On the Solution of Cooperating Stochastic Models

Gian-Luca Dei Rossi

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari, Venezia

December 13, 2013

Publications and reports

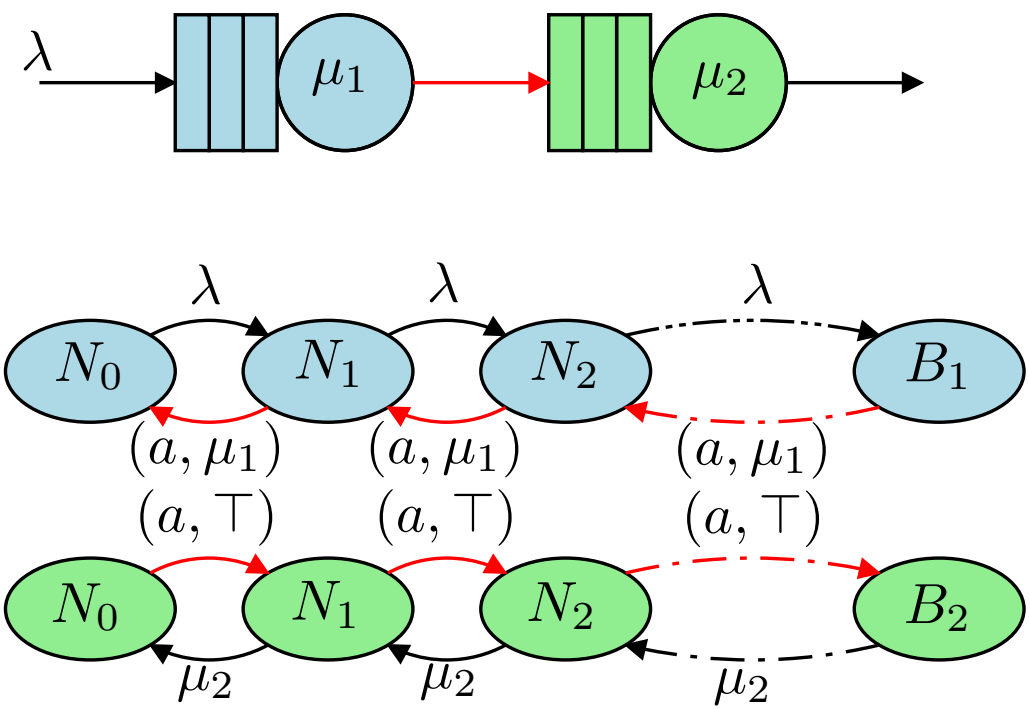
- S. Balsamo, G. Dei Rossi, A. Marin: *Queueing networks and conditional product-forms*. VALUETOOLS 2013: 7th Int. Conf. on Performance Evaluation Methodologies and Tools. ACM, to appear.
- G. Dei Rossi, L. Gallina, S. Rossi: *Performance analysis and formal verification of cognitive wireless networks*. EPEW 2013: 10th European Workshop on Performance Engineering, LNCS 8168, pp 236–250, Springer, ISSN 0302-9743, ISBN 978-3-642-40724-6.
- S. Balsamo, G. Dei Rossi, A. Marin: *Modelling Retrial-upon-conflict Systems with Product-form Stochastic Petri Nets*. ASMTA 2013: Analytical and Stochastic Modeling Techniques and Applications, LNCS 7984, pp. 52–66, Springer, ISSN 0302-9743, ISBN 978-3-642-39407-2.
- S. Balsamo, G. Dei Rossi, A. Marin, *Efficient solutions for cooperating automata based on forward and reversed lumping*. DAIS research report DAIS-2012-6, October 2012.
- S. Balsamo, G. Dei Rossi, A. Marin: *A Survey on Multi-Formalism Performance Evaluation Tools*. Proc. of the 26th annual European Simulation and Modelling Conf., ESM 2012, Essen, DE, pp. 15–23. Eurosis, ISBN 978-90-77381-73-1.
- E. Barbierato, G. Dei Rossi, M. Gribaudo, M. Iacono, A. Marin: *Exploiting product form solution techniques in multiformalism modeling*. PASM 2012: Sixth Int. Workshop on Practical Applications of Stochastic Modelling, ENTCS 296, pp. 61–77, Elsevier, ISSN 1571-0661.
- L. Gallina, G. Dei Rossi, A. Marin, S. Rossi: *Evaluating Resistance to Jamming and Casual Interception in Mobile Wireless Networks*. Proc. of MSWIM 2012: The 15th ACM Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp.151–158 ACM, ISBN 978-1-4503-1628-6.
- S. Balsamo, G. Dei Rossi, A. Marin: *Cooperating stochastic automata: approximate lumping an reversed process*. ISCIS 2012: 27th Int. Symposium on Computer and Information Sciences, Paris, FR. In Computer and Information Sciences III, pp. 131–141, Springer, ISBN 978-1-4471-4593-6 / 978-1-4471-4594-3.
- S. Balsamo, G. Dei Rossi, A. Marin: *Lumping and Reversed Processes in Cooperating Automata*. ASMTA 2012: Analytical and Stochastic Modeling Techniques and Applications, LNCS 7314/2012, pp. 212–226, Springer, ISSN 0302-9743, ISBN 978-3-642-30781-2.
- G. Dei Rossi, A. Marin, S. Balsamo: *A queueing model with batch arrivals for studying the impact of fragmentation in wireless protocols*. In Proc. of the 4th IFIP Wireless Days 2011, Niagara Falls, Ontario, CA. IEEE, ISSN 2156-9711, ISBN 978-1-4577-2028-4.
- S. Balsamo, G.Dei Rossi, A. Marin, Optimisation of Virtual Machine Garbage Collection Policies. ASMTA 2011: Analytical and Stochastic Modeling Techniques and Applications, LNCS 6751/2011, pp. 70–84 , Springer, ISSN 0302-9743, ISBN 978-3-642-21712-8. [Slides]
- G.Dei Rossi, A. Marin, M. Rosati, S. Balsamo: *A Simulation Package for an Energy-Aware Comparison of ARQ Protocols*. Proc. of the 9th Annual Industrial Simulation Conf., ISC 2011, Venice, IT, pp. 18–25, EUROSIS, ISBN 978-90-77381-63-2. [Slides]
- S. Balsamo, G. Dei Rossi, A. Marin: *Applying BCMP Multi-class Queueing Networks for the Performance Evaluation OF Hierarchical and Modular Software Systems*. Proc. of the 24th annual European Simulation and Modelling Conf., ESM 2010, Hasselt, BE, pp. 206–213, EUROSIS, ISBN 978-90-77381-57-1.
- S. Balsamo, G. Dei Rossi, A. Marin: *A numerical algorithm for the solution of product-form models with infinite state spaces*. EPEW 2010: Comp. Perf. Eng., LNCS 6342/2010, pp. 191–206, Springer, ISSN 0302-9743, ISBN 978-3-642-15783-7. Errata.
- S. Balsamo, G. Dei Rossi, A. Marin: *A tool for the numerical solution of cooperating Markov chains in product-form*. Proc. Of HET-NETs 2010, Zakopane, Poland, January 2010. pp. 311—324, IITIS, ISBN 978-83-926054-4-7

Presentation outline

Cooperating stochastic models

- Models with underlying Continuous Time Markov Chain (CTMC)
- Exploitation of compositionality in model definition
  - Each component is specified in isolation
  - Semantics of cooperation is defined so that the joint model can be algorithmically derived
- Stochastic automata considered here synchronise on the active/passive semantics
  - PEPA/EMPA active/passive synchronisation
  - Buchholz's Communicating Markov Processes
  - Plateau's stochastic automata networks (SAN) with master/slave cooperation
  - ...

Example: tandem of queues with finite capacity and repetitive service blocking



Internal transitions

Synchronising transitions

Motivation

- In general, the state-space's size of the joint model grows exponentially with the number of components
- Steady-state analysis becomes quickly unfeasible
  - Space cost
  - Time cost
  - Numerical stability issues
- Workarounds
  - Approximate analysis (e.g. fluid)
  - Exploitation of the geometry of the state space
  - Product-form decomposition
  - Lumping
  - ...

Product-form models

- Consider model S consisting of sub-models $S_1, \ldots, S_N$
- Let $m = (m_1, \ldots, m_N)$ be a state of model S and $\pi(m)$ its steady state probability
- S is in product-form with respect to $S_1, \ldots, S_N$ if:

$$\pi(m) \propto g_1(m_1) \cdot g_2(m_2) \cdots g_N(m_N)$$

  where $g_i(m_i)$ is the steady state probability distribution of S_i appropriately parametrised
- The size of the state space of S is proportional to the product of the state space cardinalities of its sub-models $\Rightarrow$ product-form models can be studied more efficiently!

Some problems...

- How to decide if a model yields a product-form solution?
  - list of instances: BCMP theorem, Coleman/Henderson Stochastic Petri Nets, G-networks...
  - general criteria: Markov implies Markov property, RCAT (and extensions),...
- How to find the correct parametrisation for the sub-models?
  - solving the linear system of traffic equations for BCMP/Jackson queueing networks
  - solving the non-linear traffic equations for G-networks
  - solving the system of equations of RCAT

Sketch of the results: INAP+

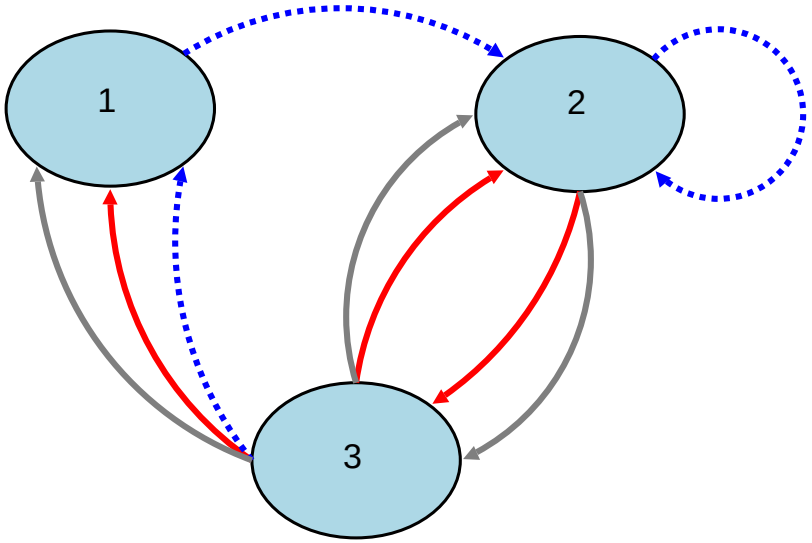
In [Balsamo et al., 2010a] we proposed an algorithm that...

