Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 151 (2006) 61–76
www.elsevier.com/locate/entcs

Process Algebraic Non-product-forms

P.G. Harrison[1]

Department of Computing
Imperial College London, England

Abstract

A generalization of the Reversed Compound Agent Theorem of Markovian process algebra is derived that yields separable, but non-product-form solutions for collections of interacting processes such as arise in multi-class queueing networks with Processor Sharing servers. It is based on an analysis of the minimal cycles in the state space of a multi-agent cooperation, which can be simply identified. The extended methodology leads to what we believe are new separable solutions and, more generally, the results represent a viable practical application of the theory of Markovian process algebras in stochastic modelling.

Keywords: Markov processes, Reversed Compound Agent Theorem, non-product form, stochastic modeling

1 Introduction

The quest for so-called *product-form solutions* for the equilibrium state probabilities in stochastic networks has been a major research area in performance modelling for over 30 years, e.g. [1,13,14]. As the name implies, such a solution is expressed as a product of terms, each of which relates to only one of a collection of interacting component processes. Most attention has been given to queueing networks and their variants such as G-networks [5], but there have also been other significant examples.

The Reversed Compound Agent Theorem (RCAT) is a compositional result that uses Markovian process algebra (MPA) to derive the reversed process of certain cooperations between two continuous time Markov chains at equilibrium. From a reversed process, together with the given, forward process, the joint state probabilities follow as a product of ratios of rates in these two processes, yielding a product-form when one exists. RCAT thereby provides an alternative methodology, with syntactically checkable conditions, which unifies many product-forms, far beyond those for queueing networks.

[1] Email: pgh@doc.ic.ac.uk

doi:10.1016/j.entcs.2006.03.012

This paper presents a significant generalisation that yields separable, but *non-product-form*, solutions in other networks in which some transition rates depend on the states of more than one synchronising process. This situation arises in multi-class networks with processor sharing (PS) queues, for example. The generalisation is based on an analysis of the *minimal cycles* in the state space of the multi-agent cooperation. Specifically, it is shown that the ratio of the products of rates around any cycle and its reversed cycle, required to establish Kolmogorov's criteria [7,9], is a product of such ratios around a set of minimal cycles. The extended methodology leads to what we believe are new separable, non-product-form solutions and constiutes a major contribution to the mechanisation of stochastic modelling tools.

In the next section, the essential background material on Markov state transition graphs and their relationship with MPA is reviewed; the basic definition of the MPA PEPA and the defining property of a reversed stationary Markov process are given in the Appendices. This section also includes a new result, which will be used in our main analysis, that relates certain parallel and synchronising processes. The main section 3 considers non-local state-dependence in multi-agent cooperations and presents a weaker version of RCAT that relies on checking products of rates around minimal cycles. This leads to the well known 'BCMP' result of Baskett, Chandy, Muntz and Palacios [1] for processor sharing queues and new product-forms. The paper concludes in section 4.

2 Preliminaries

2.1 State transition paths and split actions

Once a reversed process is known, a solution for a stationary Markov process's equilibrium probabilities follows as a product of ratios of forward and reversed rates when an appropriate path has been found from a chosen reference state to the state in question; see Appendx B.

An action α in a component that cooperates (with some action in another component) may be only a part of a 'complete' action α^+ (with higher rate) in that component considered in isolation. In the cooperation, we say this 'complete' action α^+, which represents transitions between the same pair of states, is *split* into two sub-actions, of which the one α synchronises (with a (sub-)action in the other component) and the other one proceeds independently. For example, a service completion (α^+) at a queue can cause either an external departure or the transfer of a customer to another queue (α).

In general, an action can be split into more than two sub-actions, corresponding to multiple synchronisations, and each sub-action has a well defined rate. The *reversed* sub-actions are allocated rates in proportion to their forward transition rates, their total being equal to the reversed rate of the complete action (in the isolated component); see [7].

If every cooperation involves only a sub-action in each component, with another sub-action of each respective complete action not participating, there will always be a *rectilinear* path to every state (i, j) from any chosen reference state $(0, 0)$. That is,

considering a two-component cooperation for simplicity, just follow the path from $(0,0)$ to i in the first component process with the second component in state 0, and then follow the path from 0 to j in the second component with the first in state i. Then a separable solution can be found by finding products of ratios in each cooperating component separately, with the reparameterisation of the components given by RCAT. Note that parallel (non-cooperating) agents are a special case with a null reparameterisation.

2.2 Residual actions

We can guarantee that rectilinear paths do exist, which are identical to concatenations of paths in the isolated component processes, by augmenting active synchronising actions with *residual actions* or ϵ-*actions*. These are parallel to the synchronising actions but do not participate in the cooperation.

Definition 2.1 Suppose (a,λ) is an action in some agent P. The agent $P^{a+\epsilon} = P\{(a,\lambda) \leftarrow (a,(1-\epsilon)\lambda)\} \cup P\{(a,\lambda) \leftarrow (a^\epsilon,\epsilon\lambda)\}$ for some real number $\epsilon, 0 < \epsilon < 1$, where the action type a^ϵ does not occur in P. The *residual action* $(a^\epsilon,\epsilon\lambda)$ is called an ϵ-*action*.

The agent $P^{a+\epsilon}$ denotes the Markov process with the same generator matrix as that of the Markov process underlying P, but with every element denoted by the action type a interpreted as a sum of the quantities $(1-\epsilon)\lambda$ (the original action a) and $\epsilon\lambda$ (the ϵ-action). That is, the Markov process underlying $P^{a+\epsilon}$ has the same transitions as for P except that the rate of a is reduced by a factor of $1-\epsilon$ and there are additional transitions of rate $\epsilon\lambda$ parallel to (with the same source and destination states) all those denoted by action type a. Clearly, $\lim_{\epsilon\to 0} P^{a+\epsilon} = P$. We cannot assume anything about ergodicity and its preservation in this limit, but this is not an issue here since all processes are assumed stationary. Ergodicity conditions require a separate analysis.

