

From Systems to Components: Constructive Methods for Product-Form Solutions

Andrea Marin¹ Maria Grazia Vigliotti²

¹Dipartimento di Informatica
Università Ca' Foscari di Venezia

²Department of Computing
Imperial College London

QUEST 10

What are product-form solutions?

In general terms ...

Product-form solution is a particularly efficient form of solution for determining the equilibrium distribution of a Markov process in cases where it can be written as a product of the equilibrium distributions of sub-components.

Question:

What are the sub-components in the Markov chain?

What are product-form solutions?

In general terms ...

Product-form solution is a particularly efficient form of solution for determining the **equilibrium distribution of a Markov process** in cases where it can be written as a product of the equilibrium distributions of sub-components.

Question:

What are the sub-components in the Markov chain?

What are product-form solutions?

In general terms ...

Product-form solution is a particularly efficient form of solution for determining the **equilibrium distribution of a Markov process** in cases where it can be written as a product of the equilibrium distributions of sub-components.

Question:

What are the sub-components in the Markov chain?

Brief history

- ▶ Product-form solutions in performance analysis originated in queueing theory.
- ▶ Most of the important results are formulated in this area.

What are queues?

Graphical representation of a queue



Type of queue specifies:

- Arrival of customers.
- Service of customers.

What are queues?

Graphical representation of a queue



Type of queue specifies:

- ▶ Arrival of customers.
- ▶ Service of customers.
- ▶ Discipline of the queue.
- ▶ ...

What are queues?

Graphical representation of a queue



Type of queue specifies:

- ▶ Arrival of customers.
- ▶ Service of customers.
- ▶ Discipline of the queue.
- ▶ ...

What are queues?

Graphical representation of a queue



Type of queue specifies:

- ▶ Arrival of customers.
- ▶ Service of customers.
- ▶ Discipline of the queue.
- ▶ ...

What are queues?

Graphical representation of a queue

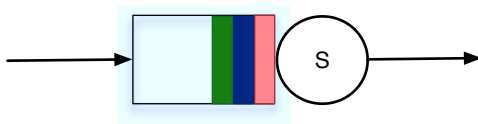


Type of queue specifies:

- ▶ Arrival of customers.
- ▶ Service of customers.
- ▶ Discipline of the queue.
- ▶ ...

What are queues?

Graphical representation of a queue

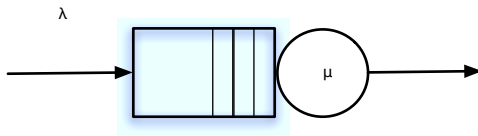


Type of queue specifies:

- ▶ Arrival of customers.
- ▶ Service of customers.
- ▶ Discipline of the queue.
- ▶ ...

More on queues

M/M/1 queue



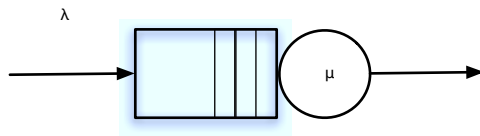
Steady state distribution

$$\begin{aligned} \lim_{t \rightarrow \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) &= \pi(n) \\ &= (1 - \rho)\rho^n \end{aligned}$$

n is the number of customers queueing and $\frac{\lambda}{\mu} = \rho < 1$.

More on queues

M/M/1 queue



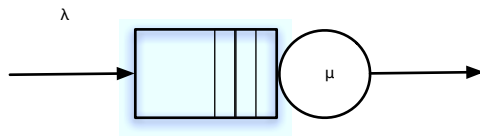
Steady state distribution

$$\begin{aligned} \lim_{t \rightarrow \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) &= \pi(n) \\ &= (1 - \rho)\rho^n \end{aligned}$$

n is the number of customers queueing and $\frac{\lambda}{\mu} = \rho < 1$.

More on queues

M/M/1 queue



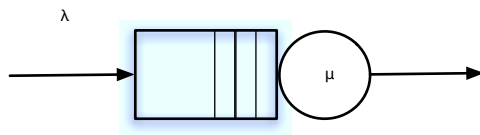
Steady state distribution

$$\begin{aligned}\lim_{t \rightarrow \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) &= \pi(n) \\ &= (1 - \rho)\rho^n\end{aligned}$$

n is the number of customers queueing and $\frac{\lambda}{\mu} = \rho < 1$.

More on queues

M/M/1 queue



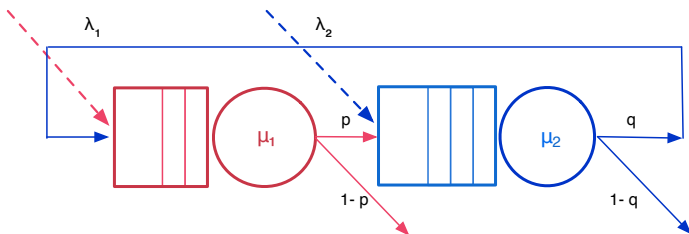
Steady state distribution

$$\begin{aligned} \lim_{t \rightarrow \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) &= \pi(n) \\ &= (1 - \rho)\rho^n \end{aligned}$$

n is the number of customers queueing and $\frac{\lambda}{\mu} = \rho < 1$.

Connecting queues

Two-node Jackson network



Traffic equations

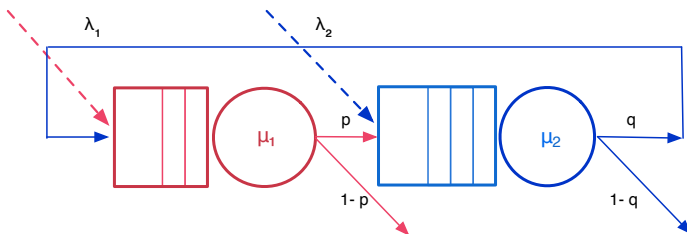
The mean rates of an arrival in each queue are:

$$\gamma_1 = \lambda_1 + q(\gamma_2)$$

$$\gamma_2 = \lambda_2 + p(\gamma_1)$$

Connecting queues

Two-node Jackson network



Product form solution

$$\pi(n, m) = C \pi_1(n) \pi_2(m)$$

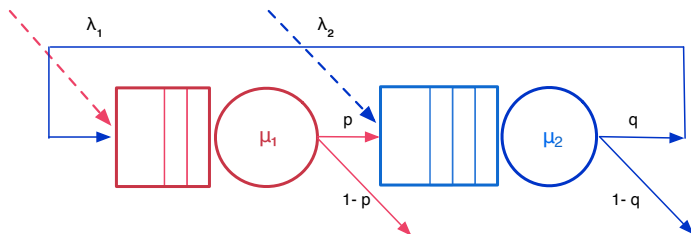
Traffic equations

The mean rates of an arrival in each queue are:



Connecting queues

Two-node Jackson network



Traffic equations

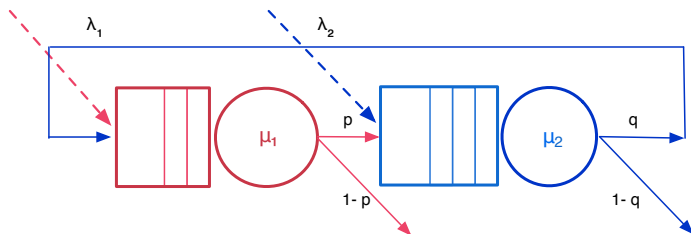
The mean rates of an arrival in each queue are:

$$\gamma_1 = \lambda_1 + q(\gamma_2)$$

$$\gamma_2 = \lambda_2 + p(\gamma_1)$$

Connecting queues

Two-node Jackson network



Traffic equations

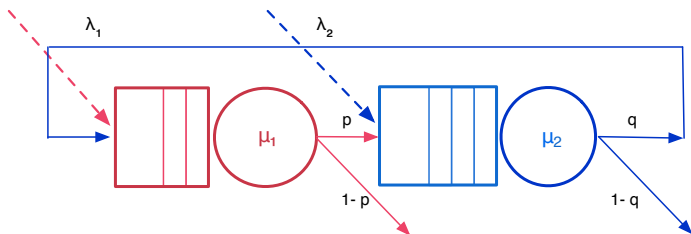
The mean rates of an arrival in each queue are:

$$\gamma_1 = \lambda_1 + q(\gamma_2)$$

$$\gamma_2 = \lambda_2 + p(\gamma_1)$$

Connecting queues

Two-node Jackson network



Traffic equations

The mean rates of an arrival in each queue are:

$$\gamma_1 = \lambda_1 + q(\gamma_2)$$

$$\gamma_2 = \lambda_2 + p(\gamma_1)$$

The result is interesting because...

Modular property

- ▶ Steady state distribution of a network can be calculated in terms of the marginal distributions of its **'components'**.
- ▶ **system** → **components**

Product-form solution is not trivial

- ▶ The two queues are not independent.
- ▶ Product-form of the solution does not hold for transient distributions.

But not only for queues . . .

Steady state distributions can be calculated for any ergodic stochastic process, not only for queues.

The result is interesting because...

Modular property

- ▶ Steady state distribution of a network can be calculated in terms of the marginal distributions of its 'components'.
- ▶ system \rightarrow components

Product-form solution is not trivial

- ▶ The two queues are not independent.
- ▶ Product-form of the solution does not hold for transient distributions.

But not only for queues . . .

Steady state distributions can be calculated for any ergodic stochastic process, not only for queues.



The result is interesting because...

Modular property

- ▶ Steady state distribution of a network can be calculated in terms of the marginal distributions of its 'components'.
- ▶ **system** → **components**

Product-form solution is not trivial

- ▶ The two queues are not independent.
- ▶ Product-form of the solution does not hold for transient distributions.

But not only for queues . . .

Steady state distributions can be calculated for any ergodic stochastic process, not only for queues.

The result is interesting because...

Modular property

- ▶ Steady state distribution of a network can be calculated in terms of the marginal distributions of its 'components'.
- ▶ system \rightarrow components

Product-form solution is not trivial

- ▶ The two queues are not independent.
- ▶ Product-form of the solution does not hold for transient distributions.

But not only for queues . . .

Steady state distributions can be calculated for any ergodic stochastic process, **not only for queues.**

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

How did he come up with the solution?

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

- ▶ How did he came up with the solution?
- ▶ Did God reveal it to him or did he found it via simulation?
- ▶ The system of linear equations above does not tell us if a network enjoys a product-form solution.

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

- ▶ How did he came up with the solution?
- ▶ Did God reveal it to him or did he found it via simulation?
- ▶ The system of linear equations above does not tell us if a network enjoys a product-form solution.

