previous section is easy to apply, in both open and closed queueing networks. Consider first the tandem pair of queues depicted in Figure 3 – a cycle of two queues is simply obtained by connecting the departures of the second queue to the arrivals of the first. The forward and reversed nodes are both shown; correspondingly, the forward and reversed sojourn times are illustrated for both nodes, as per section 4.1. We consider the joint sample paths in

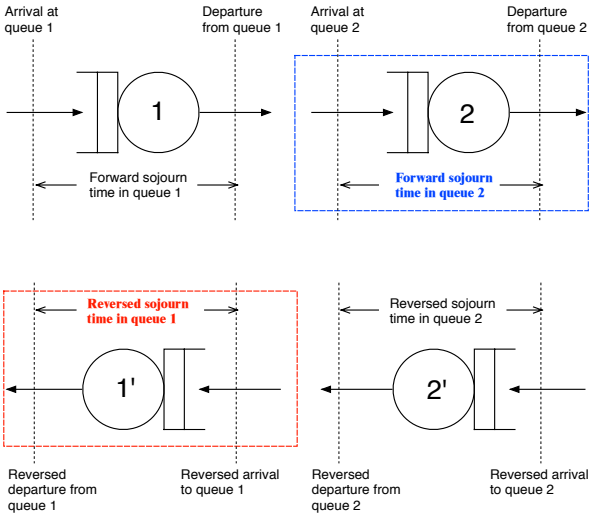

Figure 3: Two M/M/1 queues in tandem and the reversed process

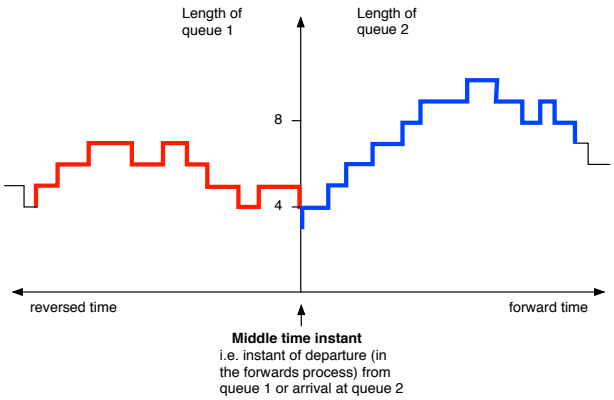

Figure 4: Forward and reversed sample paths given middle state (4,3)

the forward node 2 and reversed node 1 processes, beginning in a given middle state – (4,3) in the sample paths shown in Figure 4. For the forward response time at node 2, we look to the right of the vertical axis and for the reversed response time at node 1, we look to the left. Since forward and reversed sojourn times are identically distributed, we have:

$$\begin{aligned} J^*(\theta_1,\theta_2) &= \mathbb{E}_{S_1,S_2}[\tilde{T}_1^*(\theta_1 \mid S_1)T_2^*(\theta_2 \mid S_2)] \\ &= \mathbb{E}_{S_1,S_2}[T_1^*(\theta_1 \mid S_1)T_2^*(\theta_2 \mid S_2)] \\ &= \sum_{n_1,n_2\geq 0} \pi_{n_1 n_2} \left(\frac{\mu_1}{\mu_1+\theta_1}\right)^{n_1+1} \left(\frac{\mu_2}{\mu_2+\theta_2}\right)^{n_2+1} \end{aligned}$$

The equilibrium probabilities π are the standard product-form solution

of [19, 11, 15] and so the result simplifies to:

$$\sum_{n_1,n_2\geq 0} \pi_{n_1} \left(\frac{\mu_1}{\mu_1+\theta_1}\right)^{n_1+1} \pi_{n_2} \left(\frac{\mu_2}{\mu_2+\theta_2}\right)^{n_2+1}$$

To derive the reversed process and find the product-form solution at the same time, one could use the Reversed Compound Agent Theorem of [12]. The tandem pair of queues considered here is simple since we know what the reversed process of the first node is – the same M/M/1 queue – but we do not know this for more general nodes such as G-queues or other non-reversible processes [6, 13].