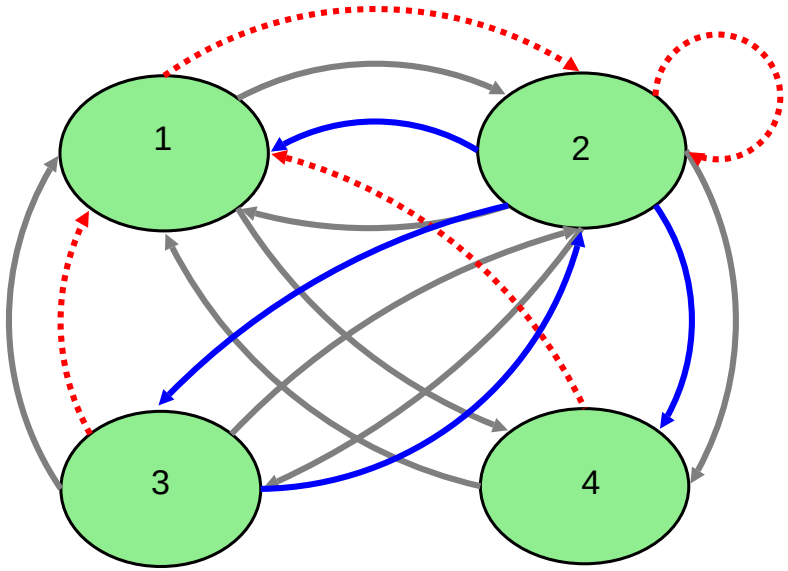
- decides if a model S has a product-form solution with respect to a set of sub-models $S_1, \ldots, S_N$
- derive the correct parametrisation for the sub-models
- compute the unnormalised steady-state solution

The Iterative Numerical Algorithm for product-forms (INAP+) is based on the Reversed Compound Agent Theorem (RCAT) [Harrison, 2003] and its extensions.

- Originally introduced in [Marin and Bulò, 2009] for models with finite state spaces (INAP)
- The improvements in INAP+ [Balsamo et al., 2010a] are:
  - possibility to deal with infinite state space models thanks to a dynamic truncation
  - faster convergence time (experimental results)
  - possibility to express the iterations in Matrix-form (simplified implementation)

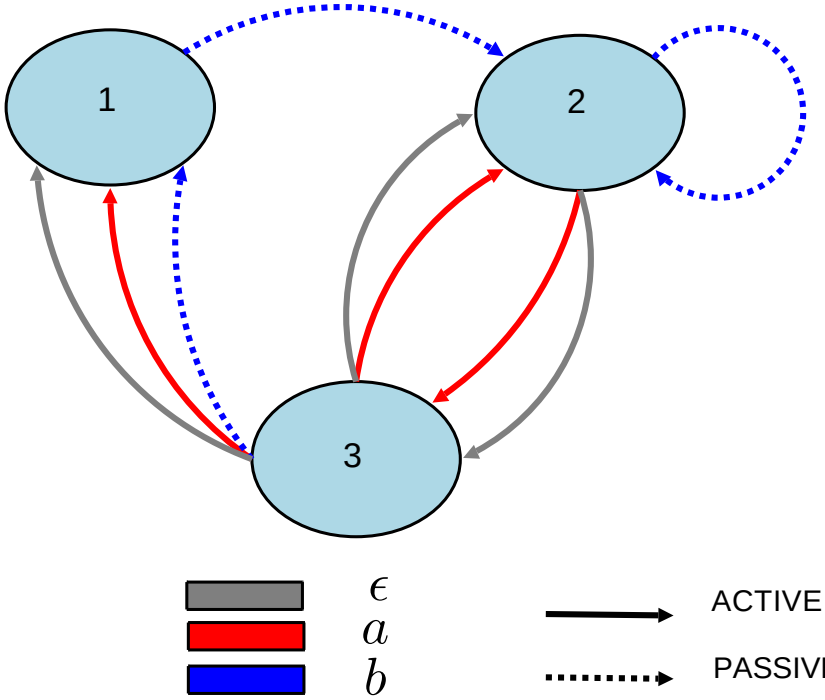
Pairwise interacting agents

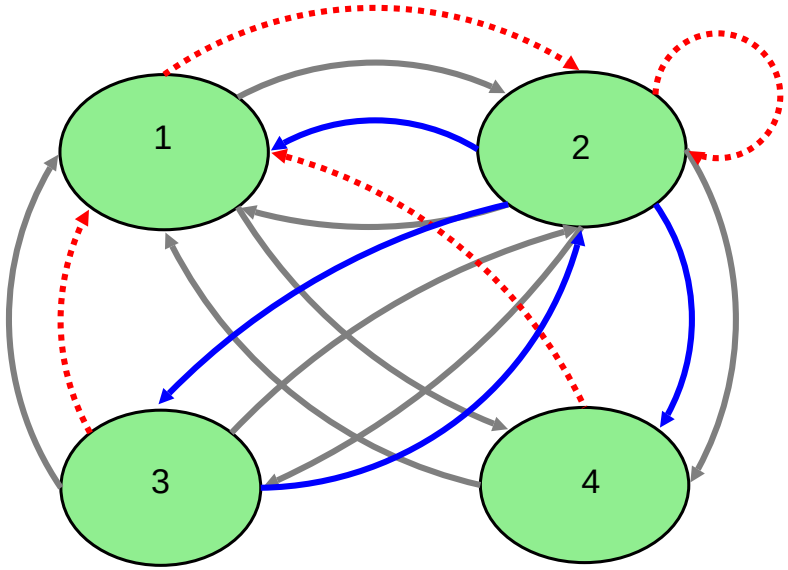




- PEPA-like cooperation with an active and a passive agent
- Active transitions have a rate
- Passive transitions have a unspecified rate
- Active/Passive transitions occur only simultaneously with the rate of the active ones
- A sub-model may be passive with respect to a cooperation and active WRT another

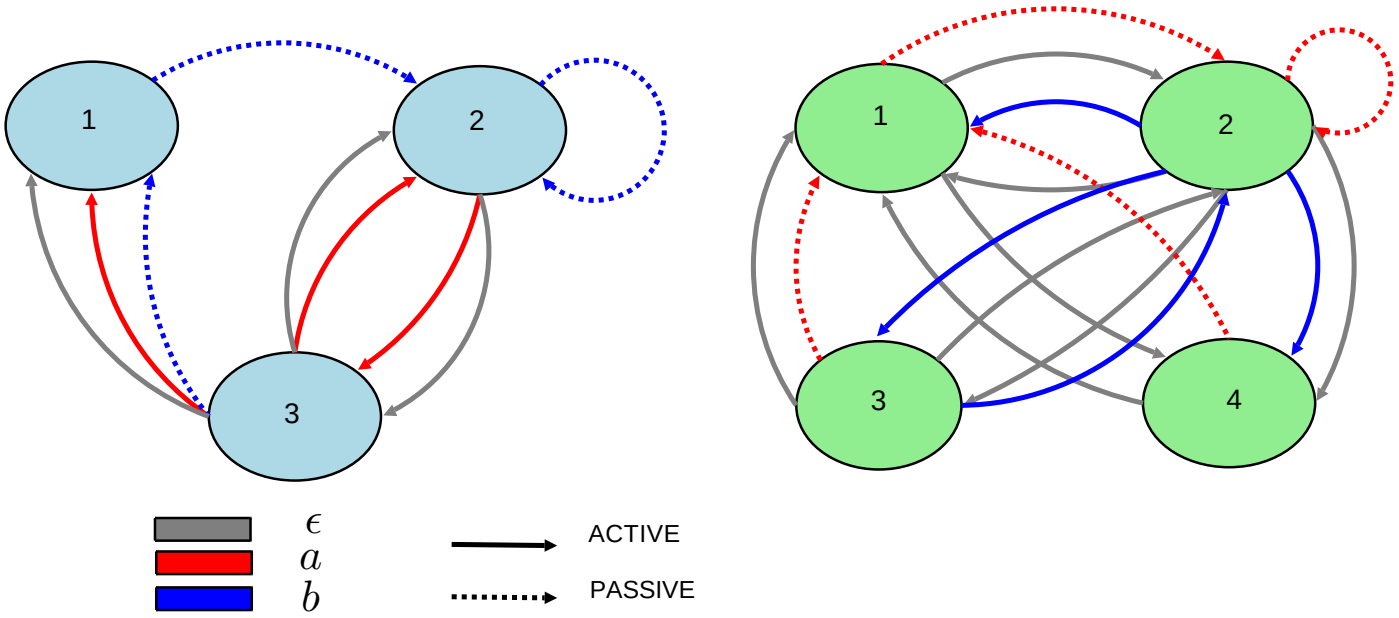
RCAT conditions





- Passive transitions outgoing from every state
- Active transitions incoming into every state
- Same reversed rate for all active transitions

RCAT parametrisation and solution



- Parametrisation: replace all the passive transitions with the reversed rate [1] of the corresponding active transitions
- Solution: let g_1 and g_2 be the solution of the parametrised agents, then the solution π of the cooperating agent is in product-form:

$$\pi \propto g_1 \cdot g_2$$

[1] $\overline{q_{ij}} = \frac{\pi(i)}{\pi(j)} q_{ij}$

Non-optimised algorithm

Initialize randomly g_k for a truncation of $\mathcal{S}'_k$ for all $k = 1, \ldots, N$
$n = 0$
repeat
  Compute the reversed rates $K_a^{(n)}$ for the active labels
  Use $K_a^{(n)}$ to close and truncate S_k for all $k = 1, \ldots, N$
  Update g_k for all $k = 1, \ldots, N$
  $n \leftarrow n + 1$
until $n > M$ **or** $\forall k = 1, \ldots, N. |g_k - g_k^{prev}| < \epsilon$;
if *the reversed rates are not constant* **then**
  fail: RCAT product-form not identified
return $\{g_k\}_{k=1,\ldots,N}$