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

- ▶ How did he came up with the solution?
- ▶ Did God reveal it to him or did he found it via simulation?
- ▶ The system of linear equations above does not tell us if a network enjoys a product-form solution.

Derivation of product-form solution

In the original proof, Jackson (1963) simply substituted the solution into the global balance equations.

Two-node network

$$\begin{aligned}\pi(n_1, n_2)\mathbf{Q} &= \mathbf{0} && \text{(stationary measure)} \\ \pi_1(n_1)\pi_2(n_2)\mathbf{Q} &= \mathbf{0}\end{aligned}$$

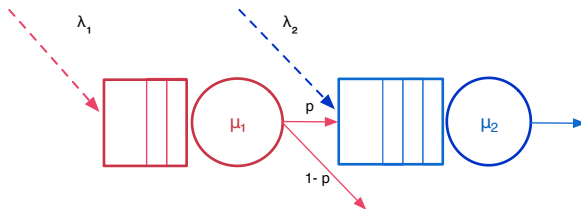
with $\sum_{(n_1, n_2) \in \mathcal{S}} \pi(n_1, n_2) = 1$ - (steady-state probability).

Search for product-form solution

- ▶ How did he came up with the solution?
- ▶ Did God reveal it to him or did he found it via simulation?
- ▶ The system of linear equations above does not tell us if a network enjoys a product-form solution.

Search for product-form solution

Product-form solution?

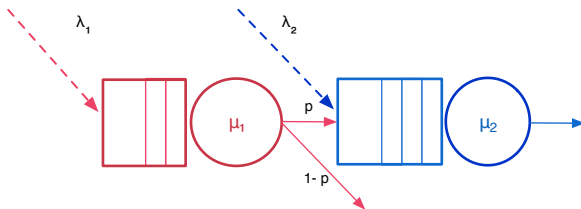


The name of the game

Find a set of sufficient (and necessary) conditions that guarantee product-form solutions in time-homogenous Continuous Time Markov Chains (CTMCs).

Search for product-form solution

Product-form solution?

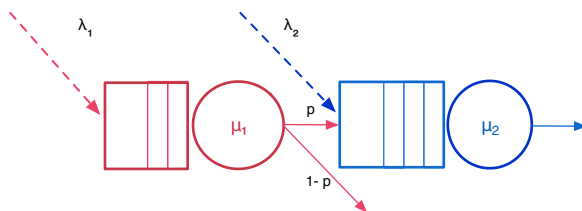


The name of the game

Find a set of sufficient (and necessary) conditions that guarantee product-form solutions in time-homogenous Continuous Time Markov Chains (CTMCs).

Search for product-form solution

Product-form solution?



The name of the game

Find a set of **sufficient** (and necessary) **conditions** that guarantee product-form solutions in time-homogenous Continuous Time Markov Chains (CTMCs).

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ G-networks (Gelenbe).
- ▶ Networks with finite buffers and blocking.
- ▶

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ G-networks (Gelenbe).
- ▶ Networks with finite buffers and blocking.
- ▶

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ **G-networks (Gelenbe).**
- ▶ Networks with finite buffers and blocking.
- ▶

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ G-networks (Gelenbe).
- ▶ **Networks with finite buffers and blocking.**
- ▶

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ G-networks (Gelenbe).
- ▶ Networks with finite buffers and blocking.
- ▶

Background

A non-exhaustive list of results for queueing theory

- ▶ Jackson networks and Gordon-Newell closed networks.
- ▶ $M \rightarrow M$ (Muntz); Local Balance (Chandy et al.)
- ▶ BCMP theorem.
- ▶ Quasi-reversibility (Kelly/Serfozo).
- ▶ G-networks (Gelenbe).
- ▶ Networks with finite buffers and blocking.
- ▶

Other product-form solution

Boucherie product-form solutions.

In summary so far ...

Traditional outlook

Product-form solutions refer to queues/networks only.

Components

In queueing networks it is natural to think of queues as components.

Yet ...

Networks are a subclass of CTMCs.
Can we talk about product-form solutions for CTMCs?

In summary so far ...

Traditional outlook

Product-form solutions refer to queues/networks only.

Components

In queueing networks it is **natural** to think of queues as components.

Yet ...

Networks are a subclass of CTMCs.
Can we talk about product-form solutions for CTMCs?

In summary so far ...

Traditional outlook

Product-form solutions refer to queues/networks only.

Components

In queueing networks it is **natural** to think of queues as components.

Yet ...

Networks are a subclass of CTMCs.
Can we talk about product-form solutions for CTMCs?

In summary so far ...

Traditional outlook

Product-form solutions refer to queues/networks only.

Components

In queueing networks it is **natural** to think of queues as components.

Yet ...

Networks are a subclass of CTMCs.
Can we talk about product-form solutions for CTMCs?

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ networks
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ **any** time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ any time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ **any** time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ **any** time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ **any** time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

What is this tutorial about?

- ▶ We shall present two main results, **Generalised Reversed Compound Agent Theorem (GRCAT)** and **Extended Reversed Compound Agent Theorem (ERCAT)** that provide general conditions for product-form solutions.
- ▶ Our results are very general. They apply to:
 - ▶ queues and
 - ▶ **any** time-homogenous CTMC.
- ▶ We shall give a formal definition of what is a **component**.
- ▶ We shall build **components** that have certain properties.
- ▶ We show that **systems** built out of those components have a product-form solution.

Interplay between System and Components

From Components to System

We shall **interconnect** components to build a **system** that enjoys product-form solution. Components must have certain properties (necessary conditions).

From System to Components

The global state of the system will also influence the **rates** of individual components -(traffic equation).

Interplay between System and Components

From Components to System

We shall **interconnect** components to build a **system** that enjoys product-form solution. Components must have certain properties (necessary conditions).

From System to Components

The global state of the system will also influence the **rates** of individual components -(traffic equation).

Outline

Labelled Markov Automata

GRCAT

Jackson Networks

Non-queueing example

Perturbation of product-form solution

Conclusion

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Introduction Labelled Markov Automata

- ▶ This is the minimal formalism necessary to precisely formulate our main results.
- ▶ This formalism is clean, clear, and precise.
- ▶ The semantics is expressed in the tradition of the operational semantics for programming languages.
- ▶ The language is not essential, transitions are important.
- ▶ There is no suitable grammar for reversed processes, but transitions are easy to revert.
- ▶ LMA has been inspired by PEPA and Labelled Automata used in model checking.

Labelled Markov Automata

$\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is a labelled Markov automaton (LMA):

- ▶ \mathbf{S} is state space with states with $s_1, s_2, \dots, s_n, \dots$
- ▶ \mathcal{A} is the set of *active labels* with $a_1, a_2, \dots, a_n, \dots$
- ▶ \mathcal{P} is the set of *passive labels* with $p_1, p_2, \dots, p_n, \dots$,
- ▶ \rightarrow is the transition relation between states defined as

$$\rightarrow: (\mathbf{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathbf{S}) \cup (\mathbf{S} \times \mathcal{P} \times \text{Var} \times \mathbf{S})$$

\mathbb{R}^+ is set of real numbers and Var is the set of variables.

Labelled Markov Automata

$\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is a labelled Markov automaton (LMA):

- ▶ \mathbf{S} is state space with states with $s_1, s_2, \dots, s_n, \dots$
- ▶ \mathcal{A} is the set of *active labels* with $a_1, a_2, \dots, a_n, \dots$
- ▶ \mathcal{P} is the set of *passive labels* with $p_1, p_2, \dots, p_n, \dots$,
- ▶ \rightarrow is the transition relation between states defined as

$$\rightarrow: (\mathbf{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathbf{S}) \cup (\mathbf{S} \times \mathcal{P} \times \text{Var} \times \mathbf{S})$$

\mathbb{R}^+ is set of real numbers and Var is the set of variables.

Labelled Markov Automata

$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is a labelled Markov automaton (LMA):

- ▶ \mathcal{S} is state space with states with $s_1, s_2, \dots, s_n, \dots$
- ▶ \mathcal{A} is the set of *active labels* with $a_1, a_2, \dots, a_n, \dots$
- ▶ \mathcal{P} is the set of *passive labels* with $p_1, p_2, \dots, p_n, \dots$,
- ▶ \rightarrow is the transition relation between states defined as

$$\rightarrow: (\mathcal{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathcal{S}) \cup (\mathcal{S} \times \mathcal{P} \times \text{Var} \times \mathcal{S})$$

\mathbb{R}^+ is set of real numbers and Var is the set of variables.

Labelled Markov Automata

$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is a labelled Markov automaton (LMA):

- ▶ \mathcal{S} is state space with states with $s_1, s_2, \dots, s_n, \dots$
- ▶ \mathcal{A} is the set of *active labels* with $a_1, a_2, \dots, a_n, \dots$
- ▶ \mathcal{P} is the set of *passive labels* with $p_1, p_2, \dots, p_n, \dots$,
- ▶ \rightarrow is the transition relation between states defined as

$$\rightarrow: (\mathcal{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathcal{S}) \cup (\mathcal{S} \times \mathcal{P} \times \text{Var} \times \mathcal{S})$$

\mathbb{R}^+ is set of real numbers and Var is the set of variables.

Labelled Markov Automata

$\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is a labelled Markov automaton (LMA):

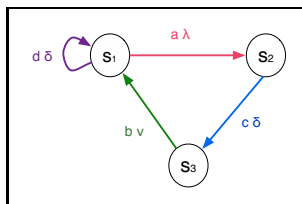
- ▶ \mathbf{S} is state space with states with $s_1, s_2, \dots, s_n, \dots$
- ▶ \mathcal{A} is the set of *active labels* with $a_1, a_2, \dots, a_n, \dots$
- ▶ \mathcal{P} is the set of *passive labels* with $p_1, p_2, \dots, p_n, \dots$,
- ▶ \rightarrow is the transition relation between states defined as

$$\rightarrow: (\mathbf{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathbf{S}) \cup (\mathbf{S} \times \mathcal{P} \times \text{Var} \times \mathbf{S})$$

\mathbb{R}^+ is set of real numbers and Var is the set of variables.

