Graph-Based Product Form

Céline Comte[1]
LAAS–CNRS, Université de Toulouse, CNRS, Toulouse, France

Isaac Grosof[1]
Industrial Engineering and Management Science, Northwestern University, Evanston, IL, USA
Electrical and Computer Engineering, University of Illinois, Urbana-Champaign, Urbana, IL, USA

Abstract

Product-form stationary distributions in Markov chains have been a foundational advance and driving force in our understanding of stochastic systems. In this paper, we introduce a new product-form relationship that we call "graph-based product form". As our first main contribution, we prove that two states of the Markov chain are in graph-based product form if and only if the following two equivalent conditions are satisfied: (i) a cut-based condition, reminiscent of classical results on product-form queueing systems, and (ii) a novel characterization that we call joint-ancestor freeness. The latter characterization allows us in particular to introduce a graph-traversal algorithm that checks product-form relationships for all pairs of states, with time complexity $O(|V|^2|E|)$, if the Markov chain has a finite transition graph $G = (V, E)$. We then generalize graph-based product form to encompass more complex relationships, which we call "higher-level product form", and we again show these can be identified via a graph-traversal algorithm when the Markov chain has a finite state space. Lastly, we identify several examples from queueing theory that satisfy this product-form relationship.

2012 ACM Subject Classification Mathematics of computing → Markov processes; Applied computing → Operations research

Keywords and phrases Markov chain, product-form, directed graph, graph cut

Funding *Isaac Grosof*: Supported by AFOSR Grant FA9550-24-1-0002 and by a startup package from Northwestern University

1 Introduction

Important classes of queueing systems and stochastic networks have been shown to have a so-called *product-form stationary distribution*, where the stationary probability of a given state has a simple multiplicative relationship to the stationary probability of other nearby states. The product-form property allows the stationary distribution of these systems to be cleanly and precisely characterized in closed-form, which is not possible for many other queueing systems. Such characterization has been instrumental in numerically evaluating performance [1, 2] and analyzing scaling regimes [3, 4, 5], and it was more recently applied in the context of reinforcement learning [6, 7].

Important queueing systems and stochastic networks that have been proven to exhibit product-form behavior include Jackson networks [8], BCMP networks [9], Whittle networks [10], and networks of order-independent queues [11] (also see [12]). Similar structures have appeared in other fields, such as statistical physics, with the zero-range process [13]. Discovering these categories of product-form systems and the underlying properties that give rise to their product-form behavior has represented a foundational advance and driving force in our understanding of stochastic systems.

Product-form results are often tied to time reversibility or quasi-reversibility properties. These can be established through the detailed balance property given the stationary

[1] Authors are joint first authors, and are written in alphabetical order.

distribution, as well as by applying Kolmogorov's criterion on the transition rates of the Markov chain [14]. In the simplest case of birth-and-death processes, both product-form and reversibility are implied by the transition diagram of the Markov chain.

However, there are important queueing systems that exhibit product-form behavior which cannot be explained under any existing product-form framework. A motivating example for this paper is the multiserver-job saturated system with two job classes, which Grosof et al. [15] demonstrated to have a product-form stationary distribution. We explore this system via our novel framework in Section 5.1.

We introduce a new kind of product-form Markov chain, *graph-based* product form. In these Markov chains, product-form arises purely from the connectivity structure of the transition graph, or in other words from the set of transitions with nonzero probability, for discrete-time Markov chains, abbreviated DTMCs; or nonzero rate, for continuous-time Markov chains, abbreviated CTMCs. If a Markov chain has graph-based product form, that product form holds regardless of its transition probabilities or rates, under a given connectivity structure. This is in contrast to most prior classes of product-form relationships, where tweaking a single transition probability or rate would remove the product-form property.

In this paper, we characterize which directed graphs hold the correct structure to give rise to graph-based product-form Markov chains. Our characterization is built up from a product-form relationship between states (i.e., nodes in the transition diagram), which exists when the ratio of the stationary probabilities for two nodes forms a simple multiplicative relationship, arising from the graph structure. If those relationships span the graph, then the whole Markov chain has graph-based product form. We therefore focus on characterizing which graphs give rise to product-form relationships between a given pair of nodes.

In our main result, Theorem 2, we give two equivalent necessary and sufficient conditions under which such a product-form relationship exists: a cut-based characterization, reminiscent to classical conditions for product-form, and a novel characterization which we call *joint-ancestor freeness*. More specifically, focusing on two particular nodes or states i and j:

- *Cut-based characterization*: For the first condition, we show that if there exists a cut (i.e. a partition of the nodes into two sets), where i is on one side of the cut and j is on the other, and where the only edges that cross the cut have either i or j as their source nodes, then i and j have a product-form relationship. We call such a cut an *i, j-sourced cut*. Unfortunately, directly searching for such cuts is inefficient and impractical, as there are exponentially many cuts in the graph.
- *Joint-ancestor freeness*: We show that the existence of such a cut is equivalent to a second, simpler-to-check property, which we call joint-ancestor freeness. We refer to a node k as a joint ancestor of i and j if there exists a path from k to i which does not go through j, and a path from k to j which does not go through i. We show that the existence of an i, j-sourced cut is equivalent to i and j having no joint ancestors, which is efficient to directly search for.

Finally, we show in Theorem 7 that this relationship is bidirectional: If there is no i, j-sourced cut, or equivalently if there is a joint ancestor k, then nodes i and j will not have a straightforward product-form relationship.

Even in graphs where the most straightforward product-form relationships do not connect every pair of nodes, a less-direct kind of product-form relationship can still exist, which we call "higher-level product form". We call the above product-form relationships "first-level product-form", and we show that a weaker, but still noteworthy, kind of product-form relationship, "second-level product-form", exists whenever there exists a cut with multiple nodes as sources on one or both sides of the cut, such that the sources on each side are connected by first-level product-form relationships. Further levels can be defined recursively. We study this higher-level product-form relationship in Section 4.3, with Section 5.2 as a motivating example.

1.1 Contributions

In Section 2.2, we define the novel concept of graph-based product form. In Section 3, and specifically Theorem 2, we prove that graph-based product form between two nodes i and j is equivalent to two graph-based properties: The existence of an i, j-sourced cut, and the absence of a joint ancestor of i and j. In Sections 4.1 and 4.2, we introduce the cut graph and its connection to graph-based product form spanning an entire Markov chain. In Section 4.3, we explore and characterize higher-level product form relationships, which we show correspond to higher-level cuts. In Sections 3.2 and 5, we give a variety of examples of graphs which do or do not have graph-based product form, and use them to illustrate our characterization.

1.2 Prior work

There is a massive literature that focuses on deriving the stationary distribution (or a stationary measure) of Markov chains with countable (finite or infinite) state spaces. In reviewing this literature, we focus on results that either provide a closed-form expression for the stationary measures or make structural assumptions on the Markov chain, or both.

Reversibility, quasi-reversibility, and partial balance

A long series of works has derived product-form stationary distributions by focusing on Markov chains where a stronger form of the balance equations holds, thus balancing the probability flow between a state and each set in a partition of its neighbors. This is often equivalent to properties of the time-reversed process [10]. For example, the Kolmogorov criterion [10, Theorem 2.8] is a necessary and sufficient condition for reversibility, which as a by-product yields a closed-form expression for the stationary distribution as a product of transition rates. Another example is quasi-reversibility, as described in [14, Chapter 3]. Among these works, many have focused on Markov chains exhibiting a specific transition diagram, e.g., multi-class queueing systems with arrivals and departures occurring one at a time, and have identified necessary and sufficient conditions on the transition rates that yield a product-form stationary distribution. This approach has therefore produced many models applicable to queueing theory and statistical physics. Reversible models and their variants involving internal routing include the celebrated Jackson networks [8], the zero-range process [13], and Whittle networks [10]. Quasi-reversibility has also given rise to multiple models, including order-independent queues [11, 16] and pass-and-swap queues [17]; see [18] for a recent survey. Other examples of queueing models that satisfy partial balance equations are token-based order-independent queues [19] and certain saturated multiserver-job queues [20, 15].

Graph-based product form

To the best of our knowledge, very few papers exploit the structure of a Markov chain's transition diagram (rather than its transition *rates*) to guarantee the existence of a product-form stationary distribution. One example is [21], which introduces *single-input super-state decomposable Markov chains*: the Markov chain's state space is partitioned into a finite number of sets, called superstates, such that all edges into a superset have the same node as endpoint. (All finite-state-space Markov chains satisfy this condition when the partition is formed by singletons.) Under this assumption, the process of deriving the Markov chain's stationary distribution can be divided into two steps, one that solves the stationary distribution of a Markov chain defined over the superstates, and another that solves the stationary distribution of a Markov chain restricted to each superstate. While the superstate decomposition has a loose resemblance to our cuts, there is no deeper similarity between the methods. In particular, our approach is nontrivial both for finite and infinite Markov chains. The superstate decomposition approach can be seen as a different approach to deriving product-forms.

Closer to our work, Grosof et al. [15] consider a multiserver-job (MSJ) model described by a CTMC and show that it has a product-form stationary distribution irrespective of the transition rates. This result is proven in more detail in a technical report [22]. This example, which inspired the present work, is discussed in detail in Section 5.1.

Matrix-geometric methods

Another well-known family of methods that exploits structural properties of Markov chains is matrix-geometric methods [23, 24, 25], developed for the analysis of quasi-birth-and-death (QBD) processes. A Markov chain is called a QBD process if its states can be partitioned into disjoint superstates indexed by $0, 1, 2, \ldots$, such that transitions occur either within a superstate (inner transitions), from a superstate $i \in \{0, 1, 2, \ldots\}$ to superstate $i + 1$ (upward transition), or from a superstate $i \in \{1, 2, \ldots\}$ to superstate $i - 1$ (downward transition). The graph-based product form we consider does have intersection with QBD processes, in the sense that there exist QBD processes that can be analyzed with the prism of graph-based product form; see Example 6 and Sections 5.1 and 5.2. However, QBD processes and graph-based product form are two different notions: there are QBD processes that do not have a graph-based product form, and there are Markov chains that exhibit graph-based product form but are not QBD processes; see Examples 2, 3, and 5, Section 5.3, and Appendix A. Furthermore, as we can see in Section 5.2, there are QBD processes for which the framework of graph-based product form allows us to derive the stationary distribution more directly, without resorting to matrix-geometric methods. Lastly, to the best of our knowledge, matrix-geometric methods are used mainly when the Markov chain shows repetitive patterns, i.e., when the inner (resp. upward, downward) transitions are similar across superstates, except for superstate 0; graph-based product form is not so much related to repetition as to exploiting the structure of the transition diagram.

Symbolic solutions

Our graph-based product-form method can also be seen as an algorithmic way to discover a particular type of product-form relationship in Markov chains, giving a clean symbolic solution for the stationary distribution. If the Markov chains are structured, as in the examples in Section 5, these relationships can be found by direct inspection. However, if algorithm searching is required, we give an algorithmic approach to discover single-source cuts in the underlying graph in Algorithm 3 in $O(|V|^2|E|)$ time, if the Markov chain has a finite transition graph $G = (V, E)$, allowing us to discover whether a product-form relationship exists.

Prior to our approach, one could symbolically find the stationary distribution for a general symbolic Markov chain in $O(|V|^2)$ time, by symbolically solving the balance equations.

However, there is no guarantee that the resulting symbolic expression would be in a simple form. Simplifying and factorizing the resulting symbolic expression, which might have $O(|V|^2)$ terms, does not have a known efficient, deterministic algorithm. In fact, polynomial factorization is a more complicated version of the polynomial identity testing (PIT) problem, for which no polynomial-time deterministic algorithm is known [26]; the two problems were recently proven equivalent, in the sense that a deterministic polynomial-time algorithm for one would imply the same for the other [27]. Finding such an algorithm has remained a major open problem.

Other related methods

Product-form stationary distributions for DTMCs or CTMCs have been studied in many different contexts, such as Stochastic Petri networks, which sometimes lead to constructive and algebraic methods that assume particular structure of the transition rates [28, 29]. Orthogonally, the graph structure of a Markov chain has also been used for other purpose than deriving a simple closed-form expression for the stationary distribution. For instance, the survey [30] focuses on iterative methods to approximate the stationary distribution of a finite Markov chain with transition matrix A using updates of the form $\pi_{t+1} = \pi_t A$. The algorithms described in [30], called *aggregation-disaggregation methods*, aim at speeding-up iterative methods by occasionally replacing π_{t+1} with $\tilde{\pi}_{t+1} = S(\pi_{t+1})$, where S is a function that exploits structure in the Markov chain's transition diagram.

2 Model and definitions

We start by introducing preliminary graph notation and terminology in Section 2.1, in particular *set-avoiding paths* and *ancestor sets*, then we introduce the key notions of a *formal Markov chain* and *graph-based product form* in Section 2.2.

2.1 Graph notation and terminology

The focus of this paper is on the directed graphs that underlie Markov chains and on cuts in these graphs. Besides recalling classical graph-theoretic notions, we introduce *set-avoiding paths* and *ancestor sets* that will be instrumental in the rest of the paper.

A *directed graph* is a pair $G = (V, E)$, where V is a countable set of nodes, and $E \subseteq V \times V$ is a set of directed edges. The graph G is called *finite* if V is finite and *infinite* if V is countably infinite. A *cut* of a directed graph $G = (V, E)$ is a pair (A, B) of nonempty sets that form a partition of V, that is, $A \cup B = V$ and $A \cap B = \emptyset$. An edge $(u, v) \in E$ is then said to cross the cut (A, B) if either $u \in A$ and $v \in B$, or $u \in B$ and $v \in A$. A *path* in a directed graph $G = (V, E)$ is a sequence $v_1, v_2, \ldots, v_n$ of *distinct* nodes in V, with $n \in \{1, 2, \ldots\}$, such that $(v_p, v_{p+1}) \in E$ for each $p \in \{1, 2, \ldots, n-1\}$; the length of the path is the number $n-1$ of edges that form it. In particular, a path of length 0 consists of a single node and no edges. A graph $G = (V, E)$ is called *strongly connected* if, for each $i, j \in V$, there exists a path from node i to node j in the graph G.

In Section 2.2, we will relate the existence of cuts that yield convenient balance equations with the following two definitions that will be instrumental in the paper.

▶ **Definition 1** (Set-avoiding subgraph and set-avoiding path). *Consider a directed graph $G = (V, E)$ and let $U \subseteq V$. The* set-avoiding subgraph *$G \backslash U = (V \backslash U, E')$ is defined with $E' = \{(i, j) \in E : i, j \notin U\}$. Given $i, j \in V \backslash U$, we let $P(i \to j \backslash U)$ denote an arbitrary path $v_1, v_2, \ldots, v_n$ in G, with $n \in \{1, 2, \ldots\}$, with source node $v_1 = i$ and destination node $v_n = j$,* *and such that $v_p \notin U$ for each $p \in \{1, 2, \dots, n\}$. Such a path is said to* avoid *the set U. Equivalently, a path $P(i \to j \backslash U)$ is a path from node i to node j in the subgraph $G \backslash U$. If $U = \{u\}$ is a singleton, we write $G \backslash u$ for $G \backslash \{u\}$ and $P(i \to j \backslash u)$ for $P(i \to j \backslash \{u\})$.*

▶ **Definition 2** (Ancestor and ancestor set). *Consider a directed graph $G = (V, E)$ and let $i, j \in V$. Node i is called an* ancestor *of node j (in G) if there exists a directed path from node i to node j (in G), i.e., if there exists a path $v_1, v_2, \dots, v_n$ (in G) with $v_1 = i$ and $v_n = j$. For each $i \in V$, $A_i(G)$ denotes the set of ancestors of node i (in G). For each $I \subseteq V$, $A_I(G) = \bigcup_{i \in I} A_i(G)$ denotes the* ancestor set *of node set I (in G).*

The ancestor set of a node contains the node itself (via a path of length zero), so that $I \subseteq A_I(G)$ for each $I \subseteq V$. A directed graph $G = (V, E)$ is strongly connected if and only if the ancestor set of each node is the whole set V. Procedure Ancestors in Algorithm 1 is a classical breadth-first-search algorithm that returns the ancestor set of a node set in a finite graph G. This algorithm can run in time $O(|E|)$ with the appropriate data structure (e.g., the graph is encoded as a list of ancestor lists for each node) because each edge is visited at most once over all executions of Algorithm 1.

Algorithm 1 Returns the ancestor set of a node set in a finite graph

procedure Ancestors(finite directed graph $G = (V, E)$, set $I \subseteq V$) $\to$ set $A \subseteq V$
  $A \leftarrow \emptyset$ ▷ Ancestor set under construction
  $F \leftarrow I$ ▷ Set of "frontier" nodes: nodes that have been visited
  ▷ but whose neighbor list has not yet been read
  while $F \neq \emptyset$ **do**
    $A \leftarrow A \cup F$
    $N \leftarrow \bigcup_{\ell \in F} \{k \in V \backslash A : (k, \ell) \in E\}$ ▷ New frontier nodes
    $F \leftarrow N$
  end while
  return A
end procedure

2.2 Markov chains and product-form relationship

As announced in Section 1, our goal is to identify necessary and sufficient conditions on a Markov chain's transition diagram G for which the associated stationary measures have a product-form relationship, for all values of the transition rates. Therefore, we start by defining a formal Markov chain, where the transition rates are free variables rather than fixed values, and we define the corresponding stationary distribution. We then specify our definition of a product-form Markov chain.

Formal Markov chain

Our goal is to understand how the structure of a Markov chain's transition diagram impacts the relationship between its transition rates and stationary measures. This motivates the definition of a formal Markov chain. As we observe below, all our results apply directly to all the stationary measures of a formal Markov chain, irrespective of whether or not the instantiations of this Markov chain are positive recurrent.

▶ **Definition 3** (Formal Markov chain). *Let $G = (V, E)$ be a (possibly infinite) strongly-connected directed graph. Define the corresponding* formal Markov chain *to have transition* *rate from node i to node j equal to $q_{i,j} > 0$ for each $(i, j) \in E$ and 0 for each $(i, j) \in (V \times V) \setminus E$. Note that $q_{i,j}$ is a free variable, not instantiated to a specific rate.*

For each strongly-connected graph G, there is a single corresponding formal Markov chain, and vice versa. We will therefore refer to the two interchangeably. The quantities $q_{i,j}$ can be interpreted either as transition rates (CTMC) or as transition probabilities (DTMC; introducing the additional assumption that $\sum_{j \in V} q_{i,j} = 1$ for each $i \in V$).

