Stochastic models in product-form: the (E)RCAT methodology

Andrea Marin\textsuperscript{1} Maria Grazia Vigliotti\textsuperscript{2}

\textsuperscript{1}Dipartimento di Informatica
Università Ca’ Foscari di Venezia

\textsuperscript{2}Department of Computing
Imperial College London

Performance 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?
M/M/1 queue

Steady state distribution

\[ \lim_{t \to \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) = \pi(n) = (1 - \rho) \rho^n \]

(\(\pi(n)\) is independent from the initial state.)

\(n\) is the number of customers queueing and \(\frac{\lambda}{\mu} = \rho < 1\).
M/M/1 queue

Steady state distribution

\[
\lim_{t \to \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) = \pi(n) = (1 - \rho) \rho^n
\]

(\(\pi(n)\) is independent from the initial state.)

\(n\) is the number of customers queueing and \(\frac{\lambda}{\mu} = \rho < 1\).
M/M/1 queue

Steady state distribution

\[
\lim_{t \to \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) = \pi(n) = (1 - \rho)\rho^n
\]

(\(\pi(n)\) is independent from the initial state.)

\(n\) is the number of customers queueing and \(\frac{\lambda}{\mu} = \rho < 1\).
M/M/1 queue

Steady state distribution

\[
\lim_{t \to \infty} \mathbb{P}(X(t) = n \mid X(0) = 0) = \pi(n) = (1 - \rho)\rho^n
\]

(\(\pi(n)\) is independent from the initial state.)

\(n\) is the number of customers queueing and \(\frac{\lambda}{\mu} = \rho < 1\).
Well-known result

Two-node Jackson network

Traffic equations

The mean rates of the arrivals in each queue are:

\[
\begin{align*}
\gamma_1 &= \lambda_1 + q(\gamma_2) \\
\gamma_2 &= \lambda_2 + p(\gamma_1)
\end{align*}
\]
Well-known result

Two-node Jackson network

Product form solution

\[ \pi(n, m) = C \pi_1(n) \pi_2(m) \]
Well-known result

Two-node Jackson network

Traffic equations

The mean rates of the arrivals in each queue are:

\[ \gamma_1 = \lambda_1 + q(\gamma_2) \]
\[ \gamma_2 = \lambda_2 + p(\gamma_1) \]
Well-known result

Two-node Jackson network

Traffic equations

The mean rates of the arrivals in each queue are:

\[
\gamma_1 = \lambda_1 + q(\gamma_2) \\
\gamma_2 = \lambda_2 + p(\gamma_1)
\]
Well-known result

Two-node Jackson network

Traffic equations

The mean rates of the arrivals in each queue are:

\[ \gamma_1 = \lambda_1 + q(\gamma_2) \]
\[ \gamma_2 = \lambda_2 + p(\gamma_1) \]
Derivation of product-form solution

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

Two-node network

\[
\begin{align*}
\pi(n_1, n_2)Q & = 0 \quad \text{(stationary measure)} \\
\pi_1(n_1)\pi_2(n_2)Q & = 0
\end{align*}
\]

with \( \sum_{(n_1, n_2)\in S} \pi(n_1, n_2) = 1 - \) (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.

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology

Performance 2010, 16/11/10
Derivation of product-form solution

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

Two-node network

\[
\pi(n_1, n_2)Q = 0 \quad \text{(stationary measure)}
\]

\[
\pi_1(n_1)\pi_2(n_2)Q = 0
\]

with \( \sum_{(n_1, n_2) \in S} \pi(n_1, n_2) = 1 \) - (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

\[
\begin{align*}
\pi(n_1, n_2)Q &= 0 \quad \text{(stationary measure)} \\
\pi_1(n_1)\pi_2(n_2)Q &= 0
\end{align*}
\]

with \( \sum_{(n_1, n_2) \in S} \pi(n_1, n_2) = 1 \) - (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

\[ \pi(n_1, n_2)Q = 0 \quad \text{(stationary measure)} \]
\[ \pi_1(n_1)\pi_2(n_2)Q = 0 \]

with \( \sum_{(n_1, n_2) \in S} \pi(n_1, n_2) = 1 \) - (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

\[
\begin{align*}
\pi(n_1, n_2)Q &= 0 \quad \text{(stationary measure)} \\
\pi_1(n_1)\pi_2(n_2)Q &= 0 \\
\end{align*}
\]

with \( \sum_{(n_1, n_2) \in S} \pi(n_1, n_2) = 1 \) - (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

\[ \pi(n_1, n_2)Q = 0 \quad \text{(stationary measure)} \]
\[ \pi_1(n_1)\pi_2(n_2)Q = 0 \]

with \( \sum_{(n_1, n_2) \in S} \pi(n_1, n_2) = 1 \) - (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

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

- M→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*

- M→ M (Muntz); Local Balance (Chandy et al.)
- BCMP theorem.
- Quasi-reversibility (Kelly/Serfozo).
- G-networks (Gelenbe).
- Networks with finite buffers and blocking.
- .....

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology
Background

A non-exhaustive list of results for queueing theory

- M→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*

- $M \rightarrow M$ (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*

- $M \rightarrow M$ (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*

- $\text{M} \rightarrow \text{M}$ (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

Bouchererie product-form solutions.
In summary so far ...

Traditional outlook

Product-form solutions refer mostly to networks of queues.

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 mostly to networks of queues.

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 mostly to networks of queues.

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 mostly to networks of queues.

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
  - 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.
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.
Outline

Labelled Markov Automata

GRCAT

Jackson Networks

Non-queueing example

Perturbation of product-form solution

New type of queue

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

\[ M = \langle S, A \cup P, \rightarrow \rangle \] is a labelled Markov automaton (LMA):

- \( S \) is state space with states with \( s_1, s_2, \ldots, s_n, \ldots \).
- \( A \) is the set of active labels with \( a_1, a_2, \ldots, a_n, \ldots \).
- \( P \) is the set of passive labels with \( p_1, p_2, \ldots, p_n, \ldots \).
- \( \rightarrow \) is the transition relation between states defined as
  \[
  \rightarrow : (S \times A \times \mathbb{R}^+ \times S) \cup (S \times P \times Var \times S)
  \]
  \( \mathbb{R}^+ \) is set of real numbers and \( Var \) is the set of variables.
Labelled Markov Automata

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

- \( \mathcal{S} \) is state space with states with \( s_1, s_2, \ldots, s_n, \ldots \)
- \( \mathcal{A} \) is the set of active labels with \( a_1, a_2, \ldots, a_n, \ldots \)
- \( \mathcal{P} \) is the set of passive labels with \( p_1, p_2, \ldots, p_n, \ldots \)
- \( \to \) is the transition relation between states defined as
  \[ \to: (\mathcal{S} \times \mathcal{A} \times \mathbb{R}^+ \times \mathcal{S}) \cup (\mathcal{S} \times \mathcal{P} \times \text{Var} \times \mathcal{S}) \]
  \( \mathbb{R}^+ \) is set of real numbers and \( \text{Var} \) is the set of variables.
Labelled Markov Automata

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

- S is state space with states with \( s_1, s_2, \ldots, s_n \).
- \( A \) is the set of active labels with \( a_1, a_2, \ldots, a_n \).
- \( P \) is the set of passive labels with \( p_1, p_2, \ldots, p_n \).
- \( \rightarrow \) is the transition relation between states defined as

