Markov Automata

Joost-Pieter Katoen

RWTH AACHEN UNIVERSITY

UNIVERSITY OF TWENTE.

Lecture 20 course MVPS, July 15, 2014

Overview

Today: Markov Automata

The *beauty* of its theory

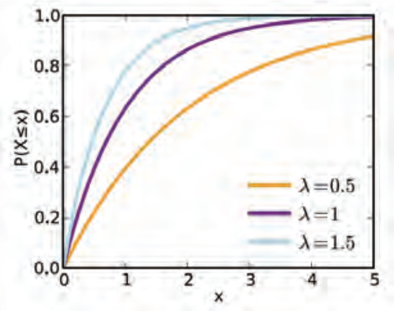
- The simplicity of the model
- Parallel composition
- Bisimulation
- Quantitative analysis

The *usage* for modeling languages

1. Process algebra
2. Stochastic Petri Nets
3. ...... not today ......
4. Architectural Analysis & Design Language
5. Dynamic Fault Trees

Overview

Exponential distributions

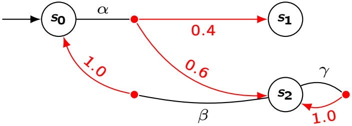


- The cdf of exponentially distributed r.v. X with rate $\lambda \in \mathbb{R}_{>0}$ is:

$$F_X(x) \;=\; 1 - e^{-\lambda \cdot x}$$

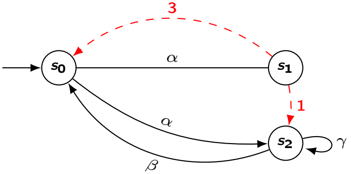
- The rate λ uniquely determines F_X
- The higher λ, the faster F_X approaches 1
- Unique memoryless continuous distribution
- Expectation = λ^{-1}

A marriage  and 



Segala's probabilistic automata

Key: a transition yields a distribution over states

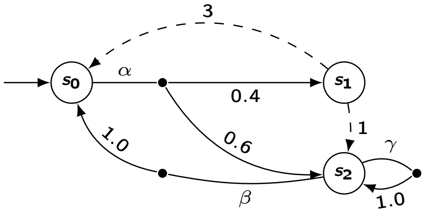


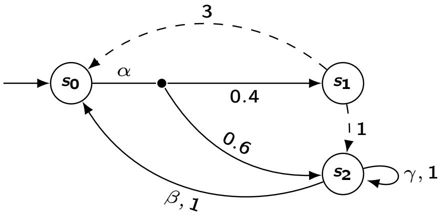
Hermanns' interactive Markov chains

Key: separated action and delay transitions

Markov automata

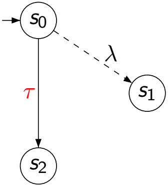
[Eisentraut *et al*, 2010]



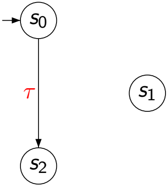


A Markov automaton M is a tuple $(S, Act, \rightarrow, \dashrightarrow, s_0)$ where

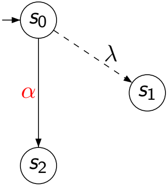
Maximal progress assumption



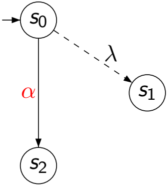
reduces to



But as visible actions may be subject to delaying by other components:



remains



Concurrent composition

The *composition* of M_1 and M_2 wrt. $A = (Act_1 \cap Act_2) \setminus \{\tau\}$ is:

$$M_1 \,||\, M_2 = (S_1 \times S_2, Act_1 \cup Act_2, \rightarrow, \dashrightarrow, (s_{0,1}, s_{0,2}))$$

where $\rightarrow$ and $\dashrightarrow$ are defined as the smallest relations satisfying:

$$\text{(SYNC)}\ \frac{s_1 \xrightarrow{\alpha}_1 \mu_1 \text{ and } s_2 \xrightarrow{\alpha}_2 \mu_2 \text{ and } \alpha \in A}{(s_1, s_2) \xrightarrow{\alpha} \mu_1 \cdot \mu_2}$$