For each formal Markov chain $G = (V, E)$, we can define the associated (formal) stationary distribution π to be the solution, as a function of the free variables $q_{i,j}$, to the balance equation and normalization requirement:

$$\pi_i \sum_{j|(i,j) \in E} q_{i,j} = \sum_{k|(k,i) \in E} \pi_k q_{k,i}, \quad i \in V, \tag{1}$$

$$\sum_{i \in V} \pi_i = 1. \tag{2}$$

Because $q_{i,j}$ are free variables, if G is an infinite graph, one cannot in general guarantee that the summation requirement (2) is satisfied. Thus for infinite graphs, we will instead consider a stationary measure and omit (2), but we will still refer to stationary distributions for simplicity.

Product-form relationship

We now come to the central concept of the paper. Definition 4 gives our definition of a product-form relationship between two nodes, while Definition 5 considers the entire graph.

▶ **Definition 4** (Graph-based product form). *Consider a formal Markov chain $G = (V, E)$ and let $i, j \in V$. Nodes i and j are in a* graph-based product-form relationship *with one another if, letting π denote the Markov chain's stationary distribution, we have*

$$\pi_i f_{i,j} = \pi_j f_{j,i}, \tag{3}$$

where $f_{i,j}$ and $f_{j,i}$ are polynomials (or more generally, rational functions) in the transition rates of the formal Markov chain. The complexity of a product-form relationship will be measured by the complexity of the associated polynomials (e.g., degree, number of monomials, arithmetic circuit complexity, arithmetic circuit depth, etc.). The term graph-based product form *will only be employed in association with a certain complexity level, as will be defined below.*

▶ **Definition 5** (Product-form distribution). *A formal Markov chain has a* product-form stationary distribution *at a given complexity level if all pairs of nodes in the graph are in a graph-based product-form relationship at that complexity level.*

For brevity, we will often say "product-form", leaving "graph-based" implicit. Note however that this notion of product-form is more specific than the general concept in the literature, as discussed in Section 1.2. We initially focus in Section 3 on two simple types of product-form relationships, which we define here. We define more complex types of product-form relationships in Section 4.3.

S-product-form: S stands for *sum*. Let $N_i := \{j \in V \mid (i, j) \in E\}$ denote the out-neighborhood of i. Focusing on $f_{i,j}$, we say that nodes i and j are in an S-product-form relationship if there exists some $S_{i,j} \subseteq N_i$, such that

$$f_{i,j} = \sum_{k \in S_{i,j}} q_{i,k}.$$

PS-product-form: P stands for *product*. Again focusing on $f_{i,j}$, we say that nodes i and j are in a PS-product-form relationship if $f_{i,j}$ is the product of sums of subsets of transition rates emerging from nodes in the graph: for some $F_{i,j} \subseteq V$, there exists $S_{a,i,j} \subseteq N_a$ for each $a \in F_{i,j}$, such that

$$f_{i,j} = \prod_{a \in F_{i,j}} \sum_{k \in S_{a,i,j}} q_{a,k}.$$

In this paper, we give necessary and sufficient conditions on the graph structure of a Markov chain for product-form relationships to exist. Moreover, we explicitly construct the rational functions which appear in these product form relationships: For S-product-form relationships, we give necessary and sufficient conditions in Theorem 2, and give the explicit rational functions in (6). For PS-product-form relationships, we give a sufficient condition in Lemma 8, and give the explicit rational functions in (18).

Arithmetic circuit complexity

We make a short parenthesis to explain how these product-form classes are related to standard complexity measures. Let us first recall the definition of *arithmetic circuit complexity classes* [31], a means of defining families of simple rational functions.

▶ **Definition 6.** *An arithmetic circuit is a directed acyclic graph in which each node with in-degree zero is an input variable and each node with positive in-degree is a basic arithmetic function (addition, multiplication, or reciprocal); there is a single node with out-degree zero called the output. Operations are performed in the direction indicated by edges, so that the output of a gate (a synonym of node in this context) is used as an input by its children. The* depth *of an arithmetic circuit is the maximum number of operations in a path from the inputs to the output.*

The product-form classes defined above correspond to classes of arithmetic circuits with a limited depth: S-product-form is a subclass of depth-1 circuits, PS-product-form is a subclass of depth-2 circuits. The width of these arithmetic circuits, or more specifically the in-degree of the gates, is also limited by the degrees of the graph's nodes. Thus simple product-form classes correspond to simple arithmetic circuits.

2.3 Cuts with a given source node set

The following lemma is borrowed from [14, Lemma 1.4]. It shows that cuts in a Markov chain's transition diagram can be exploited to derive (from the balance equations) a new set of equations, called *cut equations*, that can sometimes be used to derive the stationary distribution more easily.

▶ **Lemma 1.** *Consider a formal Markov chain $G = (V, E)$ and let π denote its stationary distribution. For each cut (A, B) of the graph G, we have*

$$\sum_{(i,j) \in E \cap (A \times B)} \pi_i q_{i,j} = \sum_{(j,i) \in E \cap (B \times A)} \pi_j q_{j,i}. \tag{4}$$

Proof. Equation (4) follows by summing the balance equations (1) over all $i \in A$ and making simplifications. More directly, (4) can be obtained by applying the strong law of large numbers for ergodic Markov chains. Indeed, the left-hand-side of (4) is the long-run rate at which the chain makes a transition from A to B, and the right-hand side is the long-run rate at which the chain makes a transition from B to A. ◀

The rest of the paper focuses on identifying necessary and sufficient conditions on the graph structure that guarantee the existence of "nice cuts" that yield a simple expression for the Markov chain's stationary distribution via (4). A simple and famous example is a birth-and-death process, as shown in Figure 1: for each $i \in \{0, 1, 2, \dots, \}$, cut i relates π_i and π_{i+1} via the cut equation $\pi_i q_{i,i+1} = \pi_{i+1} q_{i+1,i}$, so that nodes i and $i+1$ are on an S-product-form relationship. We obtain a closed-form expression for the stationary distribution up to a positive multiplicative constant:

$$\pi_i = \frac{q_{i-1,i}}{q_{i,i-1}} \frac{q_{i-2,i-1}}{q_{i-1,i-2}} \cdots \frac{q_{0,1}}{q_{1,0}} \pi_0, \quad \text{for each } i \in \{0, 1, 2, \dots\}.$$

Although the cut equations follow from the balance equations $\pi_i(q_{i,i-1}\mathbb{1}[i \geq 1] + q_{i,i+1}) = \pi_{i-1} q_{i-1,i}\mathbb{1}[i \geq 1] + \pi_{i+1} q_{i+1,i}$ for $i \in \{0, 1, 2, \dots\}$, they allow us to derive the stationary distribution more directly. As a slightly more intricate toy example, in Figure 2, the cut $(\{0, 4\}, \{1, 2, 3, 5, 6\})$ implies $\pi_4(q_{4,1} + q_{4,5}) = \pi_1 q_{1,0}$, hence nodes 1 and 4 are on an S-product-form relationship.

In general, a cut equation (4) as given in Lemma 1 is more convenient than the balance equations (1) if the set of nodes i such that π_i appears on either side of the equation is small. This set is called the *source* of the corresponding cut, as it consists of the nodes that are the sources of the edges that cross the cut.

▶ **Definition 7** (Source). *Consider a formal Markov chain $G = (V, E)$. The source of a cut (A, B) of the graph G is the pair (I, J) defined by*

$$I = \{i \in A : E \cap (\{i\} \times B) \neq \emptyset\}, \qquad J = \{j \in B : E \cap (\{j\} \times A) \neq \emptyset\}. \tag{5}$$

Equivalently, (A, B) is called an I, J-sourced cut.

Considering again the examples above, cut $i \in \{0, 1, 2, \dots\}$ in Figure 1 has source $(i, i+1)$, cut 1 in Figure 2 has source $(1, 4)$, and cut 2 in Figure 2 has source $(\{1, 4\}, 2)$. In Section 3,

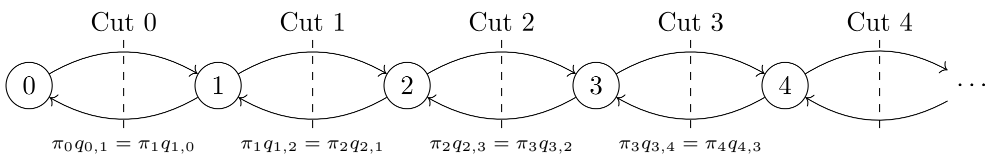


Figure 1 A birth-and-death process.

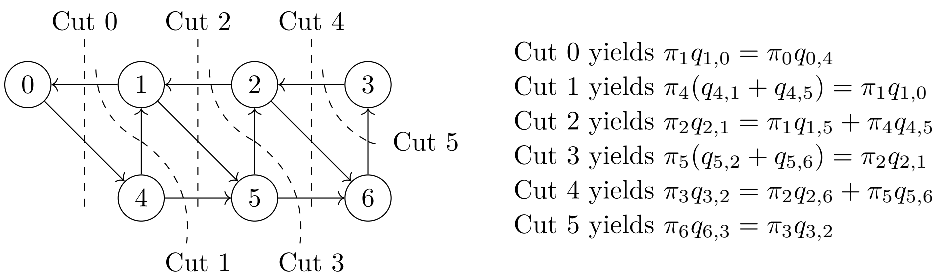


Figure 2 A simple formal Markov chain exhibiting a product form relationship between nodes 0 and 1 (via cut 0), 1 and 4 (via cut 1), 2 and 5 (via cut 3), and 3 and 6 (via cut 5). In addition, cut 2 allows us to express π_2 as a function of π_0 and π_1, and cut 4 allows us to express π_3 as a function of π_2 and π_5. All in all, these six cut equations allow us to express π_i as a function of π_0 for each $i \in \{1, 2, \dots, 6\}$.

we will focus on the special case where the source sets I and J are both singletons, $I = \{i\}$ and $J = \{j\}$. We refer to such a cut as a *single-sourced* cut. All cuts in Figure 1 are single-sourced cuts, and so are cuts 0, 1, 3, and 5 in Figure 2.

2.4 Joint-ancestor freeness

In Definition 8 below, we identify a simpler condition, called *joint-ancestor freeness*, that we will prove to be necessary and sufficient for the existence of a cut with a particular source pair in Section 3.4.

▶ **Definition 8** (Joint-ancestor freeness). *Consider a formal Markov chain $G = (V, E)$ and two disjoint nonempty sets $I, J \subsetneq V$. A node $k \in V$ is a* joint ancestor *of node sets I and J if $k \in A_I(G\backslash J) \cap A_J(G\backslash I)$, i.e., there is both a path from node k to some node in I that avoids set J and a path from node k to some node in J that avoids set I. Node sets I and J are said to be* joint-ancestor free *if $A_I(G\backslash J) \cap A_J(G\backslash I) = \emptyset$. In the special case where I and/or J are singletons, we drop the curly brackets in the notation, e.g., we write $A_i(G\backslash J)$ for $A_{\{i\}}(G\backslash J)$.*

To make this definition more concrete, lets us again consider the birth-and-death process of Figure 1. Focusing on $I = \{2\}$ and $J = \{3\}$, we have $A_2(G\backslash 3) = \{0, 1, 2\}$ and $A_3(G\backslash 2) = \{3, 4, 5, \ldots\}$, so that nodes 2 and 3 are joint-ancestor free. To see why $A_2(G\backslash 3) = \{0, 1, 2\}$, it suffices to observe that the subgraph $G\backslash 3$ consists of two strongly connected components: $\{0, 1, 2\}$ and $\{4, 5, 6, \ldots\}$. Anticipating Proposition 4 below, we observe that $(A_2(G\backslash 3), A_3(G\backslash 2)) = (\{0, 1, 2\}, \{3, 4, 5, \ldots\})$ is exactly Cut 2 in Figure 1. Similarly, $I = \{1\}$ and $J = \{2, 4\}$ are joint-ancestor free because $A_1(G\backslash\{2, 4\}) = \{0, 1\}$ and $A_{\{2,4\}}(G\backslash 1) = \{2, 3, 4, \ldots\}$. On the contrary, nodes 1 and 3 are not joint-ancestor free because $A_1(G\backslash 3) = \{0, 1, 2\}$ and $A_3(G\backslash 1) = \{2, 3, 4, \ldots\}$ have non-empty intersection $\{2\}$.

The MutuallyAvoidingAncestors procedure in Algorithm 2 returns the joint-ancestor sets $A_I(G\backslash J)$ and $A_J(G\backslash I)$ in time $O(|E|)$ in a finite graph G, by calling the Ancestors procedure from Algorithm 1. MutuallyAvoidingAncestors can be used to test if two node sets are joint-ancestor free.

Algorithm 2 Returns the mutually-avoiding ancestors of two node sets

```
procedure MutuallyAvoidingAncestors(finite directed graph G = (V, E), disjoint
  nonempty sets I, J ⊆ V) → ancestor sets A_I(G\J), A_J(G\I)
    A_I ← Ancestors(I, G\J)
    A_J ← Ancestors(J, G\I)
    return A_I, A_J
end procedure
```

3 S-product-form, cuts, and joint-ancestor freeness

In this section, we focus on the S-product-form relationship introduced in Section 2.2. Theorem 2, the main result of this section, is stated in Section 3.1 and illustrated on toy examples in Section 3.2. The proof of Theorem 2 relies on intermediary results shown in Sections 3.3–3.5. Higher-order product-form relationships, such as PS-product-form, will be considered in Section 4.

3.1 Main theorem

Theorem 2 below is our first main contribution: it gives simple necessary and sufficient conditions under which two nodes are in an S-product-form relationship. This result relies on the two graph-based notions introduced earlier, namely, cuts with a given source node (Section 2.3) and joint-ancestor freeness (Section 2.4). The rest of Section 3 will give further insights into this result.

▶ **Theorem 2.** *Consider a formal Markov chain $G = (V, E)$ and let $i, j \in V$. The following statements are equivalent:*

(i) *Nodes i and j are in an S-product-form relationship.*

(ii) *There is an i, j-sourced cut.*

(iii) *Nodes i and j are joint-ancestor free.*

If these statements are true, then the S-product-form between nodes i and j has factors

$$f_{i,j} = \sum_{\substack{k \in A_j(G\backslash i): \\ (i,k) \in E}} q_{i,k} \quad \text{and} \quad f_{j,i} = \sum_{\substack{k \in A_i(G\backslash j): \\ (j,k) \in E}} q_{j,k}. \tag{6}$$

Proof. The implication (ii) $\implies$ (i) is a classical result that will be recalled in Lemma 3 in Section 3.3. The equivalence (ii) $\iff$ (iii) will be shown in Proposition 4 in Section 3.4. The implication (i) $\implies$ (iii) will be shown in Theorem 7 in Section 3.5. Lastly, Equation (6) follows for instance by combining Lemma 3 and Proposition 4. ◀

The equivalence between conditions (i) and (ii) is reminiscent of classical sufficient conditions on the existence of a product-form relationship, except that the focus is now on the transition graph rather than on the transition rates. Now, the equivalence between conditions (ii) and (iii) can be intuitively understood as follows. If two nodes $i, j \in V$ are joint-ancestor free, meaning that $A_i(G\backslash j) \cap A_j(G\backslash i) = \emptyset$, then one can verify that $(A_i(G\backslash j), A_j(G\backslash j))$ forms a cut and that its source nodes are i and j. On the contrary, if i and j are not joint-ancestor free, there exists $k \in V\backslash\{i, j\}$ such that there are two paths $P(k \to i\backslash j)$ and $P(k \to j\backslash i)$. The existence of these two paths precludes any cut (A, B) from having source (i, j). Indeed, assuming for example that $i, k \in A$ and $j \in B$, the path $P(k \to j\backslash i)$ needs to go from part A (containing k) to part B (containing j), and it cannot do so via node i because this is an i-avoiding path. Therefore, the source of part A cannot be reduced to node i.

Thanks to Theorem 2, we can directly apply procedure MutuallyAvoidingAncestors from Algorithm 2 to verify if two nodes i and j are in an S-product-form relationship and, if yes, compute the corresponding factors, all with time complexity $O(|E|)$. This is far more efficient than directly testing each cut in the graph to see if its sources are i and j: there are $2^{|V|}$ such cuts, each of which would take $|E|$ time to check. Testing the S-product-form relationship of all pairs of nodes in the graph can be done in time $O(|V|^2|E|)$ (also see Section 4.1).

Note that Theorem 2 implies that if nodes i and j are in an S-product-form relationship, then the sets $S_{i,j}$ and $S_{j,i}$ as defined in Section 2.2 must be disjoint, as these sets are subsets of the disjoint ancestry sets $A_j(G\backslash i)$ and $A_i(G\backslash j)$, as specified in (6).

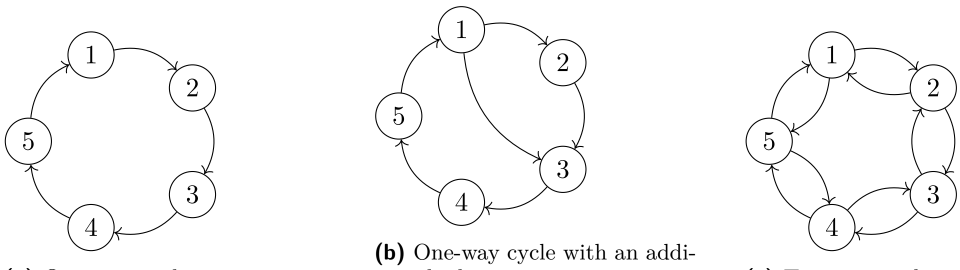
3.2 Illustrative examples

Before we prove the intermediary results that appear in the proof of Theorem 2, let us illustrate this connection between single-sourced cuts, joint-ancestor freeness, and product-form relationships.

Toy examples

We first revisit the birth-and-death process of Figure 1 already discussed in Sections 2.3 and 2.4, and then we consider the toy examples shown in Figure 3.

▶ **Example 1** (Birth-and-death process). Consider a formal Markov chain $G = (V, E)$ with $V = \{0, 1, 2, \ldots\}$ and $E = \bigcup_{i \in V} \{(i, i+1), (i+1, i)\}$, as in Figure 1. For each $i \in V$, nodes i and $i+1$ are in an S-product-form relationship through the $i, i+1$-sourced cut formed by $A_i(G \backslash i+1) = \{1, 2, \ldots, i\}$ and $A_{i+1}(G \backslash i) = \{i+1, i+2, \ldots\}$. However, for each $i, j \in V$ such that $i \le j-2$, nodes i and j are not in an S-product-form relationship because $A_i(G \backslash j) = \{0, 1, 2, \ldots, j-1\}$ and $A_j(G \backslash i) = \{i+1, i+2, \ldots, n\}$ intersect at $A_i(G \backslash j) \cap A_j(G \backslash i) = \{i+1, i+2, \ldots, j-1\}$. Observe that, unlike the next two examples, this process is (time-)reversible when it is positive-recurrent. Reversibility will be discussed again in Example 5.