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

\( \mathbb{R}^+ \) is set of real numbers and \( \text{Var} \) is the set of variables.
Labelled Markov Automata

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

is a labelled Markov automaton (LMA):

- \( S \) is state space with states with \( s_1, s_2, \ldots, s_n, \ldots \)
- \( A \) is the set of active labels with \( a_1, a_2, \ldots, a_n, \ldots \)
- \( P \) is the set of passive labels with \( p_1, p_2, \ldots, p_n, \ldots \)
- \( \rightarrow \) is the transition relation between states defined as

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

\( \mathbb{R}^+ \) is set of real numbers and \( \text{Var} \) is the set of variables.
Labelled Markov Automata

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

- S is state space with states with \( s_1, s_2, \ldots, s_n, \ldots \).
- A is the set of active labels with \( a_1, a_2, \ldots, a_n, \ldots \).
- P is the set of passive labels with \( p_1, p_2, \ldots, p_n, \ldots \).
- \( \rightarrow \) is the transition relation between states defined as

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

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

Graph

Formal definition

\[ M = \langle S, A, \rightarrow \rangle \]

\[ S_1 = \{s_1, s_2, s_3\} \quad A = \{a, b, c, d\} \]

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

\[ s_1 \xrightarrow{a,\lambda} s_2, \quad s_1 \xrightarrow{d,\delta} s_1, \quad s_2 \xrightarrow{c,\delta} s_3, \quad s_3 \xrightarrow{b,\nu} s_1 \]
Example

Graph

Formal definition

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

\[ S_1 = \{ s_1, s_2, s_3 \} \]
\[ A = \{ a, b, c, d \} \]

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

\[ s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1 \]
Example

Graph

Formal definition

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

\[ S_1 = \{ s_1, s_2, s_3 \} \quad A = \{ a, b, c, d \} \]

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

\[ s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1 \]
Example

**Graph**

```
S1 \rightarrow S2, S2 \rightarrow S3, S3 \rightarrow S1
```

**Formal definition**

\[ M = \langle S, A, \rightarrow \rangle \]

\[ S_1 = \{ s_1, s_2, s_3 \} \quad A = \{ a, b, c, d \} \]

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

Graph

Formal definition

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

\[ S_1 = \{s_1, s_2, s_3\} \quad A = \{a, b, c, d\} \]

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

\[ s_1 \rightarrow s_2, s_1 \rightarrow s_1, s_2 \rightarrow s_1, s_3 \rightarrow s_1, s_3 \rightarrow s_1 \]
Example

Graph

Formal definition

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

\[ S_1 = \{ s_1, s_2, s_3 \} \quad A = \{ a, b, c, d \} \]

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

\[ s_1 \xrightarrow{a,\lambda} s_2, \quad s_1 \xrightarrow{d,\delta} s_1, \quad s_1 \xrightarrow{c,\delta} s_3, \quad s_3 \xrightarrow{b,\nu} s_1 \]
Example

Graph

Formal definition

\[ M = \langle S, A, \rightarrow \rangle \]

\[ S_1 = \{ s_1, s_2, s_3 \} \quad A = \{ a, b, c, d \} \]

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

\[ s_1 \xrightarrow{a, \lambda} s_2, s_1 \xrightarrow{d, \delta} s_1, s_2 \xrightarrow{c, \delta} s_3, s_3 \xrightarrow{b, \nu} s_1 \]
Differences between LMAs and CTMCs

- LMA has labelled transitions.
- LMA has self-loops that are redundant in the context of the CTMCs.
Differences between LMAs and CTMCs

LMA

CTMC

Differences

- LMA has labelled transitions.
- LMA has self-loops that are redundant in the context of the CTMCs.
The associated time-homogenous CTMC

Any CTMC is fully characterised by $Q$, the generator matrix.

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

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

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology

Performance 2010, 16/11/10
The associated time-homogenous CTMC

Any CTMC is fully characterised by $Q$, the generator matrix.

- The transition rate is $q(s \rightarrow s') \overset{df}{=} \sum_{(a,\lambda): s^a,\lambda \rightarrow s'} \lambda$.

- The total (or exit) rate of a state $s$ is $q(s) \overset{df}{=} \sum_{s' \in S} q(s \rightarrow s')$.

- The diagonal of the generator matrix is $q(s \rightarrow s) \overset{df}{=} -q(s)$.

The transition rate due to action $a$ is $q(s \overset{a}{\rightarrow} s') \overset{df}{=} \sum_{\lambda: s^a,\lambda \rightarrow s'} \lambda$. 
The associated time-homogenous CTMC

Any CTMC is fully characterised by $Q$, the generator matrix.

- The transition rate is $q(s \rightarrow s') \overset{df}{=} \sum_{(a,\lambda): s^a,\lambda \rightarrow s'} \lambda.$

- The total (or exit) rate of a state $s$ is $q(s) \overset{df}{=} \sum_{s' \in S \atop s' \neq s} q(s \rightarrow s').$

- The diagonal of the generator matrix is $q(s \rightarrow s) \overset{df}{=} -q(s)$.

The transition rate due to action $a$ is $q(s \overset{a}{\rightarrow} s') \overset{df}{=} \sum_{\lambda: s^a,\lambda \rightarrow s'} \lambda.$
The associated time-homogenous CTMC

Any CTMC is fully characterised by $Q$, the generator matrix.

- The **transition rate** is $q(s \rightarrow s') \overset{df}{=} \sum_{(a, \lambda) : s^a, \lambda \rightarrow s'} \lambda$.

- The **total (or exit) rate** of a state $s$ is $q(s) \overset{df}{=} \sum_{s', \in S \atop s' \neq s} q(s \rightarrow s')$.

- The **diagonal** of the generator matrix is $q(s \rightarrow s) \overset{df}{=} -q(s)$.

- The transition rate due to action $a$ is $q(s \overset{a}{\rightarrow} s') \overset{df}{=} \sum_{\lambda : s^a, \lambda \rightarrow s'} \lambda$. 
The associated time-homogenous CTMC

Any CTMC is fully characterised by $Q$, the generator matrix.

- The transition rate is $q(s \rightarrow s') \overset{df}{=} \sum_{(a,\lambda): s\overset{a,\lambda}{\rightarrow} s'} \lambda$.

- The total (or exit) rate of a state $s$ is $q(s) \overset{df}{=} \sum_{s' \in S \atop s' \neq s} q(s \rightarrow s')$.

- The diagonal of the generator matrix is $q(s \rightarrow s) \overset{df}{=} -q(s)$.