Notice that a reversed residual action $\overline{(a^\epsilon,\epsilon\lambda)} = (\overline{a^\epsilon},\epsilon\overline{\lambda})$, i.e. its rate is the product of ϵ and the reversed rate of the unsplit action a.

In a cooperation of agents with ϵ-actions, we must split a passive action into residual and cooperating parts *before* making only the cooperating part passive – it is not meaningful to split an unspecified rate, and no action can be passive until it participates in a cooperation. For brevity, we denote an agent P, in which an action type a is made passive, by $P(a,\top) \equiv P\{(a,\lambda) \leftarrow (a,\top)\}$, where λ is matched to the rate of the action with type a in P (possibly different at each of its instances). This notation is extended in the obvious way to multiple action types $a \in S$ which each become passive in the agent $P\{(a,\lambda) \mid a \in S\}$.

We can now write $P^{a+\epsilon}(a,\top)$ to define a modified agent P with passive action type a, split to introduce a parallel residual action with rate $\epsilon\lambda$ which does not synchronise, where λ is the rate of a in P (possibly different at each instance).

We have the following simple but important property for certain cooperations with residual actions.

Lemma 2.2 *Consider agents R, S with no passive actions and let action type a in R have rate λ_a, action type a in S have rate μ_a. Let r_a and $\overline{r_a}$ be the rates of a and $\overline{a}$ at a particular instance of a in the cooperations*

$$R^{a+\epsilon} \underset{L}{\bowtie} S^{a+\epsilon}(a, \top) \quad \text{and} \quad \overline{R^{a+\epsilon}}(\overline{a}, \top) \underset{\overline{L}}{\bowtie} \overline{S^{a+\epsilon}}$$

respectively, where $a \in L$. Then

$$\frac{r_a}{\overline{r_a}} = \frac{\lambda_a \mu_a}{\overline{\lambda_a}\,\overline{\mu_a}} \quad \text{if and only if} \quad \mu_a = \overline{\lambda_a}.$$

Proof. By definition of the cooperation combinator and the splitting of the action with type a, $r_a = (1-\epsilon)\lambda_a$ and $\overline{r_a} = (1-\epsilon)\overline{\mu_a}$ and so the result follows. □

This lemma means that paths including a cooperating action are equivalent to paths that do not, in the sense of equilibrium state probabilities as follows.

Lemma 2.3 *In the notation of the previous lemma, let r_R^ϵ, r_S^ϵ be the rates of the residual action type a^ϵ in R, S respecively and let $\overline{r_R^\epsilon}$, $\overline{r_S^\epsilon}$ be the respective reversed rates of $\overline{a^\epsilon}$. Then, at a particular instance of a,*

$$\frac{r_a}{\overline{r_a}} = \frac{r_R^\epsilon\, r_S^\epsilon}{\overline{r_R^\epsilon}\,\overline{r_S^\epsilon}} \quad \text{if and only if} \quad \mu_a = \overline{\lambda_a}.$$

Proof. $r_R^\epsilon = \epsilon\lambda_a$ and $\overline{r_R^\epsilon} = \epsilon\overline{\lambda_a}$. Similarly, $r_S^\epsilon = \epsilon\mu_a$ and $\overline{r_S^\epsilon} = \epsilon\overline{\mu_a}$ so that the ϵ factors cancel in the ratio and the result follows by lemma 2.2. □

Suppose, then, that a cooperating action type a denotes a transition between states (i,j) and (i',j') in $R \underset{L}{\bowtie} S$; i.e. it also denotes transitions $i \to i'$ in R and $j \to j'$ in S. Then, the paths $(i,j) \to (i',j) \to (i',j')$, $(i,j) \to (i,j') \to (i',j')$ (via residual transitions) and $(i,j) \to (i',j')$ (via the synchronised transition) are equivalent in the sense that the products of the ratios of the forward and reversed rates of each transition in each path are equal. This is a necessary condition for $\overline{R^{a+\epsilon}}(\overline{a}, \top) \underset{L}{\bowtie} \overline{S^{a+\epsilon}}$ to be the reversed process of $R^{a+\epsilon} \underset{L}{\bowtie} S^{a+\epsilon}(a, \top)$, a property that yields a simple proof for RCAT. Moreover, it can also be used to find simple separable solutions directly for the equilibrium state probabilities of cooperations satisfying that theorem.

2.3 Multiple agent cooperations

In the PEPA cooperation $P \underset{L}{\bowtie} Q$, the subset of action types in a cooperation set L which are *passive* (i.e. have unspecified rate $\top$) with respect to an agent P is denoted by $\mathcal{P}_P$ and the subset of corresponding *active* action types by $\mathcal{A}_P = L \backslash \mathcal{P}_P$; similarly for agent Q. For RCAT, it is also assumed that, in $P \underset{L}{\bowtie} Q$, any active action in P has a corresponding passive action in Q, and vice versa; therefore, $\mathcal{P}_P = \mathcal{A}_Q$ and $\mathcal{A}_P = \mathcal{P}_Q$.

In an extension of PEPA, consider now a multiple-agent, pairwise cooperation $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$ $(n \geq 2)$, where $L = \bigcup_{k=1}^{n} L_k$ and $L_k = \mathcal{P}_k \cup \mathcal{A}_k$ is the set of synchronising action

types that occur in agent P_k (abbreviating $\mathcal{P}_{P_k}$ by $\mathcal{P}_k$ and $\mathcal{A}_{P_k}$ by $\mathcal{A}_k$). Each of the n agents cooperates with (at most) one other and so $\mathcal{P}_k \subset \bigcup_{\substack{j=1\\ j\neq k}}^{n} \mathcal{A}_j$ and $\mathcal{A}_k \subset \bigcup_{\substack{j=1\\ j\neq k}}^{n} \mathcal{P}_j$. We provide the semantics of multi-agent cooperation by defining it in terms of PEPA's cooperation combinator:

$$\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k = (\ldots((P_1 \underset{M_2}{\bowtie} P_2) \underset{M_3}{\bowtie} P_3) \underset{M_4}{\bowtie} \ldots \underset{M_{n-1}}{\bowtie} P_{n-1}) \underset{M_n}{\bowtie} P_n$$

where $M_k = L_k \cap \left(\bigcup_{j=1}^{k-1} L_j\right)$. Note the subtle change in the bowtie symbol used for multi-agent cooperations.