(a) One-way cycle

(b) One-way cycle with an additional edge

(c) Two-way cycle

Figure 3 Illustrative examples of S-product-form relationships.

▶ **Example 2** (One-way cycle). Consider a formal Markov chain $G = (V, E)$ with $V = \{1, 2, \ldots, n\}$ for some $n \ge 3$ and $E = \{(i, i+1) | i \in V\} \cup E'$, where $E' \subseteq \{(i, i) | i \in V\}$, with the convention that nodes are numbered modulo n. For instance, Figure 3a. For each $i, j \in V$, say with $i < j$, the sets $A_i(G \backslash j) = \{j+1, j+2, \ldots, n, 1, 2, \ldots, i\}$ and $A_j(G \backslash i) = \{i+1, i+2, \ldots, j\}$ are disjoint and therefore form an i, j-sourced cut. Theorem 2 implies that nodes i and j are in an S-product-form relationship with $f_{i,j} = q_{i,i+1}$ and $f_{j,i} = q_{j,j+1}$. Equivalently, we can check visually that there is no node from which there is a directed path to node i without visiting node j and vice versa. This example shows in particular that an i, j-sourced cut may exist even if there is neither an edge (i, j) nor an edge (j, i). Theorem 12 in Appendix A shows that the one-way cycle is the *only* finite formal Markov chain in which all pairs of nodes are on an S-product-form relationship.

▶ **Example 3** (One-way cycle with an additional edge). Consider the same directed cycle, but with an additional edge from node 1 to some node $k \in \{3, 4, \ldots, n\}$. For instance, Figure 3b. For each $i \in \{2, 3, \ldots, k-1\}$ and $j \in \{k, k+2, \ldots, n\}$, nodes i and j are no longer joint-ancestor free because $1 \in A_i(G \backslash j) \cap A_j(G \backslash i)$. Indeed, $1 \to 2 \to 3 \to \cdots \to i$ is a $P(1 \to i \backslash j)$ path and $1 \to k \to k+1 \to \cdots \to j$ is a $P(1 \to j \backslash i)$ path. Therefore, nodes i and j are no longer in an S-product-form relationship.

▶ **Example 4** (Two-way cycle). Lastly, consider a directed graph $G = (V, E)$ with $V = \{1, 2, \ldots, n\}$ for some $n \ge 3$ and $E = \bigcup_{i \in V} \{(i, i+1), (i+1, i)\}$, again with the convention that nodes are numbered modulo n. For instance, Figure 3c. For each $i, j \in V$, the sets $A_i(G \backslash j) = V \backslash \{j\}$ and $A_j(G \backslash i) = V \backslash \{i\}$ have nonempty intersection $A_i(G \backslash j) \cap A_j(G \backslash i) = V \backslash \{i, j\}$. Hence, there are no S-product-form relationships in this graph.

Relations with other product-form or structural conditions

We now consider two larger examples that illustrate graph-based product form and its relationship to other structural properties of Markov chains. The reader is also invited to take a look at Section 5 for examples of Markov chains that model actual queueing systems and exhibit S-product-form relationships.

▶ **Example 5** (Trees and reversibility)**.** The following extension of birth-and-death processes again has an S-product-form stationary distribution and is reversible when it is positive-recurrent; the result is from [14, Lemma 1.5] and [10, Theorem 2.2]. Consider a formal Markov chain $G = (V, E)$ and let H denote its communication graph, that is, the undirected graph $H = (V, F)$ with $F = \{\{i, j\} \subseteq V : (i, j) \in E \text{ or } (j, i) \in E\}$. Assume that (i) H is a tree, and (ii) for each $i, j \in V$, we have $(i, j) \in E$ if and only if $(j, i) \in E$. Then, for each $i, j \in V$ such that $\{i, j\} \in H$, nodes i and j are on a particularly simple form of S-product-form relationship given by $\pi_i q_{i,j} = \pi_j q_{j,i}$. Indeed, Assumptions (i) and (ii) imply that removing edges (i, j) and (j, i) from G divides G into two strongly connected subgraphs. Letting A_i (resp. A_j) denote the set of nodes in the part containing node i (resp. j), we conclude not only that (A_i, A_j) is an i, j-sourced cut, but also that the only edges across this cut are (i, j) and (j, i). Lemma 1 then yields the product-form relationship $\pi_i q_{i,j} = \pi_j q_{j,i}$.

Conversely, one can verify that the only formal Markov chains in which the S-product-form relationship implies reversibility (when the chain is positive recurrent) for all transition rates are those satisfying Assumptions (i) and (ii). Other reversible and quasi-reversible queueing models do not have an S-product-form stationary distribution (nor any sort of graph-based product form). Focusing on Whittle networks [10, Chapter 1], one can verify that, except for a single-queue network (that forms a birth-and-death process), the product-form of the distribution holds only because the departure rates satisfy the so-called *balance property*. Similarly, an order-independent queue [12] with a single class forms a birth-and-death process, but an order-independent queue with at least two classes only has a product-form stationary distribution because the departure rates satisfy the so-called *order-independence condition*.

▶ **Example 6** (Quasi birth-and-death process)**.** Another possible extension of the birth-and-death process of Example 1 is a Markov chain whose structure is a particular case of a quasi-birth-and-death (QBD) process [23, 24, 25]. Consider a formal Markov chain $G = (V, E)$ and assume V to be finite for simplicity. Assume that V can be partitioned into subsets $V_0, V_1, V_2, \dots, V_m$, called superstates, such that transitions across superstates occur only between neighboring superstates, and only via designated states. More formally, assume that, for each $j \in \{0, 1, \dots, m-1\}$, there exist $i_{j,\text{up}} \in V_j$ and $i_{j+1,\text{down}} \in V_{j+1}$ such that

$$E \cap \left(V_j \times \bigcup_{k=j+1}^{m} V_k \right) = \{i_{j,\text{up}}\} \times V_{j+1}, \qquad E \cap \left(V_{j+1} \times \bigcup_{k=0}^{j} V_k \right) = \{i_{j+1,\text{down}}\} \times V_j.$$

For instance, the toy example of Figure 2 fits this structure with $V_0 = \{0, 4\}$, $V_1 = \{1, 5\}$, $V_2 = \{2, 6\}$, and $V_3 = \{3\}$, $i_{j,\text{up}} = j + 4$ for each $j \in \{0, 1, 2\}$, and $i_{j,\text{down}} = j$ for each $j \in \{1, 2, 3\}$. In general, for each $j \in \{0, 1, \dots, m-1\}$, states $i_{j,\text{up}}$ and $i_{j+1,\text{down}}$ are on an S-product-form relationship, as the previous equation implies that $(\bigcup_{k=0}^{j} V_j, \bigcup_{k=j+1}^{m} V_j)$ is an $i_{j,\text{up}}, i_{j+1,\text{down}}$-sourced cut, so that

$$\pi_{i_{j,\text{up}}} \sum_{k \in V_{j+1}} q_{i_{j,\text{up}},k} = \pi_{i_{j+1,\text{down}}} \sum_{k \in V_j} q_{i_{j+1,\text{down}},k}.$$

While in general deriving the stationary probabilities π_i for each $i \in V$ requires solving several systems of linear equations (one for each superstate), we will see in Sections 5.1 and 5.2 that in some cases, graph-based product form can also be applied to derive the complete stationary distribution, without resorting to linear algebra.

3.3 The existence of a single-sourced cut implies S-product-form

Lemma 3 below shows the implication (ii) $\implies$ (i) from Theorem 2. This result is recalled for completeness, but it follows directly by combining the definition of a sourced cut (Definition 7) with Lemma 1.

▶ **Lemma 3.** *Consider a formal Markov chain $G = (V, E)$ and two nodes $i, j \in V$. If there is an i, j-sourced cut in the graph G, then these nodes are in an S-product-form relationship* (3)*, with factors as given in* (6)*.*

3.4 The existence of a single-sourced cut is equivalent to joint-ancestor freeness

We now prove the equivalence (ii) $\iff$ (iii) from Theorem 2, that is, joint-ancestor freeness is necessary and sufficient for the existence of a cut with a particular source pair. We show that this equivalence holds both when the source pair is a pair (i, j) of nodes (Proposition 4) and more generally for any pair (I, J) of source sets (Proposition 5). Cuts where the source pair is a general pair of source sets (I, J) give rise to higher-level cuts, as we show and discuss in Section 4.

▶ **Proposition 4.** *Consider a formal Markov chain $G = (V, E)$ and let $i, j \in V$. Nodes i and j are joint-ancestor free if and only if there exists an i, j-sourced cut. In this case, the only i, j-sourced cut is $(A_i(G\backslash j), A_j(G\backslash i))$.*

Proposition 4 is a special case of Proposition 5, which is stated and proved later in this section. To illustrate the intuition behind Proposition 4, again consider the toy example of Figure 2. Nodes 1 and 4 are joint-ancestor free because $A_1(G\backslash 4) = \{1, 2, 3, 5, 6\}$ and $A_4(G\backslash 1) = \{0, 4\}$ are disjoint and therefore form a 1, 4-sourced cut. In contrast, nodes 1 and 2 are not joint-ancestor free because $A_1(G\backslash 2) = \{0, 1, 4\}$ and $A_2(G\backslash 1) = \{2, 3, 4, 5, 6\}$ intersect at node 4. Any cut (A, B) such that $1 \in A$ and $2 \in B$ has to be crossed by an edge from a path $P(4 \to 2\backslash 1)$ (if $4 \in A$) or from a path $P(4 \to 1\backslash 2)$ (if $4 \in B$), which makes it impossible to build a cut whose sources are restricted to nodes 1 and 2. This intuition is formalized in Statement ii of Lemma 6 below. Proposition 4 implies in particular that an i, j-sourced cut is unique when it exists, hence we can say *the* i, j-sourced cut.

For Proposition 5 below, the situation is slightly more complicated. When considering joint-ancestor free sets I, J containing more than one node, cuts may not be unique, and may not have the entire sets I, J as sources. Nonetheless, there is still a bidirectional relationship between mutually-avoiding ancestor sets and cut-source sets.

▶ **Proposition 5.** *Consider a formal Markov chain $G = (V, E)$ and two disjoint nonempty sets $I, J \subseteq V$. We have the following:*

(i) *If I and J are joint-ancestor free, then the cut $(A_I(G\backslash J), A_J(G\backslash I))$ is an $\underline{I}, \underline{J}$-sourced cut, for some non-empty sets $\underline{I} \subseteq I$ and $\underline{J} \subseteq J$.*

(ii) *If $(A_I(G\backslash J), A_J(G\backslash I))$ is a cut and has source (I, J), then it is the unique cut with sources (I, J).*

(iii) *If I and J are not joint-ancestor free, then there is no I, J-sourced cut.*

Again considering the example of Figure 2, let $I = \{1, 4\}$ and $J = \{2, 5\}$. The ancestor sets $A_I(G\backslash J) = \{0, 1, 4\}$ and $A_J(G\backslash I) = \{2, 3, 5, 6\}$ are disjoint, and we can verify that they form an $\underline{I}, \underline{J}$-sourced cut with $\underline{I} = \{1, 4\} = I$ and $\underline{J} = \{2\} \subsetneq J$. For a negative example, node sets $I' = \{1\}$ and $J = \{2, 5\}$ are not joint-ancestor free because $A_{I'}(G\backslash J) = \{0, 1, 4\}$ and $A_J(G\backslash I') = \{2, 3, 4, 5, 6\}$ intersect at node 4. Correspondingly, one can verify that there is no I', J-sourced cut in the graph.

Before proving Propositions 4 and 5, we prove the following intermediary lemma, which will also be instrumental for later results.

▶ **Lemma 6.** *Consider a formal Markov chain $G = (V, E)$ and two disjoint non-empty sets $I, J \subsetneq V$. We have the following:*

(i) $A_I(G\backslash J) \cup A_J(G\backslash I) = V$.

(ii) *If (A, B) is an I, J-sourced cut, then $A_I(G\backslash J) \subseteq A$ and $A_J(G\backslash I) \subseteq B$.*

Proof of Lemma 6. Let us first prove Lemma 6i. Let $k \in V$. Since G is strongly connected and I is nonempty, there is a directed path $v_1, v_2, \ldots, v_n$ in G, with $n \geq 1$, such that $v_1 = k$ and $v_n \in I$. Let $p = \min\{q \in \{1, 2, \ldots, n\} | k_q \in I \cup J\}$. Then $k \in A_I(G\backslash J)$ if $k_p \in I$ and $k \in A_J(G\backslash I)$ if $k_p \in J$. Hence, $k \in A_I(G\backslash J) \cup A_J(G\backslash I)$ for each $k \in V$, which implies that $V = A_I(G\backslash J) \cup A_J(G\backslash I)$.

We now prove Lemma 6ii. Assume that (A, B) is an I, J-sourced cut and let $k \in A_I(G\backslash J)$: there is a directed path $v_1, v_2, \ldots, v_n$ such that $v_1 = k$, $v_n \in I$, and $v_p \notin J$ for each $p \in \{1, 2, \ldots, n\}$. Our goal is to prove that $k \in A$. If $n = 1$, we have directly $k \in I \subseteq A$. Now consider the case where $n \geq 2$. Assume for the sake of contradiction that $k \notin A$, that is, $k \in B$, so that $v_1 = k \in B$ and $v_n \in A$. Then we can define $p = \max\{q \in \{1, 2, \ldots, n-1\} | v_q \in B\}$, and we have $(v_p, v_{p+1}) \in E \cap (B \times A)$. We also know by construction of the path that $k_p \notin J$. This contradicts our assumption that J is the second source of the cut (A, B). Hence, $k \in A$. ◀

Proof of Propositions 4 and 5. Proposition 4 is a special case of Proposition 5 because the only nonempty subset of a singleton is the singleton itself. Therefore, in the remainder, we focus on proving Proposition 5.

Let us first prove Proposition 5i. Assume that I and J are joint-ancestor free, that is, $A_I(G\backslash J) \cap A_J(G\backslash I) = \emptyset$. Combining this assumption with Lemma 6i shows that $(A_I(G\backslash J), A_J(G\backslash I))$ is a cut. Assume for the sake of contradiction that the source $(\underline{I}, \underline{J})$ of the cut $(A_I(G\backslash J), A_J(G\backslash I))$ does not satisfy $\underline{I} \subseteq I$ and $\underline{J} \subseteq J$. Specifically, suppose there exists $(k, \ell) \in E \cap (A_I(G\backslash J) \times A_J(G\backslash I))$ such that $k \notin I$. Since $\ell \in A_J(G\backslash I)$, there exists a directed path $v_1, v_2, \ldots, v_n$, with $n \geq 1$, such that $v_1 = \ell$, $v_n = j \in J$, and $v_p \notin I$ for each $p \in \{1, 2, \ldots, n\}$. Since we assumed $k \notin I$, it follows that $k, v_1, v_2, \ldots, v_n$ is a directed path from k to j in $G\backslash I$, hence $k \in A_J(G\backslash I)$, which is impossible because we assumed $k \in A_I(G\backslash J)$ and $A_I(G\backslash J) \cap A_J(G\backslash I) = \emptyset$.

Next, we prove Proposition 5ii. By Lemma 6ii, any cut (A, B) with source (I, J) is such that $A_I(G\backslash J) \subseteq A$ and $A_J(G\backslash I) \subseteq B$. But we assumed $(A_I(G\backslash J), A_J(G\backslash I))$ forms a cut, and no nodes can be outside of the cut. Thus, $A_I(G\backslash J) = A$ and $A_J(G\backslash I) = B$.

Lastly, we prove Proposition 5iii. Assume that I and J are not joint-ancestor free, i.e., $A_I(G\backslash J) \cap A_J(G\backslash I) \neq \emptyset$. Assume for the sake of contradiction that there is a cut (A, B) with source (I, J). Lemma 6ii implies that $A_I(G\backslash J) \subseteq A$ and $A_J(G\backslash I) \subseteq B$, which in turn implies that $A \cap B \supseteq A_I(G\backslash J) \cap A_J(G\backslash I) \neq \emptyset$. This contradicts our assumption that (A, B) is a cut. ◀

3.5 S-product-form implies joint-ancestor freeness

Our last step in the proof of Theorem 2 is to show the implication (i) $\implies$ (iii), that is, S-product-form relationship implies joint-ancestor freeness. We have shown so far that statements (ii) and (iii) are equivalent to each other, and that they imply (i). In other words, we proved that if an i, j-sourced cut exists, or equivalently if nodes i and j have no joint ancestor, then an S-product-form relationship between i and j exists. Specifically, a product-form relationship exists where $f_{i,j}$ depends only on transition rates along edges whose source is i, and $f_{j,i}$ depends only on transition rates along edges whose source is j.

Theorem 7 below proves this condition is necessary. The intuition behind the proof is as follows. If an i, j-sourced cut does not exist, then by Proposition 4 there exist nodes which are joint ancestors of i and j, i.e., $A_i(G \setminus j) \cap A_j(G \setminus i)$ is nonempty. In particular, there must exist a node k which is an ancestor of both nodes i and j via disjoint paths. If such a node k exists, we show that the ratio $\frac{\pi_i}{\pi_j}$ depends on edge weights $q_{k,k'}$ with source k. This violates the definition of S-product-form given in Section 2.2, so Theorem 7 shows that a joint ancestor implies no S-product-form, or equivalently that S-product-form implies no joint ancestor.

▶ **Theorem 7.** *Consider a formal Markov chain $G = (V, E)$ and let $i, j \in V$. Suppose nodes i and j are not joint-ancestor free, i.e., $A_i(G\setminus j) \cap A_j(G\setminus i) \neq \emptyset$.*

Then there exists a node $k \in A_i(G\setminus j) \cap A_j(G\setminus i)$ such that the stationary probability ratio $\frac{\pi_i}{\pi_j}$ depends on at least one of the edge weights emerging from k.