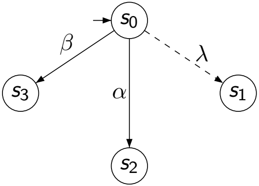
$$\text{(ASYNC)}\ \frac{s_1 \xrightarrow{\alpha}_1 \mu_1 \text{ and } \alpha \notin A}{(s_1, s_2) \xrightarrow{\alpha} \mu_1 \cdot \Delta_{s_2}}$$

$$\text{(DELAY)}\ \frac{s_1 \overset{\lambda}{\dashrightarrow}_1 s_1'}{(s_1, s_2) \overset{\lambda}{\dashrightarrow} (s_1', s_2)} \quad \text{and} \quad \frac{s_1 \overset{\lambda}{\dashrightarrow}_1 s_1 \text{ and } s_2 \overset{\lambda'}{\dashrightarrow}_2 s_2}{(s_1, s_2) \overset{\lambda+\lambda'}{\dashrightarrow} (s_1, s_2)}$$

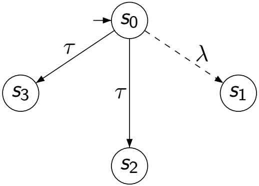
Compatibility

Parallel composition is backward compatible with parallel composition on probabilistic automata and parallel composition on labeled transition systems.

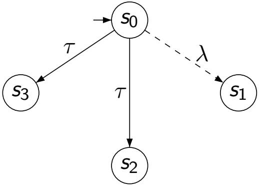
Hiding



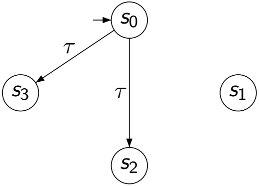
hiding $\{\alpha, \beta\}$ yields



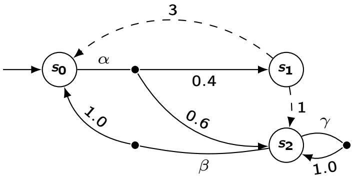
Applying maximal progress reduction yields:

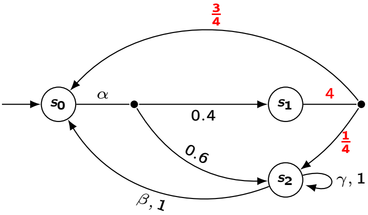


reduces to



Bisimulation





Bisimulation

Equivalence $R \subseteq S \times S$ is a *bisimulation* if for all $(s, t) \in R$:

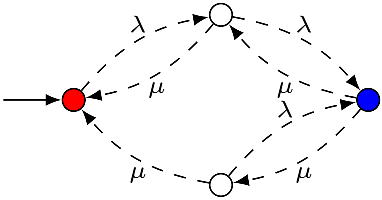
$$\forall \delta \in Act \cup \mathbb{R}_{>0}: s \xrightarrow{\delta} \mu \text{ implies } t \xrightarrow{\delta} \nu \text{ with } \forall C \in S/R : \mu(C) = \nu(C).$$

Let $\sim$ be the largest bisimulation relation.

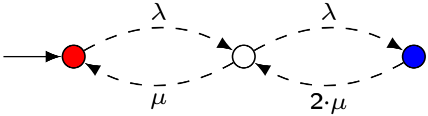
Congruence [Eisentraut *et al*, 2010]

$\sim$ is a congruence wrt. parallel composition and hiding.

Bisimulation – Example



is bisimilar to



Compatibility

Bisimulation is backward compatible with bisimulation on probabilistic automata and bisimulation on labeled transition systems.

Weak bisimulation

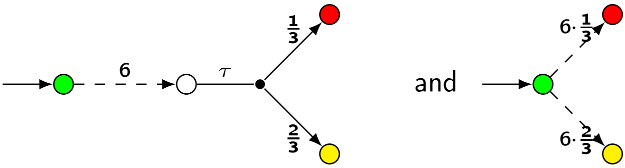
A naive attempt

Equivalence $R \subseteq S \times S$ is a *weak bisimulation* if for all $(s, t) \in R$:

$$\forall \delta \in Act \cup \mathbb{R}_{>0}\colon\ s \xrightarrow{\delta} \mu \text{ implies } t \stackrel{\delta}{\Longrightarrow} \nu \text{ with } \forall C \in S/R : \mu(C) = \nu(C)$$

where $t \stackrel{\delta}{\Longrightarrow} \mu$ means $t \xrightarrow{\tau^*} \xrightarrow{\delta} \xrightarrow{\tau^*} \nu$ (over trees).