Example

Graph



Formal definition

$$\mathcal{M} = \langle S, \mathcal{A}, \rightarrow \rangle$$

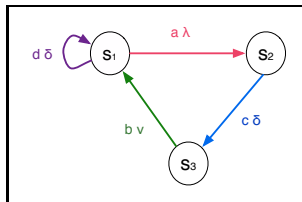
$$S_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A}, \rightarrow \rangle$$

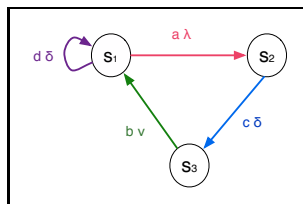
$$\mathbf{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \rightarrow \rangle$$

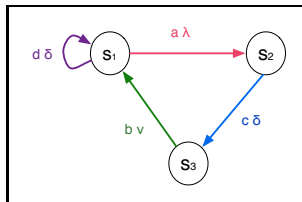
$$\mathcal{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A}, \rightarrow \rangle$$

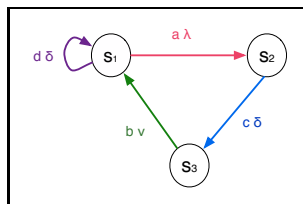
$$\mathbf{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A}, \rightarrow \rangle$$

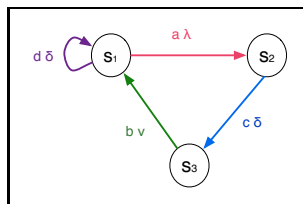
$$\mathbf{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A}, \rightarrow \rangle$$

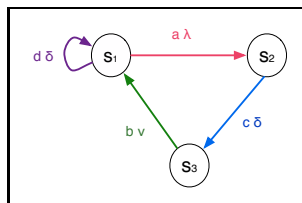
$$\mathbf{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Example

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A}, \rightarrow \rangle$$

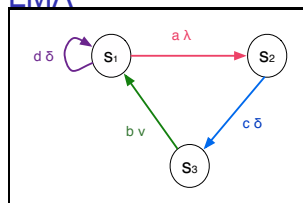
$$\mathbf{S}_1 = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b, c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, \delta, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, \delta, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1$$

Differences between LMAs and CTMCs

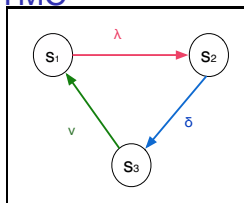
LMA



Differences

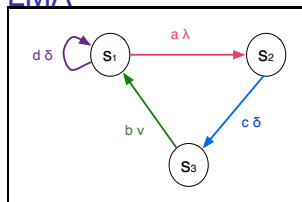
- ▶ LMA has labelled transitions.
- ▶ LMA has self-loops that are redundant CTMCs.

CTMC



Differences between LMAs and CTMCs

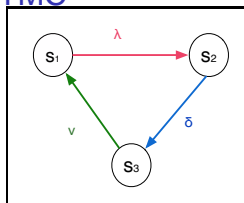
LMA



Differences

- ▶ LMA has labelled transitions.
- ▶ LMA has self-loops that are redundant CTMCs.

CTMC



The associated time-homogenous CTMC

Any CTMC is fully characterised by \mathbf{Q} , the generator matrix.

- ▶ The **transition rate** is $q(s \rightarrow s') \stackrel{df}{=} \sum_{\substack{(a,\lambda): s \xrightarrow{a,\lambda} s' \\ s \neq s'}} \lambda$.
- ▶ The **total (or exit) rate** of a state s is $q(s) \stackrel{df}{=} \sum_{\substack{s' \in \mathbf{S} \\ s' \neq s}} q(s \rightarrow s')$.
- ▶ The **diagonal** of the generator matrix is $q(s \rightarrow s) \stackrel{df}{=} -q(s)$.

LMA

The **transition rate due to action a** is $q(s \xrightarrow{a} s') \stackrel{df}{=} \sum_{\lambda: s \xrightarrow{a,\lambda} s'} \lambda$.

The associated time-homogenous CTMC

Any CTMC is fully characterised by \mathbf{Q} , the generator matrix.

- ▶ The **transition rate** is $q(s \rightarrow s') \stackrel{df}{=} \sum_{\substack{(a,\lambda): s \xrightarrow{a,\lambda} s' \\ s \neq s'}} \lambda$.
- ▶ The **total (or exit) rate** of a state s is $q(s) \stackrel{df}{=} \sum_{\substack{s' \in \mathbf{S} \\ s' \neq s}} q(s \rightarrow s')$.
- ▶ The **diagonal** of the generator matrix is $q(s \rightarrow s) \stackrel{df}{=} -q(s)$.

LMA

The **transition rate due to action a** is $q(s \xrightarrow{a} s') \stackrel{df}{=} \sum_{\lambda: s \xrightarrow{a,\lambda} s'} \lambda$.

The associated time-homogenous CTMC

Any CTMC is fully characterised by \mathbf{Q} , the generator matrix.

- ▶ The **transition rate** is $\mathbf{q}(s \rightarrow s') \stackrel{df}{=} \sum_{\substack{(a,\lambda): s \xrightarrow{a,\lambda} s' \\ s \neq s'}} \lambda$.
- ▶ The **total** (or exit) **rate** of a state s is $\mathbf{q}(s) \stackrel{df}{=} \sum_{\substack{s' \in \mathbf{S} \\ s' \neq s}} \mathbf{q}(s \rightarrow s')$.
- ▶ The **diagonal** of the generator matrix is $\mathbf{q}(s \rightarrow s) \stackrel{df}{=} -\mathbf{q}(s)$.

LMA

The **transition rate due to action a** is $\mathbf{q}(s \xrightarrow{a} s') \stackrel{df}{=} \sum_{\lambda: s \xrightarrow{a,\lambda} s'} \lambda$.

The associated time-homogenous CTMC

Any CTMC is fully characterised by \mathbf{Q} , the generator matrix.

- ▶ The **transition rate** is $\mathbf{q}(s \rightarrow s') \stackrel{df}{=} \sum_{\substack{(a,\lambda): s \xrightarrow{a,\lambda} s' \\ s \neq s'}} \lambda$.
- ▶ The **total** (or exit) **rate** of a state s is $\mathbf{q}(s) \stackrel{df}{=} \sum_{\substack{s' \in \mathbf{S} \\ s' \neq s}} \mathbf{q}(s \rightarrow s')$.
- ▶ The **diagonal** of the generator matrix is $\mathbf{q}(s \rightarrow s) \stackrel{df}{=} -q(s)$.

LMA

The **transition rate due to action a** is $\mathbf{q}(s \xrightarrow{a} s') \stackrel{df}{=} \sum_{\lambda: s \xrightarrow{a,\lambda} s'} \lambda$.

The associated time-homogenous CTMC

Any CTMC is fully characterised by \mathbf{Q} , the generator matrix.

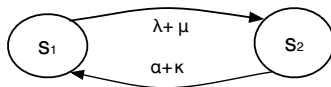
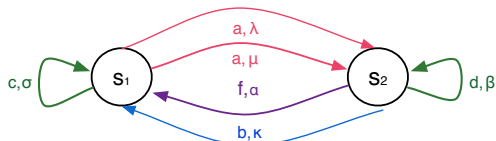
- ▶ The **transition rate** is $\mathbf{q}(s \rightarrow s') \stackrel{df}{=} \sum_{\substack{(a,\lambda): s \xrightarrow{a,\lambda} s' \\ s \neq s'}} \lambda$.
- ▶ The **total** (or exit) **rate** of a state s is $\mathbf{q}(s) \stackrel{df}{=} \sum_{\substack{s' \in \mathbf{S} \\ s' \neq s}} \mathbf{q}(s \rightarrow s')$.
- ▶ The **diagonal** of the generator matrix is $\mathbf{q}(s \rightarrow s) \stackrel{df}{=} -q(s)$.

LMA

The **transition rate due to action a** is $\mathbf{q}(s \xrightarrow{a} s') \stackrel{df}{=} \sum_{\lambda: s \xrightarrow{a,\lambda} s'} \lambda$.

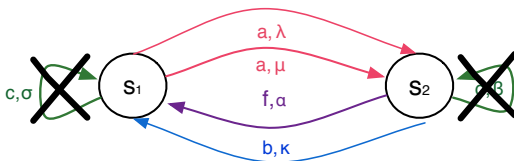
From LMAs to CTMCs

MARKOV AUTOMATON



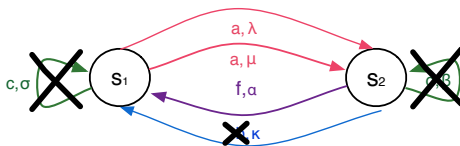
From LMAs to CTMCs

MARKOV AUTOMATON



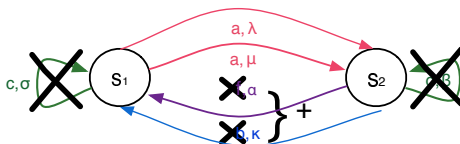
From LMAs to CTMCs

MARKOV AUTOMATON



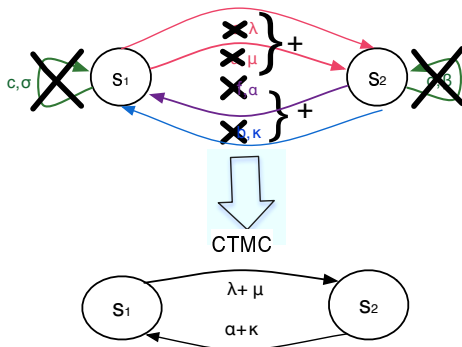
From LMAs to CTMCs

MARKOV AUTOMATON



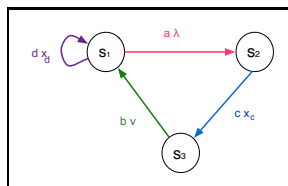
From LMAs to CTMCs

MARKOV AUTOMATON



Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle S, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$S = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

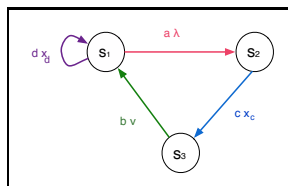
$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathcal{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

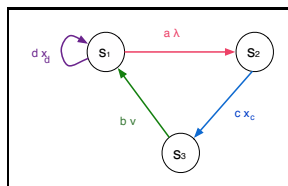
$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathcal{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

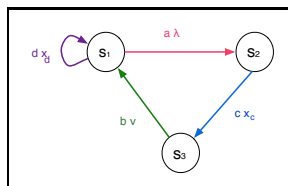
$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathcal{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

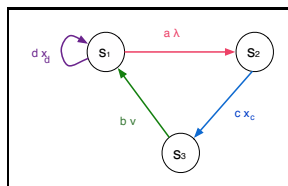
$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathcal{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

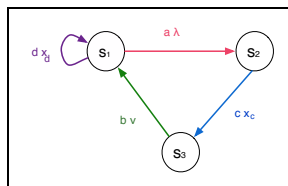
$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathbf{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

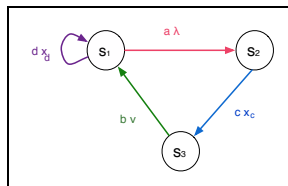
$$s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

A transition is **active** out of state s , if $s \xrightarrow{a, \lambda} s'$ for some $s' \in \mathbf{S}$ and $\lambda \in \mathbb{R}^+$.