Proof. We will demonstrate the existence of two instantiations of the vector q, denoted by q^{a} and q^{b}, such that (i) q^{a} and q^{b} differ only at edges emerging from a particular source node $k \in A_i(G\setminus j) \cap A_j(G\setminus i)$, and (ii) the stationary distributions π^{a} and π^{b} associated with q^{a} and q^{b} satisfy

$$\frac{\pi_i^{\mathsf{a}}}{\pi_j^{\mathsf{a}}} \neq \frac{\pi_i^{\mathsf{b}}}{\pi_j^{\mathsf{b}}}. \tag{7}$$

The proof is divided into 6 steps. In Step 1, we identify the vertex $k \in A_i(G\setminus j) \cap A_j(G\setminus i)$ for which the result will eventually be proven. In Step 2, we derive a convenient expression for the ratios in (7), in terms of a subchain of the original Markov chain restricted to nodes i, j, and k. As this expression is only valid for positive-recurrent Markov chains, in the remainder of the proof we will focus on instantiations of the vector q that are positive recurrent (in the sense that the corresponding Markov chain is positive recurrent). In Step 3, we build a particular vector q, and we apply the Foster-Lyapunov theorem to show that q is positive recurrent. In Step 4, we construct q^{a} and q^{b} by altering q, and we show that these vectors are positive recurrent, again using the Foster-Lyapunov theorem. In Steps 5 and 6, we finally prove that q^{a} and q^{b} satisfy (7), using the expression derived in Step 2. Throughout the proof, we see q, q^{a}, and q^{b} as transition *probability* vectors, i.e., we focus on DTMCs.

Step 1: Specify the vertex k

We choose k to be a node in $A_i(G\setminus j) \cap A_j(G\setminus i)$ such that k is an ancestor of both nodes i and j via disjoint paths. To see why such a node must exist, consider an arbitrary node k' in $A_i(G\setminus j) \cap A_j(G\setminus i)$. From any such node, there exist paths from k' to i and k' to j. Consider the shortest such paths. Either these paths are disjoint, or there is a node k'' which is a joint ancestor of i and j via shorter paths than k'. Over all nodes in $A_i(G\setminus j) \cap A_j(G\setminus i)$, let k be the node that is the ancestor of i via the shortest path length, choosing arbitrarily in case of a tie. By the above argument, the shortest paths from k to i and from k to j must be disjoint.

Step 2: Define the subchains restricted to nodes i, j, and k

Next, for an arbitrary transition probability vector q giving rise to a positive-recurrent Markov chain, we examine the probability of transitioning between nodes i, j, and k. More specifically, consider the subchain, with node set $U = \{i, j, k\}$, obtained by looking at the subsequence of states visited by the original Markov chain $G = (V, E)$ inside the set U. This subchain satisfies the Markov property, and for each $u, v \in U$, we let $p_{u,v}$ denote its transition probability from state u to state v; these are functions of the original Markov chain's transition rates $q_{i,j}$ for $i, j \in V$. Critically, letting π_i, π_j, and π_k denote the stationary probabilities for states i, j, and k in the original Markov chain under edge probabilities q, as defined by (1), we can verify that

$$\frac{\pi_i}{\pi_i + \pi_j + \pi_k}, \quad \frac{\pi_j}{\pi_i + \pi_j + \pi_k}, \quad \text{and} \quad \frac{\pi_k}{\pi_i + \pi_j + \pi_k}$$

form the stationary distribution of the embedded DTMC corresponding to the subchain. Using this definition, we will quantify the relative state probabilities π_i and π_j in terms of the subchain transition probabilities $p_{u,v}$ for $u, v \in U$.

Using the balance equations for the embedded DTMC corresponding to the subchain, we obtain successively:

$$\pi_k = p_{i,k}\pi_i + p_{j,k}\pi_j + p_{k,k}\pi_k, \tag{8}$$

$$\pi_k = \frac{p_{i,k}\pi_i + p_{j,k}\pi_j}{1 - p_{k,k}}, \tag{9}$$

$$\pi_i = p_{i,i}\pi_i + p_{j,i}\pi_j + p_{k,i}\pi_k, \tag{10}$$

$$\pi_i = p_{i,i}\pi_i + p_{j,i}\pi_j + p_{k,i}\frac{p_{i,k}\pi_i + p_{j,k}\pi_j}{1 - p_{k,k}}, \tag{11}$$

$$\pi_i\left(1 - p_{i,i} - \frac{p_{i,k}p_{k,i}}{1 - p_{k,k}}\right) = \pi_j\left(p_{j,i} + \frac{p_{j,k}p_{k,i}}{1 - p_{k,k}}\right), \tag{12}$$

$$\pi_i\left(p_{i,j} + p_{i,k} - \frac{p_{i,k}p_{k,i}}{p_{k,i} + p_{k,j}}\right) = \pi_j\left(p_{j,i} + \frac{p_{j,k}p_{k,i}}{p_{k,i} + p_{k,j}}\right), \tag{13}$$

$$\pi_i\left(p_{i,j} + p_{i,k}\frac{p_{k,j}}{p_{k,i} + p_{k,j}}\right) = \pi_j\left(p_{j,i} + p_{j,k}\frac{p_{k,i}}{p_{k,i} + p_{k,j}}\right). \tag{14}$$

Equations (8) and (10) are the balance equations of the embedded DTMC of the subchain for states k and i. Equation (9) follows by solving (8) with respect to π_k, and once injected into (10) it yields (11). Equation (12) follows by rearranging (11), and becomes (13) after injecting $p_{i,i} + p_{i,j} + p_{i,k} = 1$. Equation (14) then follows by rearranging (13).

Step 3: Construct a positive-recurrent vector q

We now construct a transition probability vector q with support E that is positive-recurrent (in the sense that the Markov chain it gives rise to is positive recurrent). We can see q as a possible instantiation of the formal Markov chain's edge weights. This vector will be our starting point for constructing q^{a} and q^{b} in the next step and showing that q^{a} and q^{b} are also positive recurrent. The reason why we want q^{a} and q^{b} to be positive recurrent, is to guarantee that their associated subchain (see Step 2) is well-defined.

We start by arbitrarily selecting a root node $r \in V$. Consider the function $h : \ell \in V \mapsto h(\ell) \in \{0, 1, 2, \ldots\}$ such that $h(\ell)$ is the directed-shortest-path distance from ℓ to the root node r, for each $\ell \in V$. We will use h first to define the vector q, and then as a Lyapunov function to prove positive recurrence of q.

For a given node of the graph m, we now define the transition probabilities $q_{m,n}$ for transitions leaving node m. Note that, for each $m \in V \backslash \{r\}$, there must be a node $m' \in N_m$ for which $h(m') = h(m) - 1$, where N_m is the neighborhood of m. We may select m' to be the first step on a shortest path from m to the root node r. Let us label the nodes in $N_m \backslash \{m'\}$ as $m_1, m_2, \ldots$ in some arbitrary order. We assign the transition probabilities

$$q_{m,m_i} = \frac{1}{2}\frac{1}{2^i}\frac{1}{h(m_i)+1} \text{ for each } i \in \{1, 2, \ldots, |N_m|\}, \qquad q_{m,m'} = 1 - \sum_{i=1}^{|N_m|} q_{m,m_i}.$$

Note that $\sum_{i=1}^{|N_m|} q_{m,m_i} \le \frac{1}{2}$, as h is non-negative, and that $|N_m| \in \{1, 2, \ldots\} \cup \{+\infty\}$. This procedure defines $q_{\ell,\ell'}$ for each $(\ell, \ell') \in E$ with $\ell \neq r$. To define $q_{r,\ell'}$ for each $(r, \ell') \in E$, we select some arbitrary $m'_0 \in N_r$ and use the same procedure.

With the transition probabilities q defined in this way, we now prove that the Markov chain is positive recurrent, using h as a Lyapunov function for the Foster-Lyapunov theorem [32, Theorem 7.1.1]. We define the finite set $F = \{r\}$ consisting only of the root node, and we let $\delta = 1/2$. We will show that the following two conditions hold:

$$\sum_{\ell' \in N_\ell} q_{\ell,\ell'} h(\ell') \le h(\ell) - \delta, \quad \text{for each } \ell \in V \backslash F, \tag{15}$$

$$\sum_{\ell' \in N_\ell} q_{\ell,\ell'} h(\ell') < \infty, \quad \text{for each } \ell \in F. \tag{16}$$

By the Foster-Lyapunov theorem, these conditions imply that q is positive recurrent.

First, let us prove the negative-drift equation (15) for each node in $V \backslash \{r\}$. We call this node m to match the notation above:

$$\begin{aligned}
\sum_{\ell' \in N_m} q_{m,\ell'} h(\ell') &\overset{\text{(i)}}{=} q_{m,m'} h(m') + \sum_{i=1}^{|N_m|} q_{m,m_i} h(m_i), \\
&\overset{\text{(ii)}}{=} q_{m,m'}(h(m) - 1) + \sum_{i=1}^{|N_m|} \frac{1}{2}\frac{1}{2^i}\frac{1}{h(m_i)+1} h(m_i), \\
&\overset{\text{(iii)}}{\le} q_{m,m'}(h(m) - 1) + \sum_{i=1}^{|N_m|} \frac{1}{2}\frac{1}{2^i}, \\
&\overset{\text{(iv)}}{=} q_{m,m'}(h(m) - 1) + \frac{1}{2} \overset{\text{(v)}}{\le} h(m) - 1 + \frac{1}{2} = h(m) - \frac{1}{2},
\end{aligned}$$

where (i) follows by enumerating the out-neighbors of m using the same notation as before, (ii) from the definition of the transition rates, (iii) because h is positive, (iv) by upper-bounding the sum over $\{1, 2, \ldots, |N_m|\}$ with the same sum over $\{1, 2, \ldots, +\infty\}$, and (v) from the fact that $q_{m,m'} \le 1$. We thus confirm (15) for all nodes m other than the root node r.

For the root node r, a similar derivation confirms the finite-expectation equation (16) by showing that

$$\sum_{\ell' \in N_r} q_{r,\ell'} h(\ell') \le h(m'_0) + \frac{1}{2}.$$

Step 4: Construct positive-recurrent vectors q^{a} and q^{b} by altering q

We have shown that q is positive recurrent, in the sense that the Markov chain defined by the graph G and the transition probability vector q is positive recurrent. We are now ready to define q^{a} and q^{b}, two transition probability vectors that differ from q only along edges with source k. We will show later in this step that q^{a} and q^{b} are positive recurrent, and in Steps 5 and 6 we will show that their associated stationary distributions π^{a} and π^{b} satisfy (7).

Let $a = k, a_2, a_3, \ldots, a_{|a|}, i$ be the shortest path from k to i and $b = k, b_2, b_3, \ldots, b_{|b|}, j$ be the shortest path from k to j. Also let $V_a = \{k, a_2, a_3, \ldots, a_{|a|}\}$ (resp. $V_b = \{k, b_2, b_3, \ldots, b_{|b|}\}$) denote the set nodes that belong to path a (resp. b), except i (resp. j). Occasionally, we will use the notation $a_{|a|+1} = i$ and $b_{|b|+1} = j$. Recall from the definition of k above that a and b share no nodes except k.

We now define transition probability vectors q^{a} and q^{b} that take the same value at all edges except (k, a_2) and (k, b_2). Let $\epsilon \in (0, 1)$ be a sufficiently small constant, smaller than a value to be precised later.

First, for each edge $(\ell, \ell') \in E \backslash \{(k, a_2), (k, b_2)\}$, we define $q^{\mathsf{a}}_{\ell,\ell'} = q^{\mathsf{b}}_{\ell,\ell'}$ as follows:

- If $\ell \in V \backslash (V_a \cup V_b)$, i.e., either $\ell \in \{i, j\}$ or ℓ is in neither a nor b, we define
$$q^{\mathsf{a}}_{\ell,\ell'} = q^{\mathsf{b}}_{\ell,\ell'} \triangleq q_{\ell,\ell'}.$$
- Next, we turn to the case where $\ell \in V_a \backslash \{k\}$, i.e., ℓ is within a but is not k or i, so that we have $\ell = a_p$ for some $p \in \{2, 3, \ldots, |a|\}$. We let, for each $p \in \{2, 3, \ldots, |a|\}$,
$$q^{\mathsf{a}}_{a_p,\ell'} = q^{\mathsf{b}}_{a_p,\ell'} \triangleq \begin{cases} 1 - \epsilon, & \text{if } \ell' = a_{p+1}, \\ \epsilon \dfrac{q_{a_p,\ell'}}{\sum_{a'' \in N_{a_p} \backslash \{a_{p+1}\}} q_{a_p,a''}} & \text{if } \ell' \in N_{a_p} \backslash \{a_{p+1}\}. \end{cases}$$
We define in the same manner weights of edges whose source node belongs to $V_b \backslash \{k\}$.
- If $\ell = k$ and $\ell' \notin \{a_2, b_2\}$, we define
$$q^{\mathsf{a}}_{k,\ell'} = q^{\mathsf{b}}_{k,\ell'} \triangleq \frac{\epsilon}{2} \frac{q_{k,\ell'}}{\sum_{\ell'' \in N_k \backslash \{a_2, b_2\}} q_{k,\ell''}}.$$

Finally, we define the two transition probabilities that differ between q^{a} and q^{b}, namely, those corresponding to the edges (k, a_2) and (k, b_2):
$$q^{\mathsf{a}}_{k,a_2} \triangleq 1 - \epsilon, \qquad q^{\mathsf{a}}_{k,b_2} \triangleq \frac{\epsilon}{2}, \qquad q^{\mathsf{b}}_{k,b_2} \triangleq 1 - \epsilon, \qquad q^{\mathsf{b}}_{k,a_2} \triangleq \frac{\epsilon}{2}.$$
One can verify that $\sum_{\ell' \in V} q_{\ell,\ell'} = 1$ for each $\ell \in V$. We assume that $\epsilon \in (0, 1)$ is small enough so that the transition probabilities in q^{a} and q^{b} defined as multiple values of ϵ are smaller than their q counterparts; in other words, we assume that
$$\begin{aligned} &q^{\mathsf{a}}_{a_p,\ell'} = q^{\mathsf{b}}_{a_p,\ell'} \le q_{a_p,\ell'} \text{ for each } p \in \{2, 3, \ldots, |a|\} \text{ and } \ell' \in N_{a_p} \setminus \{a_{p+1}\}, \\ &q^{\mathsf{a}}_{b_p,\ell'} = q^{\mathsf{b}}_{b_p,\ell'} \le q_{b_p,\ell'} \text{ for each } p \in \{2, 3, \ldots, |b|\} \text{ and } \ell' \in N_{b_p} \setminus \{b_{p+1}\}, \\ &q^{\mathsf{a}}_{k,\ell'} = q^{\mathsf{b}}_{k,\ell'} \le q_{k,\ell'} \text{ for each } \ell' \in N_k \setminus \{a_2, b_2\}, \\ &q^{\mathsf{a}}_{k,b_2} \le q_{k,b_2}, \quad \text{and} \quad q^{\mathsf{b}}_{k,a_2} \le q_{k,a_2}. \end{aligned}$$

Let us now prove that q^{a} and q^{b} are positive recurrent. Similarly to Step 3, we prove variants of (15) and (16) in order to apply the Forster-Lyapunov theorem. Compared to Step 3, we use the same Lyapunov function h, namely, $h(\ell)$ is the directed-shortest-path distance from ℓ to the same root node r, for each $\ell \in V$, and we replace the finite exclusion set F with $F' = \{r\} \cup V_a \cup V_b$. We now prove that q^{a} and q^{b} satisfy these new variants of the negative-drift equation (15) and the finite-expectation equation (16):

- For the negative-drift equation (15), note that $q^a_{\ell,\ell'} = q^b_{\ell,\ell'} = q_{\ell,\ell'}$ for all $\ell \in V \backslash F'$ and $\ell' \in N_\ell$, and we already proved that q satisfies (15) over $V \backslash F \supseteq V \backslash F'$.
- Let us now prove that q^{a} satisfies the finite-expectation equation (16); the proof for q^{b} is symmetrical. Let $\ell \in F'$. Since the paths a and b intersect only in k and each of them visits each node at most once (by definition of a path), at most one transition outside ℓ has a larger probability in q^{a} than in q, namely, the successor of ℓ in a or b, if any; in the special case where $\ell = k$, ℓ has two successors in a and b (a_2 and b_2), but we chose ϵ such that $q^{\mathsf{a}}_{k,b_2} \le q_{k,b_2}$. This observation implies that the finiteness of $\sum_{\ell' \in V} q^{\mathsf{a}}_{\ell,\ell'} h(\ell')$ follows from that of $\sum_{\ell' \in V} q_{\ell,\ell'} h(\ell')$, which follows from (15) and (16). For instance, if $\ell = a_p$ for some $p \in \{2, 3, \ldots, |a| - 1\}$, we have $q^{\mathsf{a}}_{a_p,\ell'} < q_{a_p,\ell'}$ for each $\ell' \in N_{a_p} \backslash \{a_{p+1}\}$. Thus, recalling that h is positive and $q^{\mathsf{a}}_{a_p,a_{p+1}} \le 1$, we can bound the expected value of h after one transition:

$$\sum_{\ell' \in N_{a_p}} q^{\mathsf{a}}_{a_p,\ell'} h(\ell') \le h(a_{p+1}) + \sum_{\ell' \in N_{a_p}} q_{a_p,\ell'} h(\ell').$$

  The right-hand side is finite because q satisfies (15) and (16). A similar argument confirms finite expectation for all nodes in F' under each of q^{a} and q^{b}.

This completes the proof of positive recurrence via the Foster-Lyapunov theorem.

Let π^{a} and π^{b} denote the stationary distributions associated with q^{a} and q^{b}, respectively. Because q^{a} and q^{b} are positive recurrent, we can use them to construct subchains restricted to nodes i, j, and k, as we did in Step 2, and we denote their transition probabilities by p^{a} and p^{b}, respectively. It follows from (14) in Step 2 that

$$\frac{\pi^{\mathsf{a}}_i}{\pi^{\mathsf{a}}_j} = \frac{p^{\mathsf{a}}_{j,i} + p^{\mathsf{a}}_{j,k} \frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}}}{p^{\mathsf{a}}_{i,j} + p^{\mathsf{a}}_{i,k} \left(1 - \frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}}\right)}, \qquad \frac{\pi^{\mathsf{b}}_i}{\pi^{\mathsf{b}}_j} = \frac{p^{\mathsf{b}}_{j,i} + p^{\mathsf{b}}_{j,k} \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}}}{p^{\mathsf{b}}_{i,j} + p^{\mathsf{b}}_{i,k} \left(1 - \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}}\right)}.$$

Thus, π^{a} and π^{b} satisfy (7) if and only if p^{a} and p^{b} are such that

$$\frac{p^{\mathsf{a}}_{j,i} + p^{\mathsf{a}}_{j,k} \frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}}}{p^{\mathsf{a}}_{i,j} + p^{\mathsf{a}}_{i,k} \left(1 - \frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}}\right)} \ne \frac{p^{\mathsf{b}}_{j,i} + p^{\mathsf{b}}_{j,k} \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}}}{p^{\mathsf{b}}_{i,j} + p^{\mathsf{b}}_{i,k} \left(1 - \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}}\right)}.$$

In turn, to show this inequality, it is sufficient to prove that

$$p^{\mathsf{a}}_{i,j} = p^{\mathsf{b}}_{i,j}, \quad p^{\mathsf{a}}_{i,k} = p^{\mathsf{b}}_{i,k}, \quad p^{\mathsf{a}}_{j,i} = p^{\mathsf{b}}_{j,i}, \quad p^{\mathsf{a}}_{j,k} = p^{\mathsf{b}}_{j,k}, \quad \frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}} \ne \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}}. \tag{17}$$

The equalities in (17) are shown in Step 5, and the inequality is shown in Step 6.