- N: number of agents
- $\mathcal{S}_k$, $1 \leq k \leq N$: state space of agent k
- g_k, $1 \leq k \leq N$: hypothetical stationary distribution of agent k
- n: number of the iteration being performed
- g_k^{prev}: stationary distribution computed at step $n - 1$
- M: maximum number of iterations
- ϵ: tolerance

Computing new g_k and convergence

Computing g_k and product-forms

- g_k are computed using the global balance equation system
- If g_k at step n are identical to g_k at step $n-1$
  - Constant reversed rates for active transitions $\Rightarrow$ product-form solution found
  - Otherwise $\Rightarrow$ no product-form found

Convergence

- Convergence has been proved for specific cases
- A maximum number of iterations is used to avoid infinite loops

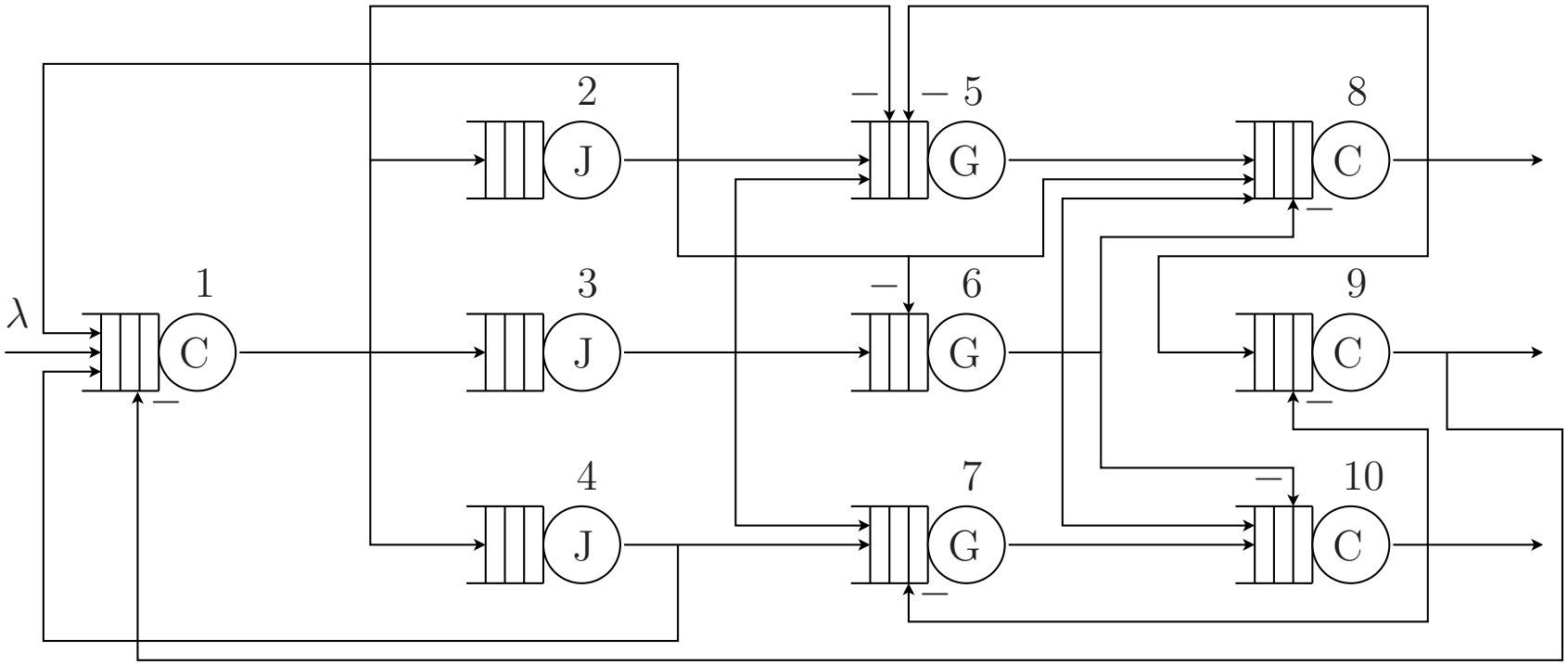
Computing the reversed rates

- Even if the RCAT product-form exists during the algorithm iteration the reversed rates of the active transitions may be non-constant
- How to compute rates $K_a^{(n)}$ for each synchronising label a at iteration n?
- $K_a^{(n)}$ is defined as the weighted mean of the reversed rates of all the active transitions labelled by a
- The weight of transition $\alpha \xrightarrow{a} \beta$ is $g_i(\beta)^{(n)}$
- Hence:

$$K_a^{(n)} = \sum_{\alpha \in \mathcal{S}_i} \frac{g_i^{(n)}(\alpha)}{g_i^{(n)}(\beta)} q_i(\alpha \xrightarrow{a} \beta) g_i^{(n)}(\beta) \quad (1)$$

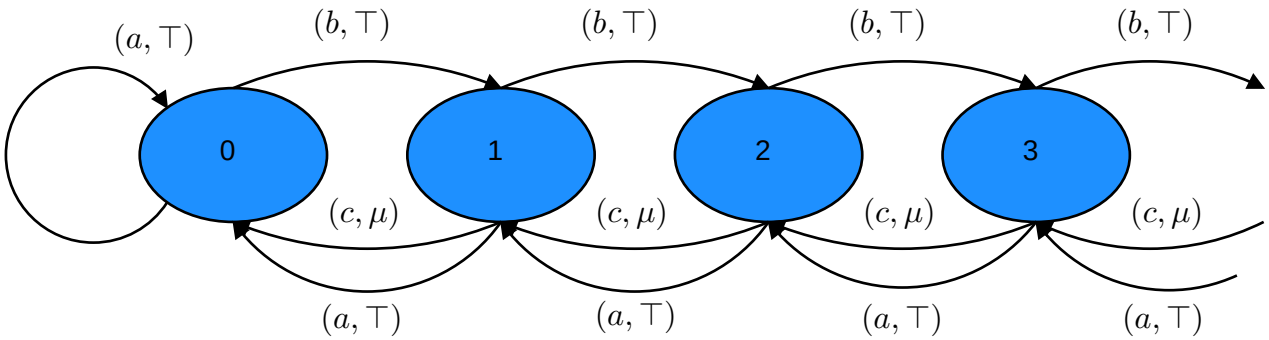
$$\approx \sum_{\alpha \in \mathcal{R}_i^{\tau}(\mathcal{S}_i)} g_i^{(n)}(\alpha) q_i(\alpha \xrightarrow{a} \beta) \quad (2)$$

Mix of negative customer and catastrophes triggers



- J: Jackson queue, C: Queue with catastrophes, G: G-queue with negative customers

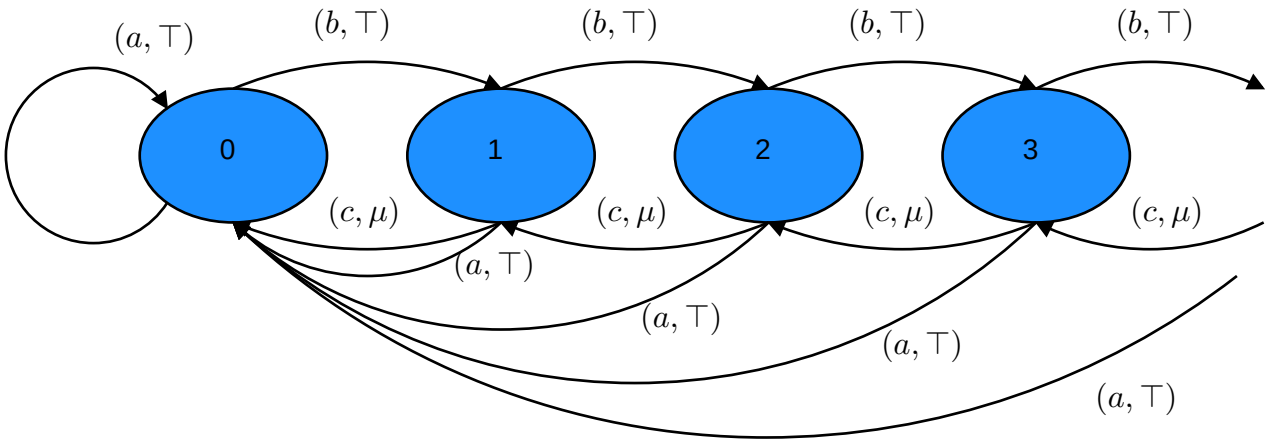
Queue with negative customers



- Transitions a cause a customer deletion, b a customer arrival
- Geometric distribution $\pi(n) \propto \rho^n$

$$\rho = \frac{K_b}{\mu + K_a}$$

Queue with catastrophes



- Transitions a cause a catastrophe, b a customer arrival
- Geometric distribution $\pi(n) \propto \rho^n$

$$\rho = \frac{K_b + \mu + K_a - \sqrt{K_b^2 + \mu^2 + K_a^2 + 2K_bK_a + 2\mu K_a - 2K_b\mu}}{2\mu}$$

Final remarks on INAP+

- Fast convergence (experimental result):
  - 6 iterations for the considered example, with a tolerance of 10^{-5}
- Convergence proved only for special cases
- Can solve heterogeneous models (consisting of Petri net blocks, and various types of queues)
- Complexity of K models with N states each: $\mathcal{O}(N^3 + K^2 N)$ for each iteration

Future works

- Proof of convergence for more general cases
- Extension of both the formalism and the algorithm to include non-pairwise cooperations (e.g. G-networks with positive and negative triggers)
- How to easily express the operator of truncation?
- Approximated product-form analysis based on [Marin and Vigliotti, 2010a, Marin and Vigliotti, 2010b]
- Development of a tool with a simple GUI to allow the user to analyse heterogeneous models...
  - ... oh, wait, we have done it :-)

A tool for the solution of product-form models

In [Balsamo et al., 2010b] we proposed a tool that...

- decides if a model M has a product-form solution with respect to a set of sub-models $M_1, \ldots, M_N$
- derives the correct parametrisation for the sub-models
- computes the unnormalised steady-state solution (INAP, INAP+)
- allows for a modular composition of the sub-models using a system or renaming
  - The user can create or use a library of sub-models that *may* be in product-form
  - sub-models are created without any knowledge about the model in which they will be instantiated
  - the same sub-model can be instantiated several times within the same model
  - composition rules avoid conflicts among labels

Name policies

Each sub-models consists of:

- a set of transitions (ϵ) that cannot synchronize
- a finite number of sets of labelled transitions, which can synchronise.