2.4 Notation

We will use the following notation, generalising that of [10]:

$\mathcal{P}_k^{i\rightarrow}$ denotes the set of action types in L_k that are *passive* in P_k and correspond to transitions *out of* state i in the Markov process of P_k;

$\mathcal{P}_k^{i\leftarrow}$ denotes the set of action types in L_k that are *passive* in P_k and correspond to transitions *into* state i in the Markov process of P_k;

$\mathcal{A}_k^{i\rightarrow}$ denotes the set of action types in L_k that are *active* in P_k and correspond to transitions *out of* state i in the Markov process of P_k;

$\mathcal{A}_k^{i\leftarrow}$ denotes the set of action types in L_k that are *active* in P_k and correspond to transitions *into* state i in the Markov process of P_k;

$\mathcal{P}^{\underline{i}\rightarrow}$ denotes the set of action types in $L = \bigcup_{k=1}^{n} L_k$ that are *passive* and correspond to transitions *out of* state $\underline{i} = (i_1, i_2, \ldots, i_n)$ in the Markov process of $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$;

$\mathcal{P}^{\underline{i}\leftarrow}$ denotes the set of action types in L that are *passive* and correspond to transitions *into* state $\underline{i}$ in the Markov process of $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$;

$\mathcal{A}^{\underline{i}\rightarrow}$ denotes the set of action types in L that are *active* and correspond to transitions *out of* state $\underline{i}$ in the Markov process of $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$;

$\mathcal{A}^{\underline{i}\leftarrow}$ denotes the set of action types in L that are *active* and correspond to transitions *into* state $\underline{i}$ in the Markov process of $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$;

$\alpha_a^{\underline{i}}$ denotes the instantaneous transition rate *out of* state $\underline{i}$ in the Markov process of $\underset{\substack{k=1\\ L}}{\overset{n}{\bowtie}} P_k$ corresponding to *active* action type $a \in L$;

$\top_a$ denotes the unspecified rate associated with the action type a in the action $(a, \top_a)$;

$\boldsymbol{x}$ denotes the vector $(x_{a_1}, \ldots, x_{a_m})$ of positive real variables x_{a_i} when $L =$

$\{a_1, \ldots, a_m\}$;

$\overline{\beta_a^{\underline{i}}}(\boldsymbol{x})$ denotes the instantaneous transition rate *out of* state $\underline{i}$ in the reversed Markov process of $\underset{L}{\overset{n}{\bowtie_{k=1}}} P_k\{\top_a \leftarrow x_a \mid a \in L\}$ corresponding to *passive* action type $a \in L$; note that a is *incoming* to state $\underline{i}$ in the forwards process. We also write $\overline{\beta_{k;a}^{i_k}}(\boldsymbol{x}) \equiv \overline{\beta_a^{\underline{i}}}(\boldsymbol{x})$ where P_k is the component in which a is passive (incoming to state i_k).

3 Non-local state dependence

The reversed process and product-form arising from RCAT requires conditions, given in [7,10] and in theorem 3.7 below, that ensure Kolmogorov's criteria are satisfied. These criteria are that:

(i) the total outgoing rate from each state (reciprocal of mean state holding time) is the same in both the forward and reversed processes;

(ii) The product of the rates around each cycle in the Markov state transition graph is the same in both the forward and reversed processes.

Inspection of the proof of RCAT shows that, even with state-dependent rates (called 'functional rates' by Hillston [12]), the total outgoing rate is the same in both the forward and reversed processes at every joint state of the cooperation. However, the second of Kolmogorov's criteria does not hold in general. Therefore, a weaker form of RCAT for cooperations with functional rates would read exactly the same but require that the products of rates around corresponding cycles in the forward and reversed processes be checked for equality. To have the first of Kolmogorov's criteria, such a generalisation would still require that the reversed rate x_a of an active action a be the same at all its instances.

3.1 Minimal cycles

To show the equality of the products of rates around *every* pair of corresponding cycles in the forward and reversed processes, it is actually only necessary to identify the *minimal cycles* and prove the equality around these. Minimal cycles are defined next and their number (also given below) is drastically less than the number of all cycles, which may be infinite. Consequently, the prospect of checking cycles individually is quite viable when the minimal cycles in the component processes are small and few in number.

Definition 3.1 A *cycle* in a Markov process with generator matrix Q is a sequence of transitions $\{i_0 \to i_1, i_1 \to i_2, \ldots, i_{n-1} \to i_n\}$, abbreviated by $i_0 \to i_1 \to \ldots \to i_{n-1} \to i_n$, where $n \geq 1$ and $i_n = i_0$. A cycle is *proper* if $i_j \neq i_k$ for $j \neq k, 1 \leq j, k \leq n-1$. A *composition* of cycles $i_0 \to i_1 \to \ldots \to i_{n-1} \to i_0$ and $j_0 \to j_1 \to \ldots \to j_{m-1} \to j_0$, where $j_0 = i_k$ for some $k, 0 \leq k \leq n-1$ is the cycle $i_0 \to i_1 \to \ldots \to i_k \to j_1 \to \ldots \to j_{m-1} \to i_k \to i_{k+1} \to \ldots \to i_{n-1} \to i_0$. A *minimal cycle* is a proper cycle that cannot be expressed as a composition of

smaller cycles.