Step 5: Verify that q^{a} and q^{b} imply the equalities in (17)

Our goal at this step is to prove the equalities in (17). These equalities follow from the fact that $q^{\mathsf{a}}_{\ell,\ell'} = q^{\mathsf{b}}_{\ell,\ell'}$ for each $(\ell,\ell') \in E$ with $\ell \ne k$. Because the argument is similar for all four equalities, we focus on the first. By definition, $p^{\mathsf{a}}_{i,j}$ and $p^{\mathsf{b}}_{i,j}$ are the sums of the probabilities of the trajectories in G that start in state i, end in state j, and never visit states i, j, or k in-between; unlike our definition of a path, a trajectory may visit the same node multiple times and may therefore have an arbitrary (but finite) length. We let $\mathcal{F}_{i,j}$ denote this set of trajectories, which is entirely determined by G. The probability of a trajectory in $\mathcal{F}_{i,j}$ is the product of the transition probabilities along the edges that compose the trajectory. By definition, none of the trajectories in $\mathcal{F}_{i,j}$ contains any transitions leaving state k. It follows that $q^{\mathsf{a}}_{\ell,\ell'} = q^{\mathsf{b}}_{\ell,\ell'}$ for any edge (ℓ,ℓ') that appears in the trajectories in $\mathcal{F}_{i,j}$, and then that $p^{\mathsf{a}}_{ij} = p^{\mathsf{b}}_{ij}$. A similar argument proves the other equalities in (17).

Step 6: Verify that q^a and q^b imply the inequality in (17)

Our goal at this step is to prove the inequality in (17). We will prove this inequality by lower-bounding the left-hand side and upper-bounding the right-hand side. First focusing on q^{a}, we have

$$p^{\mathsf{a}}_{k,i} \overset{\text{(i)}}{\geq} q^{\mathsf{a}}_{k,a_2} q^{\mathsf{a}}_{a_2,a_3} \cdots q^{\mathsf{a}}_{a_{|a|},i} \overset{\text{(ii)}}{=} (1-\epsilon)^{|a|},$$

where (i) follows by observing that path a is a trajectory that starts in state k, ends in state i, and never visits states i, j, or k in-between (i.e., $a \in \mathcal{F}_{k,i}$ with the notation of Step 6), and (ii) follows by definition of q^{a}. By applying a similar reasoning for path b under the transition probability vector q^{b}, we conclude that $p^{\mathsf{b}}_{k,j} \geq (1-\epsilon)^{|b|}$.

Let us now assume $\epsilon \leq \frac{1}{3\max(|a|,|b|)}$. One can verify that, for each $x \geq 1$, we have $(1-\frac{1}{3x})^x \in [\frac{2}{3}, e^{-\frac{1}{3}})$. It follows that $p^{\mathsf{a}}_{k,i} \geq \frac{2}{3}$ and $p^{\mathsf{b}}_{k,j} \geq \frac{2}{3}$. We conclude that

$$\frac{p^{\mathsf{a}}_{k,i}}{p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j}} \overset{\text{(i)}}{\geq} p^{\mathsf{a}}_{k,i} \overset{\text{(ii)}}{\geq} \frac{2}{3}, \qquad \frac{p^{\mathsf{b}}_{k,i}}{p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j}} \overset{\text{(iii)}}{\leq} \frac{p^{\mathsf{b}}_{k,i}}{\frac{2}{3}} \overset{\text{(iv)}}{\leq} \frac{1}{2},$$

where (i) follows from the fact that $p^{\mathsf{a}}_{k,i} + p^{\mathsf{a}}_{k,j} \leq 1$, (ii) from the fact that $p^{\mathsf{a}}_{k,i} \geq \frac{2}{3}$, (iii) from the fact that $p^{\mathsf{b}}_{k,i} + p^{\mathsf{b}}_{k,j} \geq p^{\mathsf{b}}_{k,j} \geq \frac{2}{3}$, and (iv) from the fact that $p^{\mathsf{b}}_{k,i} \leq 1 - p^{\mathsf{b}}_{k,j} \leq \frac{1}{3}$. These bounds imply the inequality in (17), completing the proof. ◀

4 Cut graph and higher-level cuts

In this section, we explore product-form relationships that go beyond the S-product-form considered in Section 3. In Section 4.1, we define the cut graph, which is an undirected graph whose edges represent S-product-form relationships between nodes. Based on this definition, in Section 4.2, we explore PS-product-form relationships, corresponding to combinations of S-product-form relationships produced by paths in the cut graph. In Section 4.3, we define SPS-product-form relationships and beyond, and we introduce and explore the corresponding higher-level cuts.

4.1 Cut graph

Let us first introduce the cut graph of a formal Markov chain.

▶ **Definition 9** (Cut graph). *Consider a formal Markov chain $G = (V, E)$. The* cut graph *of G is the undirected graph $C_1(G) = (V, R)$ where R is the family of doubletons $\{i, j\} \subseteq V$ such that states i and j are in an S-product-form relationship. In other words, $C_1(G)$ is the graph of the S-product-form binary relation.*

Recall that, by Theorem 2, two nodes i and j are in an S-product-form relationship if and only if there exists an i,j-sourced cut, that is, i and j are joint-ancestor free. Using this observation, Algorithm 3 returns the cut graph of a formal Markov chain with a finite number of states. It uses the MutuallyAvoidingAncestors procedure from Algorithm 2.

Because each call to this procedure takes time $O(|E|)$, the CutGraph procedure runs in time $O(|V|^2|E|)$. If the cut graph is connected, then the stationary distribution can be entirely computed by applying S-product-form relationships. Lastly, observe that the S-product-form relationship is not transitive, i.e., if the pairs i, j and j, k of nodes are both in an S-product-form relationship, this does not imply that nodes i, k are. Instead, we show in Section 4.2 that nodes i, k are then in a PS-product-form relationship.

Algorithm 3 Returns the cut graph $C_1(G)$ of a finite directed graph $G = (V, E)$

procedure CutGraph(finite directed graph $G = (V, E)$) $\rightarrow$ the cut graph $C_1(G)$
    $E' \leftarrow \emptyset$
    for each pair of distinct nodes $i, j \in V$ **do**
        $A_i, A_j \leftarrow$ MutuallyAvoidingAncestors(G, i, j)
        if $A_i \cap A_j = \emptyset$ **then**
            add edge $\{i, j\}$ to E'
        end if
    end for
    return (V, E')
end procedure

To help build more intuition on graph-based product form, in Appendix A, we relate the structure of a formal Markov chain G to the existence of a clique in its cut graph $C_1(G)$. This condition can be seen as an extension of Proposition 4, as an edge is a clique of size 2. Appendix A shows in particular that the one-way cycle of Example 2 is the only finite formal Markov chain whose cut graph is the complete graph.

4.2 PS-product-form relationships

The next lemma shows that, if two nodes are connected in the cut graph $C_1(G)$, they are in a PS-product-form relationship. In this way, each connected component of the cut graph forms a set of nodes that are pairwise in PS-product-form relationships. An example with a fully-connected cut graph appears in Section 5.1.

▶ **Lemma 8.** *Consider a formal Markov chain $G = (V, E)$ and two nodes $i, j \in V$. Assume that i and j belong to the same connected component of $C_1(G)$. Let d denote the distance between i and j in $C_1(G)$ and $i = k_1, k_2, \dots, k_d, k_{d+1} = j$ a path of length d in $C_1(G)$. Then*

$$\pi_i \prod_{p=1}^{d} f_{k_p,k_{p+1}} = \pi_j \prod_{p=1}^{d} f_{k_{p+1},k_p}. \tag{18}$$

where the f's are given by Equation (6) in Theorem 2. Hence, nodes i and j are in a PS-product-form relationship.

Lemma 8 follows from the fact that each edge in the cut graph represents an S-product-form relationship, as proven in Lemma 3, which chains together along the path to give a PS-product-form relationship. We can be more specific than simply saying that i and j are in a PS-product-form relationship. The arithmetic circuit (see Definition 6) associated with the left-hand side of (18) has depth 2 and size (total number of gates)

$$1 + d + \sum_{p=1}^{d} \left| E \cap \left(\{k_p\} \times A_{k_{p+1}}(G \backslash k_p) \right) \right|.$$

In particular, if the out-degree of each node on the path is upper-bounded by D, the size of the arithmetic circuit is upper-bounded by $1 + d + dD$.

4.3 Higher-level product-form relationships and cuts

We give a general definition of higher-level product-form relationships and cuts, with a special focus on the second level.

Higher-level product-form relationships

So far, we have studied S-type and PS-type product-form relationships, as defined in Section 2.2. We refer to these as "first-level" product-form relationships. We now turn to studying higher-level product-form relationships, which we define as follows:

SPS-product-form: We add another layer of alternation to PS-product-form. Each sum is over neighboring vertices, while the products are over arbitrary vertices. We also allow the terms in the products to be the *inverses* of sums, as well as direct sums. Focusing on $f_{i,j}$, we say that nodes i and j are in an SPS-product-form relationship if there exist $S_{i,j} \subseteq N_i$, $F_{k,i,j} \subseteq (V \times \{-1, 1\})$ for each $k \in S_{i,j}$, and $S_{a,k,i,j} \subseteq N_a$ for each $(a, p) \in F_{k,i,j}$, such that

$$f_{i,j} = \sum_{k \in S_{i,j}} \prod_{(a,p) \in F_{k,i,j}} \Big(\sum_{k' \in S_{a,k,i,j}} q_{a,k'} \Big)^p.$$

Higher-order: We can similarly define PSPS-product-form, SPSPS-product-form and more generally $(\text{PS})^n$ and $\text{S}(\text{PS})^n$ product form for any $n \in \{1, 2, 3, \ldots\}$.

Higher-level cuts

Corresponding to these higher-level product-form relationships, we also study higher-level cuts. Up to this point, we have focused on cuts with a single source vertex on each of side of the cut: i, j-sourced cuts, where i and j are each single vertices. We refer to such cuts as "first-level" cuts. These cuts correspond to pairs of states which are joint-ancestor free, and result in S-type product-form relationships between these states. In addition to these first-level cuts, we are also interested in *second-level* cuts. We now define such second-level cuts with reference to the cut graph $C_1(G)$ introduced in Section 4.1. In Lemma 9 we will prove that second-level cuts give rise to SPS-product-form relationships. Table 1 summarizes this section by showing what levels of cuts give rise to the graph-based product-form relationships enumerated in Section 2.2.

Nodes i and j are neighbors in $C_1(G)$	S-product-form	Theorem 2
Nodes i and j are connected by a path in $C_1(G)$	PS-product-form	Lemma 8
Nodes i and j are connected by a hyperpath in $C_2(G)$ containing at most one hyperedge made of more than two nodes	SPS-product-form	Lemma 9
Nodes i and j are connected by a hyperpath in $C_2(G)$	PSPS-product-form	

Table 1 Sufficient conditions under which two distinct states i and j of a formal Markov chain $G = (V, E)$ are in a product-form relationship. A hyperpath is defined as a sequence of distinct nodes such that each pair of consecutive nodes in the path is connected by a hyperedge (sets of 2 or more nodes).

We define two kinds of second-level cuts, closely related but subtly different.

▶ **Definition 10** (Second-level cuts). *Consider a formal Markov chain $G = (V, E)$.*
A broad second-level cut is a cut with source (I, J) such that the nodes in I are connected to one another via the cut graph $C_1(G)$, and the same is true of J. In other words, an I, J-sourced cut is a broad second-level cut if there exists a pair (K_1, K_2) of connected components of the cut graph $C_1(G)$ such that $I \subseteq K_1$ and $J \subseteq K_2$.
A narrow second-level cut is a cut arising from a joint-ancestor free relationship between two connected components of the cut graph $C_1(G)$. Specifically, the narrow second-level cut arising from a pair (K_1, K_2) of connected components of $C_1(G)$ is the cut $(A_{K_1}(G\backslash K_2), A_{K_2}(G\backslash K_1))$.

Note that a narrow second-level cut is also a broad second-level cut: its sources must be subsets of K_1 and K_2, by Proposition 5. The reverse is not as clear: the broad second-level cut with source (I, J), if it exists, is given by $(A_I(G\backslash J), A_J(G\backslash I))$, and *a priori* we may not have $A_I(G\backslash J) = A_{K_1}(G\backslash K_2)$ and $A_J(G\backslash I) = A_{K_2}(G\backslash K_1)$. Nonetheless, we conjecture that whenever a broad second-level cut exists, a corresponding narrow second-level cut also exists; see Conjecture 10 later in this section for details.

If a broad second-level cut exists in the graph, generated by $I \subseteq K_1$ and $J \subseteq K_2$, we will show in Lemma 9 that this cut gives rise to an SPS-product-form relationship between any pair of vertices $i \in K_1, j \in K_2$. It follows that, if all connected components of the cut graph are connected via broad second-level cuts, then G exhibits a PSPS-product-form; an example appears in Section 5.2. Our primary motivation for introducing narrow second-level cuts is that they can be algorithmically discovered more easily than broad second-level cuts; see the following sub-subsection for details.

One can similarly define broad third-level cuts, fourth-level cuts, and so on, which give rise to $\mathrm{S(PS)}^n$ and $\mathrm{(PS)}^n$ product-form relationships for larger n. An example of formal Markov chain with cuts of arbitrary levels will appear in Section 5.3. A broad third-level cut is a cut such that all nodes within each of the source sets I and J are connected by a combination of first-level cuts and broad second-level cuts, and so forth. One can also define narrow third-level cuts with reference to narrow second-level cuts, and so forth. Intuitively, each additional sum (S) appears by applying a cut equation, and each additional product (P) appears by combining combining several product-form relationships.

The second-level equivalent of the first-level cut graph $C_1(G)$ is the narrow second-level cut *hypergraph* $C_2(G)$. A hypergraph can contain *hyperedges*, which are sets containing 2 or more nodes, expanding the standard notion of 2-node edges. We define the narrow second-level cut graph $C_2(G)$ as follows. Starting with the first-level cut graph $C_1(G)$, for each pair of connected components K_1, K_2 which form a narrow second-level cut, we add a hyperedge containing the sources of the cut $(A_{K_1}(G\backslash K_2), A_{K_2}(G\backslash K_1))$. Assuming Conjecture 10, which claims that broad and narrow second-level cuts are equivalent, $C_2(G)$ contains all necessary information to identify all first-level and second-level cuts, and we can characterize the corresponding S, PS, SPS, and PSPS product-form relationships. We can similarly define narrow third-level and higher-level cut graphs. If Conjecture 10 fails, then we can define a distinct broad second-level cut graph, and higher-level broad cut graphs.

SPS-product-form

We show that if there exists a broad second-level cut with source (I, J), then not only are every pair of vertices in I and J in an SPS-product-form relationship, but in fact every pair of vertices in K_1 and K_2 are in an SPS-product-form relationship, where K_1 and K_2 are the connected components of the cut graph $C_1(G)$ that include I and J, respectively.

▶ **Lemma 9.** *Consider a formal Markov chain $G = (V, E)$ and two disjoint non-empty sets $I, J \subsetneq V$. If there exists an I, J-sourced broad second level cut, with $I \subseteq K_1$ and $J \subseteq K_2$, and K_1, K_2 connected components of $C_1(G)$, then every pair of vertices $i \in K_1$ and $j \in K_2$ are in an SPS-product-form relationship.*

Proof. First, recall from Lemma 8 that because K_1 is a connected component of $C_1(G)$, every pair of vertices $i, i' \in K_1$ is in a PS relationship. Letting $i = k_1, k_2, \dots k_d, k_{d+1} = i'$ be a path connecting i and i' in K_1, a PS-product-form relationship is given by:

$$f_{i,i'} = \prod_{p=1}^{d} f_{k_p,k_{p+1}}, \quad f_{i',i} = \prod_{p=1}^{d} f_{k_{p+1},k_p}, \tag{19}$$

$$\pi_i f_{i,i'} = \pi_{i'} f_{i',i}. \tag{20}$$

A similar PS-product-form relationship holds for any pair $j, j' \in K_2$.

Next, let us apply Lemma 1 to the cut with source sets (I, J). By Lemma 1, we have

$$\sum_{(i,j)\in E\cap(I\times J)} \pi_i q_{i,j} = \sum_{(j,i)\in E\cap(J\times I)} \pi_j q_{j,i}. \tag{21}$$

Note that all edges that cross the cut belong to either $I \times J$ or $J \times I$.

Let i_* and j_* be an arbitrary pair of vertices, $i_* \in K_1$ and $j_* \in K_2$. We will now explicitate the SPS-product-form relationship between i_* and j_*. Applying the PS-product-form relationships within K_1 and K_2, given by (19), we can rewrite (21) in terms of π_{i_*} and π_{j_*}:

$$\sum_{(i,j)\in E\cap(I\times J)} \pi_{i_*} \frac{f_{i_*,i}}{f_{i,i_*}} q_{i,j} = \sum_{(j,i)\in E\cap(J\times I)} \pi_{j_*} \frac{f_{j_*,j}}{f_{j,j_*}} q_{j,i},$$

$$\pi_{i_*} \sum_{(i,j)\in E\cap(I\times J)} \frac{f_{i_*,i}}{f_{i,i_*}} q_{i,j} = \pi_{j_*} \sum_{(j,i)\in E\cap(J\times I)} \frac{f_{j_*,j}}{f_{j,j_*}} q_{j,i}. \tag{22}$$

This gives the SPS-product-form relationship between i_* and j_* as desired. Note that we have made use of the flexibility of the SPS-product-form definition, which allows us to invert the sums within the SPS formula, or equivalently to invert the products within $f_{i,j}$ as defined in (19). Because i_* and j_* were an arbitrary pair of vertices in K_1, K_2, this completes the proof. ◀

Algorithmic discovery

We now discuss how to algorithmically and efficiently find all second-level cuts which exist in a given graph G, akin to Algorithm 3, which did the same for first-level cuts.