Open automaton

Graph



Formal definition

$$\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$$

$$\mathbf{S} = \{s_1, s_2, s_3\} \quad \mathcal{A} = \{a, b\} \quad \mathcal{P} = \{c, d\}$$

$$\rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), \\ (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\}$$

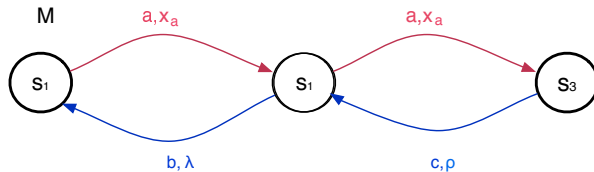
$$s_1 \xrightarrow{a, \lambda} s_2, \quad s_1 \xrightarrow{d, x_d} s_1, \quad s_2 \xrightarrow{c, x_c} s_3, \quad s_3 \xrightarrow{b, \nu} s_1.$$

Passive and active transitions

A transition is **passive** out of state s , if $s \xrightarrow{a, x_a} s'$ for some $s' \in \mathbf{S}$ and $(x_a \in \text{Var})$.

Open automaton and CTMC

Example of an open automaton

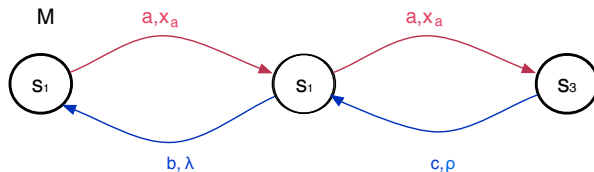


Problem

Can this *open* automaton be mapped to a CTMC ?

Open automaton and CTMC

Example of an open automaton



$$\underbrace{\mathcal{A} = \{b, c\}}_{\text{active labels}}$$

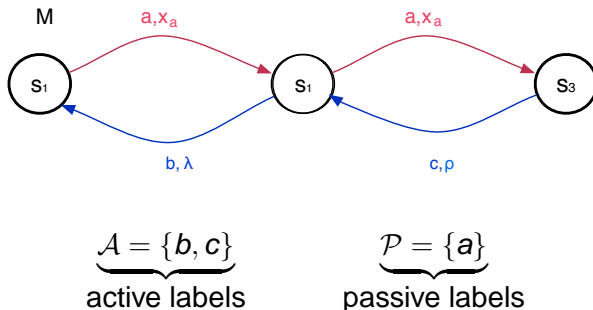
Problem

Can this *open* automaton be mapped to a CTMC?



Open automaton and CTMC

Example of an open automaton

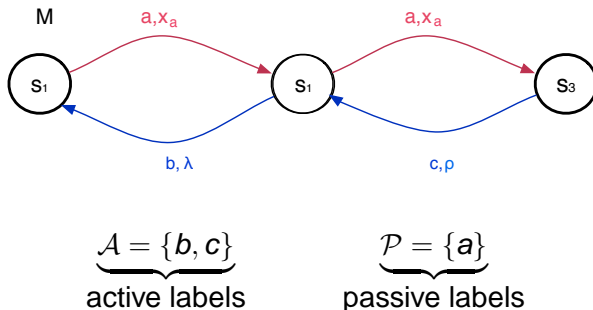


Problem

Can this *open* automaton be mapped to a CTMC?

Open automaton and CTMC

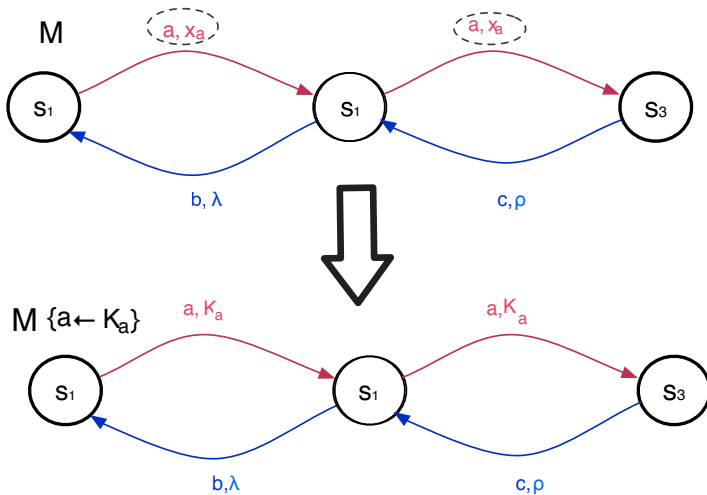
Example of an open automaton



Problem

Can this *open* automaton be mapped to a CTMC ?

Closure of open automaton



Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle \mathbf{S}, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} =$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle \mathbf{S}, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} =$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} =$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} =$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\}$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\} \\ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\}$$

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\} \\ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\}$$

Properties of closure

- ▶ Definition of closure for multiple labels is as expected.
- ▶ $\mathcal{M}\{p \leftarrow K_a : p \in \mathcal{P}\}$.

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\} \\ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\}$$

Properties of closure

- ▶ Definition of closure for multiple labels is as expected.
- ▶ $\mathcal{M}\{p \leftarrow K_a : p \in \mathcal{P}\}$.

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\} \\ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\}$$

Properties of closure

- ▶ Definition of closure for multiple labels is as expected.
- ▶ $\mathcal{M}\{p \leftarrow K_a : p \in \mathcal{P}\}$.

Closure of open automaton

Single label closure

$\mathcal{M}\{a \leftarrow K_a\} \stackrel{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} \rangle$ is the *closure* of the automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle$

$$\rightarrow_{\mathcal{M}\{a \leftarrow K_a\}} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in \mathcal{A}\} \\ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\}$$

Properties of closure

- ▶ The order in which the closures are performed is irrelevant.
- ▶ A **closed automaton** has any passive transition.

Automata Interacting

$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \mathbf{Act}, \rightarrow \rangle$
 is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\mathbf{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \qquad s_k \xrightarrow{a, x_k}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_j, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_j, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \qquad s_k \xrightarrow{a, x_k}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_j, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_j, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$
 is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \qquad s_k \xrightarrow{a, x_k}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_j, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_j, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \mathbf{Act}, \rightarrow \rangle$$

is the **interacting LMA**:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\mathbf{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \qquad s_k \xrightarrow{a, x_k}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_j, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_j, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_j \xrightarrow{a, \lambda}_i s'_j \quad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \mathbf{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\mathbf{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \qquad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_j \xrightarrow{a, \lambda}_i s'_j \quad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_i \xrightarrow{a, \lambda}_i s'_i \quad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_j \xrightarrow{a, \lambda}_i s'_j \qquad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \mathbf{Act}, \rightarrow \rangle$
 is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\mathbf{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

$$\frac{s_j \xrightarrow{a, \lambda}_i s'_j \qquad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

Automata Interacting

$$\bigoplus_L(\mathcal{M}_1, \dots, \mathcal{M}_i, \dots, \mathcal{M}_k, \dots, \mathcal{M}_n) = \langle \mathbf{S}, \text{Act}, \rightarrow \rangle$$

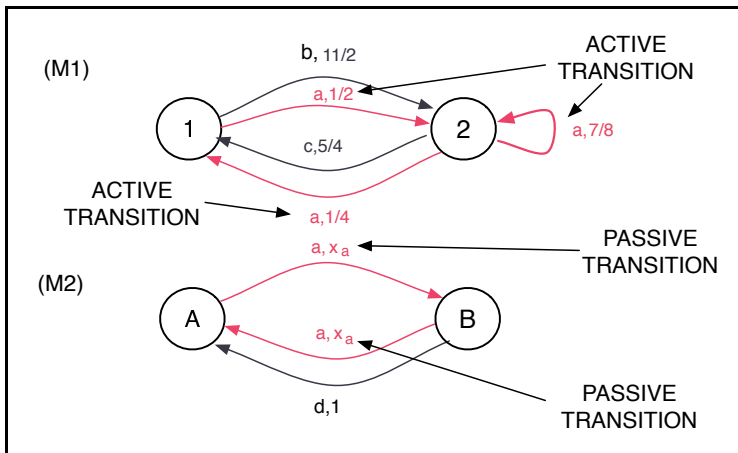
is the interacting LMA:

- ▶ $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2 \times \dots \times \mathbf{S}_n$.
- ▶ $\text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i)$ and $\mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset$.
- ▶ \rightarrow is the smallest relation defined by:

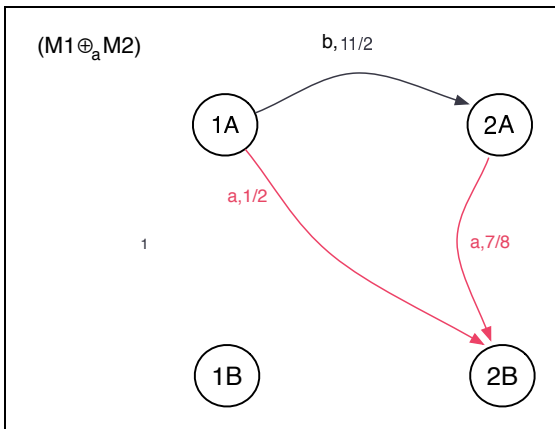
$$\frac{s_j \xrightarrow{a, \lambda}_i s'_j \qquad s_k \xrightarrow{a, x_a}_k s'_k}{(s_1, \dots, s_j, \dots, s_k, \dots, s_n) \xrightarrow{a, \lambda} (s_1, \dots, s'_j, \dots, s'_k, \dots, s_n)} \quad (a \in L)$$

$$\frac{s_i \xrightarrow{a, r}_i s'_i}{(s_1, \dots, s'_i, \dots, s_n) \xrightarrow{a, r} (s_1, \dots, s'_i, \dots, s_n)} \quad (a \notin L)$$

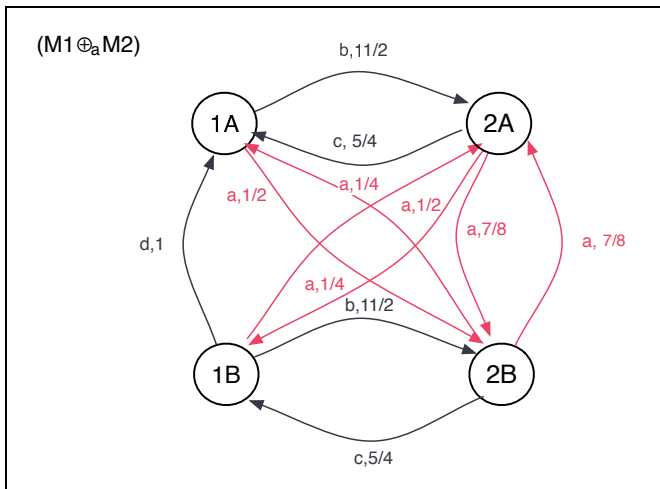
Example of interacting automata



Example of cooperating automata (three steps)



Example of cooperating automata (cont.)