We are concerned with cooperations that satisfy the condition that the reversed rate of every active action is the same at all its instances. Consequently, the agents R_k of theorem 3.7 pairwise satisfy lemmas 2.2 and 2.3. Consider now the ratios of products of rates around cycles in the cooperations of agents with residual actions, $\underset{k=1}{\overset{n}{\underset{L}{\bowtie}}} R_k^\epsilon\{(a,\top) \mid a \in \mathcal{P}_k(L_k)\}$ and $\underset{k=1}{\overset{n}{\underset{\overline{L}}{\bowtie}}} \overline{R_k^\epsilon}\{(\overline{a},\top) \mid a \in \mathcal{A}_k(L_k)\}$ where R_k^ϵ denotes the agent R_k in which all actions have been split with residual actions according to definition 2.1 [2] Here, it is not necessary to take into account synchronising transitions; these may be replaced by the appropriate pair of rectilinear transitions in single dimensions only, corresponding to the individual, synchronising components.

Definition 3.2 A *rectilinear cycle* in a Markov process defined by a cooperation is cycle in which every transition is denoted by an action with type $a \notin L$; i.e. by an action that occurs independently in a single component and does not synchronise. A *trivial* (rectilinear) cycle is one in which all transitions are in the same dimension, i.e. are denoted by actions that all occur in just one component, the states of the other component processes remaining constant. In a cooperation of n components, let $C_1, \ldots, C_n$ be minimal cycles in each distinct component process. A *basic* cycle (with respect to $C_1, \ldots, C_n$) is a non-trivial rectilinear cycle in which every transition is denoted by an action with type in one of the minimal cycles $C_k,\ 1 \leq k \leq n$.

By the preceding observations, it is sufficient to consider only rectilinear minimal cycles in RCAT-cooperations when checking the second of Kolmogorov's criteria. Notice that rectilinear cycles are *products* of cycles in each of the component processes. Some of these component cycles may be null, i.e. have no transitions; when all but one of the component cycles are null the rectilinear cycle is trivial. It is straightforward to construct the basic cycles of a cooperation from the given minimal cycles in the component processes, using a combinatorial algorithm that is easily mechanised. Their number is given by the following result.

Proposition 3.3 *The maximum number of basic cycles in a cooperation of n components with $m_1, \ldots, m_n$ minimal cycles respectively, starting at a given state in the state space, is*

$$\frac{(m_1 + \ldots + m_n)!}{m_1! \ldots, m_n!}$$

of which $m_1 + \ldots + m_n$ are trivial.

Proof. Consider a 2-component cooperation and assume that all (the actions denoting) the transitions in each cycle are enabled in every state of the cooperation. Then, if a cycle starts on a given one of its transitions, the number of basic cycles, which cannot be exceeded when some actions are disabled, is the number of distinguishable ways of arranging $m_1 + m_2$ objects of which m_1 are white and m_2 are black, i.e.

[2] Strictly it is defined recursively by $R_k^\epsilon = R_k^{|L|\epsilon}$ where $R_k^{m\epsilon} = \left(R_k^{(m-1)\epsilon}\right)^{a_m \epsilon}$ for $1 \leq m \leq |L|$, $R_k^{0\epsilon} = R_k$, $L = \{a_1, \ldots, a_{|L|}\}$.

$\begin{pmatrix} m_1+m_2 \\ m_1 \end{pmatrix}$. These include m_1+m_2 arrangements in which all black or all white occur consecutively in the cycle—corresponding to trivial rectilinear cycles. The argument extends simply to n-component cooperations with $n \geq 2$ by induction. □

The main result of this section identifies the minimal cycles of a cooperation as its basic cycles. We first define the *corners* that may occur in a cycle.

Definition 3.4 A *corner* at a state $\underline{i}$ is a pair of successive state transitions which occur in two different component processes P_j, P_k of a cooperation, e.g. one 'horizontal' and one 'vertical'. A *top-right corner*, denoted by $\rceil$, consists of successive transitions $\underline{i}_{j:-a} \to \underline{i} \to \underline{i}_{k:-b}$ or $\underline{i}_{k:-c} \to \underline{i} \to \underline{i}_{j:-d}$ for some integers $a, b, c, d \geq 1$, where $\underline{i}_{j:-a} = (i_1, \ldots, i_{j-1}, i_j - a, i_{j+1}, \ldots, i_n)$ in an $n-$component cooperation. A *bottom-left*, denoted $\lfloor$, *bottom-right*, denoted $\rfloor$ and *top-left*, denoted $\lceil$, corner are defined similarly to the $\rceil$-corner.

Proposition 3.5 *The basic cycles are the minimal cycles of a cooperation.*

Proof. First, a trivial basic cycle is clearly minimal. Next, in a non-trivial basic cycle, consider the transitions denoted by actions in one of the cooperating components. These must comprise a minimal cycle in that component's process, by definition of basic. Hence every basic cycle is minimal.

To prove the converse, consider an arbitrary finite cycle A in a cooperation of two components. We show that if this is not a basic cycle, it is a composition of simpler, ultimately basic, cycles. We enumerate the states of each compponent of the cooperation by the non-negative integers, so that the state space of the cooperation is in the upper right quadrant in two-dimensional space. We define the size $z(A)$ of a cycle A by $z(A) = \sum_{(i,j)\in A} i + j$.

In passing around any cycle, the direction of the successive transitions changes by a non-zero multiple of 2π. Each corner contributes a change of $\pm\pi/2$ and so each type of corner must occur in the cycle the same number $c > 0$ times. Consider a $\rceil$ corner (say), comprising (without loss of generality) transitions $(i, j-1) \to (i, j) \to (i-1, j)$, say, $i, j > 0$. Suppose that the transition $i \to i-1$ in component process 1 is in cycle C_1 in that process and, similarly, that the transition $j-1 \to j$ is in cycle C_2 in component process 2. Now let a rectilinear cycle C contain the $\rceil$ corner and consist of precisely all the transitions (denoted by actions) in cycles C_1 and C_2. Then A is a composition of some cycle A' and C. Moreover, $z(A') < s(A)$. This procedure cannot be repeated indefinitely since $z(A') > 0$ and so must ultimately lead to a trivial cycle A'. But by hypothesis, a trivial cycle is a composition of minimal cycles in one component process, which are basic cycles with the other component process's cycle being null.