Specifically, we find all narrow second-level cuts, via the following straightforward but efficient algorithm. We iterate over all pairs K_1, K_2 of connected components in the cut graph $C_1(G)$. For each pair of components, we use Algorithm 2 to check whether the components are joint-ancestor free, and hence form a narrow second-level cut.

In contrast, attempting to discover broad second-level cuts directly via a similar procedure is not nearly as straightforward, as one may in principle be required to search over all pairs of subsets $I \subseteq K_1, J \subseteq K_2$, which is inefficient. However, in all cases that we have examined, broad second-level cuts are only present between subsets of components that form narrow second-level cuts. This motivates the following conjecture.

Conjecture: Equivalence of broad and narrow second-level cuts

We conjecture that all broad second-level cuts have a corresponding narrow second-level cut, in the sense specified in Conjecture 10 below. As a result, we conjecture that the algorithmic procedure described above discovers all components K_1 and K_2 that contain broad second-level cuts: If there exist $I \subseteq K_1$ and $J \subseteq K_2$ which form a second-level cut, then K_1 and K_2 are joint-ancestor free, and the above procedure will discover a narrow second-level cut between K_1 and K_2, even if that cut's source is not necessarily (I, J) or (K_1, K_2).

▶ **Conjecture 10.** *Consider a formal Markov chain $G = (V, E)$. For each pair of connected components K_1 and K_2 of $C_1(G)$, if there exist two nonempty sets $I \subseteq K_1$ and $J \subseteq K_2$ such that I and J are joint-ancestor free in G, then we conjecture that K_1 and K_2 are joint-ancestor free in G. In other words, if there is a broad second-level cut with source (I, J), then we conjecture that there is a narrow second-level cut arising from K_1 and K_2.*

If Conjecture 10 holds, then we would be able to remove the distinction between broad and narrow second level cuts.

The reason this claim is nontrivial is that in principle, K_1 and K_2 may have a joint ancestor even if I and J are joint-ancestor free. However, we were not able to build such an example. Thus, to prove Conjecture 10, one must leverage the fact that K_1 and K_2 are connected components of the cut graph $C_1(G)$.

We explored the following avenue towards proving this conjecture. Given joint-ancestor free subsets $I \subseteq K_1$ and $J \subseteq K_2$, we further conjecture that there always exists an additional node by which either I or J can be expanded, while preserving the joint-ancestor free property:

▶ **Conjecture 11.** *Given two joint-ancestor free subsets $I \subseteq K_1$ and $J \subseteq K_2$, where either $I \neq K_1, J \neq K_2$, or both, there exists either*
- *$i \in K_1 \backslash I$ such that $I \cup \{i\}$ and J are joint-ancestor free, or*
- *$j \in K_2 \backslash J$ such that I and $J \cup \{j\}$ are joint-ancestor free.*

If Conjecture 11 holds, then by applying it inductively we show K_1 and K_2 must be joint-ancestor free in G whenever I and J are, which would complete the proof of Conjecture 10.

In Appendix B, we prove in Theorem 15 that in the special case where J contains a single vertex ($|J| = 1$), there exists an $i \in K_1 \backslash I$ such that $I \cup \{i\}$ and J are joint-ancestor free. However, we were not able to resolve the general conjecture, either Conjecture 11 or Conjecture 10, and leave both as open problems.

5 Examples

We now give several examples of formal Markov chains arising from queueing systems which exhibit graph-based product form.

5.1 Multiserver jobs

First, we give a practical example of a Markov chain with PS-product-form arising from its graph structure. This setting is taken from [15] and explored in more detail in a technical report [22].

Consider the following multiserver-job (MSJ) queueing system. There are two types of jobs: class-1 jobs, which each require 3 servers to enter service, and class-2 jobs, which require 10 servers to enter service. There are 30 servers in total. Servers are assigned to jobs in first-come-first-served order, with head-of-line blocking. We will specifically consider a *saturated* queueing system, meaning that fresh jobs are always available, rather than having an external arrival process. Saturated queueing systems are used to characterize the stability region [33, 34] and mean response time [35] of the corresponding open system with external arrivals.

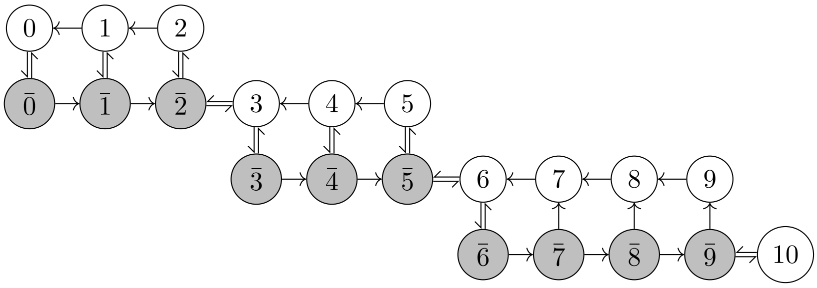
In the saturated system, fresh jobs are always available, and a new job enters into the system whenever it is possible that this fresh job might receive service, based on the number of available servers. Specifically, if at least 3 servers are available, a fresh job will enter the system. If that job is a class-2 job, and if there are not 10 servers available, the class-2 job will wait in the queue. There will always be at most one job in the queue, and only a class-2 job can be in the queue. Said differently, when a job completes service, we uncover the classes of as many fresh jobs as needed to verify that no more fresh jobs can be added to service. The service time of each job is exponentially distributed, with a rate based on the class of the job: μ_1 for class-1 jobs, and μ_2 for class-2 jobs.

There are several Markov chains corresponding to this system that we could examine at this point. For instance, we could consider the standard CTMC, or the embedded DTMC with steps at epochs where jobs complete. These two Markov chains both exhibit product-form stationary distributions, but they do not exhibit graph-based product form: the specific transition rates, not just the graph structure, produce the product-form behavior.

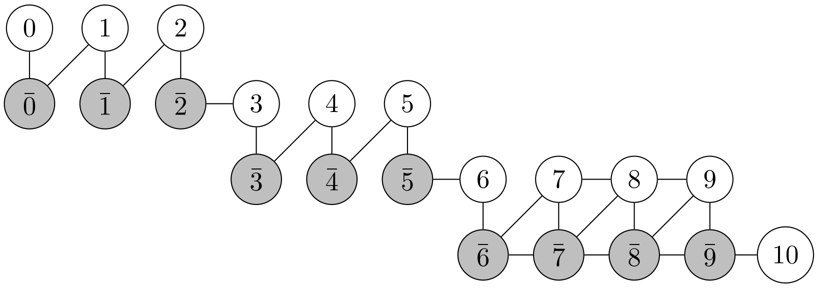
However, we instead examine a nonstandard DTMC, with epochs whenever a job either *enters the system* or completes. We call this DTMC the *arrival-and-completion DTMC*, in contrast to the more standard *completion-only DTMC* defined in [15]. In particular, whenever a job leaves upon service completion, the DTMC transitions through a the sequence of states, starting right after the job has completed and no fresh job has yet entered the system, then after a the first fresh job has entered, then the second, and so on, until there is no possibility that any further fresh jobs could immediately enter service. This DTMC *does* exhibit graph-based product form, as we will show, and this product-form behavior transfers to the other two Markov chains mentioned above.

The process of switching back from a DTMC associated with a more fine-grained embedding sequence (e.g., arrivals and completions) to a DTMC associated with a more coarse-grained embedding sequence (e.g., only completions) always preserves product-form relationships between the coarse-grained states, as we can see it as a subchain of the fine-grained DTMC restricted to a subset of the states. See Step 2 in the proof of Theorem 7 for a similar idea. However, switching embeddings may turn graph-based product-form relationships into probability-specific (non-graph-based) product-form relationships, as multiple transitions in the fine-grained DTMC may become a single transition in the coarse-grained DTMC, with transition probability written as a product of transitions in the fine-grained DTMC. This distinction is present in our setting.

More specifically, to reveal the graph-based product form, we examine the embedded DTMC of this system, examining moments at which jobs enter the system or complete. There are two kinds of states in the system: *completion states*, from which the next transition is a service completion, and *arrival states*, from which the next transition is a job arrival. In any state where at least 3 servers are available, and no job is in the queue, a fresh job enters the system. That job is a class-1 job with probability p_1, and a class-2 job otherwise. Otherwise, jobs complete. Class-1 jobs have service rate μ_1, and class-2 jobs have service rate μ_2, resulting in some probability of a completion of each class of job. After a completion, the job in the queue (if any) moves into service if enough servers are available, transitioning



(a) Transition diagram graph G.



(b) Cut graph $C_1(G)$.

Figure 4 Multiserver jobs example.

to a new state.

We denote states by the number of class-1 jobs in the system, from 0 to 10, and by whether the state is a completion state or an arrival state. This naming convention is sufficient to differentiate all states in the system (see more details in [15, Section 4]). An overbar denotes an arrival state, while the number without an overbar denotes a completion state. For instance, state 4 consists of 4 class-1 jobs and 1 class-2 job in service, for a total of 22 servers occupied, with another class-2 job in the queue. State $\bar{4}$ is the same state but without the class-2 job in the queue. Note that the arrival states such as $\bar{4}$ are present in this arrivals-and-completions embedded DTMC, and would not be present in a more standard completions-only DTMC. Such a completions-only DTMC can be obtained from the arrivals-and-completions DTMC by combining a completion transition with the immediately following arrival transitions into a single transition in the completions-only DTMC.

Figure 4a shows the graph G underlying the Markov chain for this saturated MSJ queue, with white backgrounds for completion states and grey backgrounds for arrival states. For instance, starting in state 4, a class-1 job can complete, transitioning to state 3, with 3 class-1 jobs and 2 class-2 jobs in service, or a class-2 job can complete, transitioning to state $\bar{4}$. From state $\bar{4}$, a class-1 job can enter the system, transitioning to state $\bar{5}$, with 5 class-1 jobs and 1 class-2 job in service, or a class-2 job can enter, transitioning to state 4.

Figure 4b shows the (first-level) cut graph $C_1(G)$ corresponding to this graph. For instance, there is a first-level cut between nodes 4 and $\bar{4}$. This cut partitions the graph nodes into two subsets: $A_{\bar{4}}(G\backslash 4) = \{0, \bar{0}, \ldots, 3, \bar{3}, \bar{4}\}$ and $A_4(G\backslash \bar{4}) = \{4, 5, \bar{5}, \ldots, \bar{9}, 10\}$. The only edges crossing this cut are the outgoing edges from 4 and $\bar{4}$.

Table 2 lists some of the cuts and the corresponding product-form relationships between pairs of nodes, where the transition rate from state i to state j is denoted by $q_{i,j}$. Note that each first-level cut gives rise to an S-product-form relationship, defined in Section 2.2, as shown in Lemma 3. Because the cut graph is fully connected, as shown in Figure 4b, every pair of nodes has a PS-product-form relationship, by composing S-product-form relationships

Nodes	Product-form Relation
0 and $\bar{0}$	$\pi_{\bar{0}}(q_{\bar{0},0} + q_{\bar{0},\bar{1}}) = \pi_0 q_{0,\bar{0}}$
$\bar{0}$ and 1	$\pi_1 q_{1,0} = \pi_{\bar{0}} q_{\bar{0},\bar{1}}$
1 and $\bar{1}$	$\pi_{\bar{1}}(q_{\bar{1},1} + q_{\bar{1},\bar{2}}) = \pi_1(q_{1,0} + q_{1,\bar{1}})$
$\bar{1}$ and 2	$\pi_2 q_{2,1} = \pi_{\bar{1}} q_{\bar{1},\bar{2}}$
2 and $\bar{2}$	$\pi_{\bar{2}} q_{\bar{2},3} = \pi_2(q_{2,1} + q_{2,\bar{2}})$
$\bar{2}$ and 3	$\pi_3 q_{3,\bar{2}} = \pi_{\bar{2}} q_{\bar{2},3}$
3 and $\bar{3}$	$\pi_{\bar{3}}(q_{\bar{3},3} + q_{\bar{3},\bar{4}}) = \pi_3 q_{3,\bar{3}}$
$\bar{3}$ and 4	$\pi_4 q_{4,3} = \pi_{\bar{3}} q_{\bar{3},\bar{4}}$

Table 2 First-level cuts involving states i or $\bar{i}$, for $i \in \{0, 1, 2, 3\}$, in the multiserver job example.

along a path connecting those nodes in the cut graph, as described in Lemma 8. As a result, the entire formal Markov chain has a PS-product-form stationary distribution, as follows:

$$\pi_i = \pi_0 \prod_{j=0}^{i-1} \frac{q_{j,j-1}\mathbb{1}[j \notin \{0,3,6,10\}] + q_{j,\bar{j}}}{q_{\bar{j},j} + q_{\bar{j},\overline{j+1}}\mathbb{1}[j \notin \{2,5,9\}]} \frac{q_{\bar{j},\overline{j+1}}}{q_{j+1,j}},$$

$$\pi_{\bar{i}} = \pi_0 \prod_{j=0}^{i} \frac{q_{j,j-1}\mathbb{1}[j \notin \{0,3,6,10\}] + q_{j,\bar{j}}}{q_{\bar{j},j} + q_{\bar{j},\overline{j+1}}\mathbb{1}[j \notin \{2,5,9\}]} \prod_{j=0}^{i-1} \frac{q_{\bar{j},\overline{j+1}}}{q_{j+1,j}}.$$

If the three structural parameters of the system are changed (3 servers for class-1 jobs, 10 servers for class-2 jobs, 30 servers total), PS-product-form continues to hold.

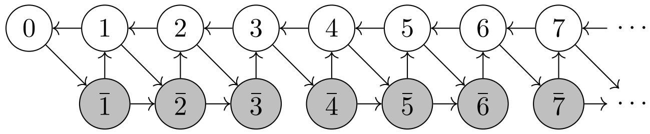
In prior work [15], this saturated queueing system was shown to have a PS-product-form stationary distribution, and that stationary distribution was given with respect to the specific transition probabilities $q_{i,j}$ for that system. This work further illuminates the system, demonstrating that the underlying graph of the Markov chain gives rise to the product-form stationary distribution. A similar product form would exist for a system with arbitrary transition rates and the same graph.

5.2 Queue with batch arrivals: version 1

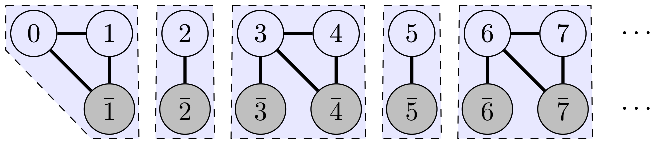
Next, we give an example of a queueing Markov chain whose associated cut graph is not connected via first-level cuts, but is connected via second-level cuts as discussed in Section 4.3, giving rise to PSPS-product-form from its graph structure. Note that for this particular Markov chain, Conjecture 10 holds, so we do not need to distinguish between broad second-level cuts and narrow second-level cuts.

Consider a single-server queueing system with structured batch arrivals. Jobs arrive in batches of size sampled from a geometric distribution, but the batch size is truncated to not bring the total number of jobs in the system above the next multiple of 3. For instance, if there were 4 jobs present prior to an arrival, the number of jobs in queue would be increased to $\min(4+N, 6)$ where $\mathbb{P}(N = n) = (1-p)p^{n-1}$ for each $n \in \{1, 2, 3, \ldots\}$, with the truncation ensuring that the total number of jobs after the batch arrival does not exceed 6. Jobs are indistinguishable, with exponentially-distributed service times. Batch arrivals occur according to a Poisson process and have i.i.d. sizes.

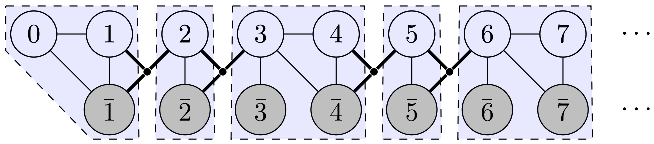
We consider the embedded DTMC of the system, examining moments when jobs enter the system or complete. Figure 5a shows the graph G underlying the Markov chain for this batch-arrivals system. It is infinite in one direction, in contrast to the finite graph we considered in Section 5.1. In the same spirit as Section 5.1, there are two kinds of states:



(a) Transition diagram graph G.



(b) First-level cut graph $C_1(G)$. Edges (solid lines) connect nodes that are in an S-product-form relationship. Dashed contours outline connected components of $C_1(G)$.



(c) Second-level cut graph $C_2(G)$. Solid lines intersecting at a dot represent second-level cuts, showing the source nodes of the cut.

Figure 5 Queue with batch arrivals, first version.

during a batch, when a job has just arrived, and in the bulk of time, when jobs may complete or a new batch may arrive. We denote states by the number of jobs in the system, and by whether the state is immediately-post-arrival, shown in grey, or spanning a nonzero amount of time, shown in white. An overbar denotes an immediately-post-arrival state. When a batch ends, the system transitions from the immediately-post-arrival state to the corresponding general-time state, such as from state $\bar{2}$ to state 2. For instance, state 4 has four jobs in the system, and jobs may complete or a new batch may begin; state $\bar{4}$ also has four jobs, but a batch of arrivals is ongoing. If a new batch begins while the Markov chain is in state 4, the Markov chain jumps state $\bar{5}$, meaning that the batch contains *at least* one job. Once it is in state $\bar{5}$, the Markov chain either moves to state 5 (if the batch is of size 1) or to state $\bar{6}$ (if the batch size is at least 2). In the latter case, the Markov chain necessarily jumps to state 6 because batches are truncated.

Figure 5b shows the first-level cut graph $C_1(G)$ corresponding to this graph. For instance, there is a first-level cut between nodes 3 and $\bar{4}$. This cut partitions the graph into two subsets: $A_{\bar{4}}(G\backslash 3) = \{\bar{4}\}$, and $A_3(G\backslash \bar{4}) = V\backslash\{\bar{4}\}$. This cut gives rise to an S-product-form relationship between nodes 3 and $\bar{4}$. More first-level cuts and corresponding S-product-form relationships are listed in Table 3.

The connected components of $C_1(G)$, illustrated with dashed outlines in Figure 5b, each contain two to four vertices. Between these components, there exist second-level cuts, as shown in Figure 5c. For instance, between components $K_1 = \{2, \bar{2}\}$ and $K_2 = \{3, \bar{3}, 4, \bar{4}\}$, there exists a cut with source (I, J), where $I = K_1 = \{2, \bar{2}\}$ and $J = \{3\}, J \subseteq K_2$. Correspondingly, K_1 and K_2 are joint-ancestor free.