LMA

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

Maria Grazia Vigliotti

Imperial College London
From LMAs to CTMCs

MARKOV AUTOMATON

S₁

S₂

c,σ

a,λ

a,μ

f,α

b,κ

d,β

CTMC

S₁

S₂

λ+μ

α+κ

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology
From LMAs to CTMCs
From LMAs to CTMCs

MARKOV AUTOMATON

S1 → S2

S2 → S1

From LMAs to CTMCs
From LMAs to CTMCs
From LMAs to CTMCs

Markov Automaton

CTMC

S1

S2

\[ \lambda + \mu \]

\[ \alpha + \kappa \]
Open automaton

Graph

Formal definition

\[ M = \langle S, A \cup P, \rightarrow \rangle \]

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

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

Passive and active transitions

\[ s_1 \xrightarrow{a, \lambda} s_2, \quad s_1 \xrightarrow{d, x_d} s_2, \quad s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1. \]
Open automaton

Formal definition

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

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

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

\[ s_1 \xrightarrow{a, \lambda} s_2, \quad s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, \quad s_3 \xrightarrow{b, \nu} s_1. \]
Open automaton

Graph

Formal definition

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

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

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

$$S_1 \xrightarrow{a, \lambda} S_2, \quad S_1 \xrightarrow{d, x_d} S_1, S_2 \xrightarrow{c, x_c} S_3, S_3 \xrightarrow{b, \nu} S_1.$$
Open automaton

Graph

Formal definition

\[ \mathcal{M} = \langle S, A \cup P, \rightarrow \rangle \]
\[ S = \{ s_1, s_2, s_3 \} \quad A = \{ a, b \} \quad P = \{ c, d \} \]
\[ \rightarrow = \{ (s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1) \} \]

\[ s_1 \xrightarrow{a,\lambda} s_2, s_1 \xrightarrow{d,x_d} s_1, s_2 \xrightarrow{c,x_c} s_3, s_3 \xrightarrow{b,\nu} s_1. \]
Open automaton

Formal definition

\[ M = \langle S, A \cup P, \rightarrow \rangle \]

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

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

Passive and active transitions

\[ s_1 \xrightarrow{a, \lambda} s_2, \quad s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1. \]
Open automaton

Graph

Formal definition

\[ M = \langle S, A \cup P, \rightarrow \rangle \]

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

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

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

Passive and active transitions

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

Formal definition

\[ M = \langle S, A \cup P, \rightarrow \rangle \]
\[ S = \{ s_1, s_2, s_3 \} \quad A = \{ a, b \} \quad P = \{ c, d \} \]
\[ \rightarrow = \{(s_1, a, \lambda, s_2), (s_2, c, x_c, s_3), (s_3, b, \nu, s_1), (s_1, d, x_d, s_1)\} \]
\[ s_1 \xrightarrow{a, \lambda} s_2, \quad s_1 \xrightarrow{d, x_d} s_1, s_2 \xrightarrow{c, x_c} s_3, s_3 \xrightarrow{b, \nu} s_1. \]

Passive and active transitions

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

Open automaton

A Markov automaton $\mathcal{M}$ is open if there exists an action $a \in \mathcal{P}$ and a state $s \in S$ such that $a$ is passively enabled out of $s$.

Close automaton

A Markov automaton $\mathcal{M}$ is closed if it is not open.

Problem
Can any open automaton be mapped to a CTMC?
Open automaton and CTMC

Open automaton

A Markov automaton $\mathcal{M}$ is **open** if there exists an action $a \in \mathcal{P}$ and a state $s \in \mathcal{S}$ such that $a$ is passively enabled out of $s$.

Close automaton

A Markov automaton $\mathcal{M}$ is **closed** if it is not open.

Problem

Can any *open* automaton be mapped to a CTMC?
Open automaton and CTMC

Open automaton

A Markov automaton $\mathcal{M}$ is **open** if there exists an action $a \in \mathcal{P}$ and a state $s \in S$ such that $a$ is passively enabled out of $s$.

Close automaton

A Markov automaton $\mathcal{M}$ is **closed** if it is not open.

Problem

Can any *open* automaton be mapped to a CTMC?
Open automaton and CTMC

Example of an open automaton

M

S₁ → a,xₐ → S₁ → S₃

b, λ

c, ρ
Open automaton and CTMC

Example of an open automaton

\[ A = \{ b, c \} \]

active labels
Open automaton and CTMC

Example of an open automaton

\[ \mathcal{A} = \{b, c\} \]

active labels

\[ \mathcal{P} = \{a\} \]

passive labels
Closure of open automaton
Closure of open automaton

Single label closure

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow \mathcal{M}\{a \leftarrow K_a\} \rangle \text{ is the closure of the automaton } \mathcal{M} = \langle S, A \cup \{a\}, \rightarrow \rangle \]

\[ \rightarrow \mathcal{M}\{a \leftarrow K_a\} = \]

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology

Performance 2010, 16/11/10
Closure of open automaton

Single label closure

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow \rangle \]

is the closure of the automaton \( \mathcal{M} = \langle S, A \cup \{a\}, \rightarrow \rangle \)

\[ \rightarrow \mathcal{M}\{a \leftarrow K_a\} = \]
Closure of open automaton

Single label closure

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

\[ \rightarrow \mathcal{M}\{a \leftarrow K_a\} \]
Closure of open automaton

Single label closure

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow \mathcal{M}\{a \leftarrow K_a\}\rangle \text{ is the closure of the automaton } \mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle \]

\[ \rightarrow \mathcal{M}\{a \leftarrow K_a\} = \]
Closure of open automaton

Single label closure
\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow, \mathcal{M}\{a \leftarrow K_a\} \rangle \] is the \textit{closure} of the automaton \( \mathcal{M} = \langle S, A \cup \{a\}, \rightarrow \rangle \)

\[ \rightarrow \mathcal{M}\{a \leftarrow K_a\} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in A\} \]
Closure of open automaton

Single label closure

\[ M\{a \leftarrow Ka\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow M\{a \leftarrow Ka\} \rangle \text{ is the closure of the automaton } M = \langle S, A \cup \{a\}, \rightarrow \rangle \]

\[ \rightarrow M\{a \leftarrow Ka\} = \{(s, b, r, s') : (s, b, r, s') \in \rightarrow, b \in A\} \]

\[ \cup \{(s, a, Ka, s') : (s, a, x_a, s') \in \rightarrow\} \]
Closure of open automaton

Single label closure

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, \mathcal{A}' \cup \emptyset, \rightarrow \mathcal{M}\{a \leftarrow K_a\} \rangle \text{ is the closure of the automaton } \mathcal{M} = \langle S, \mathcal{A} \cup \{a\}, \rightarrow \rangle \]

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

\[ \cup \{ (s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow \} \]

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

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow \rangle \]

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

\[ \cup \{ (s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow \} \]

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

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{df}{=} \langle S, A' \cup \emptyset, \rightarrow \mathcal{M}\{a \leftarrow K_a\} \rangle \text{ is the closure of the automaton } \mathcal{M} = \langle S, A \cup \{a\}, \rightarrow \rangle \]

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

\[ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\} \]

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

\[ \mathcal{M}\{a \leftarrow K_a\} \overset{\text{df}}{=} \langle S, A' \cup \emptyset, \rightarrow \mathcal{M}\{a \leftarrow K_a\} \rangle \] is the closure of the automaton \( \mathcal{M} = \langle S, A \cup \{a\}, \rightarrow \rangle \)

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

\[ \cup \{(s, a, K_a, s') : (s, a, x_a, s') \in \rightarrow\} \]