The above result generalises inductively to overtake-free paths in both open and closed networks to give the following well known result:

Proposition 1 *For overtake-free path* $\mathbf{z} = (z_1, \ldots, z_m)$ *in a queueing network of* M *nodes with state space* $\mathcal{S}$ *at equilibrium* ($1 \leq m \leq M$)*, the LST of the joint sojourn time probability distribution is*

$$J^*(\theta_1, \ldots, \theta_m) = \sum_{(n_1,\ldots,n_M)\in\mathcal{S}} \pi_{n_1,\ldots,n_M} \prod_{j=1}^{m} \left(\frac{\mu_{z_j}}{\theta_j+\mu_{z_j}}\right)^{n_{z_j}+1}$$

where $\pi_{n_1,\ldots,n_M}$ *is the equilibrium probability distribution of the network's state immediately prior to the instant of arrival of a task at any node.*

Notice that the probabilities $\pi_{n_1,\ldots,n_M}$ are well known by the arrival theorem [15], being the same as an open network's steady state probabilities (at a random time point) or the steady state probabilities of a closed network with population reduced by one, depending on whether the network in question is open or closed, respectively.

In the case of open networks, $\pi_{n_1,\ldots,n_M}$ is a product of the form $\pi_1(n_1)\ldots\pi_M(n_M)$ where $\pi_i(n_i) = (1-x_i)x_i^{n_i}$ for some constants x_i, and so the result simplifies to

$$J^*(\theta_1, \ldots, \theta_m) = \prod_{j=1}^{m} \frac{\mu_{z_j}(1-x_{z_j})}{\theta_j+\mu_{z_j}(1-x_{z_j})}$$

We observe that if service rates varied with queue length, we could not ignore tasks behind a given tagged task, even when they could not overtake, because they would influence the service rate received by the tagged task. Except in special cases, therefore, constant service rates are required.

4.3 G-queues and networks

The challenge with applying our method to G-networks is that the reversed response time in a G-queue is not as straightforward as in the M/M/1 case – it is not even a response time in the same sense, in fact. The corresponding transient analysis required in the conventional approach is even more complex [17]. We therefore begin by considering the reversed sojourn time in a single G-queue, conditional on the queue length faced by a reversed arrival, i.e. left behind by a forward departure.

4.3.1 Reversed sojourn time in a single G-queue

The sojourn time probability distribution in a single G-queue at equilibrium can be determined by finding the reversed sojourn time distribution conditioned on the queue length left behind on (forwards) departure and then deconditioning with respect to the equilibrium *departure state* probabilities. These are the same as the equilibrium state probabilities in G-queues by the Random Observer Property [15], since the reversed arrival process is also Poisson. This approach contrasts with the direct (forwards) method given in [16].

Consider a G-queue with positive arrival rate λ^+ of positive tasks, exponential service times with parameter μ and an additional Poisson arrival process of negative tasks, rate λ^-. This can be regarded as an M/M/1 queue with arrival rate λ^+ and service rate $\mu + \lambda^-$, with the departure stream split into normal service completions (rate μ) and killed tasks (rate λ^-). At equilibrium, the probability that the queue length is n is therefore $(1-\rho)\rho^n$ where $\rho = \lambda^+/(\lambda^- + \mu)$. The reversed queue is therefore also an M/M/1 queue, with service rate $\mu + \lambda^-$ and two independent Poisson arrival streams with rates $\gamma_1 = \frac{\mu\lambda^+}{\lambda^-+\mu}$, for the reversed departures, and $\gamma_2 = \frac{\lambda^-\lambda^+}{\lambda^-+\mu}$, for the reversed killings.

For a tagged task arriving at the reversed queue, let R denote the (reversed) sojourn time random variable and $R^*(\theta)$ be the Laplace-Stieltjes transform (LST) of its probability distribution function. We also define the following random variables:

- N is the queue length just before the arrival of the tagged task (i.e. just after the departure of its forward counterpart);
- B, B_i for $i = 1, 2, \ldots$ are service times;
- W, W_i for $i = 1, 2, \ldots$ are busy periods arising from reversed negative killings.