This cut partitions the graph into two subsets: $A_I(G\backslash J) = \{0, 1, \bar{1}, 2, \bar{2}\}$, and $A_J(G\backslash I) = \{3, \bar{3}, 4, \bar{4}, 5, \bar{5}, \ldots\}$. Due to this second-level cut, the graph induces an SPS-product-form relationship between each pair of vertices in K_1 and K_2, by Lemma 9. Each connected component has a second-level cut with each of its neighbors, as shown in Figure 5c and in Table 3, so the Markov chain has a PSPS-product-form. Every pair of states has a

Level	Nodes	Equation
1	0 and 1	$\pi_0 q_{0,\bar{1}} = \pi_1 q_{1,0}$
	0 and $\bar{1}$	$\pi_0 q_{0,\bar{1}} = \pi_{\bar{1}}(q_{\bar{1},1} + q_{\bar{1},\bar{2}})$
	1 and $\bar{1}$	$\pi_1 q_{1,0} = \pi_{\bar{1}}(q_{\bar{1},1} + q_{\bar{1},\bar{2}})$
	2 and $\bar{2}$	$\pi_2 q_{2,1} = \pi_{\bar{2}}(q_{\bar{2},2} + q_{\bar{2},\bar{3}})$
	3 and $\bar{3}$	$\pi_3 q_{3,2} = \pi_{\bar{3}}(q_{\bar{3},3} + q_{\bar{3},\bar{2}})$
	3 and 4	$\pi_3 q_{3,\bar{4}} = \pi_4 q_{4,3}$
	4 and $\bar{4}$	$\pi_4 q_{4,3} = \pi_{\bar{4}}(q_{\bar{4},4} + q_{\bar{4},\bar{3}})$
	5 and $\bar{5}$	$\pi_5 q_{5,4} = \pi_{\bar{5}}(q_{\bar{5},5} + q_{\bar{5},\bar{4}})$
	6 and $\bar{6}$	$\pi_6 q_{6,5} = \pi_{\bar{6}}(q_{\bar{6},6} + q_{\bar{6},\bar{5}})$
	6 and 7	$\pi_6 q_{6,\bar{7}} = \pi_7 q_{7,6}$
	7 and $\bar{7}$	$\pi_7 q_{7,6} = \pi_{\bar{7}}(q_{\bar{7},7} + q_{\bar{7},\bar{6}})$
2	$\{1, \bar{1}\}$ and 2	$\pi_1 q_{1,\bar{2}} + \pi_{\bar{1}} q_{\bar{1},\bar{2}} = \pi_2 q_{2,1}$
	$\{2, \bar{2}\}$ and 3	$\pi_2 q_{2,\bar{3}} + \pi_{\bar{2}} q_{\bar{2},\bar{3}} = \pi_3 q_{3,2}$
	$\{4, \bar{4}\}$ and 5	$\pi_4 q_{4,\bar{5}} + \pi_{\bar{4}} q_{\bar{4},\bar{5}} = \pi_5 q_{5,4}$
	$\{5, \bar{5}\}$ and 6	$\pi_5 q_{5,\bar{6}} + \pi_{\bar{5}} q_{\bar{5},\bar{6}} = \pi_6 q_{6,5}$

Table 3 Cuts associated with the single-server queue batch example of Figure 5.

PSPS-product-form relationship. For example, to find a PSPS-relationship between states 1 and 3, we can combine the first-level cuts between states 1 and $\bar{1}$ and between states 2 and $\bar{2}$, with the second level cuts between $\{1, \bar{1}\}$ and 2 and between $\{2, \bar{2}\}$ and 3, as given in Table 3. By doing so, we obtain the following PSPS-relationship between states 1 and 3:

$$\pi_1 \left(q_{1,\bar{2}} + \frac{q_{1,0} q_{\bar{1},\bar{2}}}{q_{\bar{1},1} + q_{\bar{1},\bar{2}}} \right) \left(q_{2,\bar{3}} + \frac{q_{2,1} q_{\bar{2},\bar{3}}}{q_{\bar{2},2} + q_{\bar{2},\bar{3}}} \right) = \pi_3 q_{2,1} q_{3,2}.$$

The system still has PSPS-product-form if one varies the parameters defining the system and its Markov chain, for instance by changing the multiplier 3 that truncates the batches to some other multiplier, or truncating the batches at an arbitrary sequence of cutoff values.

As in Section 5.1, the addition of the immediately-post-arrival states $\bar{\imath}$ is instrumental in obtaining a graph-based product form. To see why, consider the embedded DTMC in which the immediately-post-arrival states $\bar{\imath}$ are deleted, and transitions involving these states are replaced with transitions between general-time states whose transition probabilities take the form of a product. For instance, this embedded DTMC has a transition from state 4 to state 5 with rate $q_{4,\bar{5}} q_{\bar{5},5}$. This embedded DTMC still has a product-form stationary distribution. However, this product-form follows not only from the graph structure, but also from the fact that the arrival transition rates are products. Intuitively, by adding the immediately-post-arrival states, we encode explicitly in the transition diagram the fact that the arrival transitions can be written are products. A similar intuition will hold for the example of Section 5.3.

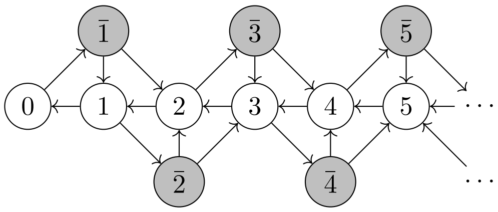
5.3 Queue with batch arrivals: version 2

Finally, we give an example of a queueing system with a positive number of first-level cuts, second-level, third-level, and so forth, but for which the entire graph is not connected via any finite level of cuts.

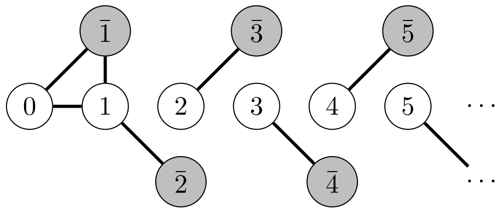
Consider a single-server queueing system with structured batch arrivals, of size either 1 or 2. Jobs arrive in batches of size 1 with probability p_1 and 2 with probability p_2, with

$p_1 + p_2 = 1$. Jobs are statistically indistinguishable, with exponential service times. Batch arrivals occur according to a Poisson process.

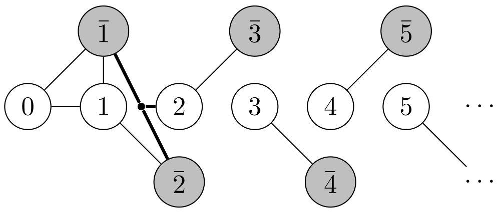
We examine the embedded DTMC of the system, examining moments when jobs enter the system or complete. We use the same state representation as in Section 5.2. Figure 6a shows the graph G underlying the Markov chain for this batch-arrivals system. Figure 6b shows the first-level cut graph $C_1(G)$ corresponding to this graph. Table 4 lists some such cuts and the corresponding S-product-form relationships.



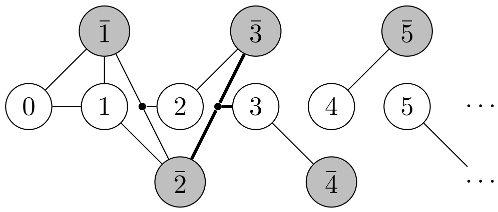
(a) Transition diagram G.



(b) First-level cut graph $C_1(G)$.



(c) Second-level cut graph $C_2(G)$.



(d) Third-level cut graph $C_3(G)$.

Figure 6 Queue with batch arrivals, second version.

In contrast to the batch setting in Section 5.2, there are very few second-level cuts in this graph. In fact, the only second level cuts are between subsets of the two leftmost components of the cut graph, namely $K_1 = \{0, 1, \bar{1}, \bar{2}\}$ and $K_2 = \{2, \bar{3}\}$, as shown in Figure 6c. A narrow second-level cut exists between these components, namely $A_{K_1}(G \backslash K_2) = K_1$ and $A_{K_2}(G \backslash K_1) = G \backslash K_1$. This cut has source $(\{\bar{1}, \bar{2}\}, \{2\})$, as shown in Figure 6c.

This cut implies an SPS product-form relationship between every pair of vertices $i \in K_1$ and $j \in K_2$. The key equation for defining that SPS relationship is the second-level equation in Table 4.

Note that other subsets of K_1 and K_2 also form broad second-level cuts, such as $\{\bar{1}, 1\} \subset K_1, \{2\} \subset K_2$. This is the behavior predicted by Conjecture 10, which states that broad second level cuts will only exist between subsets of connected components that also have

Level	Nodes	Equation
1	0 and 1	$\pi_0 q_{0,\bar{1}} = \pi_1 q_{1,0}$
	0 and $\bar{1}$	$\pi_0 q_{0,\bar{1}} = \pi_{\bar{1}}(q_{\bar{1},1} + q_{\bar{1},2})$
	1 and $\bar{1}$	$\pi_1 q_{1,0} = \pi_{\bar{1}}(q_{\bar{1},1} + q_{\bar{1},2})$
	2 and $\bar{3}$	$\pi_2 q_{2,\bar{3}} = \pi_{\bar{3}}(q_{\bar{3},3} + q_{\bar{3},4})$
	3 and $\bar{4}$	$\pi_3 q_{3,\bar{4}} = \pi_{\bar{4}}(q_{\bar{4},4} + q_{\bar{4},5})$
	4 and $\bar{5}$	$\pi_4 q_{4,\bar{5}} = \pi_{\bar{5}}(q_{\bar{5},5} + q_{\bar{5},6})$
2	$\{\bar{1}, \bar{2}\}$ and 2	$\pi_{\bar{1}} q_{\bar{1},2} + \pi_{\bar{2}}(q_{\bar{2},2} + q_{\bar{2},3}) = \pi_2 q_{2,1}$
3	$\{\bar{2}, \bar{3}\}$ and 3	$\pi_{\bar{2}} q_{\bar{2},3} + \pi_{\bar{3}}(q_{\bar{3},3} + q_{\bar{3},4}) = \pi_3 q_{3,2}$

Table 4 Cuts associated with the single-queue batch example of Figure 6.

narrow second-level cuts, but that the sources may differ.

We can recursively define third-level cuts based on the connected components of the second-level cut graph. However, as the second-level cut graph only adds a single hyperedge, the only difference between the first-level components and the second-level components is that K_1 and K_2 are combined into a single component. As result, there is only a single third-level cut, shown in Figure 6d. We can continue on to higher and higher levels of the cut graph, adding a single cut each time.

Corresponding to the fact that the nth cut graph is not fully connected for any finite level n, the Markov chain does not exhibit a $(\mathrm{PS})^n$ or $\mathrm{S}(\mathrm{PS})^n$ product-form stationary distribution for any finite level n. However, any two specific nodes $i, j \in V$ are connected in some (potentially large) level of the cut graph, and are in an $\mathrm{S}(\mathrm{PS})^n$ product-form relationship for some correspondingly large n.

Acknowledgements

Thank you to Jean-Michel Fourneau for pointing out [21] in an informal discussion on queueing systems with product-form stationary distributions. The authors are also grateful to the two anonymous reviewers for their valuable feedback on an earlier version of the paper; in particular, the alternative proof of Lemma 1 based on the strong law of large numbers for ergodic Markov chains and the suggestion of Examples 5 and 6.

A Cliques in the cut graph

As announced in Example 2 and Section 4.1, Theorem 12 below gives a necessary and sufficient condition for the existence of a clique in the cut graph of a formal Markov chain. This condition can be seen as an extension of Proposition 4, as an edge is a clique of size 2. Figure 7 shows a toy example that will be discussed after the theorem.

▶ **Theorem 12.** *Consider a formal Markov chain $G = (V, E)$ and a set $K \subseteq V$ of $n \geq 2$ nodes. Also let $V_i = A_i(G \backslash (K \backslash \{i\}))$ for each $i \in K$, and consider the directed graph $Q = (K, L)$ where $L = \{(i, j) \in K \times K : i \neq j$ and $E \cap (V_i \times V_j) \neq \emptyset\}$. Then:*

(i) $V = \bigcup_{i \in K} V_i$.

(ii) *The following statements are equivalent:*

- **a.** *K is a clique in $C_1(G)$.*
- **b.** *$(V_i)_{i \in K}$ is a partition of V and Q is a directed cycle.*

(iii) *If the equivalent statements of* (ii) *are satisfied then, for each $i, j \in K$, the i, j-sourced cut is (S, T) with $S = \bigcup_{k \in A_i(Q \backslash j)} V_k$ and $T = \bigcup_{k \in A_j(Q \backslash i)} V_k$.*

Theorem 12 is illustrated in Figure 7 with a 9-node formal Markov chain G. By definition, for each $i \in K$, V_i is the set of ancestors of node i in the subgraph of G obtained by removing all nodes in $K \backslash \{i\}$. The cut graph of G contains the clique $K = \{1, 5, 6, 8, 9\}$. Indeed, condition (iib) of Theorem 12 is satisfied: the sets V_1, V_5, V_6, V_8, and V_9 are disjoint, and the quotient graph Q as defined in Theorem 12 is a cycle visiting V_1, V_5, V_6, V_8, and V_9, in this order[2]. Intuitively, the nodes in K act as *no-return points* in the graph G in the sense that, once the set V_i has been exited (necessarily via i) for some $i \in K$, the only way of returning to this set is by traversing the graph G consistently with the cycle Q. Saying that

[2] Note that we identify each subset V_i with its representative vertex $i \in K$. While formally Q is defined with K as its set of vertices, we equivalently think of it as having the family $(V_i)_{i \in K}$ as vertices.

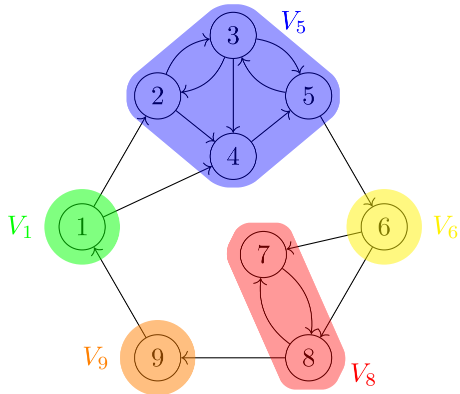


Figure 7 A formal Markov chain whose cut graph contains the clique $\{1, 5, 6, 8, 9\}$.

the sets V_1, V_5, V_6, V_8, and V_9 are pairwise disjoint (and therefore form a partition of V) is equivalent to saying that node i is the only exit point from set V_i, for each $i \in K$. Focusing for instance on nodes 5 and 8, the 5, 8-sourced cut is given by $(V_5 \cup V_1 \cup V_9, V_8 \cup V_6)$.

Proof of Theorem 12. We prove each statement one after another.

(i) Let $k \in V$. Because G is strongly connected, there exists a directed path $k_1, k_2, \ldots, k_n$, with $k_1 = k$ and $k_n \in K$. Then $k \in V_{k_p}$ with $p = \min\{q \in \{1, 2, \ldots, n\} | k_q \in K\}$.

(ii) We prove each direction of the equivalence separately.
First assume that (iia) is satisfied: K is a clique in $C_1(G)$. By Proposition 4, it means that $A_i(G \backslash j) \cap A_j(G \backslash i) = \emptyset$ for each $i, j \in K$. We now verify the two parts of (iib):

- $(V_i)_{i \in K}$ is a partition of V: For each $i, j \in K$, we have $V_i \cap V_j = \emptyset$ because $V_i \subseteq A_i(G \backslash j)$, $V_j \subseteq A_j(G \backslash i)$, and $A_i(G \backslash j) \cap A_j(G \backslash i) = \emptyset$. Therefore, $(V_i)_{i \in K}$ is a family of pairwise disjoint sets. Combining this with (i) implies that $(V_i)_{i \in K}$ is a partition of V.
- Q is a directed cycle: Q is strongly connected because G is and $(V_i)_{i \in V}$ covers V. It remains to be proved that, for each $i \in K$, there is at most one $j \in K \backslash \{i\}$ such that $(i, j) \in L$. First observe that, for each $i, j \in K$, we have $E \cap (V_i \times V_j) \subseteq \{i\} \times V_j$ because $V_i \subseteq A_i(G \backslash j)$, $V_j \subseteq A_j(G \backslash i)$, and (iia) implies that $(A_i(G \backslash j), A_j(G \backslash i))$ is an i, j-sourced cut. Now assume for the sake of contradiction that there are $j, j' \in K$, with $j \neq j'$, such that $(i, j) \in L$ and $(i, j') \in L$. Combined with the previous observation, it follows there exist $k \in V_j$ and $k' \in V_{j'}$ such that $(i, k) \in E$ and $(i, k') \in E$. Recalling the definitions of V_j and $V_{j'}$, we conclude that $i \in A_j(G \backslash j') \cap A_{j'}(G \backslash j)$, By Proposition 4, this contradicts our assumption that there is a j, j'-sourced cut.

Now assume that (iib) is satisfied, i.e., $(V_i)_{i, \in K}$ is a partition of V and Q is a directed cycle. We proceed step-by-step.

(A) We first verify that, for each $(i, j) \in L$, we have $E \cap (V_i \times V_j) \subseteq \{i\} \times V_j$, i.e., edges in G from V_i to V_j necessarily have source node i. Let $(i, j) \in L$. Assume for the sake of contradiction that there exist $k \in V_i \backslash \{i\}$ and $\ell \in V_j$ such that $(k, \ell) \in E$. Then there exists a path from k to j (through ℓ) that does not visit any node in $K \backslash \{j\}$, meaning that $k \in V_j$, which is impossible because $k \in V_i$ and $V_i \cap V_j = \emptyset$.

Now let $i, j \in K$, $S = \bigcup_{k \in A_i(Q \backslash j)} V_k$, and $T = \bigcup_{k \in A_j(Q \backslash i)} V_k$. Our end goal is to prove that (S, T) is an i, j-sourced cut.

(B) We know that (S, T) is a partition of V because $(A_i(Q \backslash j), A_j(Q \backslash i))$ is a partition of K (because Q is a directed cycle) and $(V_k)_{k \in k}$ is a partition of V.

(C) Let us prove that, for each $k \in A_i(Q\backslash j)$, every path in G from any node in V_k to node j visits node i. Let $k \in A_i(Q\backslash j)$ and $\ell \in V_k$. Consider any path $\ell_1, \ell_2, \dots, \ell_n$ in G such that $\ell_1 = \ell$ and $\ell_n = j$. For each $p \in \{1, 2, \dots, n\}$, let k_p denote the unique node in K such that $\ell_p \in V_{k_p}$; we have in particular $k_1 = k$ and $k_n = j$. By definition of Q, for each $p \in \{1, 2, \dots, n-1\}$, we have either $k_p = k_{p+1}$ or $(k_p, k_{p+1}) \in L$, i.e., the sequence of distinct nodes in $k_1, k_2, \dots, k_n$ forms a path in Q. Since Q is a directed cycle and $k \in A_i(Q\backslash j)$, the only path in Q from $k_1 = k$ to $k_n = j$ visits i. Hence, there is $q \in \{1, 2, \dots, n-1\}$ such that $k_q = i$, and we can define $p = \max\{q \in \{1, 2, \dots, n\} : k_q = i\}$. We know that $p \le n-1$ because $k_n = j \ne i$. We obtain $(k_p, k_{p+1}) \in L$, so that by A we have necessarily $\ell_p = i$.