This completes the proof for two-component cooperations. The result now follows by induction for cooperations of $n \geq 2$ components by the inductive definition of a multiple cooperation itself in terms of cooperations of two components. □

Kolmogorov's second criterion always holds for trivial cycles by the hypothesis that the reversed component processes $\overline{R_k}$ are given. Hence any rectilinear cycle is a composition of minimal (rectilinear) cycles and so, to verify the weaker theorem 3.7 with state-dependent rates, it is only necessary to consider the basic cycles directly; contrast the state-independent case where these automatically satisfy the second of Kolmogorov's criteria.

3.2 A weaker, more general, multi-agent RCAT

Before stating the main theorem, RCAT extended to multiple agents and functional rates, we first make the notion of a state-dependent rate more rigorous.

Definition 3.6 An action (a, λ) in a component P_k $(1 \leq k \leq n)$ of a cooperation $\underset{\substack{k=1\\L}}{\overset{n}{\bowtie}} P_k$ has a *functional rate* if λ depends on at least one of the derivatives of $\{P_j \mid j \neq k\}$.

The action type a may or may not be in the cooperation set L. In the Markov process denoted by a cooperation with functional rates, the transitions corresponding to an action with a functional rate are *state-dependent*. Some such cooperations still have separable equilibrium state probabilities, as given by the following:

Theorem 3.7 *(WMARCAT)*
Suppose that the cooperation $\underset{\substack{k=1\\L}}{\overset{n}{\bowtie}} P_k$ *of agents* P_k*, with functional rates, has a derivation graph with an irreducible subgraph* G *and that, for* $1 \leq k \leq n$*,*

- $R_k = P_k\{\top_a \leftarrow x_a \mid a \in \mathcal{P}_k(L_k)\}$;
- *every instance of a reversed action, type* $\overline{a}$*, of an* active *action type* $a \in \mathcal{A}_k(L_k)$ *has the same rate* $\overline{r_a}$ *in* $\overline{R_k}$;
- $\{x_a\}$ *are the unique solutions of the rate equations* $x_a = \overline{r_a}$*,* $a \in L$*.*

Then the agent

$$\underset{\substack{k=1\\\overline{L}}}{\overset{n}{\bowtie}} \overline{R_k}\{(\overline{a}, \top) \mid a \in \mathcal{A}_k(L_k)\}$$

with derivation graph containing the reversed subgraph $\overline{G}$*, is the reversed agent* $\overline{\underset{\substack{k=1\\L}}{\overset{n}{\bowtie}} P_k}$*, provided that*

(i) *every instance of each reversed action has the same rate (as noted above);*

(ii) $$\sum_{a \in \mathcal{P}^{\underline{i}\rightarrow}} x_a - \sum_{a \in \mathcal{A}^{\underline{i}\leftarrow}} x_a = \sum_{a \in \mathcal{P}^{\underline{i}\leftarrow} \setminus \mathcal{A}^{\underline{i}\leftarrow}} \overline{\beta_a^{\underline{i}}(\boldsymbol{x})} - \sum_{a \in \mathcal{A}^{\underline{i}\rightarrow} \setminus \mathcal{P}^{\underline{i}\rightarrow}} \alpha_a^{\underline{i}} \ ;$$

(iii) *The product of the transition rates around every non-trivial basic cycle* C *in the Markov process denoted by* $\underset{\substack{k=1\\L}}{\overset{n}{\bowtie}} P_k$ *is equal to the product of the transition rates around the corresponding reversed cycle* $\overline{C}$ *in the Markov process denoted by* $\underset{\substack{k=1\\\overline{L}}}{\overset{n}{\bowtie}} \overline{R_k}\{(\overline{a}, \top) \mid a \in \mathcal{A}_k(L_k)\}$.

Furthermore, assuming the cooperation set L is finite, the cooperation has separable solution

$$\pi(\underline{i}) \propto \prod_{k=1}^{n} \pi_k(i_1, \ldots, i_k, 0, \ldots, 0)$$

for the equilibrium probability of state $\underline{i} = (i_1, \ldots, i_n)$, where $\pi_k(\underline{i})$ is proportional to the equilibrium probability of state i_k in the process denoted by R_k when the state of each other component process $j \neq k$ is fixed at i_j.

Proof. The proof that the first of Kolmogorov's criteria is satisfied is essentially the same as for RCAT in the extended form of [10]—the functional rates are not significant since we only consider one joint state and the reversed rates of active actions are constant. The second criterion is satisfied by the analysis in the preceding section—note that we need not consider trivial (basic) cycles since these satisfy it by the hypothesis that each agent $\overline{R_k}$ is known.

For the second part of the theorem, we consider rectilinear paths from state $\mathbf{0}$ to state $\underline{i}$, following the state-space dimensions in the order $1, 2, \ldots, n$. In the path segment in dimension k, the ratio of forward to reversed rates is then (by hypothesis) $\pi_k(i_1, \ldots, i_k, 0, \ldots, 0)$ and the result follows. □

3.3 Queueing networks with state-dependent rates

Consider an M-node Jackson network in which the service rate at each queue may depend on the lengths of any of the queues. Let the M nodes have respective constant external arrival rates $\lambda_1, \ldots, \lambda_M$, state-dependent service rates $\mu_1(\underline{i})$, …, $\mu_M(\underline{i})$ in state $\underline{i} = (i_1, \ldots, i_M)$, and routing probability p_{ij} from node i to node j $(1 \leq i \neq j \leq M)$, where $p_{ii} = 0$. Tasks leave the network from node i with probability $p_{i0} = 1 - \sum_{j=1}^{M} p_{ij}$. 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 is easily specified in PEPA with functional rates as: $\underset{L}{\overset{M}{\bowtie}}_{k=1} P_{k,0}$ (starting with an empty network), where, for $1 \leq k \leq M$:

$$\begin{aligned}
P_{k,n} &= (e_k, \lambda_k) P_{k,n+1} && n \geq 0 \\
P_{k,n} &= (a_{jk}, \top_{jk}) P_{k,n+1} && n \geq 0, 1 \leq j \neq k \leq M \\
P_{k,n} &= (d_k, p_{k0}\mu_k(\underline{i})) P_{k,n-1} && n > 0 \\
P_{k,n} &= (a_{kj}, p_{kj}\mu_k(\underline{i})) P_{k,n-1} && n > 0, 1 \leq j \neq k \leq M
\end{aligned}$$

with $L_k = \{a_{kj} \mid j \neq k\} \cup \{a_{jk} \mid j \neq k\}$. The functional rates imply that the service rate of server k is $\mu_k(\underline{i})$ when component P_j of the cooperation is at derivative P_{j,i_j}, i.e. when its underlying Markov process is in state i_j or the queue length at node j is i_j, $1 \leq j \leq M$.