This relation is backward compatible but too fine, as it distinguishes:



Weak bisimulation over distributions [Doyen *et al.*, 2008]

Definition 10 (Weak bisimulation [20]). *A symmetric relation $\mathcal{R}$ on subdistributions over S is called a* weak bisimulation *if and only if whenever $\mu_1 \mathcal{R} \mu_2$ then for all $\alpha \in \mathbb{R} \cup \{\varepsilon\}$: $|\mu_1| = |\mu_2|$ and for all $s \in Supp(\mu_1)$ there exist ${\mu_2}^{\rightarrow}, {\mu_2}^{\Delta}$: $({\mu_2}^{\rightarrow}, {\mu_2}^{\Delta}) \in \mathrm{split}(\mu_2)$ and*

(i) $\mu_1(s)\delta_s \;\mathcal{R}\; {\mu_2}^{\rightarrow}$ *and* $(\mu_1 \ominus s) \;\mathcal{R}\; {\mu_2}^{\Delta}$

(ii) whenever $s \xrightarrow{\alpha} \mu_1'$ *for some* μ_1' *then* ${\mu_2}^{\rightarrow} \overset{\alpha}{\Longrightarrow}_C \mu''$ *and* $(\mu_1(s) \cdot \mu_1') \;\mathcal{R}\; \mu''$

Two subdistributions μ and γ are weak bisimilar, denoted by $\mu \approx \gamma$, if the pair (μ, γ) is contained in some weak bisimulation.

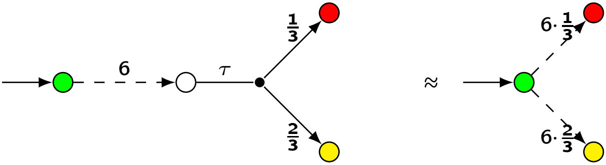
Congruence [Eisentraut et. al., 2010]

$\approx$ is a congruence wrt. parallel composition and hiding.

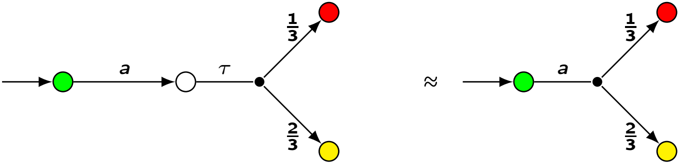
Theorem [Deng & Hennessy, 2011]

$\approx$ is the coarsest "reasonable" notion of weak bisimulation.

Backward incompatibility



Similarly, one obtains:



But Segala's weak bisimulation distinguishes these PA

Overview

A process algebra for PA

GSPNs: historical perspective

1973 Timed Petri Nets [Noe & Nutt]

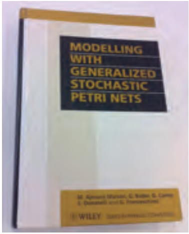
1980 Stochastic Petri Nets [Molloy, Natkin, Symons]

1984 Generalized Stochastic Petri Nets [Ajmone Marsan, Conte & Balbo]

1995 Modeling with Generalized Stochastic Petri Nets [Ajmone Marsan *et al.*]

A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems

MARCO AJMONE MARSAN and GIANNI CONTE
Politecnico di Torino, Turin, Italy
and
GIANFRANCO BALBO
Universita' di Torino, Turin, Italy

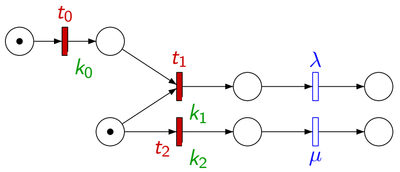


Generalized stochastic Petri nets [Ajmone Marsan et al, 1984]

What is a GSPN?

A Petri net with

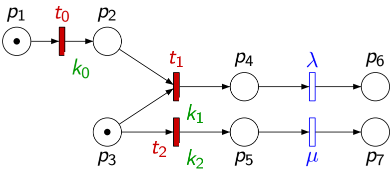
- Timed transitions
- Immediate transitions
- Natural weights



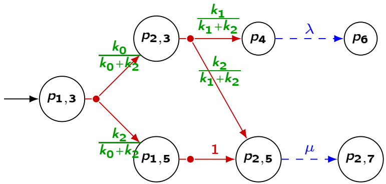
Two-phase semantics

1. Determine enabled transitions and their probability
    - Maximal progress: immediate transitions have priority
2. Determine the underlying stochastic process

GSPN semantics by example

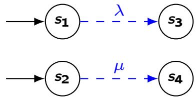


Token game and probabilities



Isn't this a Markov automaton?