A cooperation between two sub-models requires to specify:

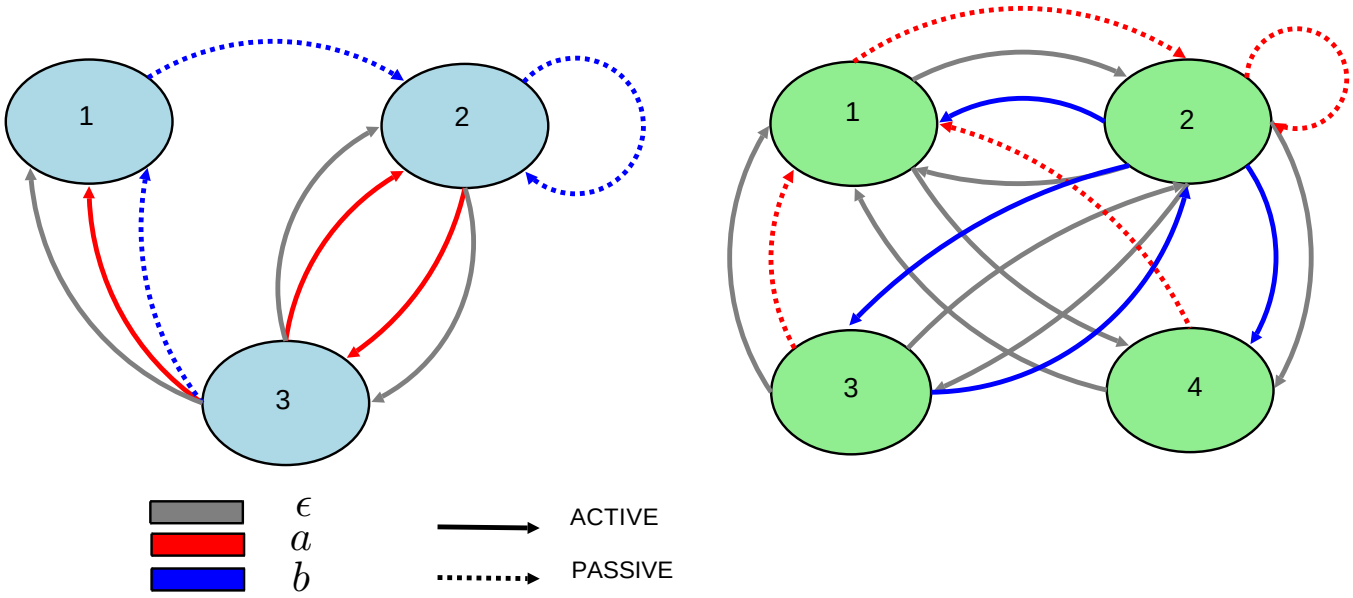
- name of the instance of the active sub-model
- name of the instance of the passive sub-model
- name assigned to the transitions corresponding to the cooperation in the joint model

Example:

$$M_1 \underset{(a,b)}{\overset{c}{\times}} M_2$$

specifies a cooperation between active label a in model M_1 with passive label b in model M_2. The resulting transitions are labelled by c

Example



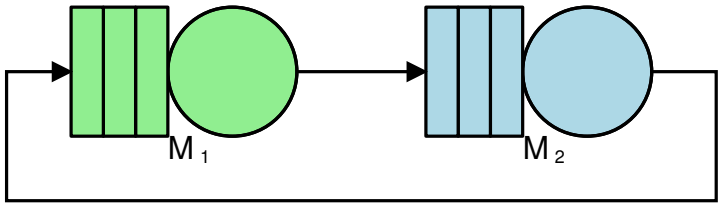
This cooperation can be specified as follows:

$$M_1 \underset{(a,a)}{\overset{c}{\times}} M_2$$

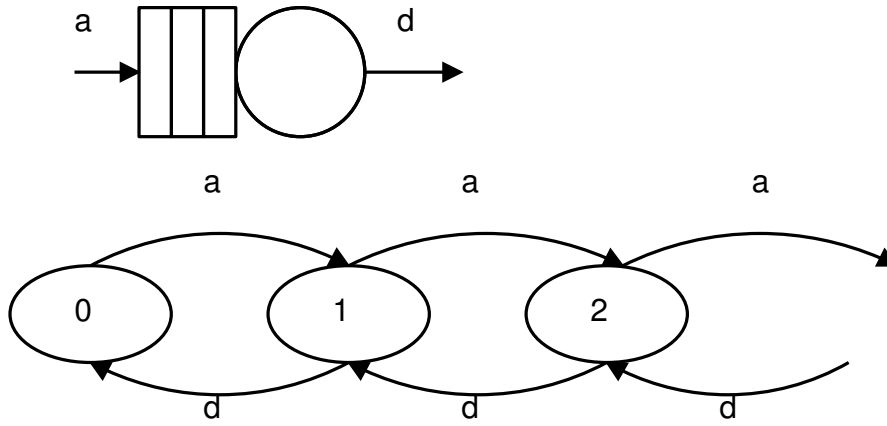
$$M_2 \underset{(b,b)}{\overset{d}{\times}} M_1$$

Example 1-1 cooperation

COOPERATING MODEL (TWO INSTANCES OF THE SAME SUB-MODEL)



SINGLE SUB-MODEL IN LIBRARY



STATE TRANSITION DIAGRAM

This cooperation can be specified as follows (departures are active and arrivals are passive):

$$M_1 \underset{(d,a)}{\overset{s_1}{\times}} M_2 \quad \text{and} \quad M_2 \underset{(d,a)}{\overset{s_2}{\times}} M_1$$

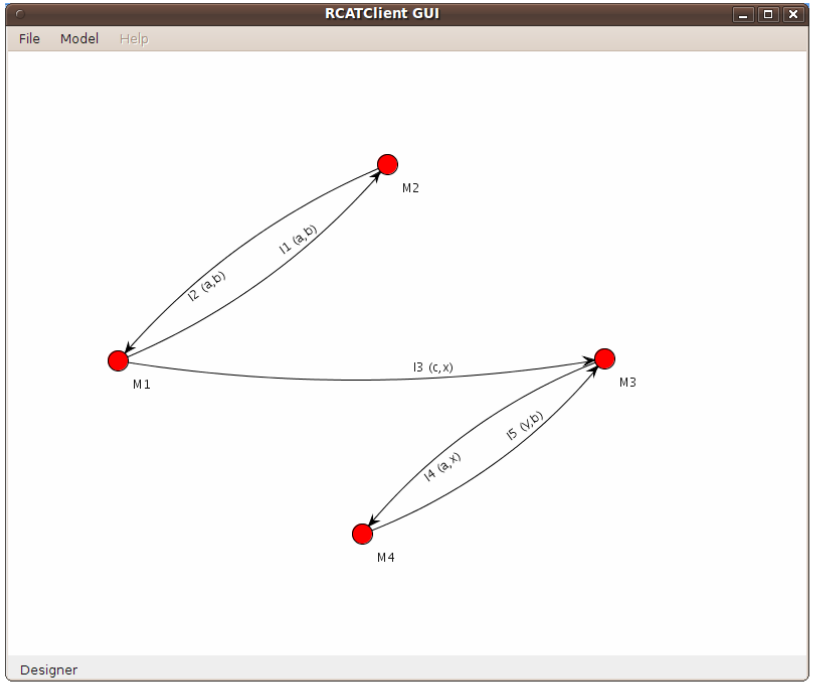
Other cooperation features

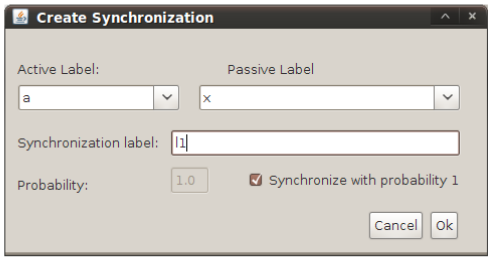
- Probabilistic cooperation
  - an active transition synchronises with a passive one with a certain probability
  - useful to model probabilistic routing in queueing networks
- More than one active label synchronise on the same passive label
  - useful to model multiple arrival flows to the same queueing station
- The modifications on the models required by these types of synchronisations are performed automatically by the tool

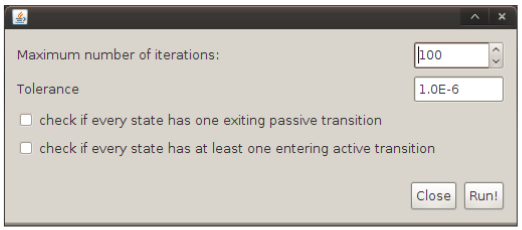
The architecture

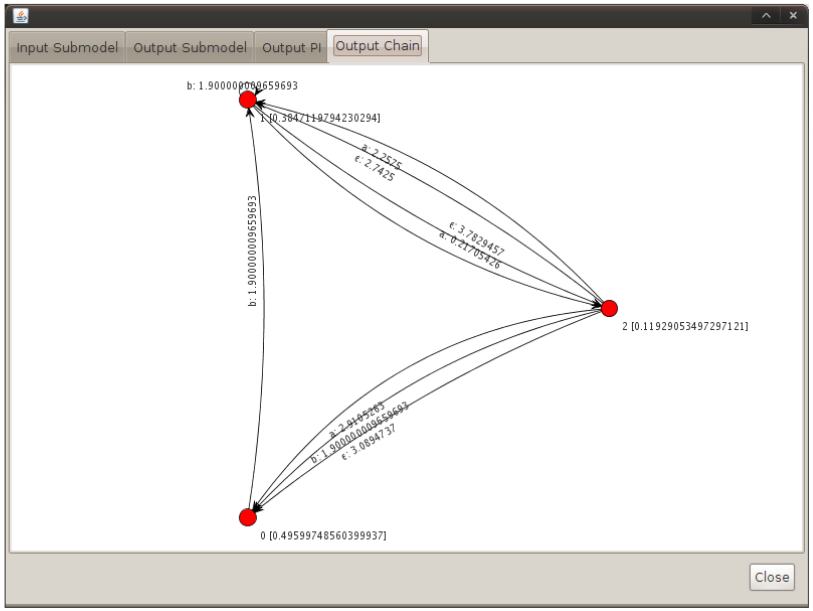
- Client-server architecture
- Server:
  - keeps in memory the sub-model instances
  - stores the cooperations (possibly modifying the instances)
  - runs the algorithm to compute the stationary solution
  - allows for multiple, independent, parallel sessions
- Client:
  - translates the user-friendly operations performed by the user into commands for the server
  - several clients can be created using different interfaces