Every occurrence of the reversed action of an active action, type a_{kj} say, in $\mathcal{A}_k$ is a constant fraction of the constant net arrival rate λ_k, since each component is an M/M/1 queue. Hence, condition 1 of WMARCAT is satisfied and we obtain the

following rate equations for agent $P_{j,0}$ $(j = 1, \ldots, M)$:

$$x_{ij} = p_{ij}\big(\lambda_i + \sum_{k=1}^{M} x_{ki}\big) \quad (i = 1, \ldots, M)$$

(We use the abbreviation x_{ij} for $x_{a_{ij}}$, $1 \leq i \neq j \leq M$.)

Now let $v_i = \lambda_i + \sum_{k=1}^{M} x_{ki}$ for $1 \leq i \leq M$ so that:

$$x_{ij} = v_i p_{ij} \qquad 1 \leq i, j \leq M$$

These are precisely the traffic equations for the internal flows, where x_{ij} is the internal traffic rate from node i to node j. A solution therefore always exists in an irreducible network. In fact, summing over i we obtain

$$v_j - \lambda_j = \sum_{i=1}^{M} v_i p_{ij}$$

which are the usual traffic or 'visitation rate' equations for the network, v_i being the average number of visits made to node i in unit time at equilibrium in an open network—and proportional to this quantity in a closed network.

The second condition of WMARCAT holds trivially, because all passive actions are enabled in every state in both the forward and reversed processes. For the third condition, relating to minimal cycles, we have to check the non-trivial basic cycles formed from every pair of component processes. Each component denotes an M/M/1 queue and so has just one minimal cycle of the form $i-1 \to i \to i-1$ for $i \geq 1$. There are therefore only two (parameterised) basic cycles, squares of the form:

Anti-clockwise: $(i-1, j-1) \to (i, j-1) \to (i, j) \to (i-1, j) \to (i-1, j-1)$ and

Clockwise: $(i-1, j-1) \to (i-1, j) \to (i, j) \to (i, j-1) \to (i-1, j-1)$

where $i, j \geq 1$ are states in the respective component processes. Furthermore, since the reversed process of an M/M/1 queue is the same M/M/1 queue (easily checked since this queue satisfies detailed balance [14]), the clockwise and anticlockwise squares are reversed cycles of each other. Hence it is sufficient to prove that the products of the rates around these two squares are equal. Now, the rate $i-1 \to i$ in the queue denoted by component R_k in WMARCAT is the constant traffic rate v_k defined above—it does not depend on the service rate function. Hence we have to show, for the basic cycles derived from components h and k, $1 \leq h \neq k \leq n$, that

$$v_h v_k \mu_h(\underline{i}) \mu_k(\underline{i}_h) = v_k v_h \mu_k(\underline{i}) \mu_h(\underline{i}_k)$$

where $\underline{i}_k = (i_1, \ldots, i_{k-1}, i_k - 1, i_{k+1}, \ldots, i_n)$—$\underline{i}_{k:-1}$ in previous notation. This simplifies to

$$\frac{\mu_k(\underline{i}_h)}{\mu_k(\underline{i})} = \frac{\mu_h(\underline{i}_k)}{\mu_h(\underline{i})} \tag{1}$$

for $1 \leq h \neq k \leq n$ and valid states $\underline{i}$. Thus, any set of service rate functions that satisfy these equations will also validate the third condition of WMARCAT. The reversed PEPA agent of $\underset{k=1}{\overset{M}{\bowtie}}\limits_{L} P_{k,0}$ now follows directly as $\underset{k=1}{\overset{M}{\bowtie}}\limits_{\overline{L}} X_{k,0}$, where, for $1 \leq j, k \leq M$:

$$X_{k,n} = (\overline{e_k}, \frac{\lambda_k}{v_k}\mu_k(\underline{i}))X_{k,n-1} \qquad n > 0$$

$$X_{k,n} = (\overline{a_{jk}}, \frac{x_{jk}}{v_k}\mu_k(\underline{i}))X_{k,n-1} \qquad n > 0$$

$$X_{k,n} = (\overline{d_k}, (1 - \sum_{j \neq k} p_{kj})v_k)X_{k,n+1} \qquad n \geq 0$$

$$X_{k,n} = (\overline{a_{kj}}, \top)X_{k,n+1} \qquad n \geq 0$$

with $\overline{L_k} = \{\overline{a_{kj}} \mid j \neq k\} \cup \{\overline{a_{jk}} \mid j \neq k\}$.

The rates for the reversed actions are easily calculated using the rule for apportioning rates to reversed sub-actions; see section 2. For example, consider the reversed external arrivals at node 1, which have type $\overline{e_1}$. The total departure rate of node 1 is $\mu_1(\underline{i})$ and the proportion of e_1 in the forward process is $\frac{\lambda_1}{v_1}$. Hence the rate for the reversed action $\overline{e_1}$ is $\frac{\lambda_1}{v_1}\mu_1$.

A separable solution for the network's equilibrium probabilities follows similarly. This is a very general result, but what suitable service rate functions exist, if any?