Properties of closure

- The order in which the closures are performed is irrelevant.
- A closed automaton has no passive transition.
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i & \xrightarrow{a, \lambda} S_i' & S_k & \xrightarrow{a, x_a} S_k' \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n) \\
S_i & \xrightarrow{a, r} S_i' & (a \notin L) \\
(s_1, \ldots, s_i', \ldots, s_n) & \xrightarrow{a, r} (s_1, \ldots, s_i, \ldots, s_n)
\end{align*}
\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i & \xrightarrow{a, \lambda} S_i \quad & S_k & \xrightarrow{a, x_a} S_k' \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n) \\
S_i & \xrightarrow{a, r} S_i' \quad & (a \notin L) \\
(s_1, \ldots, s_i', \ldots, s_n) & \xrightarrow{a, r} (s_1, \ldots, s_i', \ldots, s_n)
\end{align*}
\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_I = P_h \cap P_I = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i^{a,\lambda} & \rightarrow_i S'_i \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n)^{a,\lambda} & \rightarrow (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) \\
S_k^{a,x_a} & \rightarrow_k S'_k \\
(s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n)^{a,x_a} & \rightarrow (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n)
\end{align*}
\]

\((a \in L)\)

\[
\begin{align*}
S_i^{a,r} & \rightarrow_i S'_i \\
(s_1, \ldots, s'_i, \ldots, s_n)^{a,r} & \rightarrow (s_1, \ldots, s'_i, \ldots, s_n)
\end{align*}
\]

\((a \notin L)\)
Automata Interacting

\[ \bigoplus_L (M_1, \ldots, M_i, \ldots, M_k, \ldots, M_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^{n} A_i) \cup (\bigcup_{i=1}^{n} P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i & \xrightarrow{a, \lambda} S_i' \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n) (a \in L) \\
S_k & \xrightarrow{a, x_a} S_k' \\
(s_1, \ldots, s_i', \ldots, s_k, \ldots, s_n) & \xrightarrow{a, x_a} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n) (a \notin L)
\end{align*}
\]
Automata Interacting

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

is the interacting LMA:

- $S = S_1 \times S_2 \times \ldots \times S_n$.
- $Act = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i)$ and $A_h \cap A_i = P_h \cap P_i = \emptyset$.
- $\rightarrow$ is the smallest relation defined by:

\[
\begin{align*}
S_i \xrightarrow{a, \lambda}_i S'_i \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) \xrightarrow{a, \lambda} (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) \\
S_i \xrightarrow{a, x_a}_i S'_i \\
(s_1, \ldots, s'_i, \ldots, s_n) \xrightarrow{a, r} (s_1, \ldots, s'_i, \ldots, s'_n) \\
\end{align*}
\]

\[(a \in L) \quad (a \notin L)\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n \mathcal{A}_i) \cup (\bigcup_{i=1}^n \mathcal{P}_i) \) and \( \mathcal{A}_h \cap \mathcal{A}_l = \mathcal{P}_h \cap \mathcal{P}_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
  s_i \xrightarrow{a, \lambda} & s'_i \\
  (s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) \\
  s_i \xrightarrow{a, r} & s'_i \\
  (s_1, \ldots, s'_i, \ldots, s_n) & \xrightarrow{a, r} (s_1, \ldots, s'_i, \ldots, s_n)
\end{align*}
\]

\((a \in L)\)

\((a \notin L)\)
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \[ S = S_1 \times S_2 \times \ldots \times S_n. \]
- \[ \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \text{ and } A_h \cap A_l = P_h \cap P_l = \emptyset. \]
- \[ \rightarrow \text{ is the smallest relation defined by:} \]

\[
\begin{align*}
S_i & \xrightarrow{a, \lambda}_i S_i' & S_k & \xrightarrow{a, x_a}_k S_k' \quad (a \in L) \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) \\
S_i & \xrightarrow{a, r}_i S_i' \quad (a \notin L) \\
(s_1, \ldots, s'_i, \ldots, s_n) & \xrightarrow{a, r} (s_1, \ldots, s'_i, \ldots, s_n)
\end{align*}
\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i \xrightarrow{a,\lambda} i S'_i & \quad & S_k \xrightarrow{a,x_a} k S'_k (a \in L) \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) \xrightarrow{a,\lambda} (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) \\
S_i \xrightarrow{a,r} i S'_i & \quad & (a \notin L) \\
(s_1, \ldots, s'_i, \ldots, s_n) \xrightarrow{a,r} (s_1, \ldots, s'_i, \ldots, s_n)
\end{align*}
\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = (S, \text{Act}, \rightarrow) \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i & \xrightarrow{a, \lambda} S_i' \\
S_k & \xrightarrow{a, x_a} S_k' \quad (a \in L) \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) & \xrightarrow{a, \lambda} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n) \\
S_i & \xrightarrow{a, r} S_i' \quad (a \notin L) \\
(s_1, \ldots, s_i', \ldots, s_n) & \xrightarrow{a, r} (s_1, \ldots, s_i', \ldots, s_n)
\end{align*}
\]
Automata Interacting

\[ \bigoplus_L (M_1, \ldots, M_i, \ldots, M_k, \ldots, M_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i \xrightarrow{a, \lambda}_i S_i' \\
S_k \xrightarrow{a, x_a}_k S_k' (a \in L)
\end{align*}
\]

\[
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) \xrightarrow{a, \lambda} (s_1, \ldots, s_i', \ldots, s_k', \ldots, s_n)
\]

\[
S_j \xrightarrow{a, r}_j S_j' (a \notin L)
\]

\[
(s_1, \ldots, s_i', \ldots, s_n) \xrightarrow{a, r} (s_1, \ldots, s_i', \ldots, s_n)
\]
Automata Interacting

\[ \bigoplus_L (\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_k, \ldots, \mathcal{M}_n) = \langle S, \text{Act}, \rightarrow \rangle \]

is the interacting LMA:

- \( S = S_1 \times S_2 \times \ldots \times S_n \).
- \( \text{Act} = (\bigcup_{i=1}^n A_i) \cup (\bigcup_{i=1}^n P_i) \) and \( A_h \cap A_l = P_h \cap P_l = \emptyset \).
- \( \rightarrow \) is the smallest relation defined by:

\[
\begin{align*}
S_i \xrightarrow{a,\lambda} S_i' & \quad & S_k \xrightarrow{a,xa} S_k' \\
(s_1, \ldots, s_i, \ldots, s_k, \ldots, s_n) \xrightarrow{a,\lambda} (s_1, \ldots, s'_i, \ldots, s'_k, \ldots, s_n) & \quad & (a \in L) \\
S_i \xrightarrow{a,r} S_i' & \quad & (s_1, \ldots, s'_i, \ldots, s_n) \xrightarrow{a,r} (s_1, \ldots, s'_i, \ldots, s_n) \\
\end{align*}
\]

\( (a \notin L) \)
Example of interacting automata

(M1)

(M2)
Example of cooperating automata (three steps)

\[(M_1 \oplus_a M_2)\]
Example of cooperating automata (cont.)
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 $s$.

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

   $\forall s, s', s'' \in S, s \xrightarrow{a,x} s' \land s \xrightarrow{a,x} s'' \implies s' = s''$. 