Well-formed automaton

An automaton $\mathcal{M} = \langle \mathbf{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is well formed if:

1. For any label $a \in (\mathcal{A} \cup \mathcal{P})$ the transitions labeled a are either active or passive.

$$\mathcal{A} \cap \mathcal{P} = \emptyset.$$

2. If $a \in \mathcal{P}$ then for every state s of the automaton there exists **exactly** one passive transition a out of state s .

$$\forall a \in \mathcal{P} \forall s \exists s' \in \mathbf{S} \text{ such that } s \xrightarrow{a, x_a} s'$$

$$\forall s, s', s'' \in \mathbf{S}, s \xrightarrow{a, x_a} s' \wedge s \xrightarrow{a, x_a} s'' \implies s' = s''.$$

Well-formed automaton

An automaton $\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is well formed if:

1. For any label $a \in (\mathcal{A} \cup \mathcal{P})$ the transitions labeled a are either active or passive.

$$\mathcal{A} \cap \mathcal{P} = \emptyset.$$

2. If $a \in \mathcal{P}$ then for every state s of the automaton there exists **exactly** one passive transition a out of state s .

$$\forall a \in \mathcal{P} \forall s \exists s' \in \mathcal{S} \text{ such that } s \xrightarrow{a, x_a} s'$$

$$\forall s, s', s'' \in \mathcal{S}, s \xrightarrow{a, x_a} s' \wedge s \xrightarrow{a, x_a} s'' \implies s' = s''.$$

Well-formed automaton

An automaton $\mathcal{M} = \langle \mathcal{S}, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is well formed if:

1. For any label $a \in (\mathcal{A} \cup \mathcal{P})$ the transitions labeled a are either active or passive.

$$\mathcal{A} \cap \mathcal{P} = \emptyset.$$

2. If $a \in \mathcal{P}$ then for every state s of the automaton there exists **exactly** one passive transition a out of state s .

$$\forall a \in \mathcal{P} \forall s \exists s' \in \mathcal{S} \text{ such that } s \xrightarrow{a, x_a} s'$$

$$\forall s, s', s'' \in \mathcal{S}, s \xrightarrow{a, x_a} s' \wedge s \xrightarrow{a, x_a} s'' \implies s' = s''.$$

Well-formed automaton

An automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is well formed if:

1. For any label $a \in (\mathcal{A} \cup \mathcal{P})$ the transitions labeled a are either active or passive.

$$\mathcal{A} \cap \mathcal{P} = \emptyset.$$

2. If $a \in \mathcal{P}$ then for every state s of the automaton there exists **exactly** one passive transition a out of state s .

$$\forall a \in \mathcal{P} \forall s \exists s' \in S \text{ such that } s \xrightarrow{a, x_a} s'$$

$$\forall s, s', s'' \in S, s \xrightarrow{a, x_a} s' \wedge s \xrightarrow{a, x_a} s'' \implies s' = s''.$$

Well-formed automaton

An automaton $\mathcal{M} = \langle S, \mathcal{A} \cup \mathcal{P}, \rightarrow \rangle$ is well formed if:

1. For any label $a \in (\mathcal{A} \cup \mathcal{P})$ the transitions labeled a are either active or passive.

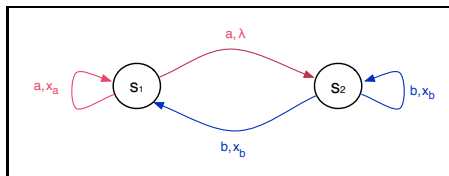
$$\mathcal{A} \cap \mathcal{P} = \emptyset.$$

2. If $a \in \mathcal{P}$ then for every state s of the automaton there exists **exactly** one passive transition a out of state s .

$$\forall a \in \mathcal{P} \forall s \exists s' \in S \text{ such that } s \xrightarrow{a, x_a} s'$$

$$\forall s, s', s'' \in S, s \xrightarrow{a, x_a} s' \wedge s \xrightarrow{a, x_a} s'' \implies s' = s''.$$

Non-well-formed automaton

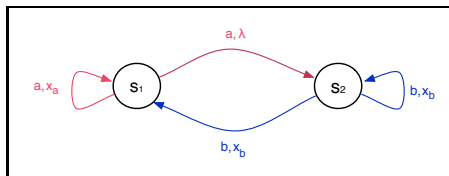


Violation Condition (1) The label a is both active and passive.

Violation Condition (2) Two passive transitions labelled b out of s_2 . No passive transitions labelled a out of s_2 .

Violation Condition (2) No passive transitions labelled b out of state s_1 .

Non-well-formed automaton

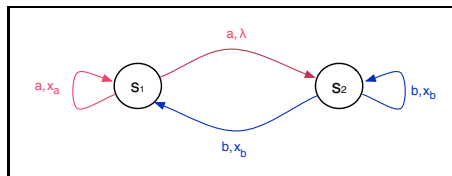


Violation Condition (1) The label a is **both active and passive**.

Violation Condition (2) **Two passive** transitions labelled b out of s_2 . **No passive** transitions labelled a out of s_2 .

Violation Condition (2) **No passive** transitions labelled b out of state s_1 .

Non-well-formed automaton

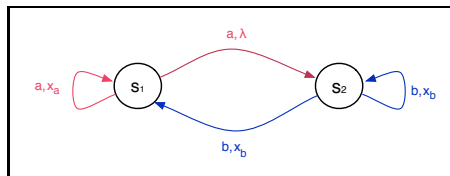


Violation Condition (1) The label a is **both active and passive**.

Violation Condition (2) **Two passive** transitions labelled b out of s_2 . **No passive** transitions labelled a out of s_2 .

Violation Condition (2) **No passive** transitions labelled b out of state s_1 .

Non-well-formed automaton

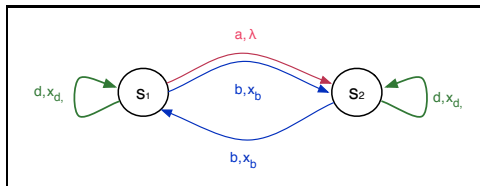


Violation Condition (1) The label a is **both active and passive**.

Violation Condition (2) **Two passive** transitions labelled b out of s_2 . **No passive** transitions labelled a out of s_2 .

Violation Condition (2) **No passive** transitions labelled b out of state s_1 .

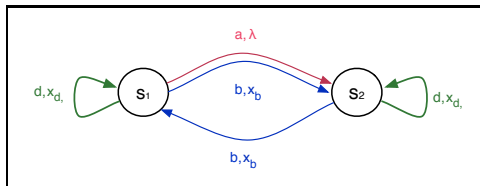
Well-formed automaton



Condition (1) Since $\mathcal{A} = \{a\}$ and $\mathcal{P} = \{b, d\}$, therefore $\mathcal{A} \cap \mathcal{P} = \emptyset$.

Condition (2) There is exactly one transition out of each state s_1, s_2 with label either b or d .

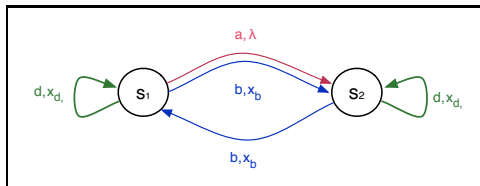
Well-formed automaton



Condition (1) Since $\mathcal{A} = \{a\}$ and $\mathcal{P} = \{b, d\}$, therefore $\mathcal{A} \cap \mathcal{P} = \emptyset$.

Condition (2) There is exactly one transition out of each state s_1, s_2 with label either b or d .

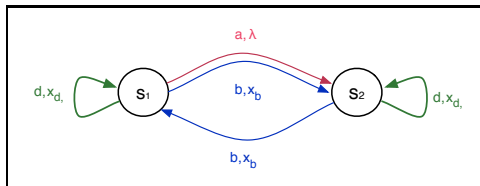
Well-formed automaton



Condition (1) Since $\mathcal{A} = \{a\}$ and $\mathcal{P} = \{b, d\}$, therefore
 $\mathcal{A} \cap \mathcal{P} = \emptyset$. ✓

Condition (2) There is exactly one transition out of each state s_1, s_2 with label either b or d .

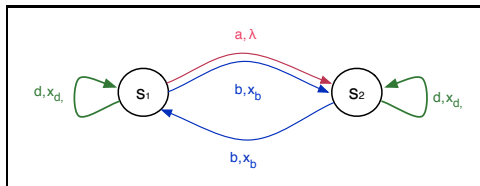
Well-formed automaton



Condition (1) Since $\mathcal{A} = \{a\}$ and $\mathcal{P} = \{b, d\}$, therefore $\mathcal{A} \cap \mathcal{P} = \emptyset$. ✓

Condition (2) There is exactly one transition out of each state s_1, s_2 with label either b or d .

Well-formed automaton



Condition (1) Since $\mathcal{A} = \{a\}$ and $\mathcal{P} = \{b, d\}$, therefore $\mathcal{A} \cap \mathcal{P} = \emptyset$. ✓

Condition (2) There is exactly one transition out of each state s_1, s_2 with label either b or d . ✓

Trivial product-form solution

- ▶ Two (or more) closed automata are independent, if they do not interact, i.e. if there is no set of co-operating transitions among components ...

$$\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2$$

- ▶ Trivially, $\pi(\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2)$.
- ▶ For the rest of the tutorial we shall consider non-trivial interactions between (among various) automata, i.e.

$$\mathcal{M}_1 \oplus_L \mathcal{M}_2 \quad L \neq \emptyset$$

Trivial product-form solution

- ▶ Two (or more) closed automata are independent, if they do not interact, i.e. if there is no set of co-operating transitions among components ...

$$\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2$$

- ▶ Trivially, $\pi(\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2)$.
- ▶ For the rest of the tutorial we shall consider non-trivial interactions between (among various) automata, i.e.

$$\mathcal{M}_1 \oplus_L \mathcal{M}_2 \quad L \neq \emptyset$$

Trivial product-form solution

- ▶ Two (or more) closed automata are independent, if they do not interact, i.e. if there is no set of co-operating transitions among components ...

$$\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2$$

- ▶ Trivially, $\pi(\mathcal{M}_1 \oplus_{\emptyset} \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2)$.
- ▶ For the rest of the tutorial we shall consider non-trivial interactions between (among various) automata, i.e.

$$\mathcal{M}_1 \oplus_L \mathcal{M}_2 \quad L \neq \emptyset$$

GRCAT

Assumption 1

$\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N$ are well formed and the global state space \mathcal{S} of the automaton $\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)$ is irreducible.