Induced stochastic process



Initial distribution $\mu(s_1) = \frac{k_0}{k_0+k_2} \cdot \frac{k_1}{k_1+k_2}$, and

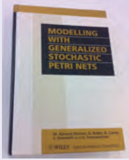
$$\mu(s_2) = \frac{k_2}{k_0+k_2} + \frac{k_0}{k_0+k_2} \cdot \frac{k_1}{k_1+k_2}$$

Isn't this weakly bisimilar?

A caveat

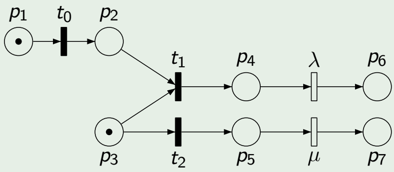
The presence of confused subnets
of immediate transitions within a GSPN
is an undesirable property of the model.

Ajmone Marsan *et al.* (1995)



Confusion

A simple confused GSPN



- Transitions t_0 and t_2 are concurrent
- If t_2 fires first, no conflict arises
- If t_0 fires first, a conflict $t_1 \frown t_2$ arises

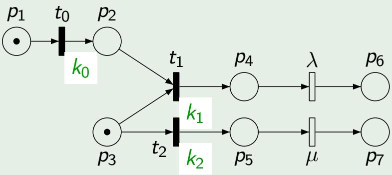
In marking $p_2 + p_7$ one cannot conclude whether a conflict had to be resolved.

This situation is called confusion.

Classical GSPN approach: Avoid confusion. Resolve nondeterminism by weights.

Weighted immediate transitions

A simple weighted GSPN



- Transition t_i has weight $k_i \in \mathbb{N}_{>0}$
- t_2 fires first with probability $\frac{k_2}{k_0+k_2}$
- t_0 fires first with probability $\frac{k_0}{k_0+k_2}$
- Concurrency is thus resolved probabilistically

$$\Pr\{\Diamond(p_2+p_7)\} = \underbrace{\frac{k_2}{k_0+k_2}}_{t_2 \text{ before } t_0} + \underbrace{\frac{k_0}{k_0+k_2} \cdot \frac{k_2}{k_1+k_2}}_{\substack{t_0 \text{ before } t_2 \\ \text{and } t_2 \text{ before } t_1}}$$

Note the influence of k_0 on this quantity.

Drawbacks of weights

How to get adequate weights?

For conflicting transitions this is mostly simple, but not for confused ones.

But: weight values are fundamental for the quantitative evaluation.

Biased analysis

Quantitative results are subject to a specific weight assignment.

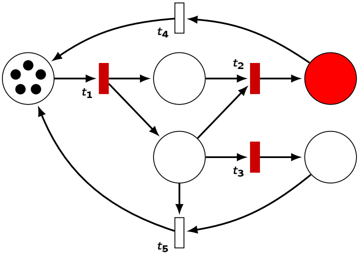
This bias is mostly neglected.

Unexpected effects

Splitting or deleting an immediate transition "has drastic effects on the values of the results obtained from the quantitative evaluation".

Weights are not innocent

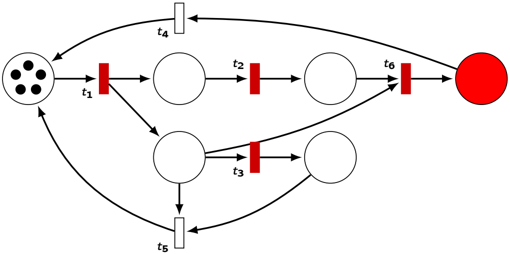
A sample GSPN



Long-run average (2 tokens in red)

$= \mathbf{0.31}\ldots$

Adding an immediate transition



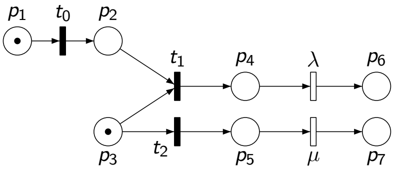
Long-run average (2 tokens in red)

$= \mathbf{0.\underline{0}39}\ldots$

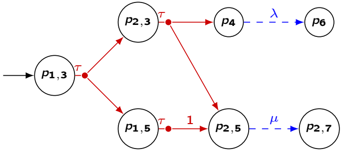
Quantitative results differ almost one order of magnitude!