Mary Grazia Vigliotti  
Imperial College London  
Stochastic models in product-form: the (E)RCAT methodology  
Performance 2010, 16/11/10
Well-formed automaton

An automaton \( M = \langle S, A \cup P, \rightarrow \rangle \) is well formed if:

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

\[ A \cap P = \emptyset. \]

2. If \( a \in P \) then for every state \( s \) of the automaton there exists exactly one passive transition \( a \) out of state \( s \).

\[ \forall a \in P \quad \forall s \exists s' \in S \quad \text{such that} \quad s \xrightarrow{a,x_a} s' \]

\[ \forall s, s', s'' \in S, \quad s \xrightarrow{a,x_a} s' \land s \xrightarrow{a,x_a} s'' \implies s' = s''. \]
Well-formed automaton

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

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

\[ A \cap P = \emptyset. \]

2. If \( a \in P \) then for every state \( s \) of the automaton there exists exactly one passive transition \( a \) out of state \( s \).

\[ \forall a \in P \forall s \exists s' \in S \text{ such that } s \xrightarrow{a,x_a} s'. \]

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

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

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

   $\forall a \in \mathcal{P} \ \forall s \exists s' \in S \ \text{such that} \ s \xrightarrow{a,x_a} s' \ \forall s, s', s'' \in S, s \xrightarrow{a,x_a} s' \land s \xrightarrow{a,x_a} s'' \Longrightarrow s' = s''$. 
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 state $s_2$. 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 $A = \{a\}$ and $P = \{b, d\}$, therefore $A \cap P = \emptyset$.

**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 $A = \{a\}$ and $P = \{b, d\}$, therefore $A \cap P = \emptyset$. √

**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 the 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 the there is no set of co-operating transitions among components ...

\[ M_1 \oplus \emptyset M_2 \]

- Trivially, \( \pi(M_1 \oplus \emptyset M_2) = \pi_1(M_1)\pi_2(M_2). \)

- For the rest of the tutorial we shall consider non-trivial interactions between (among various) automata, i.e.

\[ M_1 \oplus_L M_2 \quad L \neq \emptyset \]
Assumption 1
The closed automaton $\bigoplus_L (\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_N)$ has irreducible global state space $S$ and it is composed by well-formed automata $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_N$.

Assumption 2
For each label $L = \{a_1, a_2, \ldots, a_M\}$ there exists a set of rates $\{K_{a_1}, \ldots, K_{a_M}\}$ such that:

$$\forall a \in (A_i \cap L), \forall s \in S_i, 1 \leq i \leq N$$

$$\frac{\sum_{s' \in S_i} q(s' \xrightarrow{a} s) \pi_i(s')}{\pi_i(s)} = K_a \quad (3)$$
Assumption 1
The closed automaton $\bigoplus_L (M_1, M_2, \ldots, M_N)$ has irreducible global state space $S$ and it is composed by well-formed automata $M_1, M_2, \ldots, M_N$.

Assumption 2
For each label $L = \{a_1, a_2, \ldots, a_M\}$ there exists a set of rates $\{K_{a_1}, \ldots, K_{a_M}\}$ such that:

$$\forall a \in (A_i \cap L), \forall s \in S_i, \ 1 \leq i \leq N$$

$$\frac{\sum_{s' \in S_i} q(s' \xrightarrow{a} s) \pi_i(s')}{\pi_i(s)} = K_a \quad (3)$$
Stationary measure

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

Product form of the steady-state distribution

If $\sum_{s \in S_i} \pi_i(M_i^i) = 1$ then $\pi(\bigoplus_L (M_1, M_2, \ldots M_N))$ is the steady-state distribution.
Stationary measure

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

$$\pi\left(\bigoplus_{L}(M_1, M_2, \ldots M_N)\right) = \prod_{i=1}^{N} \pi_i(M_i^c)$$

Product form of the steady-state distribution

If $\sum_{s \in S_i} \pi_i(M_i^c) = 1$ then $\pi\left(\bigoplus_{L}(M_1, M_2, \ldots M_N)\right)$ is the steady-state distribution.
Stationary measure

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

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

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

Product form of the steady-state distribution

If $\sum_{s \in S_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi(\bigoplus_{L}(\mathcal{M}_1, \mathcal{M}_2, \ldots \mathcal{M}_N))$ is the steady-state distribution.
Stationary measure

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

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

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

Product form of the steady-state distribution

If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi\left(\bigoplus_{L}(\mathcal{M}_1, \mathcal{M}_2, \ldots \mathcal{M}_N)\right)$ is the steady-state distribution.
Stationary measure
Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

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

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

Product form of the steady-state distribution
If $\sum_{s \in \mathcal{S}_i} \pi_i(\mathcal{M}_i^c) = 1$ then $\pi\left(\bigoplus_{L}(\mathcal{M}_1, \mathcal{M}_2, \ldots \mathcal{M}_N)\right)$ is the steady-state distribution.
Stationary measure

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

$$\pi\left(\bigoplus_{L}(M_1, M_2, \ldots, M_N)\right) = \prod_{i=1}^{N} \pi_i(M_i^c)$$

$M_i^c = M_i\{a \leftarrow K_a : a \in P_i\}$ (closed automata).

Product form of the steady-state distribution

If $\sum_{s \in S_i} \pi_i(M_i^c) = 1$ then $\pi\left(\bigoplus_{L}(M_1, M_2, \ldots, M_N)\right)$ is the steady-state distribution.
Stationary measure
Under these assumptions the stationary measure $\pi(\cdot)$ has the product form:

$$\pi(\bigoplus_L (M_1, M_2, \ldots M_N)) = \prod_{i=1}^{N} \pi_i(M_i^c)$$

$M_i^c = M_i\{a \leftarrow K_a : a \in P_i\}$ (closed automata).

Product form of the steady-state distribution
If $\sum_{s \in S_i} \pi_i(M_i^c) = 1$ then $\pi(\bigoplus_L (M_1, M_2, \ldots M_N))$ 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 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 in the composition.
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 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 in the composition.
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 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 in the composition.
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 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 in the composition.
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.
Two-node Jackson network

Arrival Process
Arrival rate: $\lambda_1$.
Service Process
Service rate: $\mu_1$.
Routing $r_{1,2} = p$ $r_{1,0} = 1 - p$.

Arrival Process
Arrival rate: $\lambda_2$.
Service Process
Service rate: $\mu_2$.
Routing $r_{2,1} = q$ $r_{2,0} = 1 - q$. 
Two-node Jackson network

Arrival Process  Arrival rate: $\lambda_1$.
Service Process  Service rate: $\mu_1$.
Routing  $r_{1,2} = p$  $r_{1,0} = 1 - p$.

Arrival Process  Arrival rate: $\lambda_2$.
Service Process  Service rate: $\mu_2$.
Routing  $r_{2,1} = q$  $r_{2,0} = 1 - q$. 
Two-node Jackson network

Arrival Process  Arrival rate: $\lambda_1$.
Service Process  Service rate: $\mu_1$.
Routing  $r_{1,2} = p$, $r_{1,0} = 1 - p$.

Arrival Process  Arrival rate: $\lambda_2$.
Service Process  Service rate: $\mu_2$.
Routing  $r_{2,1} = q$, $r_{2,0} = 1 - q$. 