A formalism-agnostic product-form analyser









Some remarks

- We presented a tool to detect and solve product form models
  - Application: integration with SIMTHESys and MARCAT conditions checking [Barbierato et al., 2012] for multi-formalism performance modelling and analysis.
- What if the model is NOT in product form?
- We can still try to reduce the state space through lumping
- We found some interesting relations between lumping and product form

Condition of lumping for CTMC

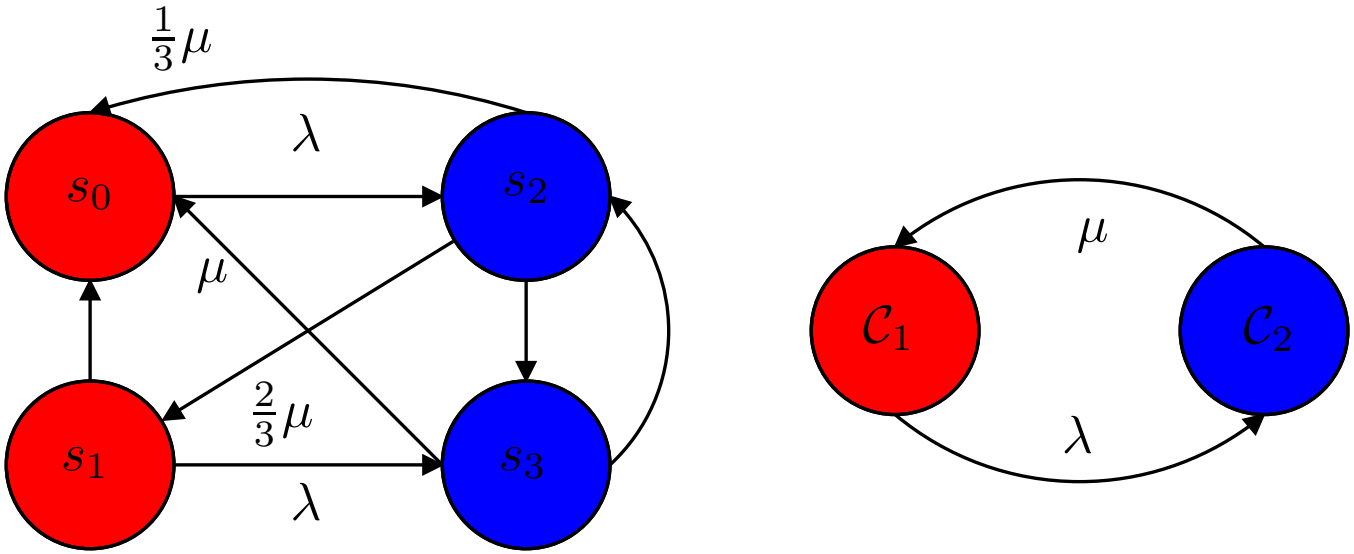
- Notation:
  - $\mathcal{S}$ state space
  - $s \in \mathcal{S}$ state
  - $q(s \to s')$ transition rate from s to s'
  - $\mathcal{C} = (\mathcal{C}_1, \mathcal{C}_2.., \mathcal{C}_N)$ partition of states
- $\mathcal{C}$ induces a lumping on the chain if:

$$\forall \mathcal{C}_i, \mathcal{C}_j, i \neq j, \forall s \in \mathcal{C}_i \sum_{s' \in \mathcal{C}_j} q(s \to s') = \tilde{q}(\mathcal{C}_i \to \mathcal{C}_j)$$

- If π^* is the steady-state distribution of the lumped process and π that of the original, then:

$$\forall \mathcal{C}_i, \pi^*(\mathcal{C}_i) = \sum_{s \in \mathcal{C}_i} \pi(s)$$

Example



$$\mathcal{C}_1 = \{s_0, s_1\} \qquad \mathcal{C}_2 = \{s_2, s_3\}$$

Main idea

- Following the idea of PEPA *strong equivalence* we aim at lumping the automata before the cooperation with other ones
- We want to reduce the solution complexity for the system of Global Balance Equations
  - Suppose the two automata have both a spate-space of size M
  - Time cost reduces from $\mathcal{O}(M^6)$ to $\mathcal{O}((NM)^3)$, where N is the number of clusters in the lumping
- Intuition: *for each synchronising label the original and lumped automata must behave (in steady-state) equivalently*

Lumped automaton

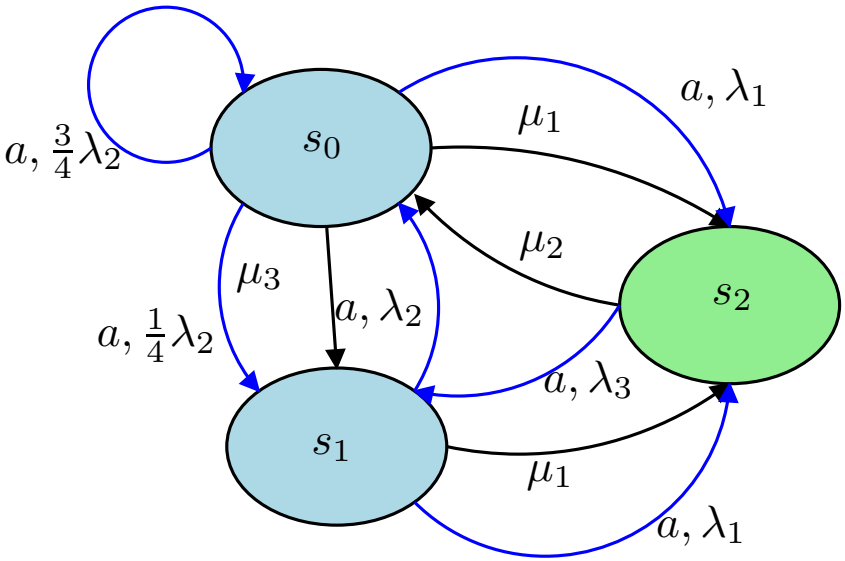
Definition (Lumping condition)

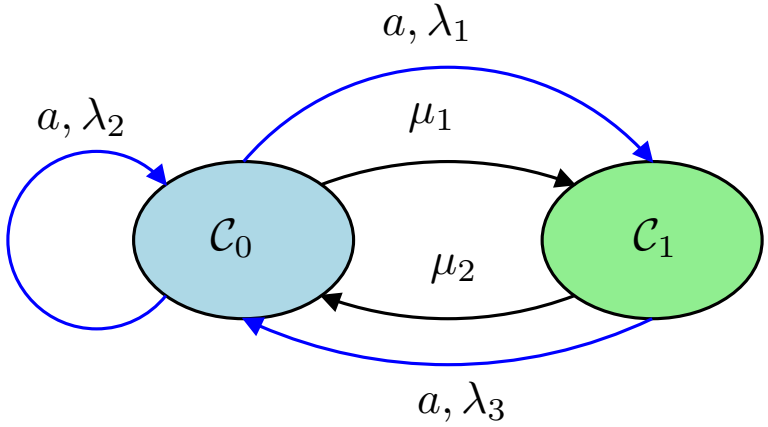
Given active automaton M_1, a set of labels $\mathcal{T}$, and a partition of the states of M_1 into N_1 clusters $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{N_1}\}$, we say that $\mathcal{C}$ is an exact lumping for M_1 if:

1. $\forall \mathcal{C}_i, \mathcal{C}_j,\ \mathcal{C}_i \neq \mathcal{C}_j,\ \forall s_1 \in \mathcal{C}_i \quad \sum_{s_1' \in \mathcal{C}_k} q_1(s_1 \to s_1') = \tilde{q}_1(\mathcal{C}_i \to \mathcal{C}_j)$ not synchronising label
2. $\forall t \in \mathcal{T}, \forall \mathcal{C}_i, \mathcal{C}_j\ ,\ \forall s_1 \in \mathcal{C}_i \quad \sum_{s_1' \in \mathcal{C}_k} q_1^t(s_1 \to s_1') = \tilde{q}_1^t(\mathcal{C}_i \to \mathcal{C}_k)$

- lumped automaton is derived straightforwardly (see next example)
- condition on synchronising labels may generate self-loops in lumped automaton!

Example


s_0
s_1
s_2
$a, \frac{3}{4}\lambda_2$
$a, \frac{1}{4}\lambda_2$
μ_3
a, λ_2
μ_1
a, λ_1
μ_2
a, λ_3
μ_1
a, λ_1


