STOCHASTIC PROCESS ALGEBRA:

linking process descriptions with performance

Ed Brinksma

joint work with :

P. D'Argenio, H. Hermanns, J.-P. Katoen

Formal methods & Tools

University of Twente

The Netherlands

Contents

- Introduction to Stochastic Process Algebra
  motivation, concepts of PA & SPA
- Markovian Process Algebra
  Interactive Markov Chains   TIPPtool
- Non-Markovian Process Algebra
  GSMPs, Discrete Event Simulation: ♠ spades.
- Conclusion
  current developments

MOTIVATION

Central Issue

Can the qualitative and quantitative aspects of reactive systems be modelled and analysed within one compositional framework?

- increasing importance of quantitative behaviour
- need for integrated design disciplines
- cross-fertilization
- theory of approximate correctness

Process Algebra

- a formalism to specify the behaviour of systems in a
  - systematic,
  - modular, and
  - hierarchical way.
- building blocks
  - processes,
  - actions,

    atomic activities that processes can perform
- process algebra provides compositionality, by means of
  - operators to compose processes out of smaller ones, and
  - operators and transformations to reduce internal complexity

⇒ Modelling of complex systems becomes manageable

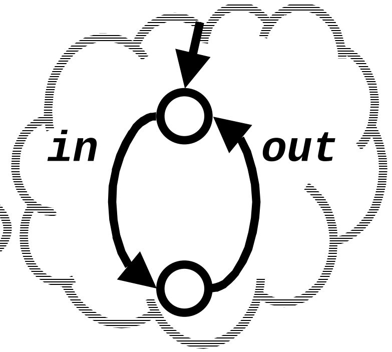
Basic Process Algebraic Operators

• inaction:	**stop**
• action-prefix:	a ; B or τ ; B
• choice:	$B + C$ or $\Sigma_I B_i$
• composition:	$B \,\|\|_A C$ or B |[A]| C
• hiding:	$B \setminus A$ or **hide** A **in** B
• definition:	$p := B$
• application:	p

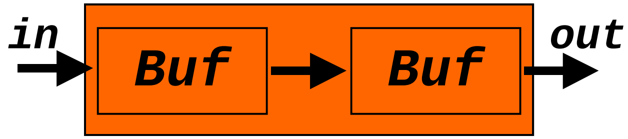
A very basic example

- A simple one-place buffer

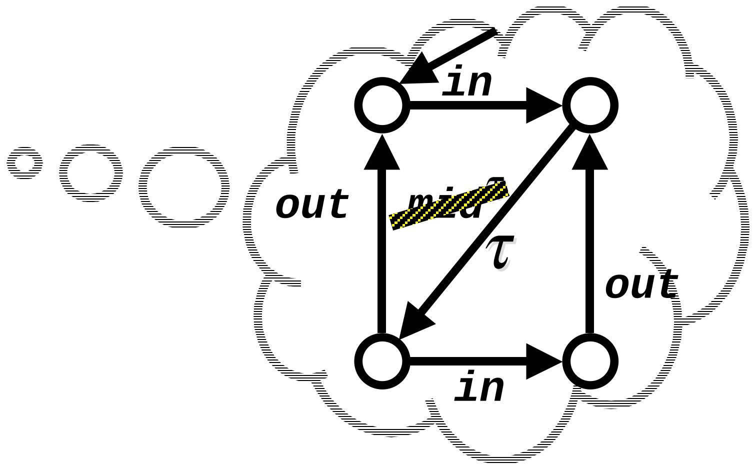
```
Buf:= in ; out ; Buf
```



- Two instances of this buffer



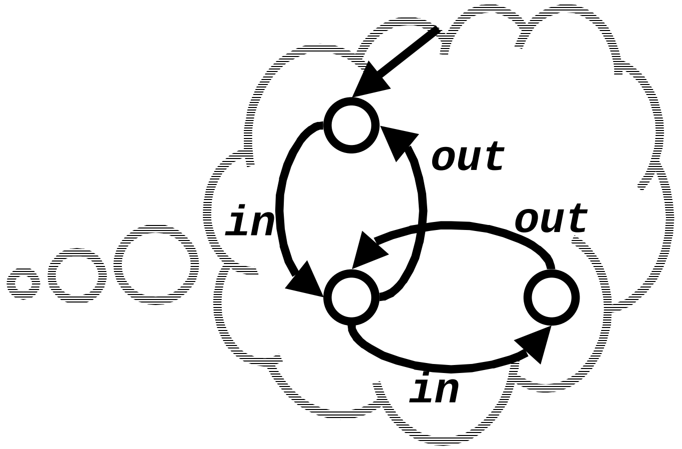

```
hide mid in
    Buf[out/mid]
      |[mid]|
    Buf[in/mid]
```



A very basic example II

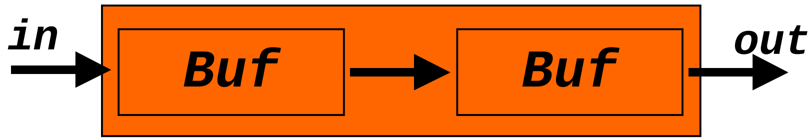
- A two-place buffer


```
Buf2:= in; Half
Half:= in; Full + out; Buf2
Full:= out; Half
```

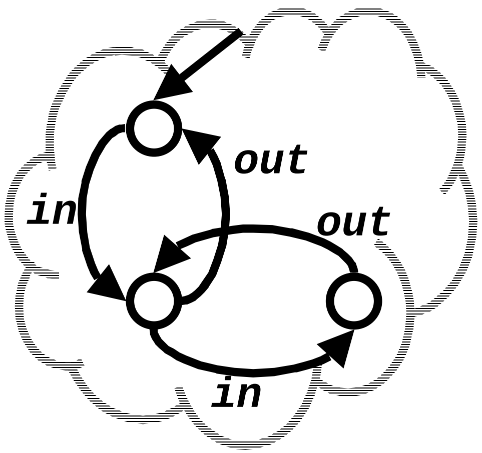


Equivalence

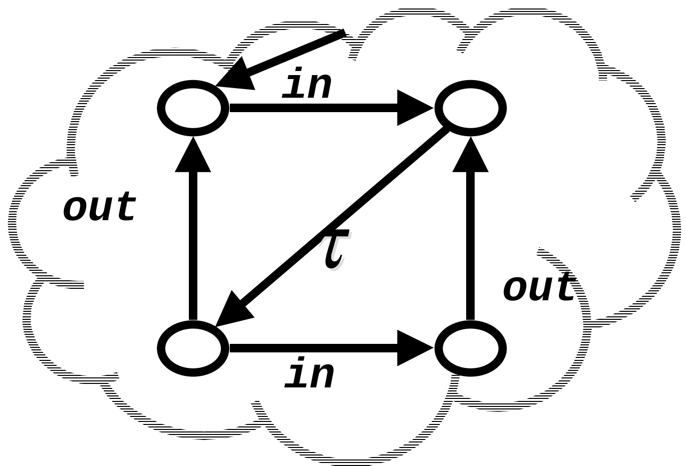
Two ways to represent a two-place buffer:



- by enumerating the detailed behaviour



- by coupling two one place buffers



Examples for the need to study equivalences

Equivalence

- Process algebraic equivalences are based on different answers to the question:

  What is the **observable** part of process behaviour?