3.3.1 Network-load dependent servers

Suppose that the service rate at node k is modified multiplicatively according to both the global state of the network and the local state of node k. That is,

$$\mu_k(\underline{i}) = g(\underline{i})\mu'_k(i_k)$$

for certain functions g and μ'_k—g being the same for all component nodes. Equation 1 then implies, for all $\underline{i}$ with $i_h, i_k > 0$, $g(\underline{i}_h) = g(\underline{i}_k)$. Applying this equation repeatedly therefore leads to

$$\begin{aligned} g(\underline{i}) &= g(i_1 + 1, i_2 - 1, i_3, \ldots, i_n) \\ &= g(i_1 + i_2, 0, i_3, \ldots, i_n) \\ &= g(i_1 + i_2 + \ldots + i_n, 0, \ldots, 0) \end{aligned}$$

which is a function only of the total population of the network, which we abbreviate to $g(i_1 + i_2 + \ldots + i_n)$. Applying WMARCAT we obtain the following separable state probabilities when equilibrium exists:

Proposition 3.8 *A steady state Markovian queueing network with constant arrival rates and state-dependent service rates of the form $\mu_k(\underline{i}) = g(N)\mu'_k(i_k)$ at node $k = 1, \ldots, M$ in state $\underline{i}$, where $N = \sum_{k=1}^{n} i_k$ is the network population, has equilibrium probabilities*

$$\pi(\underline{i}) \propto \frac{\prod_{k=1}^{n} \left(v_k^{i_k} / \prod_{j=1}^{i_k} \mu'_k(j) \right)}{\prod_{j=1}^{N} g(j)}$$

Proof. Applying WMARCAT, we find

$$\pi(\underline{i}) \propto \prod_{k=1}^{M} \left(v_k^{i_k} / \prod_{j=1}^{i_k} g(i_1 + \ldots + i_{k-1} + j) \mu'_k(j) \right)$$

and the result follows. □

3.3.2 Generalised processor sharing in Coxian queues

It is fairly well known that a node with a global resource-sharing queueing discipline contributes a separable factor in a product-form solution for the equilibrium probabilities in a network containing that node. This factor is duly given by the result of the previous section, with $g(\underline{i}) = 1/\sum_{k=1}^{M} i_k$. However, it is not so well known that the functional service rate dependence can be *any* function of the current network population, not just inverse proportion. For example, the rate might decrease less rapidly as the population increases, such as inversely with its square root or logarithm, or perhaps increase linearly or quadratically, or more exotically, as would be given by $g(\underline{i}) = \sin\left(\sum_{k=1}^{M} i_k\right)$.

An S-phase Coxian random variable is usually thought of as the truncated sum of a finite series of $S \geq 1$ exponential delays. The probability of truncating after s delays is $a_1 a_2 \ldots a_{s-1}(1 - a_s)$, where $a_S = 0$. Thus, a queueing node with processor sharing (PS) queueing discipline and S-phase Coxian service times can be modelled as a standard, tandem, Jackson network of S nodes in which departures from the network after service at node s occur with probability $1 - a_s$, $1 \leq s \leq S$. All customers receive service concurrently at a rate inversely proportional to the number at the Coxian node, i.e. to the number in the S-node Jackson network. Any number of customers can be in each stage at the same time since there is no blocking of customers. Each customer at stage s receives service at rate $\mu_s/(i_1 + \ldots + i_S)$ in state $\underline{i}$, giving a service rate function at that stage of $i_s\mu_s/(i_1 + \ldots + i_S)$. However, the dependence on the global state could be *any* function of the Coxian node population, not just inverse proportion, giving a service rate function $i_s\mu_s g(i_1 + \ldots + i_S)$ for the chosen function g. In this way, we obtain the queue length distribution at a Coxian node of:

$$\pi(\mathbf{i}) \propto \frac{\lambda^N}{i_1! \ldots i_S! \prod_{j=1}^{N} g(j)} \prod_{k=1}^{S} \left(\frac{a_1 \ldots a_{k-1}}{\mu_i} \right)^{n_i}$$

where $N = i_1 + \ldots + i_S$. In the special case of conventional PS discipline, this becomes

$$\pi(\mathbf{i}) \propto \frac{N! \lambda^N}{i_1! \ldots i_S!} \prod_{k=1}^{S} \left(\frac{a_1 \ldots a_{k-1}}{\mu_i} \right)^{n_i}$$

Summing over $i_1, \ldots, i_S$ such that $\sum_{k=1}^{S} i_k = N$ then yields the equilibrium queue length probability (by a routine application of the multinomial theorem)

$$\pi(N) \propto \rho^N$$

where $\rho = \lambda/\overline{\mu}$ and $\overline{\mu}^{-1} = \sum_{k=1}^{S} a_1 \dots a_{k-1}\mu_k^{-1}$ is the mean service time of the coxian server.

We can now apply theorem 3.7 to obtain a product-form, cf. [1], for a network of queues with either FCFS queueing discipline and exponential service time or GPS discipline and Coxian service time. Last come first served (LCFS) queueing discipline with Coxian service times can also be included in the RCAT framework as described in [10], and also infinite servers (IS) analogously to PS. In every case, all passive actions are enabled in every state of both the forward and reversed cooperations, the required reversed rates x_a are given by the traffic equations and so WMARCAT can be applied, giving the known product-form. Extension to the multi-class case is also straightforward, as discussed in [10].

4 Conclusion

The Weak Multiple Agents Reversed Compound Agent Theorem (WMARCAT) greatly simplifies the use of its predecessor, RCAT, for cooperations of an arbitrary number of agents. More significantly, allowing global state dependence in synchronised actions' rates, i.e. functional rates, leads to the weaker form of WMARCAT, based on direct analysis of the minimal cycles in state transition graphs. This benefits from a simple proof using residual actions and directly yields separable, but non-product, forms when reversed processes can be found. The main application of this result is a new, mechanisable derivation of the multiclass BCMP theorem for networks of queues with PS servers, which generalises to a wider class of queueing networks with subnetworks of globally state-dependent servers. New separable solutions were also found, to the authors' best knowledge.