$\mathcal{C}_0$
$\mathcal{C}_1$
a, λ_2
a, λ_1
μ_1
μ_2
a, λ_3

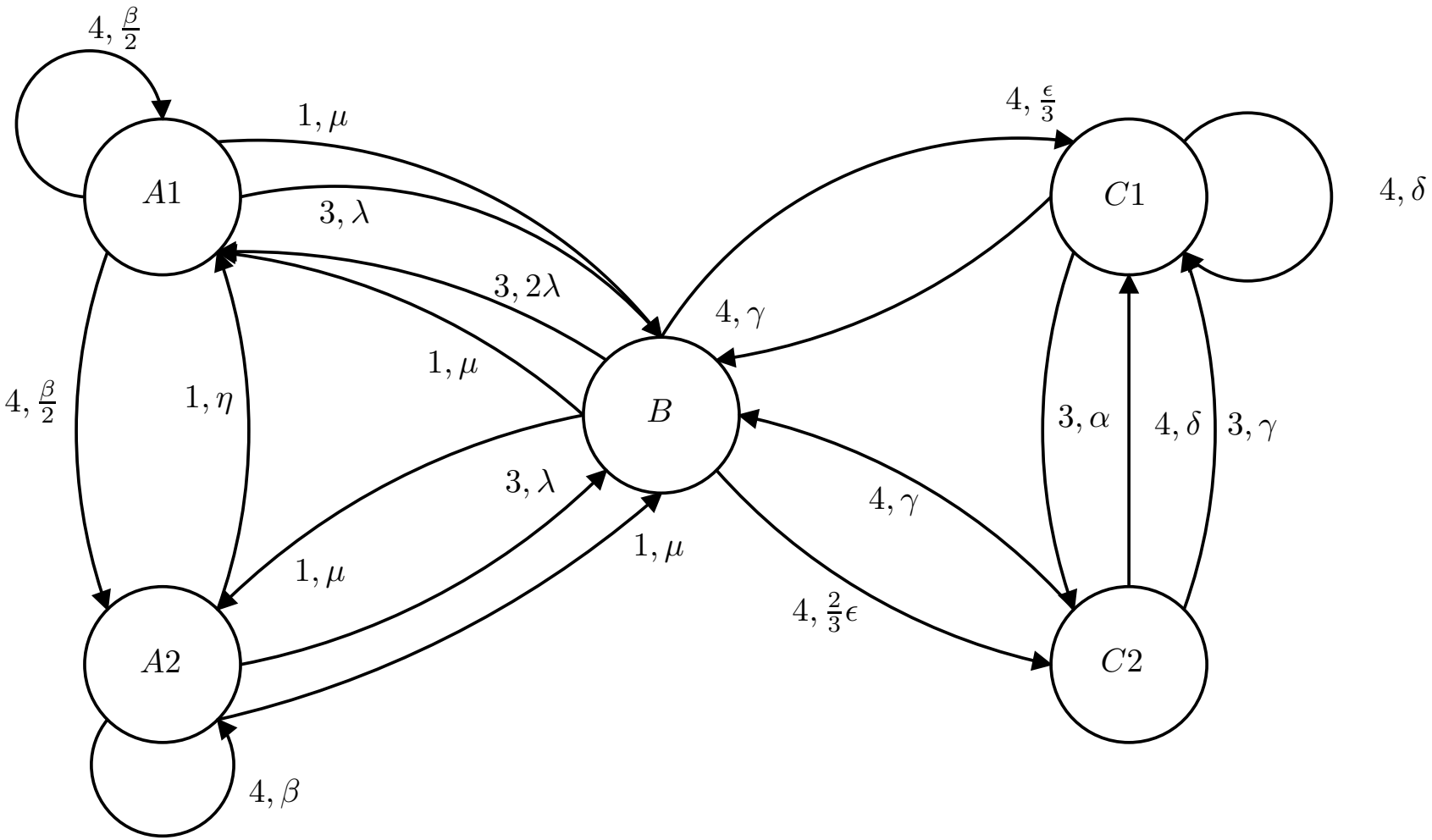
Marginal distribution and lumping

Theorem

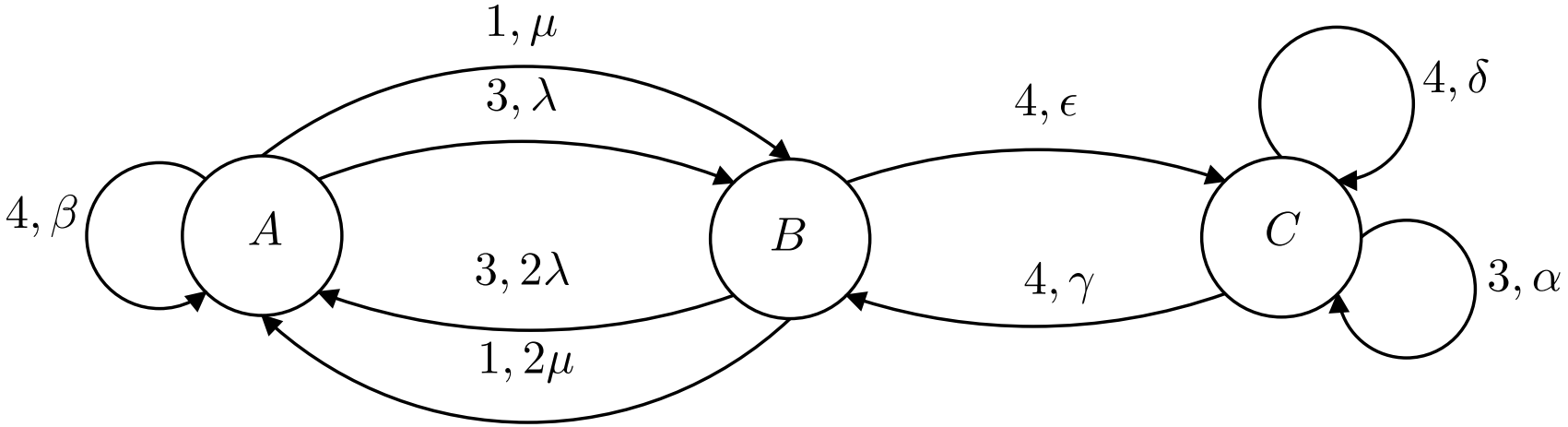
Let M_1 and M_2 be two cooperating automata. If $\tilde{M}_1$ is a lumped automaton of M_1, then the marginal steady state distribution of M_2 in the cooperations $M_1 \otimes M_2$ and $\tilde{M}_1 \otimes M_2$ are the same.

Note that ergodicity is assumed and the state-space of the joint process is the Cartesian product of the single automata state-spaces.

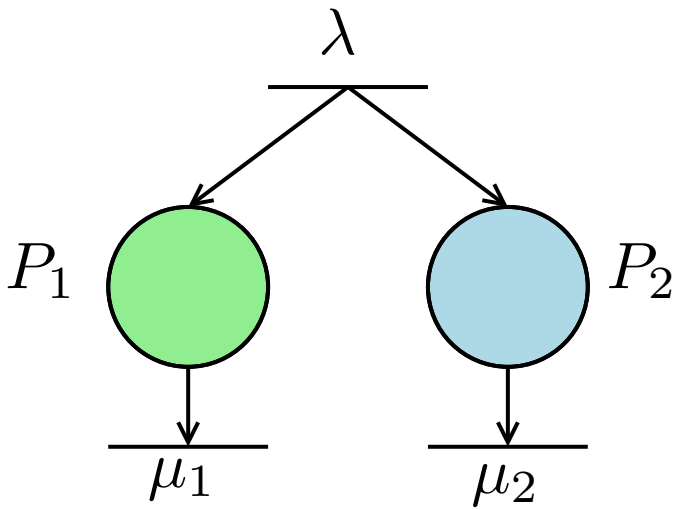
Example: original process

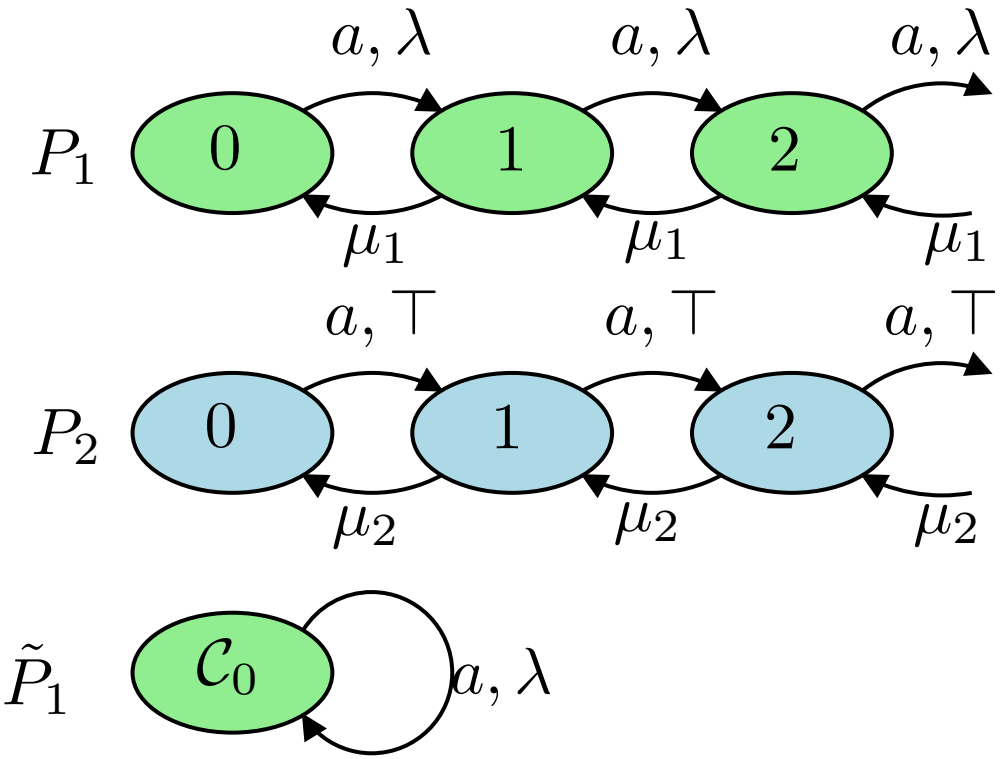


Example: lumped process



A trivial example





$$\pi_1(n) = \left(1 - \frac{\lambda}{\mu_1}\right)\left(\frac{\lambda}{\mu_1}\right)^n \qquad \pi_2(n) = \left(1 - \frac{\lambda}{\mu_2}\right)\left(\frac{\lambda}{\mu_2}\right)^n$$

Reversed processes

- An ergodic CTMC can always be reversed
  - Same state space
  - A transition from s to s' with rate $q(s \to s')$ becomes a transition from s' to s with rate $q^R(s' \to s) = \pi(s)/\pi(s')q(s \to s')$
- The forward and reversed processes share the same steady-state distribution

Conditional product-forms

Theorem (Conditional product-forms)

Given the model $M_1 \otimes M_2$, in a feed-forward and non-blocking synchronisation. Let M_1^R be the reversed automaton of M_1 and let $\tilde{M}_1^R$ be an exact lumping of M_1^R whose clusters are $\mathcal{S} = \{\tilde{1}, \ldots, \tilde{N}_1\}$ and let $\tilde{M}_1^R$ be reversible. Then, under ergodicity assumption, the following $\tilde{N}_1$-order product-form expression holds:

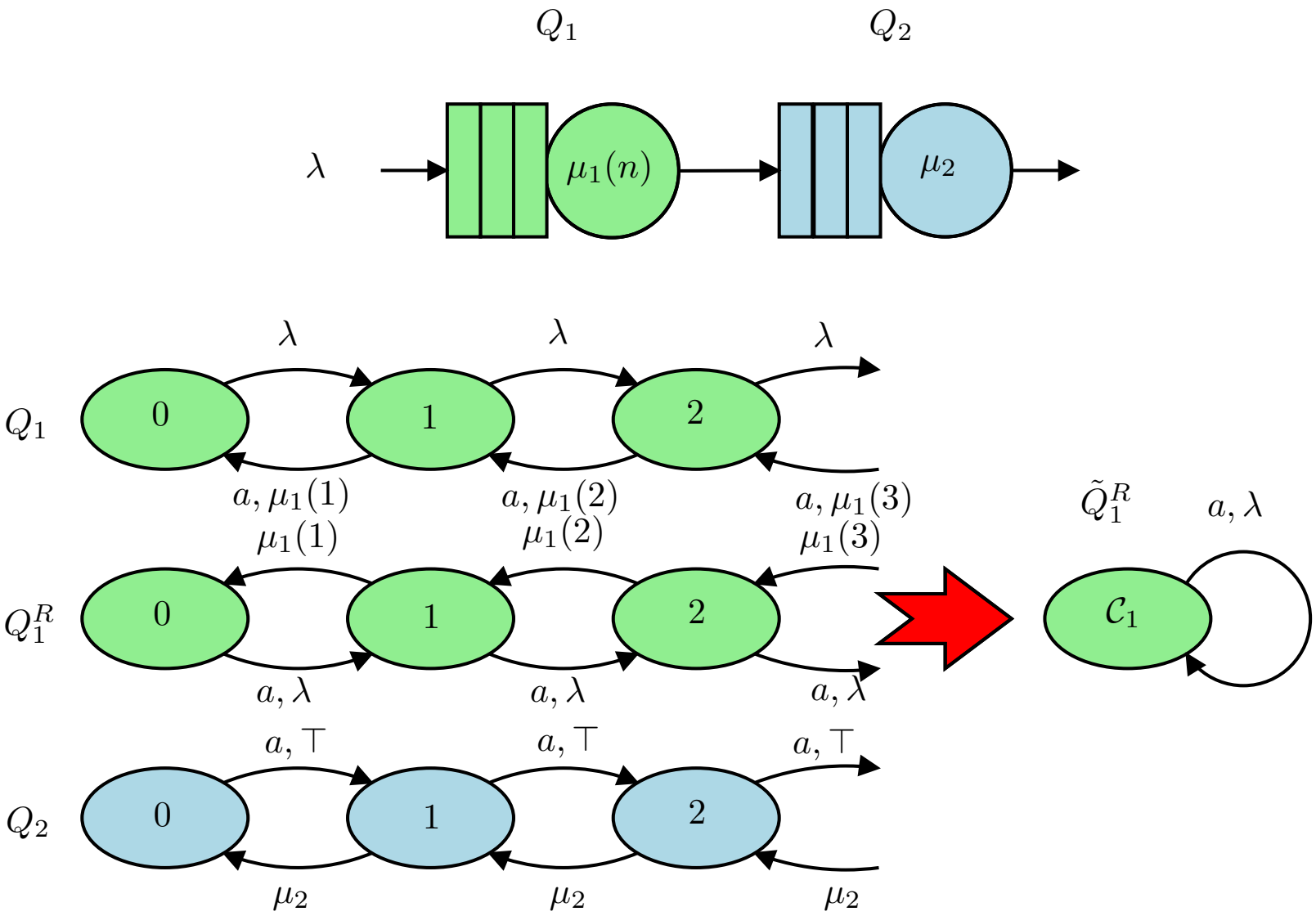
$$\pi(s_1, s_2) = \tilde{\pi}^R_{M_2|\tilde{M}_1^R}(s_2|\tilde{s}_1)\pi_1(s_1)\,, \tag{3}$$

where π is the steady-state distribution of $M_1 \otimes M_2$ and $\tilde{\pi}^R$ that of $\tilde{M}_1^R \otimes M_2$, π_1 that of M_1 and:

$$\tilde{\pi}^R_{M_2|\tilde{M}_1^R}(s_2|\tilde{s}_1) = \frac{\tilde{\pi}^R(\tilde{s}_1, s_2)}{\tilde{\pi}_1(\tilde{s}_1)}\,,$$

where, since the stochastic process underlying $\tilde{M}_1^R$ is a lumping of that underlying M_1^R, we have $\tilde{\pi}_1(\tilde{s}_1) = \sum_{s_1 \in \tilde{s}_1} \pi_1(s_1)$.

Another trivial example



Reversed lumping and product-forms

Corollary (Marginal distributions)

The marginal distribution of M_2 may be computed as:

$$\pi_2(s_2) = \sum_{\tilde{s}_1 = \tilde{1}}^{\tilde{N}_1} \pi(\tilde{s}_1, s_2)\,.$$

Corollary (Product-forms)

A synchronisation is in product-form if the reversed active automaton can be lumped into a single state

Remarks on lumping and CPF

- Relaxation of strong equivalence conditions for component-wise lumping
- Introduction of the concept of *Conditional Product Form*.

Current development

- Application of conditional product form to real-world examples (see [Balsamo et al., 2013]).
- Automatic detection of models in conditional product form.

Approximation of marginal SSD through aggregation

- With our theorem we can reduce the cost to compute marginal steady state distributions of a cooperating automaton *if we're able to find an exact lumping of the other one*.
- What if this is not feasible or even possible?
  - We could try to find an approximated lumping.
  - Can be applied also to the reversed process.
- How to evaluate the quality of an approximation?
- How to adapt clustering algorithms to use our definition of (approximated) exact lumping?

Evaluating the quality of an approximate lumping

How close is an arbitrary state partition $\mathcal{W}$ to an exact lumping?

- We measure the coefficient of variation of the outgoing fluxes $\phi_1^t(s_1)$ of the states in $\tilde{s}_1$.
- We further refine that measurement.

ϵ-error

Definition (ϵ-error)

Given model M_1 and a partition of states $\mathcal{W} = \{\tilde{1}, \ldots, \tilde{N}_1\}$, for all $\tilde{s}_1 \in \mathcal{W}$ and $t > 2$, we define:

$$\begin{aligned} \overline{\phi}_1^t(\tilde{s}_1) &= \frac{\sum_{s_1 \in \tilde{s}_1} \pi_1(s_1)\phi_1^t(s_1)}{\sum_{s_1 \in \tilde{s}_1} \pi_1(s_1)} \\ \epsilon^t(\tilde{s}_1) &= 1 - \exp\left(-\sqrt{\sum_{s_1 \in \tilde{s}_1} \frac{\pi_1(s_1)(\phi_1^t(s_1) - \overline{\phi}_1^t(\tilde{s}_1))^2}{\sum_{s \in \tilde{s}_1} \pi_1(s_1)}}\right). \end{aligned}$$

where $\phi_1^t(s_1) = \sum_{s_1'=1}^{N_1} q_1^t(s_1, s_1')$.

δ-error

Definition (δ-error)

Given model M_1 and a partition of states $\mathcal{W} = \{\tilde{1}, \ldots, \tilde{N}_1\}$, for all $\tilde{s}_1, \tilde{s}'_1 \in \mathcal{W}$, we define:

$$\overline{\varphi}_1^t(\tilde{s}_1, \tilde{s}'_1) = \begin{cases} 0 & \tilde{s}_1 = \tilde{s}'_1 \wedge t = 1 \\ \frac{\left(\sum_{s_1 \in \tilde{s}_1} \pi_1(s_1) \varphi_1^t(s_1, \tilde{s}'_1)\right)}{\sum_{s_1 \in \tilde{s}_1} \pi_1(s_1)} & \text{otherwise} \end{cases}$$

$$\left(\sigma^t(\tilde{s}_1, \tilde{s}'_1)\right)^2 = \sum_{s_1 \in \tilde{s}_1} \frac{\pi_1(s_1)(\varphi_1^t(s_1, \tilde{s}'_1) - \overline{\varphi}_1^t(\tilde{s}_1, \tilde{s}'_1))^2}{\sum_{s \in \tilde{s}_1} \pi_1(s)}$$

$$\delta^t(\tilde{s}_1, \tilde{s}'_1) = 1 - e^{-\sigma(\tilde{s}_1, \tilde{s}'_1)}$$

where $\varphi_1^t(s_1, \tilde{s}'_1) = \sum_{s'_1 \in \tilde{s}'_1} q_1^t(s_1, s'_1)$.

An ideal algorithm

Definition (Ideal algorithm)

- **Input:** automata M_1, M_2, $\mathcal{T}$, tolerances $\epsilon \geq 0$, $\delta \geq 0$
- **Output:** marginal distribution π_1 of M_1; approximated marginal distribution of M_2

1. Find the minimum $\tilde{N}_1'$ such that there exists a partition $\mathcal{W} = \{\tilde{1}, \ldots, \tilde{N}_1'\}$ of the states of M_1 such that $\forall t \in \mathcal{T}, t > 2$ and $\forall \tilde{s}_1 \in \mathcal{W}\ \epsilon(\tilde{s}_1) \leq \epsilon$
2. Let $\mathcal{W}' \leftarrow \mathcal{W}$
3. Check if partition $\mathcal{W}'$ is such that $\forall t \in \mathcal{T}$, $\forall \tilde{s}_1, \tilde{s}_2 \in \mathcal{W}$, $\tilde{s}_1 \neq \tilde{s}_2$, $\delta^t(\tilde{s}_1, \tilde{s}_1') \leq \delta$. If this is true then return the marginal distribution of M_1 and the approximated of M_2 by computing the marginal distribution of $\tilde{M}_1 \otimes M_2$ and terminate.
4. Otherwise, refine partition $\mathcal{W}$ to obtain $\mathcal{W}^{new}$ such that the number of clusters of $\mathcal{W}^{new}$ is greater than the number of clusters in $\mathcal{W}'$. $\mathcal{W}' \leftarrow \mathcal{W}^{new}$. Repeat from Step 3

Constructing the approximate lumped automata

Definition (Approx. lumped automata)

Given active automaton M_1, a set of transition types $\mathcal{T}$, and a partition of the states of M_1 into $\tilde{N}_1$ clusters $\mathcal{W} = \{\tilde{1}, \tilde{2}, \ldots, \tilde{N}_1\}$, then we define the automaton $M_1^{\simeq}$ as follows:

$$\begin{aligned}
\tilde{\mathbf{E}}_{11}(\tilde{s}_1, \tilde{s}_1') &= \begin{cases} \overline{\varphi}_1^1(\tilde{s}_1, \tilde{s}_1')\tilde{\lambda}_1^{-1} & \text{if } \tilde{s}_1 \neq \tilde{s}_2 \\ 0 & \text{otherwise} \end{cases} \\
\tilde{\mathbf{E}}_{12} &= \mathbf{I}, \\
\tilde{\mathbf{E}}_{1t}(\tilde{s}_1, \tilde{s}_1) &= \overline{\varphi}_1^t(\tilde{s}_1, \tilde{s}_1')\lambda_t^{-1} \quad t > 2
\end{aligned}$$

where

$$\tilde{\lambda}_t = \max_{\tilde{s}_1 = 1, \ldots, \tilde{N}_1} \left(\sum_{\tilde{s}_1'=1}^{\tilde{N}_1} \overline{\varphi}_1^t(\tilde{s}_1, \tilde{s}_1') \right)$$

are the rates associated with the transition types in the cooperation between $M_1^{\simeq}$ and M_2.