(D) Let us now prove that $A_j(G\backslash i) \subseteq T$, i.e., if $\ell \in S = V\backslash T$ then $\ell \notin A_j(G\backslash i)$. Let $\ell \in S$. By definition of S, there is $k \in A_i(Q\backslash j)$ so that $\ell \in V_k$. By C, every path in G from node ℓ to node j visits node i (and such a path exists because G is strongly connected). By definition of $A_j(G\backslash i)$, this means that $\ell \notin A_j(G\backslash i)$.

(E) Putting all the pieces together, we have that $A_j(G\backslash i) \subseteq T$ (by D), $A_i(G\backslash j) \subseteq S$ (by symmetry), (S, T) is a partition of V (by B), and $A_i(G\backslash j) \cup A_j(G\backslash i) = V$ (by Lemma 6i). It follows that $A_i(G\backslash j) = S$ and $A_j(G\backslash i) = T$, and that $A_i(G\backslash j) \cap A_j(G\backslash i) = \emptyset$, i.e., nodes i and j are joint-ancestor free. By Proposition 4, we conclude that $(A_i(G\backslash j), A_j(G\backslash i)) = (S, T)$ is the i, j-sourced cut.

(iii) This is a by-product of the proof of (ii). ◀

B Second-level cuts: Progress towards Conjecture 10

In Appendix B.1, we prove intermediary results that are then applied in Appendix B.2 to prove Theorem 15.

B.1 Paths in the cut graph

Let us first study the behavior of paths $i_1, i_2, \dots, i_n$ in the cut graph $C_1(G)$, where each pair of neighboring vertices in the path is joint-ancestor free. We are interested in the joint-ancestor freeness of the two terminal nodes in the path, i_1 and i_n. We show that the endpoints i_1 and i_n cannot share a joint ancestor in the subgraph graph $G\backslash\{i_2, i_3, \dots i_{n-1}\}$.

First, for the purpose of induction, we prove this claim for length-3 paths:

▶ **Lemma 13.** *Consider a directed graph $G = (V, E)$ and three nodes $i_1, i_2, i_3 \in V$ such that i_1 and i_2 are joint-ancestor free (i.e., $A_{i_1}(G\backslash i_2) \cap A_{i_2}(G\backslash i_1) = \emptyset$) and i_2 and i_3 are joint-ancestor free (i.e., $A_{i_2}(G\backslash i_3) \cap A_{i_3}(G\backslash i_2) = \emptyset$). Then $A_{i_1}(G\backslash\{i_2, i_3\}) \cap A_{i_3}(G\backslash\{i_1, i_2\}) = \emptyset$.*

Proof. Assume for the sake of contradiction that $A_{i_1}(G\backslash\{i_2, i_3\}) \cap A_{i_3}(G\backslash\{i_1, i_2\}) \ne \emptyset$, and let $k \in A_{i_1}(G\backslash\{i_2, i_3\}) \cap A_{i_3}(G\backslash\{i_1, i_2\})$. We will prove that either $k \in A_{i_2}(G\backslash i_1)$ or $k \in A_{i_2}(G\backslash i_3)$, which will contradict our assumption that i_1 and i_2 are joint-ancestor free and i_1 and i_3 are joint-ancestor free. Since G is strongly connected, there is a path $p = p(k \to i_2)$. We now make a case disjunction.

First, suppose that $p(k \to i_2)$ either does not visite node i_1 or does not visit node i_3. If $p(k \to i_2)$ does not visit node i_1, then $k \in A_{i_2}(G\backslash i_1)$. Since $k \in A_{i_1}(G\backslash\{i_2, i_3\}) \subseteq A_{i_1}(G\backslash i_2)$, this contradicts our assumption that nodes i_1 and i_2 are joint-ancestor free. If $p(k \to i_2)$ does not visit node i_3, then $k \in A_{i_2}(G\backslash i_3)$, which leads to a similar contradiction regarding nodes i_2 and i_3.

On the other hand, suppose that $p(k \to i_2)$ visits both i_1 and i_3. Let us consider p', the last portion of p beginning at the last visit to either i_1 or i_3. Without loss of generality, suppose p' begins at i_1 and reaches i_2 without visiting i_3. Since we assumed that $k \in A_{i_1}(G\backslash\{i_2, i_3\})$, concatenating $p(k \to i_1\backslash\{i_2, i_3\})$ and p' gives us a path from k to i_2 without visiting i_3. So $k \in A_{i_2}(G\backslash i_3)$, which as explained before leads to a contradiction.

In every case, we have a contradiction to either our assumption that i_1 and i_2 are joint-ancestor free, or that i_2 and i_3 are joint-ancestor free. Thus, our initial assumption much be wrong: There is no node k in $A_{i_1}(G\backslash\{i_2, i_3\}) \cap A_{i_3}(G\backslash\{i_1, i_2\})$, as desired. ◀

Next, we build inductively on this result to handle paths of arbitrary lengths.

▶ **Lemma 14.** *Consider a directed graph $G = (V, E)$ and a sequence of $n \geq 3$ distinct nodes $i_1, i_2, \ldots, i_n$ such that i_p and i_{p+1} are joint-ancestor free for each $p \in \{1, 2, \ldots, n-1\}$. Then $A_{i_1}(G\backslash\{i_2, i_3, \ldots, i_n\}) \cap A_{i_n}(G\backslash\{i_1, i_2, \ldots, i_{n-1}\}) = \emptyset$.*

Proof. We make a proof by induction over n, using Lemma 13 as a base case for $n = 3$.

Let $n \geq 4$ and assume the induction assumption is true for each $p \in \{3, 4, \ldots, n-1\}$. Assume for the sake of contradiction that the conclusion does not hold, i.e., there exist $k \in V$ and two paths $p_{(a)} = p(k \to i_1\backslash\{i_2, i_3, \ldots, i_n\})$ and $p_{(b)} = p(k \to i_n\backslash\{i_1, i_2, \ldots, i_{n-1}\})$.

Since G is strongly connected, we know that at least one of the following is true: there is a path $p_{(1)} = p(i_1 \to i_{n-1}\backslash i_n)$, or there is a path $p_{(2)} = p(i_n \to i_{n-1}\backslash i_1)$. To see why, consider a path x from i_1 to i_{n-1}. If the path x avoids i_n, x is $p_{(1)}$, and we have the first case. If the path visits i_n, the tail of the path x starting at its visit to i_n is $p_{(2)}$, satisfying the second case.

Let us consider each case in turn:

- Suppose there is a path $p_{(1)} = p(i_1 \to i_{n-1}\backslash i_n)$. In this case, concatenating $p_{(a)}$ and $p_{(1)}$ gives us a path $p_{(3)} = p(k \to i_{n-1}\backslash i_n)$. The existence of paths $p_{(3)}$ and $p_{(b)}$ implies that that $k \in A_{i_{n-1}}(G\backslash i_n) \cap A_{i_n}(G\backslash i_{n-1})$, which contradicts our assumption that i_{n-1} and i_n are joint-ancestor free.
- On the other hand, suppose that there is a path $p_{(2)} = p(i_n \to i_{n-1}\backslash i_1)$. Let $q \in \{2, 3, \ldots, n-1\}$ so that i_q is the first node in $p_{(2)}$ that belongs to $\{i_2, i_3, \ldots, i_{n-1}\}$. This provides us with a path $p_{(4)} = p(i_n \to i_q\backslash\{i_1, \ldots, i_{q-1}, i_{q+1}, \ldots, i_{n-1}\})$ Concatenating $p_{(b)}$ and $p_{(4)}$ gives us a path $p_{(5)} = p(k \to i_q\backslash\{i_1, \ldots, i_{q-1}, i_{q+1}, \ldots, i_{n-1}\})$. The existence of paths $p_{(5)}$ and $p_{(a)}$ implies that $k \in A_{i_1}(G\backslash\{i_2, i_3, \ldots, i_{q-1}\}) \cap A_{i_q}(G\backslash\{i_1, \ldots, i_{q-1}\})$. If $q = 2$, this contradicts our assumption that i_1 and i_2 are joint-ancestor free. If $q \geq 3$, this contradicts the induction assumption. ◀

B.2 Special case: Expanding I when $|J| = 1$

Now building on Lemma 14, we are ready to prove Theorem 15, which we see as a stepping stone to prove Conjecture 10, that any joint-ancestor free subsets $I \subseteq K_1$ and $J \subseteq K_2$ can be expanded to joint-ancestor freeness of the entire connected components of the cut graph, K_1 and K_2. In other words, that any broad second-level cut gives rise to a narrow second-level cut. Here, we only focus on the case of expanding I when J is a single node $\{j\}$.

▶ **Theorem 15.** *Consider a directed graph $G = (V, E)$. Let K_1 and K_2 denote two connected components of $C_1(G)$. Assume that there is a nonempty strict subset I of K_1 and a vertex $j \in K_2$ such that I and j are joint-ancestor free (in G). Then there exists $i \in K_1\backslash I$ such that $I \cup \{i\}$ and j are also joint-ancestor free.*

Proof. First, because I and j are joint-ancestor free and because G is strongly connected, there is $\ell \in A_I(G\backslash j)$ such that $(j, \ell) \in E$: In particular, there must be a path from j to some node in I, and we may take ℓ to be the second node on that path, after j.

Next, because $\ell \in A_I(G\backslash j)$, there is a path $p(\ell \to i'\backslash j)$ for some $i' \in I$. In particular, by ending when the path first enters I, there must exist a path $p_{(1)}(\ell \to i'\backslash(\{j\} \cup I\backslash\{i'\})$ for some $i' \in I$.

Now, we will switch to viewing paths in the cut graph $C_1(G)$. In particular, we will think about I and K_1, the connected component of $C_1(G)$ that I lies within. Because K_1 is a connected component of $C_1(G)$, there is a path in the cut graph going from i' to an arbitrary node $i \in K_1\backslash I$. In particular, there is a path in the cut graph that stays within I until it visits i as the first node in the path outside of I and in K_1. In other words, this is a cut graph path in $I \cup \{i\}$.

Now, assume for the sake of contradiction that $I \cup \{i\}$ and j are not joint-ancestor free, i.e., there exists $k \in A_{I\cup\{i\}}(G\backslash j) \cap A_j(G\backslash(I \cup \{i\}))$. Because I and j are joint-ancestor free, we necessarily have $k \in A_i(G\backslash j) \cap A_j(G\backslash(I \cup \{i\}))$. Hence, there exists a path (in G) $p(k \to i\backslash j)$ and a path $p(k \to j\backslash(I \cup \{i\})$.

Now, note that the path $p(k \to i\backslash j)$ does not visit I, so it is also a path $p(k \to i\backslash I)$. To see why, note that if this path did visit I, then taking the portion from k to the first visit to I would give a path $p(k \to I\backslash j)$. But because we also have a path $p(k \to j\backslash(I \cup \{i\}))$, it follows that $k \in A_I(G\backslash j) \cap A_j(G\backslash I)$, which contradicts our assumption that I and j are joint-ancestor free.

Now, let us return to the path $p_{(1)}(\ell \to i'\backslash(\{j\}\cup I\backslash\{i'\}))$. We claim that the path $p_{(1)}(\ell \to i'\backslash(\{j\} \cup I\backslash\{i'\}))$ does not visit node i, so in particular it is a path $p_{(1)}(\ell \to i'\backslash(I \cup \{i\}\backslash\{i'\})$. To see why, note that if $p_{(1)}$ did visit i prior to visiting i', then by taking the portion of the path just after visiting i, there is a path $p(i \to I\backslash j)$. Concatenating this path with the $p(k \to i\backslash j)$ path mentioned previously, we now have a path $p(k \to I\backslash j)$. Thus, k is a joint ancestor of I and j, contradicting our assumption.

Now, we're ready to put it all together. Concatenating the path $p(k \to j\backslash(I \cup \{i\})$, the edge (j, ℓ), and then to the path $p_{(1)}(\ell \to i'\backslash(I\cup\{i\}\backslash\{i'\})$ yields a path $p(k \to i'\backslash(I\cup\{i\}\backslash\{i'\})$. Therefore, we have a path $p(k \to i\backslash I)$ and a path $p(k \to i'\backslash(I \cup \{i\}\backslash\{i'\})$. By Lemma 14, this contradicts the fact that there is a path between i and i' through I in the cut graph $C_1(G)$.

Thus, our assumption was false, and $I \cup \{i\}$ and j are joint-ancestor free, as desired. ◀

References

1 Jeffrey P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, 16(9):527–531, September 1973.

2 M. Reiser and S. S. Lavenberg. Mean-value analysis of closed multichain queuing networks. *Journal of the ACM*, 27(2):313–322, April 1980.

3 L. R. van Kreveld, O. J. Boxma, J. L. Dorsman, and M. R. H. Mandjes. Scaling limits for closed product-form queueing networks. *Performance Evaluation*, 151:102220, November 2021.

4 Ellen Cardinaels, Sem Borst, and Johan S. H. van Leeuwaarden. Heavy-traffic universality of redundancy systems with assignment constraints. *Operations Research*, 72(4):1539–1555, July 2024.

5 Ellen Cardinaels, Sem Borst, and Johan S.H. van Leeuwaarden. Multi-dimensional state space collapse in non-complete resource pooling scenarios. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 8(2):24:1–24:52, May 2024.

6 Jonatha Anselmi, Bruno Gaujal, and Louis-Sébastien Rebuffi. Learning optimal admission control in partially observable queueing networks. *Queueing Systems*, 108(1):31–79, October 2024.

7 Céline Comte, Matthieu Jonckheere, Jaron Sanders, and Albert Senen-Cerda. Score-aware policy-gradient and performance guarantees using local lyapunov stability. *Journal of Machine Learning Research*, 26(132):1–74, 2025.

8 James R. Jackson. Networks of waiting lines. *Operations Research*, 5(4):518–521, 1957.

9 Forest Baskett, K. Mani Chandy, Richard R. Muntz, and Fernando G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, apr 1975.

10 Richard Serfozo. *Introduction to stochastic networks.* Springer, 1999.

11 S. A. Berezner, C. F. Kriel, and A. E. Krzesinski. Quasi-reversible multiclass queues with order independent departure rates. *Queueing Systems*, 19(4):345–359, December 1995.

12 A. E. Krzesinski. Order independent queues. In Richard J. Boucherie and Nico M. van Dijk, editors, *Queueing networks: A fundamental approach*, number 154 in International Series in Operations Research & Management Science, pages 85–120. Springer US, 2011.

13 Enrique Daniel Andjel. Invariant measures for the zero range process. *The Annals of Probability*, 10(3), August 1982.

14 F. P. Kelly. *Reversibility and stochastic networks.* Cambridge University Press, 2011.

15 Isaac Grosof, Mor Harchol-Balter, and Alan Scheller-Wolf. New stability results for multiserver-job models via product-form saturated systems. *ACM SIGMETRICS Performance Evaluation Review*, 51(2):6–8, October 2023.

16 S. A. Berezner and A. E. Krzesinski. Order independent loss queues. *Queueing Systems*, 23(1):331–335, March 1996.

17 Céline Comte and Jan-Pieter Dorsman. Pass-and-swap queues. *Queueing Systems*, 98(3):275–331, August 2021.

18 Kristen Gardner and Rhonda Righter. Product forms for FCFS queueing models with arbitrary server-job compatibilities: an overview. *Queueing Systems*, 96(1):3–51, October 2020.

19 Urtzi Ayesta, Tejas Bodas, Jan-Pieter L. Dorsman, and Ina Maria Verloop. A token-based central queue with order-independent service rates. *Operations Research*, 70(1):545–561, 2022.

20 Alexander Rumyantsev and Evsey Morozov. Stability criterion of a multiserver model with simultaneous service. *Annals of Operations Research*, 252(1):29–39, May 2017.

21 Brion N. Feinberg and Samuel S. Chiu. A method to calculate steady-state distributions of large Markov chains by aggregating states. *Operations Research*, 35(2):282–290, April 1987.

22 Isaac Grosof, Mor Harchol-Balter, and Alan Scheller-Wolf. Stability for two-class multiserver-job systems. *arXiv preprint arXiv:2010.00631*, 2020.

23 Peter G. Harrison and Naresh M. Patel. *Performance modelling of communication networks and computer architectures.* Wokingham, England ; Reading, Mass. : Addison-Wesley, 1993.

24 G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling.* ASA-SIAM Series on Statistics and Applied Mathematics. Society for Industrial and Applied Mathematics, jan 1999.

25 Juan F. Pérez and Benny Van Houdt. Quasi-birth-and-death processes with restricted transitions and its applications. 68(2):126–141, February 2011.

26 Manindra Agrawal and Ramprasad SaptharishiI. Classifying polynomials and identity testing. Indian Academy of Sciences, 2009.

27 Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka. Equivalence of polynomial identity testing and polynomial factorization. *Computational Complexity*, 24(2):295–331, June 2015.

28 Susanna Donatelli and Matteo Sereno. On the product form solution for stochastic Petri nets. In K. Jensen, editor, *Application and Theory of Petri Nets 1992*, pages 154–172. Springer, 1992.

29 Simonetta Balsamo, Peter G. Harrison, and Andrea Marin. Methodological construction of product-form stochastic Petri nets for performance evaluation. *Journal of Systems and Software*, 85(7):1520–1539, July 2012.

30 Moshe Haviv. Aggregation/disaggregation methods for computing the stationary distribution of a Markov chain. *SIAM Journal on Numerical Analysis*, 24(4):952–966, August.

31 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*, 5(3–4):207–388, 2010.

32 Pierre Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation and Queues*, volume 31 of *Texts in Applied Mathematics*. Springer International Publishing.

33 François Baccelli and Serguei Foss. On the saturation rule for the stability of queues. *Journal of Applied Probability*, 32(2):494–507, 1995.

34 Serguei Foss and Takis Konstantopoulos. An overview of some stochastic stability methods. *Journal of the Operations Research Society of Japan*, 47(4):275–303, 2004.

35 Isaac Grosof, Yige Hong, Mor Harchol-Balter, and Alan Scheller-Wolf. The RESET and MARC techniques, with application to multiserver-job analysis. *Performance Evaluation*, 162:102378, 2023.