GSPN marking graphs are Markov automata!

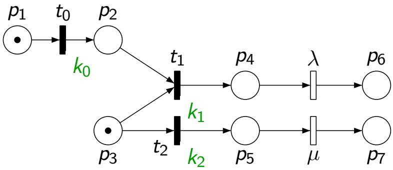
A confused GSPN:



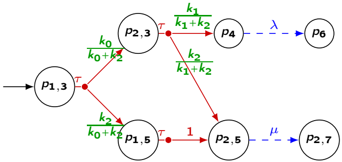
Its semantics:



A weighted GSPN:



Its semantics:



Well-defined nets

Backward compatibility [Eisentraut *et al.*, 2013]

The MA semantics of a well-defined GSPN is weak bisimilar to its standard GSPN semantics.

GSPNs go non-deterministic

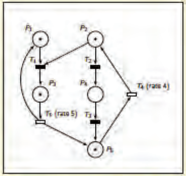
Advantages of MA semantics

- It is truly simple
- It is intuitive
- It is compositional
- It is backward compatible
- No restrictions on net level

This solves a long-standing open issue in stochastic Petri nets

Tool support

GSPN (.pnml)

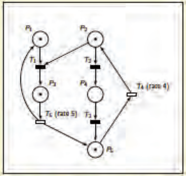


`reach P1 = 1 & P5 = 2`

`http://wwwhome.cs.utwente.nl/~timmer/mama/`

Tool support

GSPN (.pnml)



```
reach P1 = 1 & P5 = 2
```

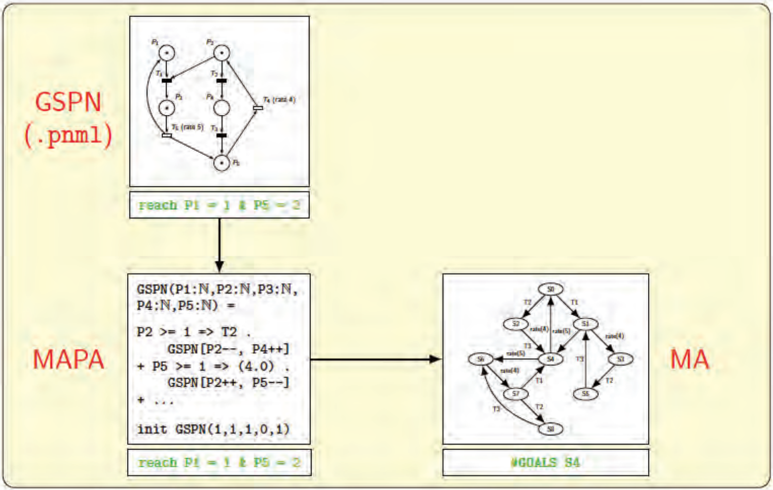
MAPA

```
GSPN(P1:ℕ,P2:ℕ,P3:ℕ,
P4:ℕ,P5:ℕ) =

P2 >= 1 => T2 .
    GSPN[P2--, P4++]
+ P5 >= 1 => (4.0) .
    GSPN[P2++, P5--]
+ ...

init GSPN(1,1,1,0,1)
```

```
reach P1 = 1 & P5 = 2
```

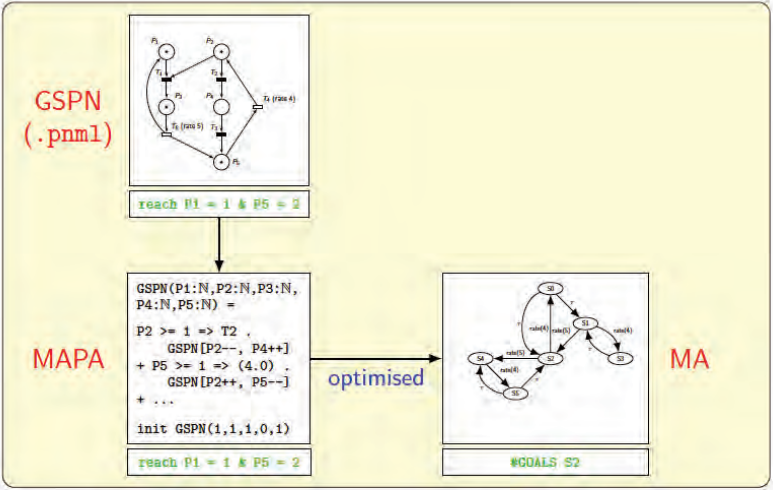
http://wwwhome.cs.utwente.nl/~timmer/mama/