- Various notions have been studied [van Glabbeek]

Examples:

- trace equivalence
- testing equivalence
- bisimulation equivalence

Distinguishing features:

- strong vs. weak equivalences
- congruence property

Algebraic Laws

Equivalences (congruences) induce algebraic laws

- B+C = C+B
- (B+C)+D = B+(C+D)
- B+**stop** = B
- B+B = B
- $B \,||_A\, C = C \,||_A\, B$
- $(B \,||_A\, C) \,||_A\, D = B \,||_A\, (C \,||_A\, D)$

Expansion Laws

In the interleaving interpretation parallelism

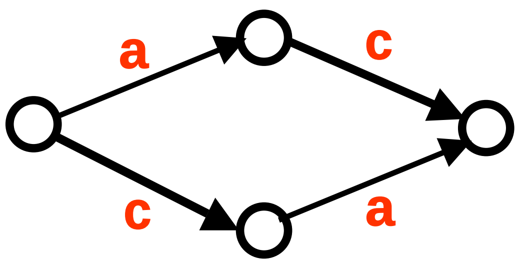
can be removed step by step:

Let $B = \Sigma_k\, a_k\ ;\ B_k$ **and** $C = \Sigma_l\, c_l\ ;\ C_l$

$$B \parallel_A C = \Sigma\{a_k\ ;(B_k \parallel_A C\ )\ |\ a_k \notin A\} + \Sigma\{c_l\ ;(B \parallel_A C_l\ )\ |\ c_l \notin A\} + \Sigma\{d\ ;(B_k \parallel_A C_l)\ |\ d = a_k = c_l \in A\}$$

Example:

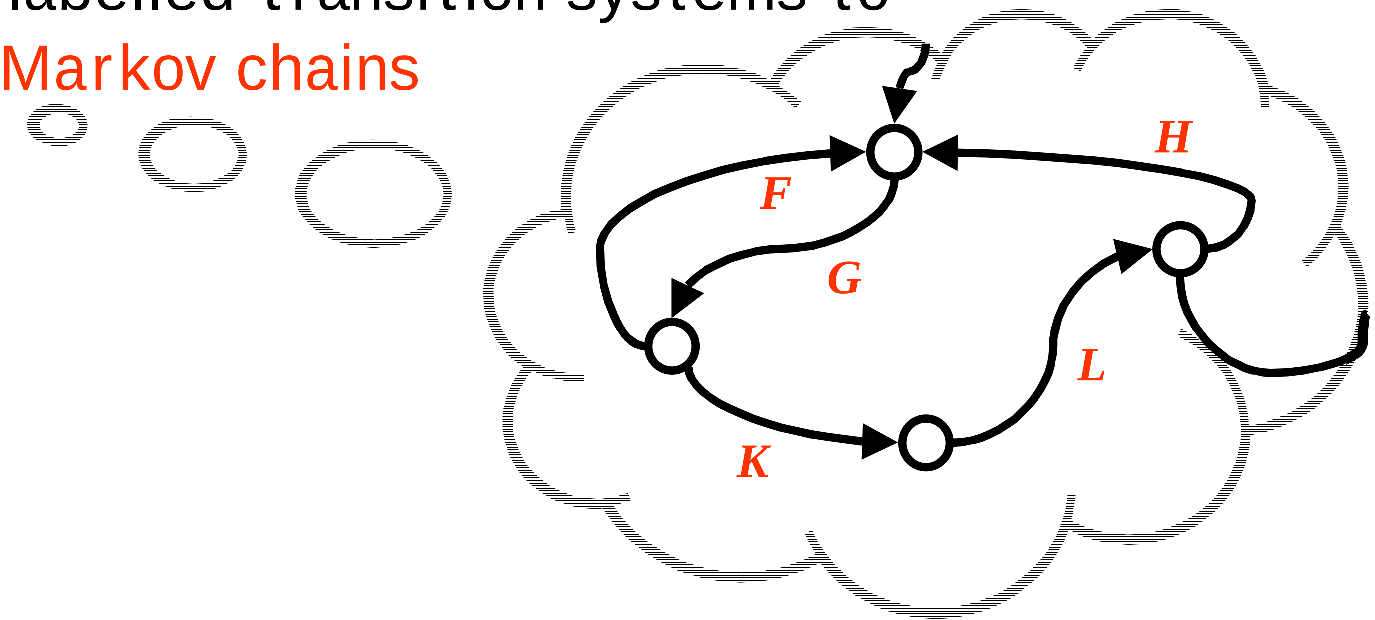
$$a\ ;\ \mathbf{stop} \parallel_\emptyset c\ ;\ \mathbf{stop} = a\ ;\ c\ ;\ \mathbf{stop} + c\ ;\ a\ ;\ \mathbf{stop}$$



Adding Stochastic Features

Naive idea: decorate actions with distribution functions:

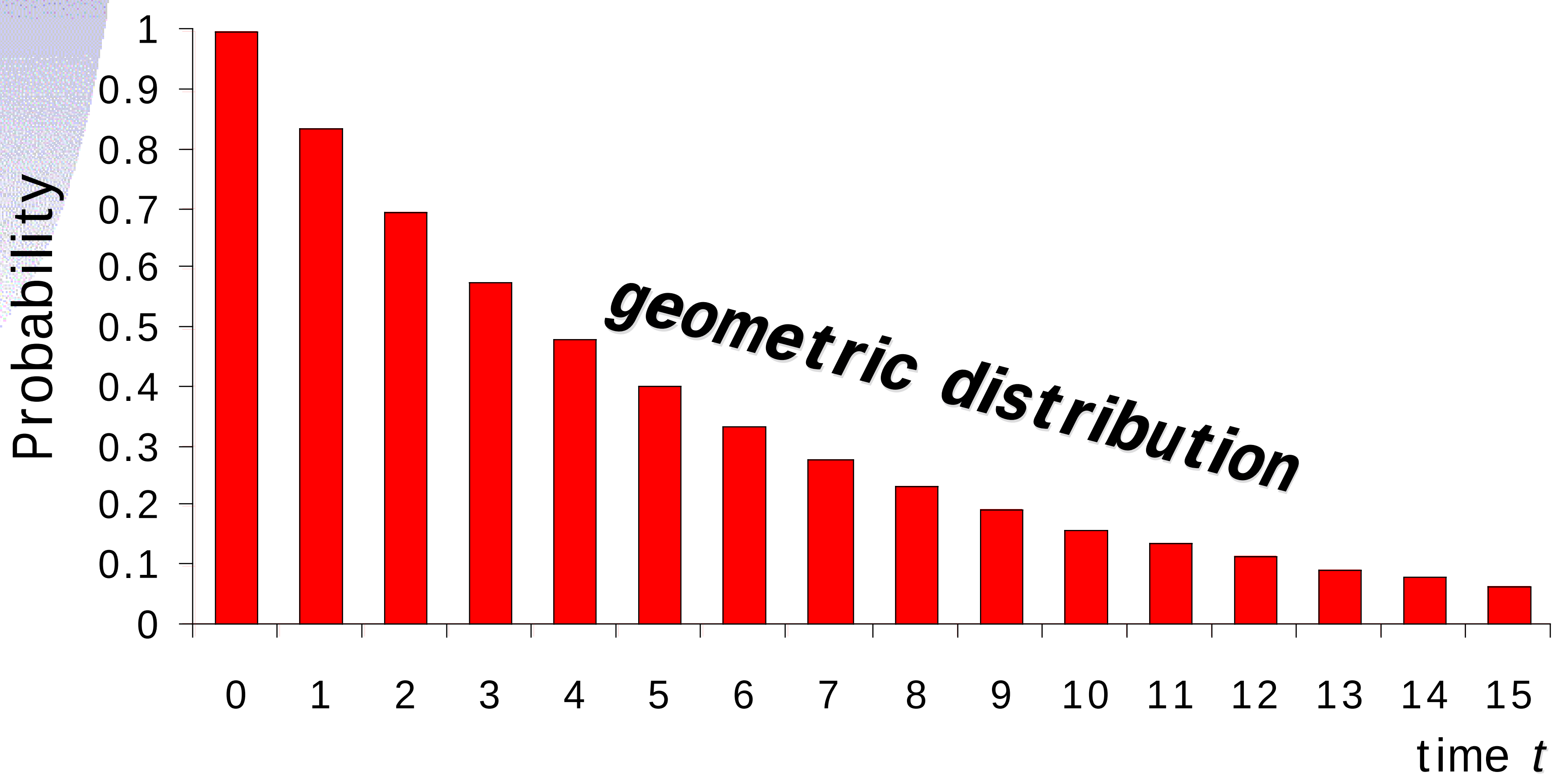
- a_F the time between enabling and occurrence of a is distributed according to F
- linking labelled transition systems to (semi) Markov chains



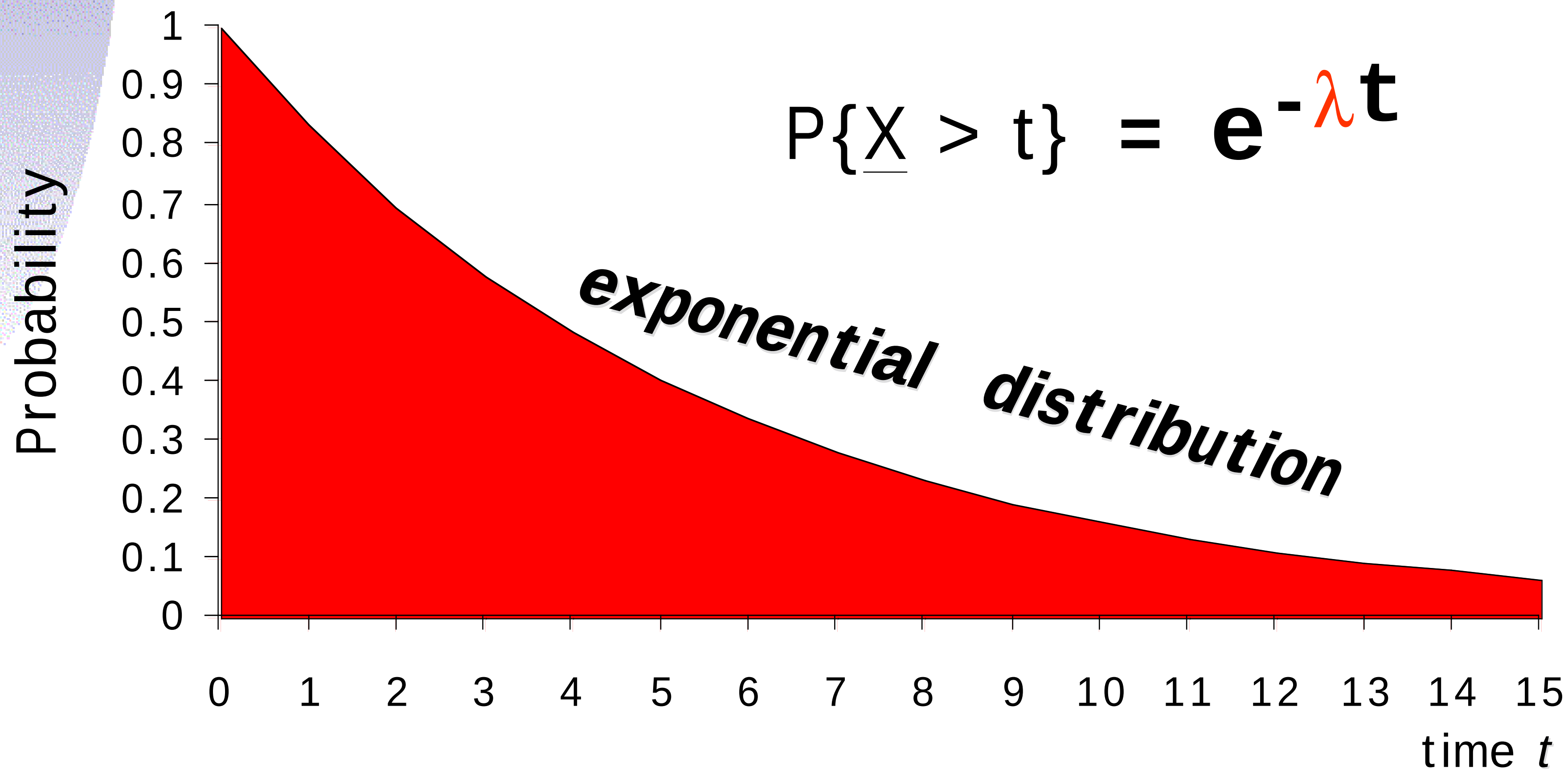
Issues in SPA

- What distributions can be allowed?
  - memoryless versus general distributions
- What is the meaning of choice?
  - nondeterminism versus race conditions
- What is the meaning of synchronization?
  - how to synchronize distributions
- What is the meaning of concurrency?
  - how to expand parallelism