Proposition 2 *Given queue length N on arrival, the reversed sojourn time probability distribution function has LST*

$$R^*_N(\theta) = w(\theta)^{N+1}$$

and the unconditional sojourn time LST is $R^*(\theta) = \frac{(1-\rho)w(\theta)}{1-\rho w(\theta)}$, where $w(\theta)$ is the smaller root of the equation

$$\gamma_2 w^2 - (\theta + \mu + \lambda^- + \gamma_2)w + (\mu + \lambda^-) = 0$$

Proof: The derivation of $R^*(\theta)$ is based on the observation that the queue, of length N, left behind by a tagged task in the forwards process comprises precisely those tasks that arrived *after* that task and were not killed during its (forwards) sojourn time. This queue length is the same as that faced by the corresponding, arriving, reversed tagged task. The reversed sojourn time, conditional on the arrival queue length N, is therefore the sum of $N+1$ service times together with the service times of all those reversed killing departures

that arrive during these service times. This is just the sum of $N+1$ busy periods in an M/M/1 queue with arrival rate γ_2 and service rate $\mu+\lambda^-$, i.e. $W_1+\ldots+W_{N+1}$.

It is routine to show that the LST of the probability distribution of a generic busy period W is $B^*(\theta+\gamma_2(1-W^*(\theta)))$. We can write $W^*(\theta)=\mathbb{E}[e^{-\theta W}]=\mathbb{E}[e^{-\theta(B+W_1+\ldots+W_A}]$ where A is the number of arrivals in the service period B. Hence

$$\begin{aligned} W^*(\theta) &= \mathbb{E}[e^{-\theta B}\mathbb{E}[e^{-\theta(W_1+\ldots+W_A)} \mid B]] \\ &= \mathbb{E}[e^{-\theta B}\mathbb{E}[\mathbb{E}[e^{-\theta(W_1+\ldots+W_A)} \mid A,B] \mid B]] \\ &= \mathbb{E}[e^{-\theta B}\mathbb{E}[W^*(\theta)^A \mid B]] \\ &= \mathbb{E}[e^{-\theta B}e^{-\gamma_2(1-W^*(\theta))B}] \\ &= B^*(\theta+\gamma_2(1-W^*(\theta))) \end{aligned}$$

For exponential service times, this implies that $W^*(\theta)$ is a solution for w of the quadratic equation $w=\frac{\mu+\lambda^-}{\theta+\mu+\lambda^-+\gamma_2(1-w)}$, i.e.

$$\gamma_2 w^2-(\theta+\mu+\lambda^-+\gamma_2)w+(\mu+\lambda^-)=0$$

Only the smaller root is valid, the larger one being greater than unity.

The conditional reversed sojourn time LST, given queue length N on arrival, now follows as $R^*_N(\theta)=\mathbb{E}\ [e^{-\theta(W_1+\ldots+W_{N+1})} \mid N]=W^*(\theta)^{N+1}$ and so the unconditional LST is

$$R^*(\theta)=\mathbb{E}[\mathbb{E}[W^*(\theta)^{N+1} \mid N]]=W^*(\theta)G_N(W^*(\theta))$$

where $G_N(z)=(1-\rho)z/(1-\rho z)$ is the probability generating function (pgf) of the queue length faced by a tagged task on arrival in steady state, which is the same as the pgf of the equilibrium state at an arbitrary time by the Random Observer Property, see [15, 23, 22] for example. Thus, $R^*(\theta)=\frac{(1-\rho)w}{1-\rho w}$, where w is the smaller root of the above quadratic equation. ♠

Notice that the reversed sojourn time depends solely on the queue length faced on arrival, the reversed node being an ordinary M/M/1 queue with two independent Poisson arrival streams, in contrast to a (forwards) G-queue. This property is crucial when analysing networks that include G-queues.