Assumption 2

For each label $L = \{a_1, a_2, \dots, a_M\}$ there exists a set of rates $\{K_{a_1}, \dots, K_{a_M}\}$ such that:

$$\forall a \in (\mathcal{A}_i \cap L), \forall s \in \mathcal{S}_i, 1 \leq i \leq N \quad \frac{\sum_{s' \in \mathcal{S}_i} q(s' \xrightarrow{a} s) \pi_i(s')}{\pi_i(s)} = K_a \quad (3)$$

GRCAT

Assumption 1

$\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N$ are well formed and the global state space \mathbf{S} of the automaton $\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)$ is irreducible.

Assumption 2

For each label $L = \{a_1, a_2, \dots, a_M\}$ there exists a set of rates $\{K_{a_1}, \dots, K_{a_M}\}$ such that:

$$\forall a \in (\mathcal{A}_i \cap L), \forall s \in \mathbf{S}_i, 1 \leq i \leq N \quad \frac{\sum_{s' \in \mathbf{S}_i} q(s' \xrightarrow{a} s) \pi_i(s')}{\pi_i(s)} = K_a \quad (3)$$

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^s) = 1$ then $\pi(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N))$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N))$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L (\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

$\mathcal{M}_i^c = \mathcal{M}_i\{a \leftarrow K_a : a \in \mathcal{P}_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi(\bigoplus_L (\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N))$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

$\mathcal{M}_i^c = \mathcal{M}_i\{a \leftarrow K_a : a \in \mathcal{P}_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N))$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

$\mathcal{M}_i^c = \mathcal{M}_i\{a \leftarrow K_a : a \in \mathcal{P}_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right)$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

$\mathcal{M}_i^c = \mathcal{M}_i\{a \leftarrow K_a : a \in \mathcal{P}_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right)$ is the steady-state distribution.

GRCAT

Stationary measure

Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right) = \prod_{i=1}^N \pi_i(\mathcal{M}_i^c)$$

$\mathcal{M}_i^c = \mathcal{M}_i\{a \leftarrow K_a : a \in \mathcal{P}_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi\left(\bigoplus_L(\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_N)\right)$ is the **steady-state distribution**.

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The set of states is bounded (finite).
 3. The transition is constant.
 4. The set of states is bounded (finite).
 5. The transition is constant.

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (system \rightarrow components).

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (system \rightarrow components).

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (system \rightarrow components).

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (system \rightarrow components).

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (**components** → **system**) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (system → components).

In summary....

- ▶ We have defined the notion of **component** as **Markov labelled automaton**.
- ▶ The steady-state distribution of interacting LMA can be written as the product of steady-state distributions of each single automata (components \rightarrow system) provided that
 1. Each automata is well formed.
 2. The total incoming 'weighted flux' into each state due to an active transition is constant.
 3. The constant 'weighted flux' is the new rate of the passive transition of each single open automata (**system \rightarrow components**).

Two-node Jackson network

Can we model well-known product-form solutions?

- ▶ We show how to model Jackson networks.
- ▶ For the purpose of the tutorial we shall consider a two-node network, but generalisation to any finite number of nodes is possible.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a routing probability r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a routing probability r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a routing probability r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a routing probability r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a **routing probability** r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

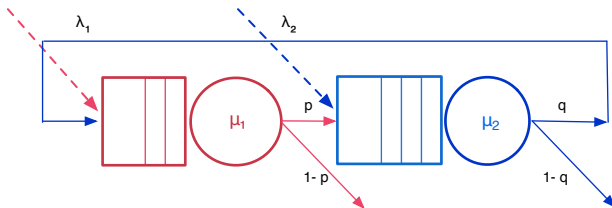
- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a **routing probability** r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Jackson networks in queueing theory

In queueing theory each node in an open Jackson network is specified as follows:

- ▶ The time between arrivals is exponentially distributed with constant rate.
- ▶ Service is exponentially distributed with constant rate.
- ▶ There is no limit on queue capacity in any node.
- ▶ Customers are selected on a first-come first-serve discipline.
- ▶ After being served, customers in node j join another queue k with a **routing probability** r_{jk} , which is independent from the state of the network.
- ▶ After service, customers at node j leave the network with probability r_{j0} (0 is the *outside node*).
- ▶ For each node j , $\sum_l r_{jl} = 1$.

Two-node Jackson network



Arrival Process Arrival rate: λ_1 .

Service Process Service rate: μ_1 .

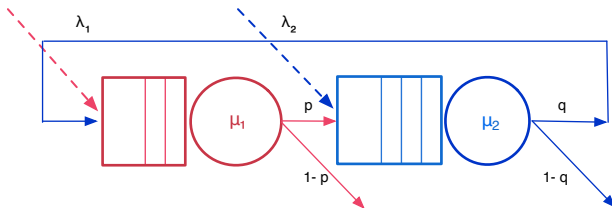
Routing $r_{1,2} = p$ $r_{1,0} = 1 - p$.

Arrival Process Arrival rate: λ_2 .

Service Process Service rate: μ_2 .

Routing $r_{2,1} = q$ $r_{2,0} = 1 - q$.

Two-node Jackson network



Arrival Process Arrival rate: λ_1 .

Service Process Service rate: μ_1 .

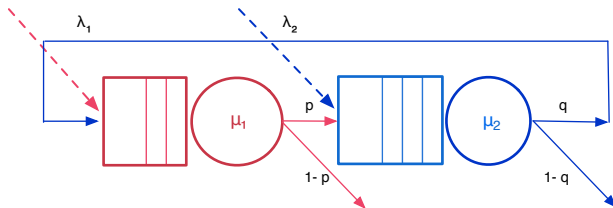
Routing $r_{1,2} = p$ $r_{1,0} = 1 - p$.

Arrival Process Arrival rate: λ_2 .

Service Process Service rate: μ_2 .

Routing $r_{2,1} = q$ $r_{2,0} = 1 - q$.

Two-node Jackson network



Arrival Process Arrival rate: λ_1 .

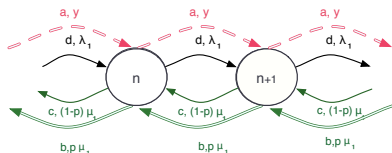
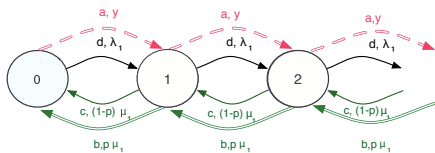
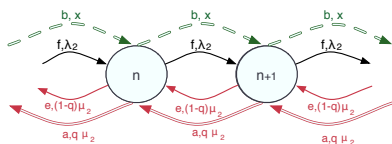
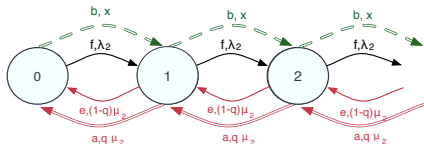
Service Process Service rate: μ_1 .

Routing $r_{1,2} = p$ $r_{1,0} = 1 - p$.

Arrival Process Arrival rate: λ_2 .

Service Process Service rate: μ_2 .

Routing $r_{2,1} = q$ $r_{2,0} = 1 - q$.

\mathcal{M}_1  \mathcal{M}_2 

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset$ ✓.
- ▶ Only one passive action a out of each state n ✓.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$
- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in \mathcal{S}_1$

$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in \mathcal{S}_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset$ ✓.
- ▶ Only one passive action a out of each state n ✓.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$
- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in \mathcal{S}_1$

$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in \mathcal{S}_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset$ ✓.
- ▶ Only one passive action a out of each state n ✓.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$
- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in \mathcal{S}_1$
 $\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1)$.
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in \mathcal{S}_2$
 $\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1)$.

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset$ ✓.
- ▶ Only one passive action a out of each state n ✓.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$
- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in S_1$

$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in S_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset$ ✓.
- ▶ Only one passive action a out of each state n ✓.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$
- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in S_1$

$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in S_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \quad \checkmark$.
- ▶ Only one passive action a out of each state $n \quad \checkmark$.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$

Verification that the “weighted fluxes” are constant:

- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in S_1$
 $\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1)$.
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in S_2$
 $\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1)$.

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \quad \checkmark$.
- ▶ Only one passive action a out of each state $n \quad \checkmark$.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$

Verification that the “weighted fluxes” are constant:

- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in S_1$

$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in S_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

Product-form solution

$$JN = \mathcal{M}_1 \oplus_{\{a,b\}} \mathcal{M}_2$$

Consider the automaton \mathcal{M}_1 , we verify:

- ▶ $\mathcal{A}_1 = \{c, b, d\}$, $\mathcal{P}_1 = \{a\}$ and $\mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \quad \checkmark$.
- ▶ Only one passive action a out of each state $n \quad \checkmark$.
- ▶ Similar reasoning for \mathcal{M}_2 . $\mathcal{A}_2 = \{e, f, a\}$ and $\mathcal{P}_2 = \{b\}$

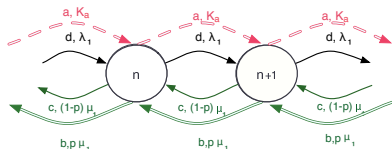
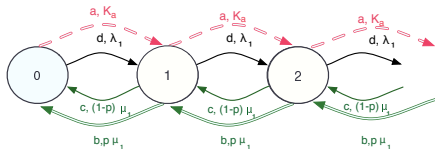
Verification that the “weighted fluxes” are constant:

- ▶ $b \in \{a, b\} \cap \mathcal{A}_1$ then for all $n \in \mathcal{S}_1$

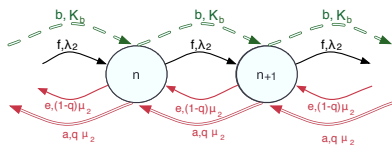
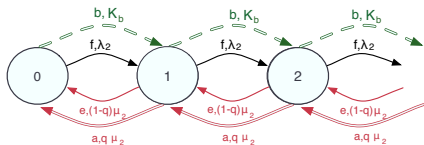
$$\frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1).$$
- ▶ $a \in \{a, b\} \cap \mathcal{A}_2$ then for all $n \in \mathcal{S}_2$

$$\frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1).$$

$$\mathcal{M}_1\{a \leftarrow K_a\}$$



$$\mathcal{M}_2\{b \leftarrow K_b\}$$



Traffic equations

The transition rates of arrival into each state for each closed LMA are:

$$\begin{aligned}
 \mathbf{q}_1(n \rightarrow n+1) &= \lambda_1 + y \\
 &= \lambda_1 + \mathbf{q} \mathbf{q}_2(n \rightarrow n+1) \\
 \mathbf{q}_2(n \rightarrow n+1) &= \lambda_2 + x \\
 &= \lambda_2 + \mathbf{p} \mathbf{q}_1(n \rightarrow n+1)
 \end{aligned}$$

where $y = \mathbf{q}_1(n \xrightarrow{a} n+1)$ and $x = \mathbf{q}_2(n \xrightarrow{b} n+1)$ for $n \geq 0$.