Discrete time, no memory

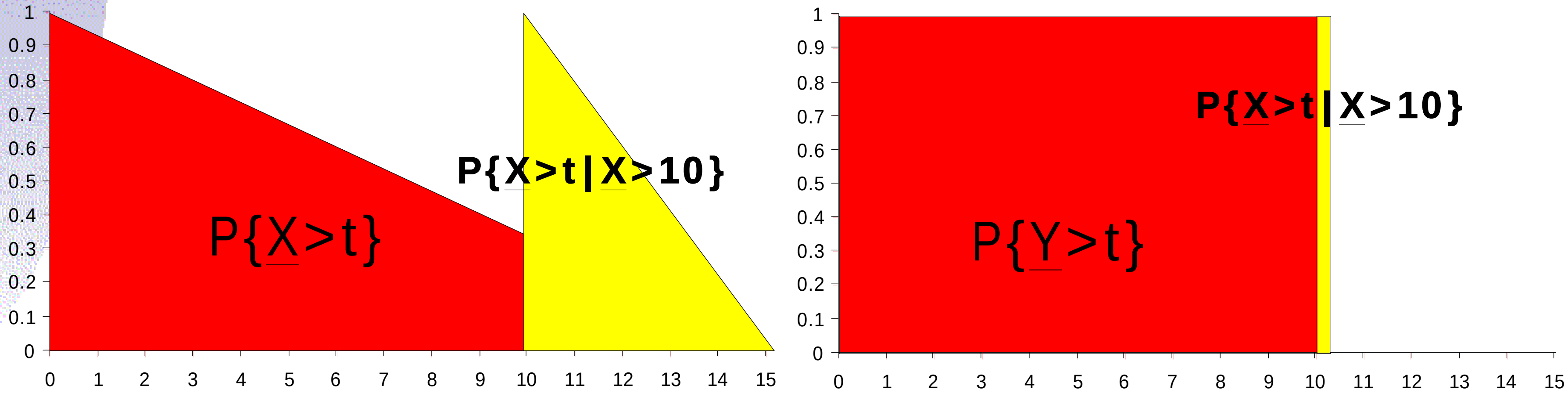


Continuous time, no memory



stochastic models are usually developed in a continuous time domain.

Continuous time with memory



- and many others
- absence of memory is rare,
- it makes modelling and analysis a lot simpler.

Choice or Summation

- In ordinary PA choice is nondeterministic, i.e. we choose one behaviour or the other
  - the operator is idempotent: B+B = B
  - we may refine nondeterminism: a;B refines a;B+a;C

- In SPA choice is capacitative, i.e. both arguments add capacity to the behaviour
  Markovian nondeterminism is additive a_{λ}; B+a_{μ};B = $a_{\lambda+\mu}$;B

  as a function of the exponential rates:

Interleaving revisited

For general distributions we do not have the usual interleaving laws, e.g.:

$$a_F ; B \mid\mid\mid c_G ; C \neq a_F ;(B \mid\mid\mid c_G ; C) + c_G ;(a_F ; B \mid\mid\mid C)$$

The occurrence of a after c generally has another distribution than a occurring initially.

Solutions

- restrict to the Markovian case

$$a_\lambda \,;\, B \;|||\; c_\mu \,;\, C = a_\lambda \,;(B \;|||\; c_\mu \,;\, C) +$$
$$c_\mu \,;(a_\lambda \,;\, B \;|||\; C_\mu\,)$$

Problem: less general

- separate actions from stochastic durations

$$\mathrm{set}_{\{F,G\}}(F\!\rightarrow a \,;\, B\,|||\, G\!\rightarrow c \,;\, C) =$$
$$\mathrm{set}_{\{F,G\}}(F\!\rightarrow a \,;(B\,|||\, G\!\rightarrow c \,;\, C) +$$
$$G\!\rightarrow c \,;(F\!\rightarrow a \,;\, B\,|||\, C\,)\,)$$

This solution is elaborated in the rest of this talk

Alternatives

- drop the interleaving law

  uses so-called partial order semantics

  Problem: more complicated,
  but smaller state spaces

- use conditional distributions

  $a_{\underline{X}} \,;\, B \;|||\; c_{\underline{Y}} \,;\, C =$

  $$a_{\underline{X}} \,;(B \;|||\; c_{(\underline{Y}-\underline{X} \mid \underline{X}<\underline{Y})} \,;\, C) \;+$$

  $$c_{\underline{Y}} \,;(a_{(\underline{X}-\underline{Y} \mid \underline{X}>\underline{Y})} \,;\, B \;|||\; C)$$

  Problem: costly and complicated

Synchronization

What should be the result of synchronizing stochastic actions?

$$a_{\underline{X}} ; B \,||\, a_{\underline{Y}} ; C = a_{\underline{X} * \underline{Y}} ; (B \,||\, C)$$

Choices for * :

- the maximum of the distributions of $\underline{X}$ and $\underline{Y}$
- the average of $\underline{X}$ and $\underline{Y}$
- ?