Maria Grazia Vigliotti
Stochastic models in product-form: the (E)RCAT methodology
Imperial College London
Performance 2010, 16/11/10
Jackson Networks

Non-queueing example
Perturbation of product-form solution
New type of queue
Conclusion

$M_1$

$M_2$

Maria Grazia Vigliotti
Imperial College London

Stochastic models in product-form: the (E)RCAT methodology

Performance 2010, 16/11/10
Product-form solution

\[ JN = M_1 \oplus \{a, b\} M_2 \]

- \( A_1 = \{c, b, d\}, \ P_1 = \{a\} \) and \( A_1 \cap P.M_1 = \emptyset \), \( \sqrt{\text{✓}} \).
- Only one passive action \( a \) out of each state \( n \), \( \sqrt{\text{✓}} \).
- Similar reasoning for \( M_2 \). \( A_2 = \{e, f, a\} \) and \( P_2 = \{b\} \).
- \( b \in \{a, b\} \cap A_1 \) then for all \( n \in S_1 \):
  \[ \frac{\pi_1(n+1)p_{\mu_1}}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1). \]
- \( a \in \{a, b\} \cap A_2 \) 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

\[ \text{JN} = \mathcal{M}_1 \oplus \{a, b\} \mathcal{M}_2 \]

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

- \( \mathcal{A}_1 = \{c, b, d\}, \quad \mathcal{P}_1 = \{a\} \) and \( \mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \) \( \checkmark \).
- Only one passive action \( a \) out of each state \( n \) \( \checkmark \).
- 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 \( n \in S_1 \)
  \[ \frac{\pi_1(n+1)p_{\mu_1}}{\pi_1(n)} = K_b = pq_1(n \to n+1). \]

- \( a \in \{a, b\} \cap \mathcal{A}_2 \) then for all \( n \in S_2 \)
  \[ \frac{\pi_2(n+1)q_{\mu_2}}{\pi_2(n)} = K_a = qq_2(n \to n+1). \]
Product-form solution

\[ JN = \mathcal{M}_1 \oplus \{a,b\} \mathcal{M}_2 \]

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

- \( A_1 = \{c, b, d\}, \quad P_1 = \{a\} \) and \( A_1 \cap P \mathcal{M}_1 = \emptyset \quad \checkmark \).
- Only one passive action \( a \) out of each state \( n \) \( \checkmark \).
- Similar reasoning for \( \mathcal{M}_2 \). \( A_2 = \{e, f, a\} \) and \( P_2 = \{b\} \)

- \( b \in \{a, b\} \cap A_1 \) then for all \( n \in S_1 \)
  \[ \frac{\pi_1(n+1) \rho_{\mu_1}}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1). \]
- \( a \in \{a, b\} \cap A_2 \) 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:

- \( A_1 = \{c, b, d\} \), \( P_1 = \{a\} \) and \( A_1 \cap P_\mathcal{M}_1 = \emptyset \) \( \checkmark \).
- Only one passive action \( a \) out of each state \( n \) \( \checkmark \).
- Similar reasoning for \( \mathcal{M}_2 \). \( A_2 = \{e, f, a\} \) and \( P_2 = \{b\} \)

- \( b \in \{a, b\} \cap A_1 \) then for all \( n \in S_1 \)
  \[ \frac{\pi_1(n+1)p_{\mu_1}}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1). \]

- \( a \in \{a, b\} \cap A_2 \) then for all \( n \in S_2 \)
  \[ \frac{\pi_2(n+1)q_{\mu_2}}{\pi_2(n)} = K_a = qq_2(n \rightarrow n+1). \]
Consider the automaton $\mathcal{M}_1$, we verify:

- $A_1 = \{c, b, d\}$, $P_1 = \{a\}$ and $A_1 \cap PM_1 = \emptyset \quad \sqrt{\vee}$.
- Only one passive action $a$ out of each state $n \quad \sqrt{\vee}$.
- Similar reasoning for $\mathcal{M}_2$. $A_2 = \{e, f, a\}$ and $P_2 = \{b\}$

- $b \in \{a, b\} \cap A_1$ then for all $n \in S_1$
  $\frac{\pi_1(n+1)p_{\mu_1}}{\pi_1(n)} = K_b = pq_1(n \rightarrow n+1)$.

- $a \in \{a, b\} \cap A_2$ 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

\[ \text{JN} = \mathcal{M}_1 \oplus \{a, b\} \mathcal{M}_2 \]

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

- \( \mathcal{A}_1 = \{c, b, d\} \), \( \mathcal{P}_1 = \{a\} \) and \( \mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \)  \( \checkmark \).
- Only one passive action \( a \) out of each state \( n \)  \( \checkmark \).
- 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 \( n \in S_1 \)
  \[ \frac{\pi_1(n+1)p_{\mu_1}}{\pi_1(n)} = K_b = \rho q_1(n \rightarrow n + 1) . \]

- \( a \in \{a, b\} \cap \mathcal{A}_2 \) then for all \( n \in S_2 \)
  \[ \frac{\pi_2(n+1)q_{\mu_2}}{\pi_2(n)} = K_a = q q_2(n \rightarrow n + 1) . \]
Product-form solution

\[ JN = M_1 \oplus \{a,b\} M_2 \]

Consider the automaton \( M_1 \), we verify:

- \( A_1 = \{c, b, d\} \), \( P_1 = \{a\} \) and \( A_1 \cap P M_1 = \emptyset \)  \( \sqrt{\text{.}} \)
- Only one passive action \( a \) out of each state \( n \) \( \sqrt{\text{.}} \)
- Similar reasoning for \( M_2 \). \( A_2 = \{e, f, a\} \) and \( P_2 = \{b\} \)

Verification that the “weighted fluxes” are constant:

- \( b \in \{a, b\} \cap A_1 \) then for all \( n \in S_1 \)
  \[
  \frac{\pi_1(n+1)\rho_{\mu_1}}{\pi_1(n)} = K_b = \rho q_1(n \rightarrow n + 1) .
  \]

- \( a \in \{a, b\} \cap A_2 \) then for all \( n \in S_2 \)
  \[
  \frac{\pi_2(n+1)q_{\mu_2}}{\pi_2(n)} = K_a = q q_2(n \rightarrow n + 1) .
  \]

Maria Grazia Vigliotti

Imperial College London

Stochastic models in product-form: the (E)RCAT methodology

Performance 2010, 16/11/10
Product-form solution

\[ \text{JN} = \mathcal{M}_1 \oplus \{a, b\} \mathcal{M}_2 \]

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

- \( \mathcal{A}_1 = \{c, b, d\} \), \( \mathcal{P}_1 = \{a\} \) and \( \mathcal{A}_1 \cap \mathcal{P}\mathcal{M}_1 = \emptyset \) \( \checkmark \).
- Only one passive action \( a \) out of each state \( n \) \( \checkmark \).
- 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 \( n \in S_1 \)
  
  \[ \frac{\pi_1(n+1)p\mu_1}{\pi_1(n)} = K_b = \rho q_1(n \rightarrow n + 1) \].