The equations above are exactly the traffic equations for Jackson network; they represent the mean arrival rate into each queue.

Traffic equations

The transition rates of arrival into each state for each closed LMA are:

$$\begin{aligned}
 \mathbf{q}_1(n \rightarrow n+1) &= \lambda_1 + y \\
 &= \lambda_1 + \mathbf{q}\mathbf{q}_2(n \rightarrow n+1) \\
 \mathbf{q}_2(n \rightarrow n+1) &= \lambda_2 + x \\
 &= \lambda_2 + \mathbf{p}\mathbf{q}_1(n \rightarrow n+1)
 \end{aligned}$$

where $y = \mathbf{q}_1(n \xrightarrow{a} n+1)$ and $x = \mathbf{q}_2(n \xrightarrow{n} n+1)$ for $n \geq 0$.

The equations above are exactly the traffic equations for Jackson network; they represent the mean arrival rate into each queue.

Traffic equations

The transition rates of arrival into each state for each closed LMA are:

$$\begin{aligned}
 \mathbf{q}_1(n \rightarrow n+1) &= \lambda_1 + y \\
 &= \lambda_1 + \mathbf{q}\mathbf{q}_2(n \rightarrow n+1) \\
 \mathbf{q}_2(n \rightarrow n+1) &= \lambda_2 + x \\
 &= \lambda_2 + \mathbf{p}\mathbf{q}_1(n \rightarrow n+1)
 \end{aligned}$$

where $y = \mathbf{q}_1(n \xrightarrow{a} n+1)$ and $x = \mathbf{q}_2(n \xrightarrow{n} n+1)$ for $n \geq 0$.

The equations above are exactly the **traffic equations for Jackson network**; they represent the mean arrival rate into each queue.

Concluding two-node Jackson network

Normalising equation

$$\frac{\mathbf{q}_i(n \rightarrow n+1)}{\mathbf{q}_i(n+1 \rightarrow n)} < 1 \quad n \geq 0, i = 1, 2$$

then the network has a steady-state distribution.

Solution to traffic equations

Existence of a solution or an algorithm computing the solution of the traffic equations is outside the scope of GRCAT or ERCAT.

Concluding two-node Jackson network

Normalising equation

$$\frac{\mathbf{q}_i(n \rightarrow n+1)}{\mathbf{q}_i(n+1 \rightarrow n)} < 1 \quad n \geq 0, i = 1, 2$$

then the network has a steady-state distribution.

Solution to traffic equations

Existence of a solution or an algorithm computing the solution of the traffic equations is outside the scope of GRCAT or ERCAT.

Concluding two-node Jackson network

Normalising equation

$$\frac{\mathbf{q}_i(n \rightarrow n+1)}{\mathbf{q}_i(n+1 \rightarrow n)} < 1 \quad n \geq 0, i = 1, 2$$

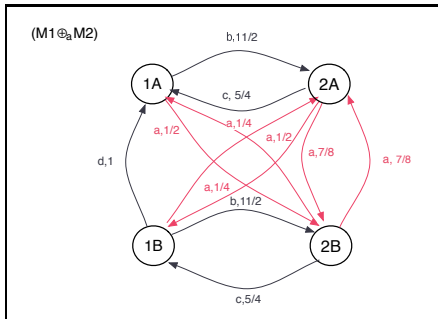
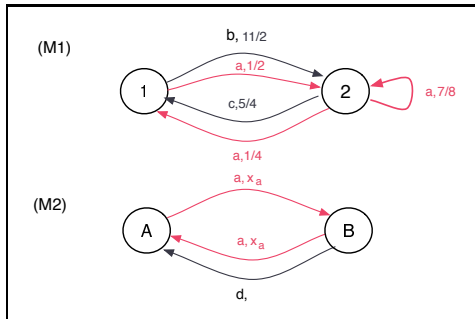
then the network has a steady-state distribution.

Solution to traffic equations

Existence of a solution or an algorithm computing the solution of the **traffic equations** is outside the scope of GRCAT or ERCAT.

Product form for any CTMC

Our method(s) can be applied to any CTMC.



Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

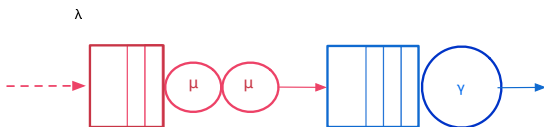
- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{x_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{x_a \leftarrow 1\})$.

Product-form of $\pi(\mathcal{M}_1, \mathcal{M}_2)$

- ▶ We observe that \mathcal{M}_1 is closed.
- ▶ Steady-state probabilities of $\pi_1(\mathcal{M}_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- ▶ $K_a = 1$
- ▶ $\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\}$.
- ▶ Steady-state probabilities of $\pi_2(\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- ▶ $\pi(\mathcal{M}_1, \mathcal{M}_2) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{\mathbf{x}_a \leftarrow 1\})$.

Mixed network

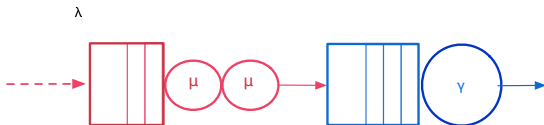
Two-node Erlang network



- ▶ First node two-phase exponential service rate μ .
- ▶ Second node ordinary M/M/1 queue.
- ▶ There is no product-form solution.

Mixed network

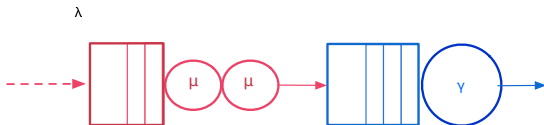
Two-node Erlang network



- ▶ First node two-phase exponential service rate μ .
- ▶ Second node ordinary M/M/1 queue.
- ▶ There is no product-form solution.

Mixed network

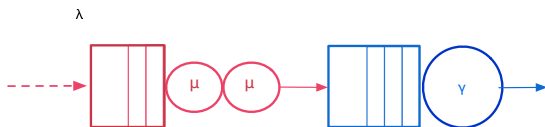
Two-node Erlang network



- ▶ First node two-phase exponential service rate μ .
- ▶ Second node ordinary M/M/1 queue.
- ▶ There is no product-form solution.

Mixed network

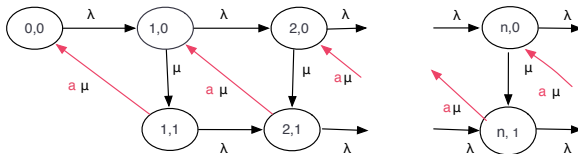
Two-node Erlang network



- ▶ First node two-phase exponential service rate μ .
- ▶ Second node ordinary M/M/1 queue.
- ▶ There is no product-form solution.

Erlang node as LMA

ER_1



$$\blacktriangleright \sum_{s \in \mathbf{S}} \frac{\pi_1(s) \mathbf{q}(s \xrightarrow{a} (n,1))}{\pi_1(n,1)} = 0 .$$

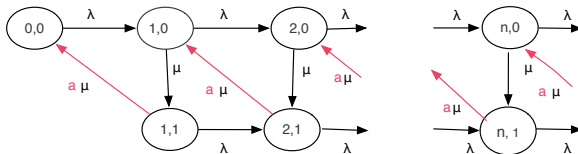
$$\blacktriangleright \frac{\pi_1(n+1,1) \mathbf{q}((n+1,1) \xrightarrow{a} (n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1) \mu}{\pi_1(n,0)} > 0 .$$

\blacktriangleright Equation (3) of GRCAT cannot be satisfied.

\blacktriangleright We can remedy by making minimal changes to the queue.

Erlang node as LMA

ER_1



$$\blacktriangleright \sum_{s \in \mathbf{S}} \frac{\pi_1(s) \mathbf{q}(s \xrightarrow{a} (n, 1))}{\pi_1(n, 1)} = 0 .$$

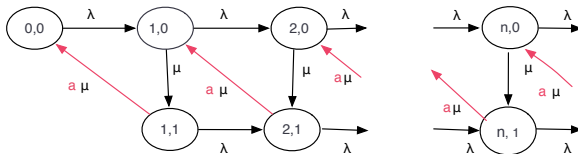
$$\blacktriangleright \frac{\pi_1(n+1, 1) \mathbf{q}((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = \frac{\pi_1(n+1, 1) \mu}{\pi_1(n, 0)} > 0 .$$

▶ Equation (3) of GRCAT cannot be satisfied.

▶ We can remedy by making minimal changes to the queue.

Erlang node as LMA

ER_1



$$\blacktriangleright \sum_{s \in \mathbf{S}} \frac{\pi_1(s) \mathbf{q}(s \xrightarrow{a} (n,1))}{\pi_1(n,1)} = 0 .$$

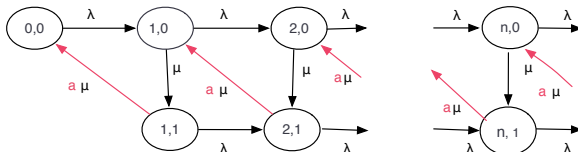
$$\blacktriangleright \frac{\pi_1(n+1,1) \mathbf{q}((n+1,1) \xrightarrow{a} (n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1) \mu}{\pi_1(n,0)} > 0 .$$

▶ Equation (3) of GRCAT cannot be satisfied.

▶ We can remedy by making minimal changes to the queue.