Synchronization & Expansion

Problem: race condition interferes with classical expansion

- no classical expansion [Hillston:PEPA]

  apparent rates

- passive components [Gorrieri, Bernardo,MPA]

  master/slave synchronization

- defining $\lambda * \mu = \lambda.\mu$ [Herzog e.a.,TIPP;Buchholz]
- separate rates from actions [Hermanns,IMC]

Contents

- Introduction to Stochastic Process Algebra ✓
  motivation, concepts of PA & SPA
- Markovian Process Algebra ←
  Interactive Markov Chains

Interactive Markov chains

- inaction: **stop**
- prefix: (λ); *B* or *a* ; *B* or τ ; *B*
- choice: *B* + *C* or $\Sigma_I B_i$
- definition: *p* := *B*
- application: *p*
- composition: $B \; ||_A C$ or *B* **|[A]|** *C*
- hiding: *B* \ *A* or **hide** *A* **in** *B*

Algebraic Laws for IMC

- **B** + **C** = **C** + **B**
- (**B** + **C**) + **D** = **B** + (**C** + **D**)
- **B** + **stop** = **B**
- **a;B** + **a;B** = **a;B**
- (λ);**B** + (μ);**B** = $(\lambda+\mu)$;**B**

These are the algebraic laws for strong Markovian bisimulation, a straightforward combination of strong bisimulation and lumpability.

Algebraic Laws for IMC

- $B + C = C + B$
- $(B + C) + D = B + (C + D)$
- $B + \mathbf{stop} = B$
- $a;B + a;B = a;B$

- $a;\tau;B = a;B$
- $B + \tau;B = \tau;B$
- $a;(B + \tau;C) + a;C = a;(B + \tau;C)$

- $(\lambda);B + (\mu);B = (\lambda+\mu);B$

- $(\lambda);\tau;B = (\lambda);B$
- $\tau;B + (\lambda);C = \tau;B$

"maximal progress"

These are the algebraic laws for weak Markovian bisimulation, a (not so straightforward) combination of weak bisimulation and lumpability.

Expansion in IMC

The delay actions can be treated as non-synchronizing actions:

Let $B = \Sigma_k\, a_k \;;\; B_k + \Sigma_m\, (\lambda_m) \;;\; B_m$

and $C = \Sigma_l\, c_l \;;\; C_l + \Sigma_n\, (\mu_n) \;;\; B_n$

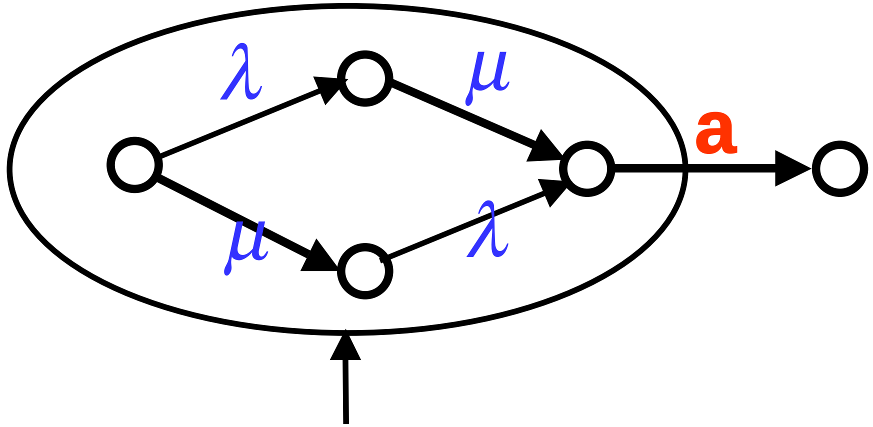
then

$$B \;||_A\; C = \Sigma\{\, a_k \;;(B_k \;||_A\; C) \mid a_k \notin A \,\} + \Sigma_m\{\, (\lambda_m);(B_m \;||_A\; C) \,\} + \Sigma\{\, c_l \;;(B \;||_A\; C_l) \mid c_l \notin A \,\} + \Sigma_n\{\, (\mu_n);(B \;||_A\; C_n) \,\} + \Sigma\{\, d \;;(B_k \;||_A\; C_l) \mid d = a_k = c_l \in A \,\}$$

Example

(λ); **a ; stop** || (μ); **a ; stop =**

(λ); (μ); **a ; stop +** (μ); (λ); **a ; stop**



This corresponds to delaying with the maximum
of two exponential delays, e.g. waiting for the slowest

Queuing Systems in IMC

```
hide enter,serve in
  CUSTOMER  |[enter]| QUEUE(0) |[serve]| SERVER
```

arriving customers:

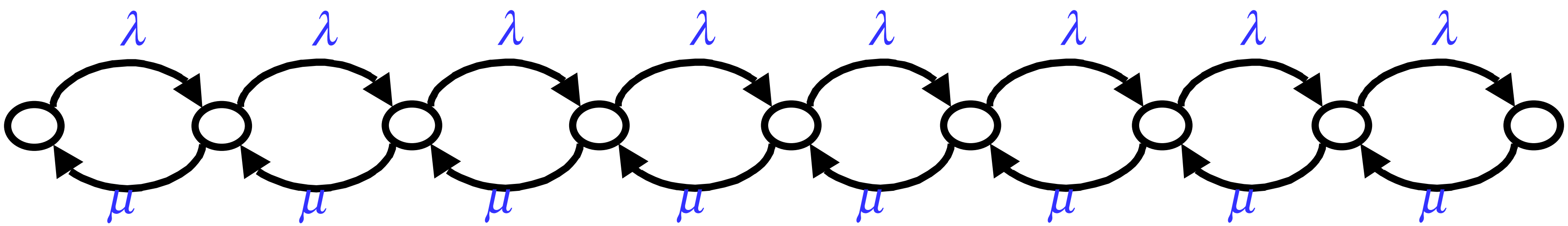
```
process CUSTOMER := (λ); enter ; CUSTOMER
endproc
```

queue:

```
process QUEUE(i) := [i<6]->  enter; QUEUE(i+1)
                    [i>0]->  serve; QUEUE(i-1)
endproc
```