4.3.2 Joint sojourn times in a pair of G-queues

Suppose now that we have a tandem network comprising an M/M/1 queue and a G-queue that has an additional external arrival stream of negative tasks that remove the last task in the FCFS queue when it is non-empty [6]. Suppose first that the G-queue is the first node.

Proposition 3 *A tandem pair of nodes consisting of a G-queue at node 1, with positive arrival rate λ_1^+, negative arrival rate λ_1^- and service rate μ_1, and an M/M/1 queue at node 2, with service rate μ_2 and arrivals comprising the*

service completions from node 1, has response time distribution (for positive tasks) with LST

$$\frac{(\mu_2 - \rho_1\mu_1)R_1^*(\theta)}{\mu_2 - \rho_1\mu_1 + \theta}$$

where $\rho_1 = \lambda_1^+/(\lambda_1^- + \mu_1)$, $R_1^*(\theta) = \frac{(1-\rho_1)w(\theta)}{1-\rho_1 w(\theta)}$ *and* $w(\theta)$ *is the smaller root of the equation* $\rho_1\lambda_1^- w^2 - (\theta + \mu_1 + \lambda_1^- + \rho_1\lambda_1^-)w + (\mu_1 + \lambda_1^-) = 0$.

Proof: The conditions in Section 4.1 are satisfied since the network is separable [13]), the second node is an M/M/1 queue with sojourn time depending only on the queue length existing on arrival, and the reversed sojourn time in the first node depends only on the queue length at the (forward) departure instant – see Proposition 2. Moreover, the Random Observer Property holds at the middle instant (of entry into the middle state) since the departure process from the first node is Poisson.

The response time distribution therefore has LST which is the product of that for the G-queue and that for the M/M/1 queue with arrival rate equal to the positive throughput from queue 1, i.e. to the product of the external positive arrival rate and the probability of a task not being 'killed'. ♠

It is interesting to note that the response time LST is separable when node 1 is the G-queue, whereas an M/G/1 queue must be second if paired with an M/M/1, when there are no negative customers. Notice that if an M/G/1 queue were paired first with an M/M/1 queue, with FCFS queueing discipline, the network is not separable – it has long been known that no product-form then exists for the equilibrium queue length probabilities.

Finally, in a tandem pair comprising an M/M/1 queue as the first node and a G-queue as the second, it is easy to find the reversed process of the first node – the same M/M/1 queue – but the forward response time in the second node is more problematic. This is because the tagged task's progress there is influenced by the state of the first node, departures from which offer this task a degree of "protection" in that they become the target of the next killing by a negative arrival to the second queue. Thus, condition 3 of Theorem 1 is not satisfied. Nevertheless, the conditional forwards response time can be investigated by considering the tagged task's progress in terms of the states of both nodes, given their initial joint state at the middle instant. The ensuing analysis is then essentially the same as the latter part of the entirely forwards method of [17]; note, however, that the initial part of that method, relating to the sojourn time at the first node, is more complex than our using the reversed sojourn time – whether we consider an M/M/1 queue or a G-queue as the first node, paired with the G-queue as the second node.

5 Conclusion

Stochastic process algebra provides a natural unifying formalism for many stochastic modelling methodologies, a claim supported by the use of a PEPA-based MPA to find many classes of product-form solutions through the Re-

versed Compound Agent Theorem. Current applications range from multi-class queueing networks, through the many variants of G-networks, to networks with mutual exclusion and blocking in critical sections [3]. An automatic implementation is currently restricted to the original RCAT of [12] and hence to actions that can cooperate in only two components at a time, such as departures from one queue passing to another as arrivals. However, the method of [13] can be applied to handle multiple, instantaneous transitions in chains of components. Performance engineering environments supported by stochastic modelling, akin to methodologies used in traditional engineering disciplines, could be built around RCAT. Such environments should further be integrated into those of software engineering, supported by formal methods of computer science.