Tool support



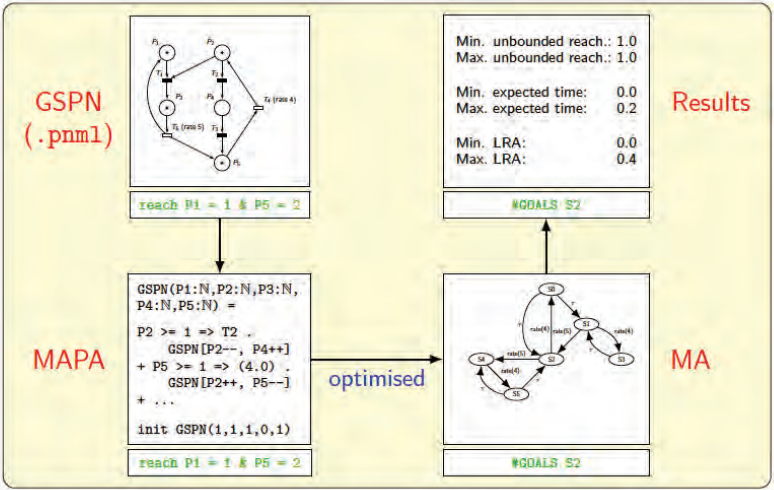
http://wwwhome.cs.utwente.nl/~timmer/mama/

Tool support



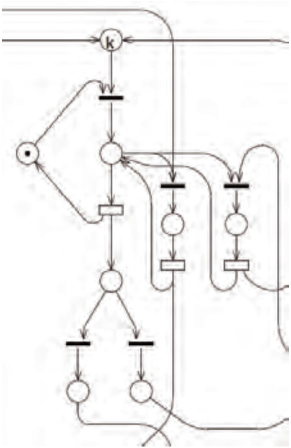
http://wwwhome.cs.utwente.nl/~timmer/mama/

Tool support



http://wwwhome.cs.utwente.nl/~timmer/mama/

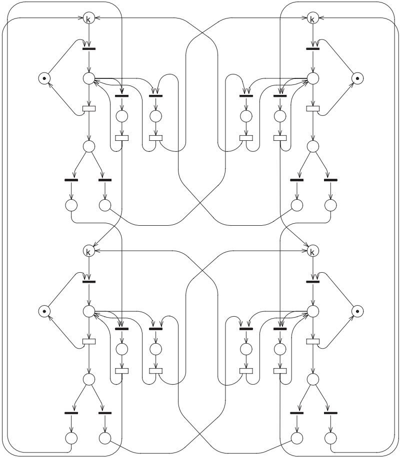
GSPN model of multi-processor system [Ajmone Marsan et. al., 1994]



GSPN of a single processor

- A 2×2 multi-processor grid
- Multi-tasking of k tasks/processor
- Two-phase task execution:
    1. local processing (1)
    2. co-operative processing (10)
- Selection policy for neighbour
- Pipelining of tasks per processor
- Co-operation has priority

Multi-processor system



Presence of immediate transitions excludes usage GSPN tools

Processor throughput

k	# states	# transitions	generation (s)	tp processor 1	tp processor 2	tp processor 4
2	2508	3215	14.5	.9031	ditto	ditto
3	10852	14379	64.7	.9086	ditto	ditto
4	31832	42879	193.0	.9090	ditto	ditto

Scenario one: uniform weight assignment

2	as above	4254	0.8	[.9031,.9055]	[.8585,.9479]	[.9029,.9032]
3	as above	19089	3.2	[.9081,.9089]	[.8633,.9541]	[.9086,.9087]
4	as above	56704	9.8	[.9089,.9091]	[.8636,.9545]	[.9090,.9091]

Scenario two: processor one selects non-deterministically

2	as above	4698	0.6	[.8110,.9956]	ditto	ditto
3	as above	20872	2.7	[.8173,.9998]	ditto	ditto
4	as above	62356	7.9	[.8181,1.0]	ditto	ditto

Scenario three: fully non-deterministic