Erlang node as LMA

ER_1



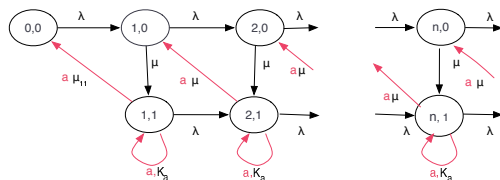
$$\blacktriangleright \sum_{s \in \mathbf{S}} \frac{\pi_1(s) \mathbf{q}(s \xrightarrow{a} (n,1))}{\pi_1(n,1)} = 0 .$$

$$\blacktriangleright \frac{\pi_1(n+1,1) \mathbf{q}((n+1,1) \xrightarrow{a} (n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1) \mu}{\pi_1(n,0)} > 0 .$$

- ▶ Equation (3) of GRCAT cannot be satisfied.
- ▶ We can remedy by making minimal changes to the queue.

Perturbed Erlang queue as LMA

ER_1



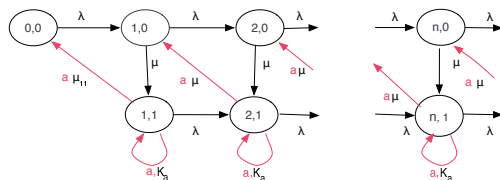
- ▶ Introduced self-loops in states $(n, 0)$ with rate K_a .
- ▶ Changed $q((1, 1) \xrightarrow{a} (0, 0)) = \mu_{11}$.
- ▶ Find steady-state probabilities imposing

$$\frac{\pi_1(n+1, 1)q((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = K_a$$

- ▶ The steady-state distribution exists - see [Harrison-Vigliotti

Perturbed Erlang queue as LMA

ER_1



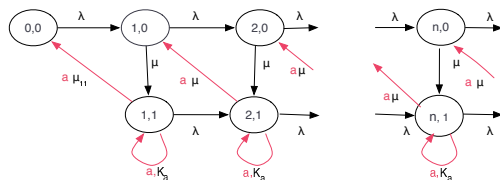
- ▶ Introduced self-loops in states $(n, 0)$ with rate K_a .
- ▶ Changed $q((1, 1) \xrightarrow{a} (0, 0)) = \mu_{11}$.
- ▶ Find steady-state probabilities imposing

$$\frac{\pi_1(n+1, 1)q((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = K_a$$

- ▶ The steady-state distribution exists - see [Harrison-Vigliotti

Perturbed Erlang queue as LMA

ER_1



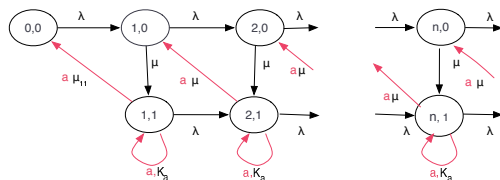
- ▶ Introduced self-loops in states $(n, 0)$ with rate K_a .
- ▶ Changed $\mathbf{q}((1, 1) \xrightarrow{a} (0, 0)) = \mu_{11}$.
- ▶ Find steady-state probabilities imposing

$$\frac{\pi_1(n+1, 1) \mathbf{q}((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = K_a$$

- ▶ The steady-state distribution exists - see [Harrison-Vigliotti

Perturbed Erlang queue as LMA

ER_1



- ▶ Introduced self-loops in states $(n, 0)$ with rate K_a .
- ▶ Changed $\mathbf{q}((1, 1) \xrightarrow{a} (0, 0)) = \mu_{11}$.
- ▶ Find steady-state probabilities imposing

$$\frac{\pi_1(n+1, 1) \mathbf{q}((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = K_a$$

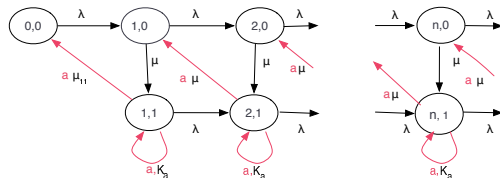
- ▶ The steady-state distribution exists - see [Harrison-Vigliotti

'09]



Perturbed Erlang queue as LMA

ER_1



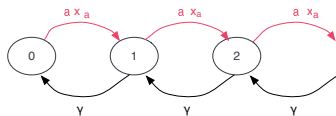
- ▶ Introduced self-loops in states $(n, 0)$ with rate K_a .
- ▶ Changed $\mathbf{q}((1, 1) \xrightarrow{a} (0, 0)) = \mu_{11}$.
- ▶ Find steady-state probabilities imposing

$$\frac{\pi_1(n+1, 1)\mathbf{q}((n+1, 1) \xrightarrow{a} (n, 0))}{\pi_1(n, 0)} = K_a$$

- ▶ The steady-state distribution exists - see [Harrison-Vigliotti '09]

Product-form solution

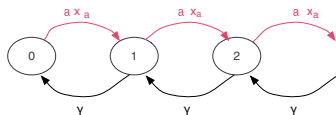
MM1



- ▶ The two automata ER_1 and $MM1$ are well formed (exercise).
- ▶ $MM1\{a \leftarrow K_a\}$ is closed and steady-state distribution $\pi_2(\cdot)$ exist only if $\frac{K_a}{\gamma} < 1$.
- ▶ The network $ER_1 \oplus_a MM1\{a \leftarrow K_a\}$ has product-form solution: equation (3) of GRCAT is satisfied because it was built this way!!!

Product-form solution

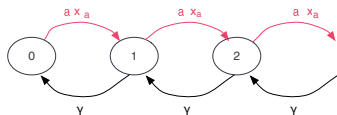
MM1



- ▶ The two automata ER_1 and $MM1$ are well formed (exercise).
- ▶ $MM1 \{a \leftarrow K_a\}$ is closed and steady-state distribution $\pi_2(\cdot)$ exist only if $\frac{K_a}{\gamma} < 1$.
- ▶ The network $ER_1 \oplus_a MM1 \{a \leftarrow K_a\}$ has product-form solution: equation (3) of GRCAT is satisfied because it was built this way!!!

Product-form solution

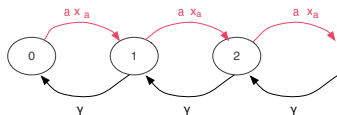
MM1



- ▶ The two automata ER_1 and $MM1$ are well formed (exercise).
- ▶ $MM1\{a \leftarrow K_a\}$ is closed and steady-state distribution $\pi_2(\cdot)$ exist only if $\frac{K_a}{\gamma} < 1$.
- ▶ The network $ER_1 \oplus_a MM1\{a \leftarrow K_a\}$ has product-form solution: equation (3) of GRCAT is satisfied because it was built this way!!!

Product-form solution

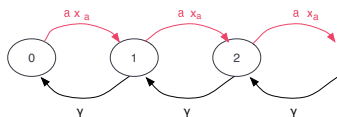
MM1



- ▶ The two automata ER_1 and $MM1$ are well formed (exercise).
- ▶ $MM1\{a \leftarrow K_a\}$ is closed and steady-state distribution $\pi_2(\cdot)$ exist only if $\frac{K_a}{\gamma} < 1$.
- ▶ The network $ER_1 \oplus_a MM1\{a \leftarrow K_a\}$ has product-form solution: equation (3) of GRCAT is satisfied because it was built this way!!!

Product-form solution

MM1



- ▶ The two automata ER_1 and $MM1$ are well formed (exercise).
- ▶ $MM1\{a \leftarrow K_a\}$ is closed and steady-state distribution $\pi_2(\cdot)$ exist only if $\frac{K_a}{\gamma} < 1$.
- ▶ The network $ER_1 \oplus_a MM1\{a \leftarrow K_a\}$ has product-form solution: equation (3) of GRCAT is satisfied because it was built this way!!!

$$\pi(ER_1 \oplus_a MM1\{a \leftarrow K_a\}) = \pi_1(ER)\pi_2(MM1).$$

Conclusion

- ▶ We have presented a general result for product-form solutions in Markovian models.
- ▶ New product-form solutions in queues can be found - see [Balbo-Vigliotti '10].
- ▶ The conditions are general enough to capture a very large class of models that are not derived from queueing theory.
- ▶ Stochastic automata make product-form solution amenable to model checking - for steady-state distribution.
- ▶ Standard related state reduction techniques such as lump-ability can be used as well.

Conclusion

- ▶ We have presented a general result for product-form solutions in Markovian models.
- ▶ New product-form solutions in queues can be found - see [Balbo-Vigliotti '10].
- ▶ The conditions are general enough to capture a very large class of models that are not derived from queueing theory.
- ▶ Stochastic automata make product-form solution amenable to model checking - for steady-state distribution.
- ▶ Standard related state reduction techniques such as lump-ability can be used as well.

Conclusion

- ▶ We have presented a general result for product-form solutions in Markovian models.
- ▶ New product-form solutions in queues can be found - see [Balbo-Vigliotti '10].
- ▶ The conditions are general enough to capture a very large class of models that are not derived from queueing theory.
- ▶ Stochastic automata make product-form solution amenable to model checking - for steady-state distribution.
- ▶ Standard related state reduction techniques such as lump-ability can be used as well.

Conclusion

- ▶ We have presented a general result for product-form solutions in Markovian models.
- ▶ New product-form solutions in queues can be found - see [Balbo-Vigliotti '10].
- ▶ The conditions are general enough to capture a very large class of models that are not derived from queueing theory.
- ▶ Stochastic automata make product-form solution amenable to model checking - for steady-state distribution.
- ▶ Standard related state reduction techniques such as lump-ability can be used as well.

Conclusion

- ▶ We have presented a general result for product-form solutions in Markovian models.
- ▶ New product-form solutions in queues can be found - see [Balbo-Vigliotti '10].
- ▶ The conditions are general enough to capture a very large class of models that are not derived from queueing theory.
- ▶ Stochastic automata make product-form solution amenable to model checking - for steady-state distribution.
- ▶ Standard related state reduction techniques such as lump-ability can be used as well.

Further reading

- ▶ Andrea Marin and Maria G. Vigliotti."A General Result for Deriving Product-Form Solutions in Markovian Models ". In proceedings of First Joint WOSP/SIPEW International Conference on Performance Engineering.[2010].
- ▶ Peter G. Harrison and Maria G. Vigliotti."Perturbation of a non-product-form network into a new product-form: equilibrium state probabilities and response time density". In Proceedings of (VALUETOOLS'09).[2009]
- ▶ Andrea Marin and Maria G. Vigliotti."On product-form approximations of cooperating stochastic models ", ISCIS [2010].
- ▶ Gianfranco Balbo and Maria G. Vigliotti. "New product-form solutions for loss networks that serve bulks of customers". [2010]