- \( a \in \{a, b\} \cap \mathcal{A}_2 \) then for all \( n \in S_2 \)
  
  \[ \frac{\pi_2(n+1)q\mu_2}{\pi_2(n)} = K_a = q q_2(n \rightarrow n + 1) \].
\[ M_1\{a \leftarrow K_a\} \]

\[ M_2\{b \leftarrow K_b\} \]

\[ JN = M_1 \oplus \{a, b\} \quad M_2 = M_1\{a \leftarrow K_a\} \oplus M_2\{b \leftarrow K_b\} \]
Traffic equations

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

\[ q_1(n \rightarrow n+1) = \lambda_1 + y = \lambda_1 + q q_2(n \rightarrow n+1) \]

\[ q_2(n \rightarrow n+1) = \lambda_2 + x = \lambda_2 + p q_1(n \rightarrow n+1) \]

where \( y = q_1(n \xrightarrow{a} n+1) \) and \( y = q_2(b \xrightarrow{n} n+1) \) 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:

\[ q_1(n \rightarrow n + 1) = \lambda_1 + y = \lambda_1 + qq_2(n \rightarrow n + 1) \]
\[ q_2(n \rightarrow n + 1) = \lambda_2 + x = \lambda_2 + pq_1(n \rightarrow n + 1) \]

where \( y = q_1(n \rightarrow n + 1) \) and \( y = q_2(n \rightarrow n + 1) \) 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:

\[ q_1(n \rightarrow n+1) = \lambda_1 + y = \lambda_1 + q_2(n \rightarrow n+1) \]

\[ q_2(n \rightarrow n+1) = \lambda_2 + x = \lambda_2 + pq_1(n \rightarrow n+1) \]

where \( y = q_1(n \xleftarrow{a} n+1) \) and \( y = q_2(b \xrightarrow{n} n+1) \) 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

$\frac{q_i(n \to n+1)}{q_i(n+1 \to n)} < 1 \quad n \geq 0, i = 1, 2$

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

\[
\frac{q_i(n \rightarrow n+1)}{q_i(n+1 \rightarrow n)} < 1 \quad n \geq 0, i = 1, 2
\]

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

\[
\frac{q_i(n \rightarrow n + 1)}{q_i(n + 1 \rightarrow n)} < 1 \quad n \geq 0, i = 1, 2
\]

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 $\pi(\oplus a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus a(M_1, M_2))$

- We observe that $M_1$ is closed.
- Steady-state probabilities of $\pi_1(M_1)$ are $\pi_1(1) = \frac{1}{5}$ and $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $M_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(M_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus a(M_1, M_2)) = \pi_1(M_1)\pi_2(M_2\{a \leftarrow 1\})$. 
Product-form of $\pi(\oplus_a(\mathcal{M}_1, \mathcal{M}_2))$

- 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 $\pi_1(2) = \frac{4}{5}$.
- $K_a = 1$
- $\mathcal{M}_2\{a \leftarrow 1\}$.
- Steady-state probabilities of $\pi_2(\mathcal{M}_2\{a \leftarrow 1\})$ are $\pi_2(A) = \frac{2}{3}$ and $\pi_2(B) = \frac{1}{3}$.
- $\pi(\oplus_a(\mathcal{M}_1, \mathcal{M}_2)) = \pi_1(\mathcal{M}_1)\pi_2(\mathcal{M}_2\{a \leftarrow 1\})$. 
Mixed network

Two-node Erlang network

- First node two-phase exponential service rate $\mu$.
- 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 $\mu$.
- 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 $\mu$.
- 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 $\mu$.
- Second node ordinary M/M/1 queue.
- There is no product-form solution.
Erlang node as LMA

\[ ER_1 \]

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

\[
\frac{\pi_1(n+1,1)q((n+1,1) \xrightarrow{a}(n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1)\mu}{\pi_1(n,0)} > 0. 
\]

Equation (3) of GRCAT cannot be satisfied.

We can remedy by making minimal changes to the queue.
Erlang node as LMA

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

\[ \frac{\pi_1(n+1,1)q((n+1,1) \xrightarrow{a} (n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1)\mu}{\pi_1(n,0)} > 0. \]

Equation (3) of GRCAT cannot be satisfied.

We can remedy by making minimal changes to the queue.
Erlang node as LMA

$$ER_1$$

\[
\begin{align*}
\sum_{s \in S} \frac{\pi_1(s)q(s \xrightarrow{a}(n,1))}{\pi_1(n,1)} &= 0. \\
\frac{\pi_1(n+1,1)q((n+1,1) \xrightarrow{a}(n,0))}{\pi_1(n,0)} &= \frac{\pi_1(n+1,1)\mu}{\pi_1(n,0)} > 0.
\end{align*}
\]

Equation (3) of GRCAT cannot be satisfied.

We can remedy by making minimal changes to the queue.
Erlang node as LMA

\( ER_1 \)

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

\[ \frac{\pi_1(n+1,1)q((n+1,1) \xrightarrow{a}(n,0))}{\pi_1(n,0)} = \frac{\pi_1(n+1,1)\mu}{\pi_1(n,0)} > 0. \]

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 '09].
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 '09].
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 '09].
Perturbed Erlang queue as LMA

\(\text{ER}_1\)

- 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 '09].
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 ’09].
Product-form solution

\[ M/M/1 \]

- The two automata \( ER_1 \) and \( M/M/1 \) are well formed (exercise).
- \( M/M/1\{a \leftarrow K_a\} \) 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

$M/M/1$

- The two automata $ER_1$ and $M/M/1$ are well formed (exercise).
- $M/M/1\{a \leftarrow K_a\}$ 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

\[ \text{M/M/1} \]

- The two automata \( ER_1 \) and \( M/M/1 \) are well formed (exercise).
- \( M/M/1 \{ a \leftarrow K_a \} \) 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!!!
The two automata $ER_1$ and $M/M/1$ are well formed (exercise).

$M/M/1\{a \leftarrow K_a\}$ 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

\[ M/M/1 \]

- The two automata \( ER_1 \) and \( M/M/1 \) are well formed (exercise).
- \( M/M/1 \{ a \leftarrow K_a \} \) 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!!!

\[ \pi(ER_1 \oplus_a MM1 \{ a \leftarrow K_a \}) = \pi_1(ER)\pi_2(M/M/1) \]
New queue with batches

- Arrival: Poisson process with constant rate $\lambda$;
- We consider the batches of size 2
- Service is exponentially distributed:
  - with rate $\mu_0$, if the queue length is equal to 1;
  - with rate $\mu_1$ for single customer with queue length is > 1;
  - with parameter $\delta$ for bulks of customers of size 2.

Generalisation to service bulk of customer arbitrary size $b$ is possible, but lengthy to present here.
New queue with batches

- Arrival: Poisson process with constant rate $\lambda$;
- We consider the batches of size 2
- Service is exponentially distributed:
  - with rate $\mu_0$, if the queue length is equal to 1;
  - with rate $\mu_1$ for single customer with queue length is $> 1$
  - with parameter $\delta$ for bulks of customers of size 2.

Generalisation to service bulk of customer arbitrary size $b$ is possible, but lengthy to present here.
New queue with batches

- Arrival: Poisson process with constant rate $\lambda$;
- We consider the batches of size 2;
- Service is exponentially distributed:
  - with rate $\mu_0$, if the queue length is equal to 1;
  - with rate $\mu_1$ for single customer with queue length is $> 1$;
  - with parameter $\delta$ for bulks of customers of size 2.

Generalisation to service bulk of customer arbitrary size $b$ is possible, but lengthy to present here.
New queue with batches

- Arrival: Poisson process with constant rate $\lambda$;
- We consider the batches of size 2;
- Service is exponentially distributed:
  - with rate $\mu_0$, if the queue length is equal to 1;
  - with rate $\mu_1$ for single customer with queue length is $> 1$;
  - with parameter $\delta$ for bulks of customers of size 2.