Furthermore, response time distributions – more generally, joint node-sojourn time distributions – can be derived much more simply and generally than previously using the reversed process of a separable network, often determinable by means of RCAT. In this way, most of the known separable solutions for the LSTs of response time distributions in queueing networks can be obtained. Moreover, many other special cases of such product-form solutions can be explained, and new ones derived. A particularly exciting prospect is to obtain stochastic bounds on the error of approximate response time analyses, by comparing with solutions obtained for modified networks, amenable to our separable approach. The methodology provides a handle for such problems and certainly is conducive to automation.

References

[1] F. Baskett, K.M. Chandy, R.R. Muntz, and F.G. Palacios. Open, closed and mixed networks of queues with different classes of customers. *J.ACM*, 22(2):248–260, 1975.

[2] G. Bolch, S. Greiner, H. de Meer, and K.S. Trivedi. *Queueing Networks and Markov Chains, Modeling and Performance Evaluation with Computer Science Applications*. Wiley, 2006.

[3] R.J. Boucherie. A characterisation of independence for competing markov chains withapplications to stochastic petri nets. *IEEE Transactions on Software Engineering*, 20(7):536–544, 1994.

[4] J.P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, 16:527–531, 1973.

[5] J-M. Fourneau, E. Gelenbe, and R. Suros. G-networks with multiple classes of positive and negative customers. *Theoretical Computer Science*, 155:141–156, 1996.

[6] E. Gelenbe. Random neural networks with positive and negative signals and product form solution. *Neural Computation*, 1(4):502–510, 1989.

[7] E. Gelenbe. Queueing networks with negative and positive customers. *Journal of Applied Probability*, 28:656–663, 1991.

[8] E. Gelenbe. G-networks with triggered customer movement. *Journal of Applied Probability*, 30:742–748, 1993.

[9] E. Gelenbe. G-networks with triggered customer movement. *Journal of Applied Probability*, 30:742–748, 1993.

[10] E. Gelenbe (ed.). Feature issue on G-networks. *European Journal of Operations Research*, 126, 2000.

[11] D. Gross and C. M. Harris. *Fundamentals of Queueing Theory*. Wiley-Sons, 1985.

[12] P.G. Harrison. Turning back time in markovian process algebra. *Theoretical Computer Science*, 290(3):1947–1986, January 2003.

[13] P.G. Harrison. Compositional reversed Markov processes, with applications to G-networks. *Performance Evaluation*, December 2004.

[14] P.G. Harrison and T.T. Lee. Separable equilibrium state probabilities via time reversal inmarkovian process algebra. *Theoretical Computer Science*, 2005.

[15] P.G. Harrison and Naresh M. Patel. *Performance Modelling of Communication Networks and Computer Architectures*. Addison-Wesley, 1992.

[16] P.G. Harrison and E. Pitel. Sojourn times in single server queues with negative customers. *Journal of Applied Probability*, 30:943–963, 1993.

[17] P.G. Harrison and E. Pitel. Response time distributions in tandem G-networks. *Journal of Applied Probability*, 32:224–246, 1995.

[18] J. Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, University of Edinburgh, 1994.

[19] J.R. Jackson. Jobshop-like queueing systems. *Management Science*, 10(1):131–142, 1963.

[20] F.P. Kelly. *Reversibility and stochastic networks*. Wiley, 1979.

[21] F.P. Kelly and P.K. Pollett. Sojourn times in closed queueing networks. *Advances in Applied Probability*, 15:638–656, 1983.

[22] I. Mitrani. Response time problems in communication networks. *Journal of the Royal Statistical Society B*, 47(3):396–406, 1985.

[23] I. Mitrani. *Probabilistic Modelling*. Cambridge University Press, 1998.

[24] U. Herzog N. Götz and M. Rettelbach. Multiprocessor and distributed system design: The integration of functional specification and performance analysis using stochastic process algebras. In *Tutorial proceedings of PERFORMANCE '93*, number 794 in Lecture Notes in Computer Science. Springer-Verlag, 1993.