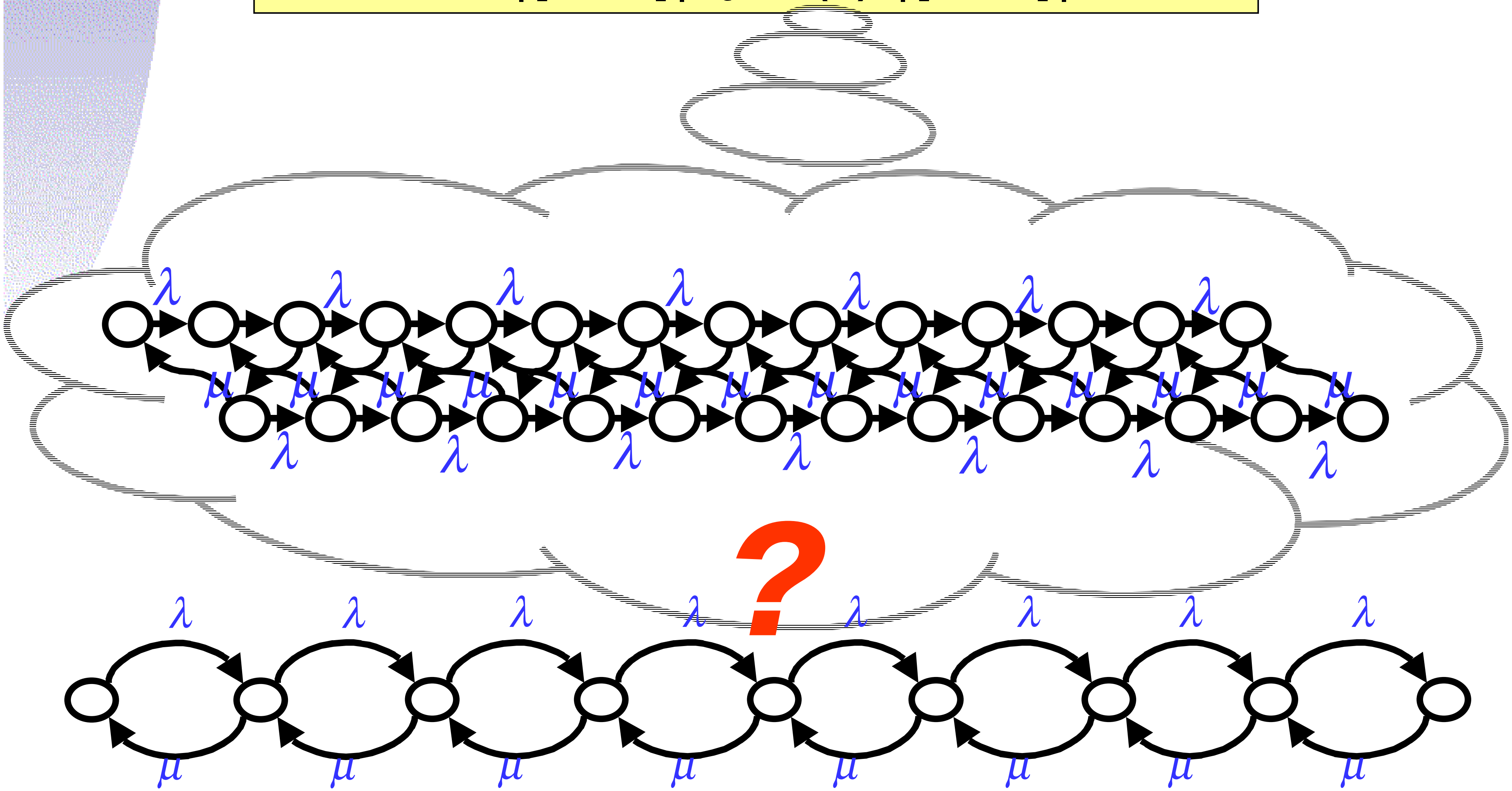
service clerk:

```
process SERVER := serve ; (μ) ; SERVER
endproc
```



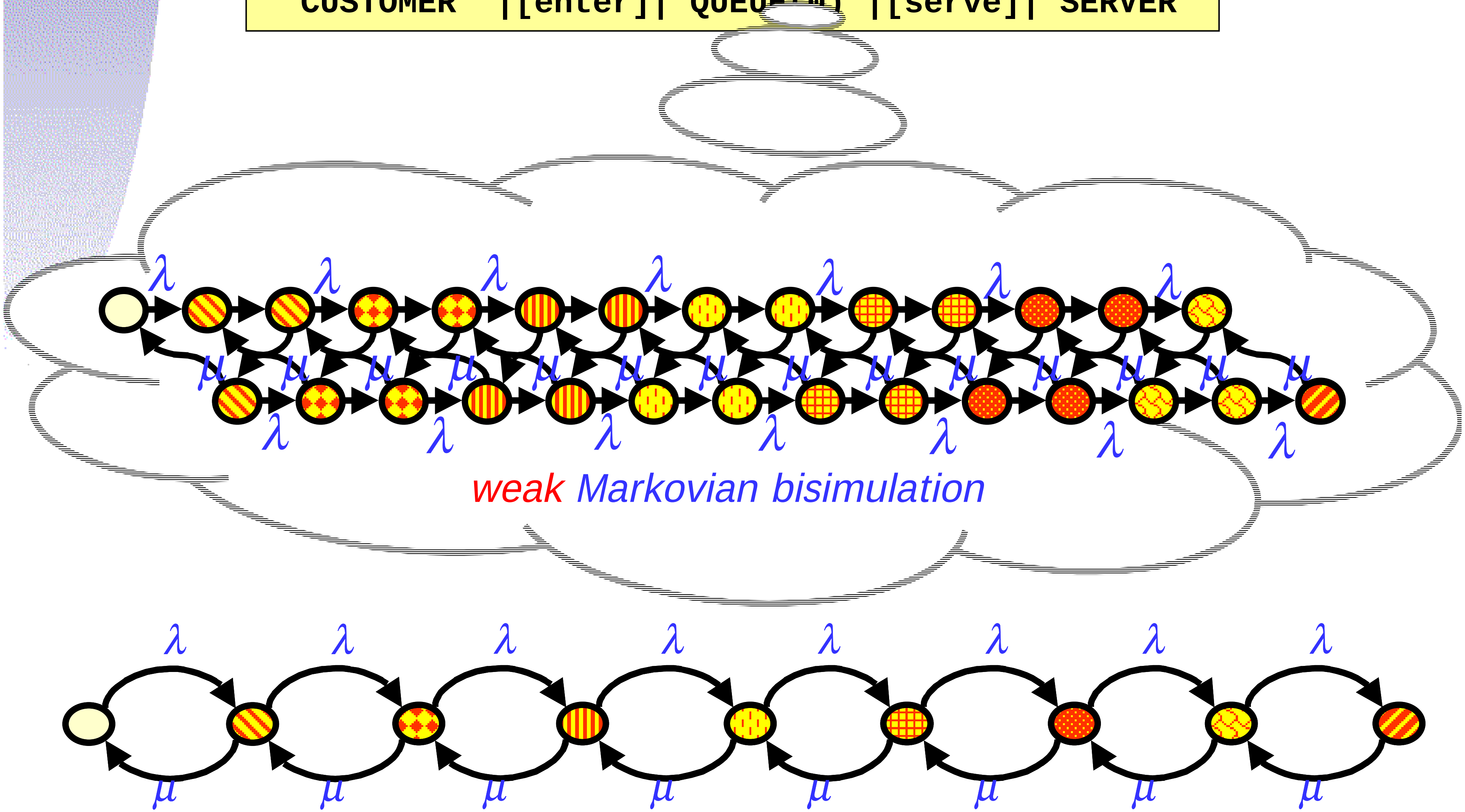
Queuing Systems in IMC

```
hide enter,serve in
  CUSTOMER  |[enter]| QUEUE(0) |[serve]| SERVER
```