The methodology can be automated and its newly generalised, multi-agent form facilitates the uniform derivation of many diverse separable solutions, as considered just for two-component cooperations in [9,10]. These applications range from multiclass queueing networks, through the numerous variants of G-networks, to networks with mutual exclusion and blocking in critical sections.

References

[1] F. Baskett, K. M. Chandy, R. R. Muntz and F. Palacios, Open, Closed and Mixed Networks of Queues with Different classes of Customers, J. ACM, 22(2):248-260, 1975.

[2] M. Bernardo, L. Donatiello and R. Gorrieri. Integrating performance and functional analysis of concurrent systems with EMPA, Proc. of the 1st Workshop on Distributed Systems: Algorithms, Architectures and Languages, pp. 5-6, Levico (Italy), June 1996.

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

[4] X. Chao, M. Miyazawa and M. Pinedo. *Queueing networks: customers, signals and product form solutions*. Wiley, 1999

[5] E. Gelenbe, The first decade of G-networks, European Journal of Operational Research, 126(2): 231-232, October 2000.

[6] William J. Gordon and Gordon F. Newell, Closed Queueing Systems with Exponential Servers, Operations Research, 15(2): 254-265, Mar-Apr 1967.

[7] Peter G Harrison. Turning Back Time in Markovian Process Algebra. Theoretical Computer Science, 290(3):1947-1986, January 2003.

[8] Peter G Harrison. Mechanical Solution of G-networks via Markovian Process Algebra. In Proc. International Conference on Stochastic Modelling and the IV International Workshop on Retrial Queues, Cochin, India, December 2002, Notable Publications, 2002.

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

[10] Peter G. Harrison. Reversed processes, product forms and some non-product forms. *J. Linear Algebra Applications*, to appear, 2004.

[11] H. Hermanns, M. Rettelbach, and T. Wei. Formal Characterisation of Immediate Actions in SPA with Nondeterministic Branching. The Computer Journal, 38(7):530-541, 1995.

[12] Jane Hillston. A Compositional Approach to Performance Modelling. PhD thesis, University of Edinburgh, 1994.

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

[14] Frank P. Kelly. Reversibilities and Stochastic Networks. John Wiley & Sons Ltd, 1979.

Appendix A: A PEPA-based MPA

We use a Markovian process algebra language that defines *agents*, which denote continuous time Markov chains. Agents evolve through the execution of *actions*, which have exponentially distributed durations. An action is a pair, the first component of which is its *type* (or name) and the second of which is its *rate*. Thus, agents and actions in an MPA specification correspond to states and transitions respectively in the underlying Markov process. MPA describes systems at a higher level than explicit state-transition diagrams. In particular, the *cooperation* combinator of PEPA defines precisely how agents interact in a concise manner, using generic descriptions of their actions' rates. The precise semantics of the original PEPA language is given in [12], and defines the Markov process denoted by a PEPA agent. Notice that the term 'agent' is *syntactic*, part of the MPA, whereas 'process' is a semantic entity with a well defined value in the domain of continuous time Markov chains. However, the terms are essentially isomorphic.

In this paper, we use only the *prefix* and *cooperation* combinators of the MPA PEPA (generalised straightforwardly in the body of the paper to multiple cooperations):

(i) The prefix combinator defines an agent $(a, \lambda).P$ that carries out action (a, λ) of *type* (or 'name') a at *rate* λ and subsequently behaves as agent P;

(ii) The agent describing the cooperation of two agents P and Q, which synchronise over actions with types in a specified set L, is written $P \underset{L}{\bowtie} Q$.

In the cooperations considered in this paper, every action type in L is *active*, i.e. has a specified real valued rate, in exactly one of the agents P, Q and is *passive*, i.e. 'waits', in the other. The rate of the joint action in the cooperation is then that specified for the active action. A passive action is indicated by an *unspecified* rate, denoted $\top$, essentially infinite in the sense that the action will proceed instantly once its synchronising action is ready. Any action with type in L can only proceed simultaneously in both of the cooperating agents. The Markov process denoted

by a cooperation has a state space with two dimensions, corresponding to each component of the cooperation respectively; relating to WMARCAT (theorem 3.7 in section 3.2) where $n \geq 2$ components cooperate, the state space has n dimensions, similarly corresponding to each component.

New agents are defined using an *assignment combinator*, $A = P$, and the *relabeling*, $P\{y \leftarrow x\}$, denotes the process P in which all occurences of the symbol y are changed to x, which may be an expression. Thus, for example, $((a, \lambda).P)\{\lambda \leftarrow \mu\}$ denotes the agent $(a, \mu).P\{\lambda \leftarrow \mu\}$. Choice is denoted by multiple assignments to a process name rather than the separate combinator symbol + of conventional PEPA. Reversed entities (agents, actions, action types, action rates) are denoted with an overbar.

Appendix B: Reversed processes

A stochastic process's *reversed process* is simply the process obtained by looking 'backwards in time'. Its key property is that the reversed Markov process of a stationary Markov process $\{X_t\}$ with state space S, generator matrix Q and stationary probabilities $\boldsymbol{\pi}$ has generator matrix Q' defined by

$$q'_{ij} = \pi_j q_{ji}/\pi_i \quad (i, j \in S)$$

and the same stationary probabilities $\boldsymbol{\pi}$.

This result is standard, see for example [14], and immediately yields a product-form solution for $\boldsymbol{\pi}$. This is because, in an irreducible Markov process, we may choose a reference state 0 arbitrarily, find a sequence of connected states, in either the forward or reversed process, $0, \ldots, j$ (i.e. with either $q_{i,i+1} > 0$ or $q'_{i,i+1} > 0$ for $0 \leq i \leq j-1$) for any state j and calculate

$$\pi_j = \pi_0 \prod_{i=0}^{j-1} \frac{q_{i,i+1}}{q'_{i+1,i}} = \pi_0 \prod_{i=0}^{j-1} \frac{q'_{i,i+1}}{q_{i+1,i}}$$