Initial clustering and refinement phase

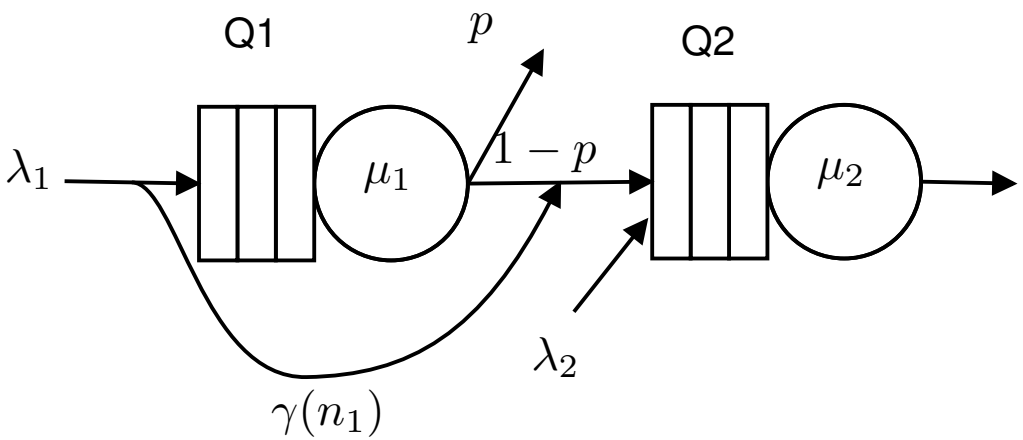
Initial clustering:

- similarity measure can be Euclidean distance between $(\phi_1^3(s_1), \ldots, \phi_1^T(s_1))$ and $(\phi_1^3(s_1'), \ldots, \phi_1^T(s'))$
- can be implemented using various algorithm
  - hierarchical clustering
  - K-means (but number of clusters must be decided a priori...)
  - ...

Refinement phase:

- using the tolerance constant δ
- distances between clusters depend on clusters themselves $\Longrightarrow$ K-means cannot be used.
- spectral analysis or iterative algorithms

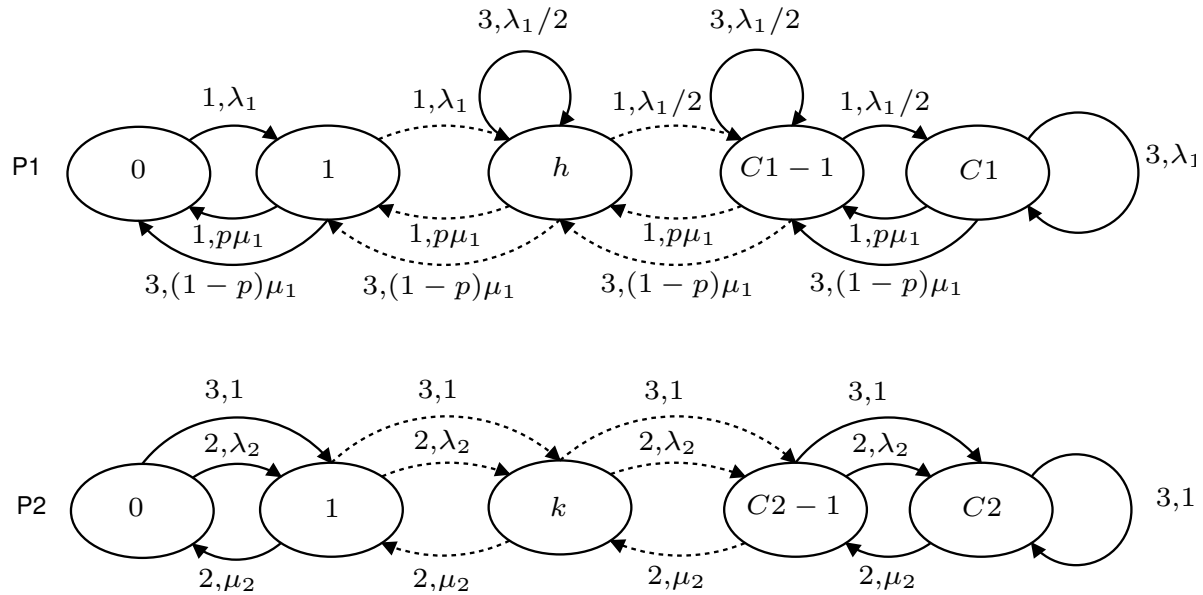
Example



where

$$\gamma(n_1) = \begin{cases} 0 & \text{if } n_1 \leq \left\lfloor \frac{C1}{2} \right\rfloor \\ \frac{\lambda_1}{2} & \text{if } \left\lfloor \frac{C1}{2} \right\rfloor < n_1 < C1 \\ \lambda_1 & \text{if } n_1 = C1 \end{cases}$$

Example



Not exactly lumpable. For
$C1 = 20, C2 = 20, \lambda_1 = 6, \lambda_2 = 1, \mu_1 = 4, \mu_2 = 4, p = 0.7$, $\epsilon = 10^{-13}$
and $\delta = 0.95$ we could find

- $L_1 = \{0\}$, $L_2 = \{1, \ldots, 10\}$, $L_3 = \{11, \ldots, 19\}$ and $L_4 = \{20\}$ on the forward process
- $\overline{L_1} = \{0, \ldots, 10\}$, $\overline{L_2} = \{11, 12\}$, $\overline{L_3} = \{13, \ldots, 19\}$ and $\overline{L_4} = \{20\}$ on the reversed one

Comparison

	FW-Lump	RV-Lump	APF	FPA	Exact
KL div.	0.0065	0.0045	0.0451	0.0112	0
$E[N]$	11.62	11.55	9.990	11.80	11.33
Rel. err.	0.0259	0.0200	0.1178	0.0424	0

Where

- APF is the Approximated Product Form of order 4 [Buchholz, 2010]
- PFA is the Fixed Point Approximation [Miner et al., 2000]

Remarks on lumping approximations

- Lumping of stochastic automata can be applied to other formalisms
- Approximate lumpings can be used to derive approximate marginal distributions
  - Several examples show that in case of queueing networks lumping the reversed automata gives better approximations!
- Future works: definition of efficient algorithms
  - The algorithm proposed in [Gilmore et al., 2001] based on strong equivalence can be adapted to consider our notion of lumpability
    - also for reversed automata
    - optimality issues

Conclusion

- We have shown some novel techniques to efficiently solve some classes of cooperating models
  - Automatic product form detection and solution
  - Compositional lumping techniques
  - Conditional Product Forms
- Current research
  - real-world conditional product-forms
  - Better approximate aggregations
- What I didn't show:
  - Multi-formalism product form analysis
  - Applications of product-form SPNs

References

[Balsamo et al., 2010a] Balsamo, S., Dei Rossi, G., and Marin, A. (2010a). A numerical algorithm for the solution of product-form models with infinite state spaces. In *Proc. of EPEW 2010: Comp. Perf. Eng.*, pages 191–206, Bertinoro, Italy. LNCS 6342/2010.

[Balsamo et al., 2013] Balsamo, S., Dei Rossi, G., and Marin, A. (2013). Queueing networks and conditional product-forms. In *Proc. of Valuetools 2013 Conf.*, page to appear, Turin, IT. ICST.

[Balsamo et al., 2010b] Balsamo, S., Dei Rossi, G., and Marin, A. (January, 2010b). A tool for the numerical solution of cooperating Markov chains in product-form. In *Proc. Of Int. Conf. HET-NETs*, pages 311–324, Zakopane, PL.

[Barbierato et al., 2012] Barbierato, E., Dei Rossi, G., Gribaudo, M., Iacono, M., and Marin, A. (2012). Exploiting product form solution techniques in multiformalism modeling. In *ENTCS, Proc. of Int. Workshop PASM*, page to appear, London, UK. Elsevier.

[Buchholz, 2010] Buchholz, P. (2010). Product form approximations for communicating Markov processes. *Perf. Eval.*, 67(9):797 – 815. Special Issue: QEST 2008.

[Gilmore et al., 2001] Gilmore, S., Hillston, J., and Ribaudo, M. (2001). An Efficient Algorithm for Aggregating PEPA Models. *IEEE Trans. on Software Eng.*, 27(5):449–464.

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

[Marin and Bulò, 2009] Marin, A. and Bulò, S. R. (September 2009). A general algorithm to compute the steady-state solution of product-form cooperating Markov chains. In *Proc. of MASCOTS 2009*, pages 515–524, London, UK.

[Marin and Vigliotti, 2010a] Marin, A. and Vigliotti, M. G. (2010a). A general result for deriving product-form solutions of markovian models. In *Proc. of First Joint WOSP/SIPEW Int. Conf. on Perf. Eng.*, pages 165–176, San Josè, CA, USA. ACM.

[Marin and Vigliotti, 2010b] Marin, A. and Vigliotti, M. G. (2010b). On product-form approximations of cooperating stochastic models. In *Proc. of 25th Int. Symp. on Computer and Information Sciences*, pages 65–70, The Royal Society, London. LNEE, Springer.

[Miner et al., 2000] Miner, A. S., Ciardo, G., and Donatelli, S. (2000). Using the exact state space of a markov model to compute approximate stationary measures. In *Proc. of ACM SIGMETRICS*, pages 207–216, New York, NY, USA. ACM.

Thanks!

Thanks for the attention

any question?