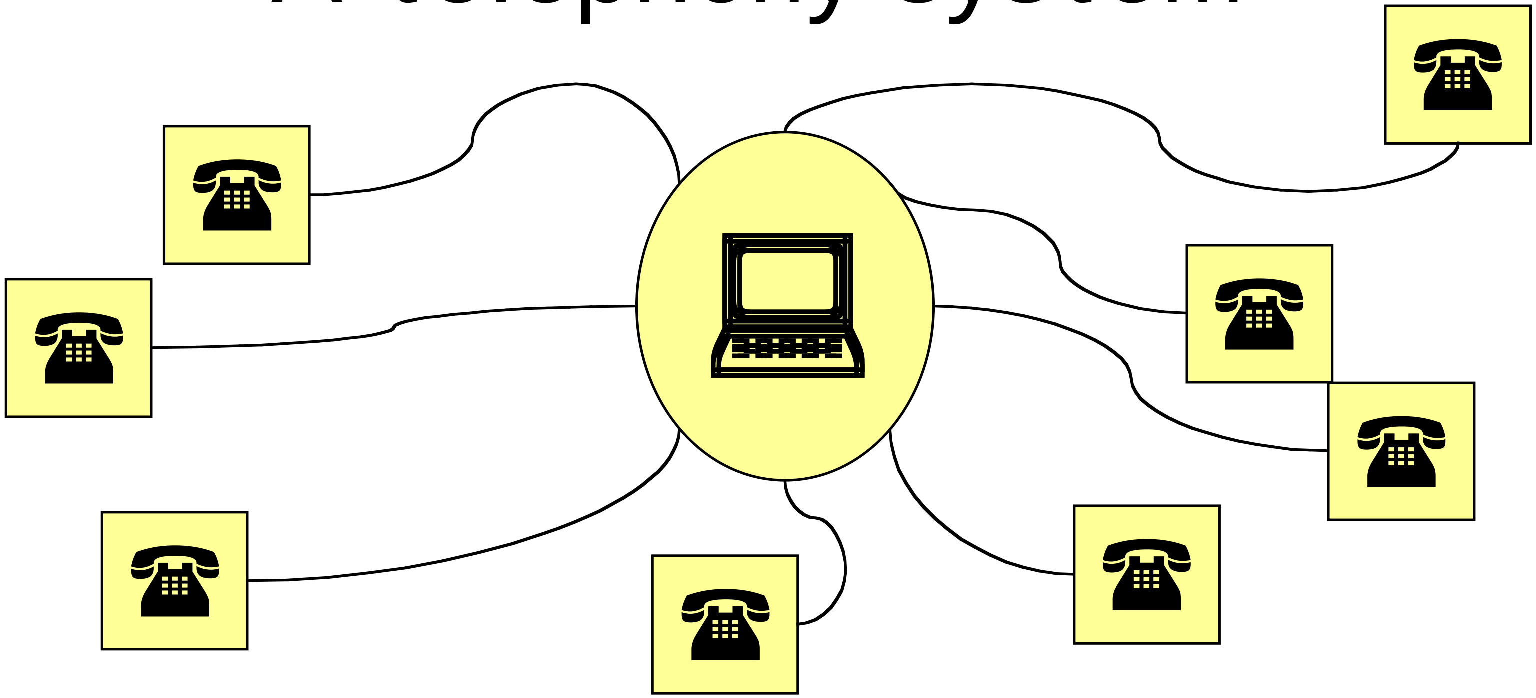


Queuing Systems in IMC

hide `enter,serve` **in**
`CUSTOMER |[enter]| QUEUE(0) |[serve]| SERVER`

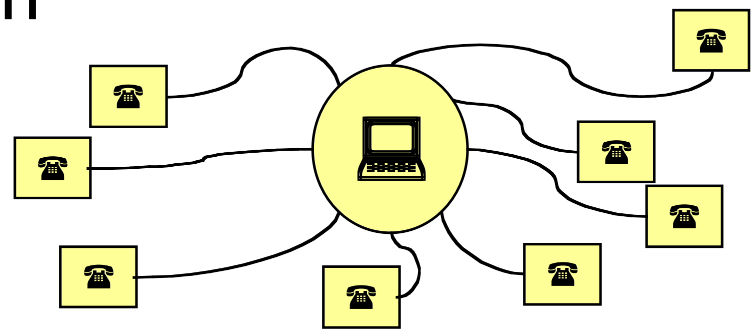


A telephony system



- **Original specification developed by P. Ernberg (SICS), further studied in the French/Canadian *Eucalyptus* project: more than 1500 lines of LOTOS.**
- **Extensively verified using state-of-the-art techniques**
  - **model checking**
  - **equivalence checking**

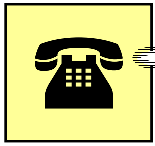
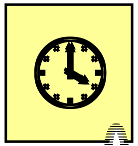
Performance analysis of the telephony system

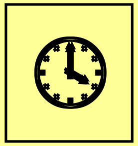
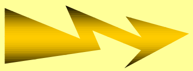