Generalisation to service bulk of customer arbitrary size $b$ is possible, but lengthy to present here.
Graphical representation of the queue

LMA of the queue with bulk size 2

LMA of the queue with bulk size $B = b + 1$
Steady-state distribution

Assumption

Imposing the condition on the weighted outgoing flux

\[ K_a = \frac{\pi_1(n + 2) \delta}{\pi_1(n)} = \kappa' \quad \text{for all } n \quad (1) \]

Geometric steady-state distribution

\[ \pi_1(n) = (1 - \frac{(\lambda_1 - \kappa')}{\mu_0}) \frac{(\lambda_1 - \kappa')^n}{\mu_0^n} \quad (2) \]

only if \( \mu_0 > \mu_1 \) and \( \delta = \mu_0 - \mu_1 \).
Steady-state distribution

Assumption

Imposing the condition on the weighted outgoing flux

\[ K_a = \frac{\pi_1(n + 2)\delta}{\pi_1(n)} = \kappa' \quad \text{for all } n \quad (1) \]

Geometric steady-state distribution

\[ \pi_1(n) = \left(1 - \frac{(\lambda_1 - \kappa')}{\mu_0}\right) \frac{(\lambda_1 - \kappa')^n}{\mu_0^n} \quad (2) \]

only if \( \mu_0 > \mu_1 \) and \( \delta = \mu_0 - \mu_1 \).
Steady-state distribution

**Assumption**

Imposing the condition on the weighted outgoing flux

\[ K_a = \frac{\pi_1(n + 2)\delta}{\pi_1(n)} = \kappa' \quad \text{for all } n \quad (1) \]

**Geometric steady-state distribution**

\[ \pi_1(n) = (1 - \frac{(\lambda_1 - \kappa')}{\mu_0}) \left(\frac{\lambda_1 - \kappa'}{\mu_0}\right)^n \quad (2) \]

only if \( \mu_0 > \mu_1 \) and \( \delta = \mu_0 - \mu_1 \).
Steady-state distribution

Assumption

Imposing the condition on the weighted outgoing flux

\[ K_a = \frac{\pi_1(n + 2)\delta}{\pi_1(n)} = \kappa' \quad \text{for all } n \]  \hspace{1cm} (1)

Geometric steady-state distribution

\[ \pi_1(n) = (1 - \frac{(\lambda_1 - \kappa')}{\mu_0})(\frac{(\lambda_1 - \kappa')}{\mu_0})^n \]  \hspace{1cm} (2)

only if \( \mu_0 > \mu_1 \) and \( \delta = \mu_0 - \mu_1 \).
Meaning of $\kappa'$

How do we find $\kappa'$?

$\kappa'$ is the smallest positive solution of the equation:

$$
\kappa' = \frac{(\lambda_1 - \kappa')^2}{\mu_0^2} \delta
$$

General case size $B$ an equation of degree $B$ is obtained, which has always a positive solution smaller than $\lambda$.

The role of $\kappa'$ in the workload

- Since $\frac{(\lambda_1 - \kappa')^2}{\mu_0} < 1$ then $\lambda > \mu_0$.
- Take an $M/M/1$ with arrival rate $\lambda$ and service rate $\mu_0$ then $\lambda < \mu_0$. 
Meaning of $\kappa'$ in graphs

Steady-state distribution of customers in the queue with different values of $\kappa'$. From right to left: $\kappa' = 0.009, \kappa' = 0.1, \kappa' = 0.2, \kappa' = 0.3$. 
Meaning of $\kappa'$ in graphs

Steady-state distribution of customers in the queue with different values of $\kappa'$. From right to left: $\kappa' = 0.009, \kappa' = 0.1, \kappa' = 0.2, \kappa' = 0.3$. 
Meaning of $\kappa'$ in graphs

Steady-state distribution of customers in the queue with different values of $\kappa'$. From right to left: $\kappa' = 0.009$, $\kappa' = 0.1$, $\kappa' = 0.2$, $\kappa' = 0.3$. 
Meaning of $\kappa'$ in graphs

Steady-state distribution of customers in the queue with different values of $\kappa'$. From right to left: $\kappa' = 0.009, \kappa' = 0.1, \kappa' = 0.2, \kappa' = 0.3$. 
Meaning of $\kappa'$ in graphs

Steady-state distribution of customers in the queue with different values of $\kappa'$. From right to left: $\kappa' = 0.009, \kappa' = 0.1, \kappa' = 0.2, \kappa' = 0.3$. 
New product form solution

Mixed Network with feedback

\[ \gamma_1 = \lambda_1 + p(\lambda_2 + \kappa') \]
\[ \gamma_2 = \lambda_2 + \kappa' \]

Product form solution

\[ \pi_1(n, m) = \pi_1(n)\pi_2(m) \]
\[ = (1 - \rho_1)\rho^n(1 - \rho_2)\rho^m \]

where \( \rho_1 = \frac{\gamma_1}{\mu_0} \) and \( \rho_2 = \frac{\gamma_2}{\mu_2} \)
New product form solution

Mixed Network with feedback

\[
\begin{align*}
\gamma_1 &= \lambda_1 + p(\lambda_2 + \kappa') \\
\gamma_2 &= \lambda_2 + \kappa'
\end{align*}
\]

Product form solution

\[
\begin{align*}
\pi_1(n, m) &= \pi_1(n)\pi_2(m) \\
&= (1 - \rho_1)\rho^n(1 - \rho_2)\rho^m
\end{align*}
\]

where \( \rho_1 = \frac{\gamma_1}{\mu_0} \) and \( \rho_2 = \frac{\gamma_2}{\mu_2} \)
Conclusion

Where are the strengths of the method presented?

- **Queuing network** You can take any queue that satisfy the conditions of our theorem and form a network that preserves product-form solution.

- **Markov chains** The conditions are general enough to capture a very large class of models that do not fall into queueing theory.
Conclusion

Where are the strengths of the method presented?

► **Queuing network** You can take any queue that satisfy the conditions of our theorem and form a network that preserves product-form solution.

► **Markov chains** The conditions are general enough to capture a very large class of models that do not fall into queuing theory.
We have shown how to derive product form solution for general complex systems-starting from the components.

Equally given a complex system, where we have a rough idea of the components, we know how to shape each sub-chain to obtain product-form solution.

New product-form solutions in queues can be found - see [Balbo-Vigliotti ’10].

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.
We have shown how to derive product form solution for general complex systems-starting from the components.

Equally given a complex system, where we have a rough idea of the components, we know how to shape each sub-chain to obtain product-form solution.

New product-form solutions in queues can be found - see [Balbo-Vigliotti ’10].

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.
We have shown how to derive product form solution for general complex systems-starting from the components.

Equally given a complex system, where we have a rough idea of the components, we know how to shape each sub-chain to obtain product-form solution.

New product-form solutions in queues can be found - see [Balbo-Vigliotti ’10].

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.
We have shown how to derive product form solution for general complex systems-starting from the components.

Equally given a complex system, where we have a rough idea of the components, we know how to shape each sub-chain to obtain product-form solution.

New product-form solutions in queues can be found - see [Balbo-Vigliotti ’10].

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.
We have shown how to derive product form solution for general complex systems-starting from the components.

Equally given a complex system, where we have a rough idea of the components, we know how to shape each sub-chain to obtain product-form solution.

New product-form solutions in queues can be found - see [Balbo-Vigliotti ’10].

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