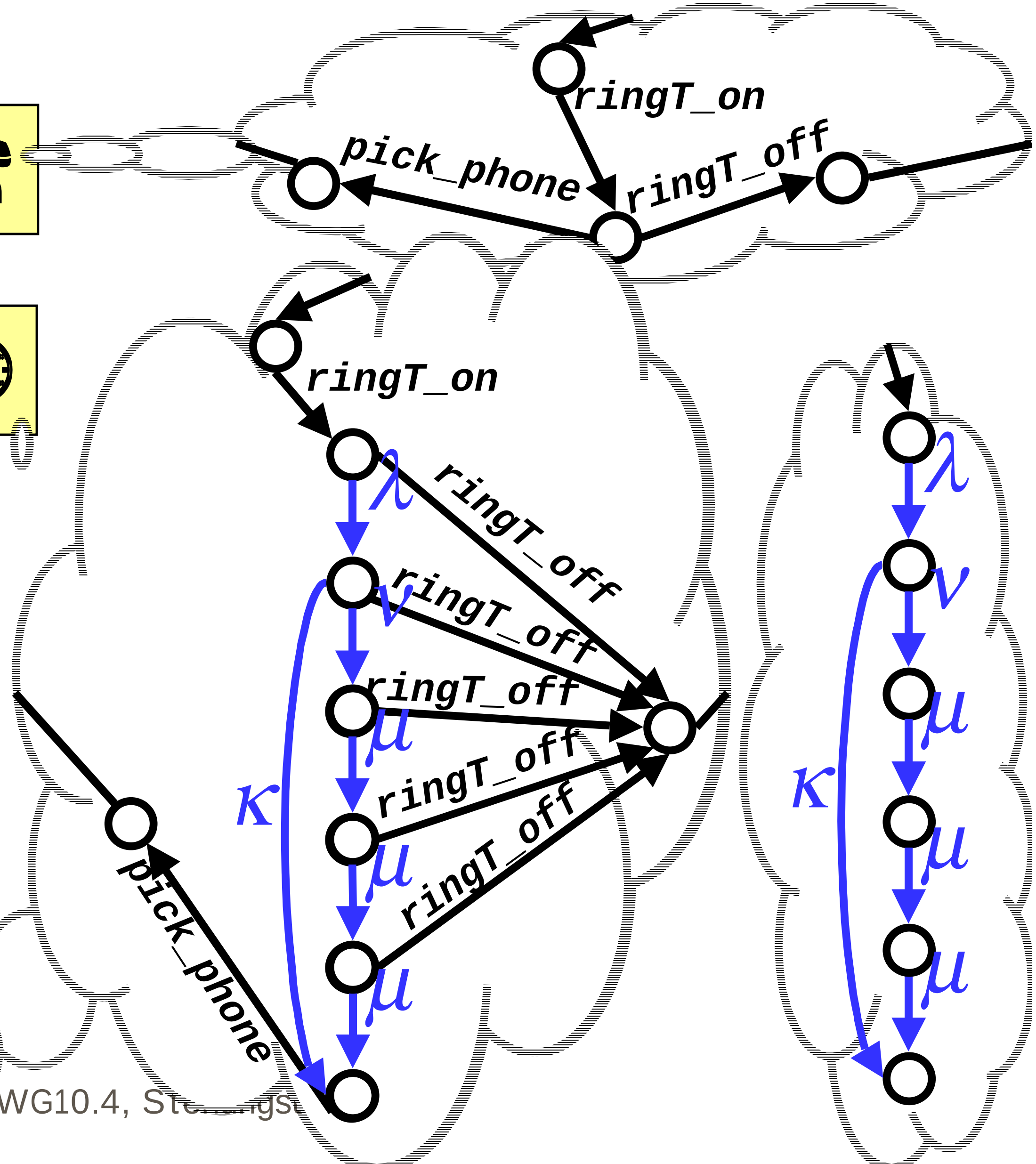
using a dedicated operator, time constraints

- Takes the original specification *without changes.*
- Stochastic delays are incorporated
  - in a compositional way,
    i.e. as additional constraints
    imposed on the specification.
  - *exponential, Erlang* and *phase-type* distributions.
- *Weak bisimulation* is used to factor out nondeterminism.
- State space **> 10^7** leads to a Markov Chain of **720** states with a *highly irregular* structure.

Time constraints

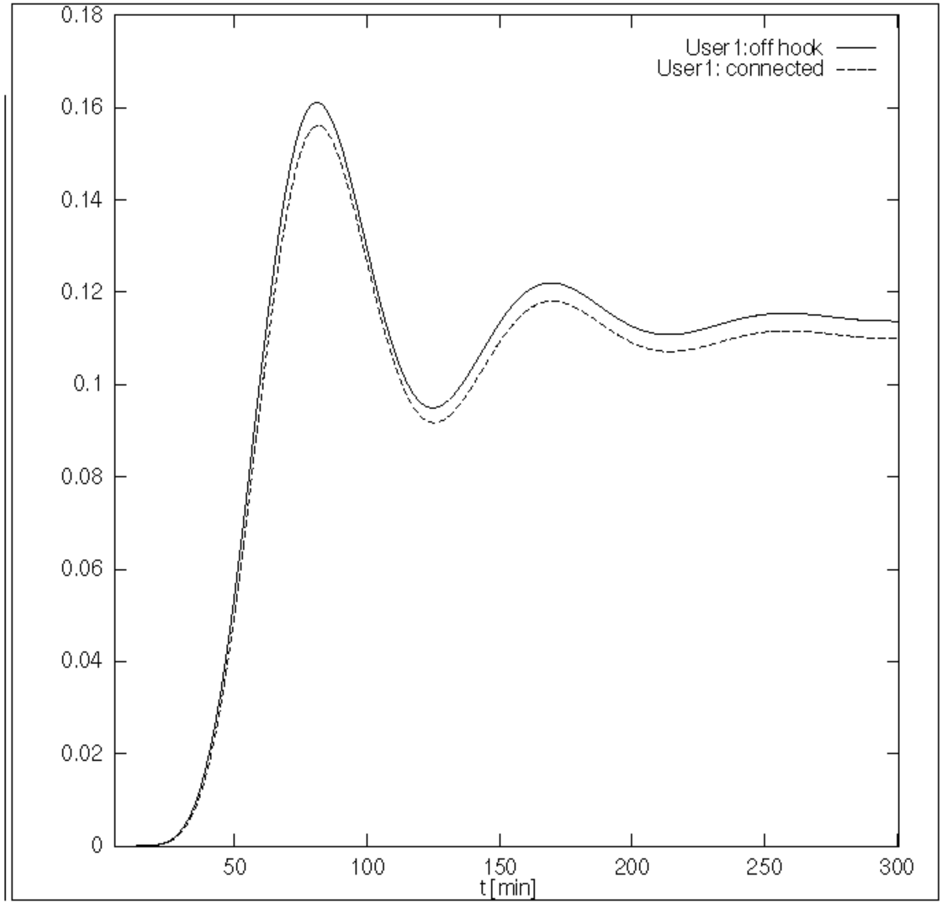
- A particular phone: 
- The time it takes to pick up the phone: 
- The phone with time constraints:

on ***ringT_on***
delay ***pick_phone***
in  by 
 ***ringT_off***



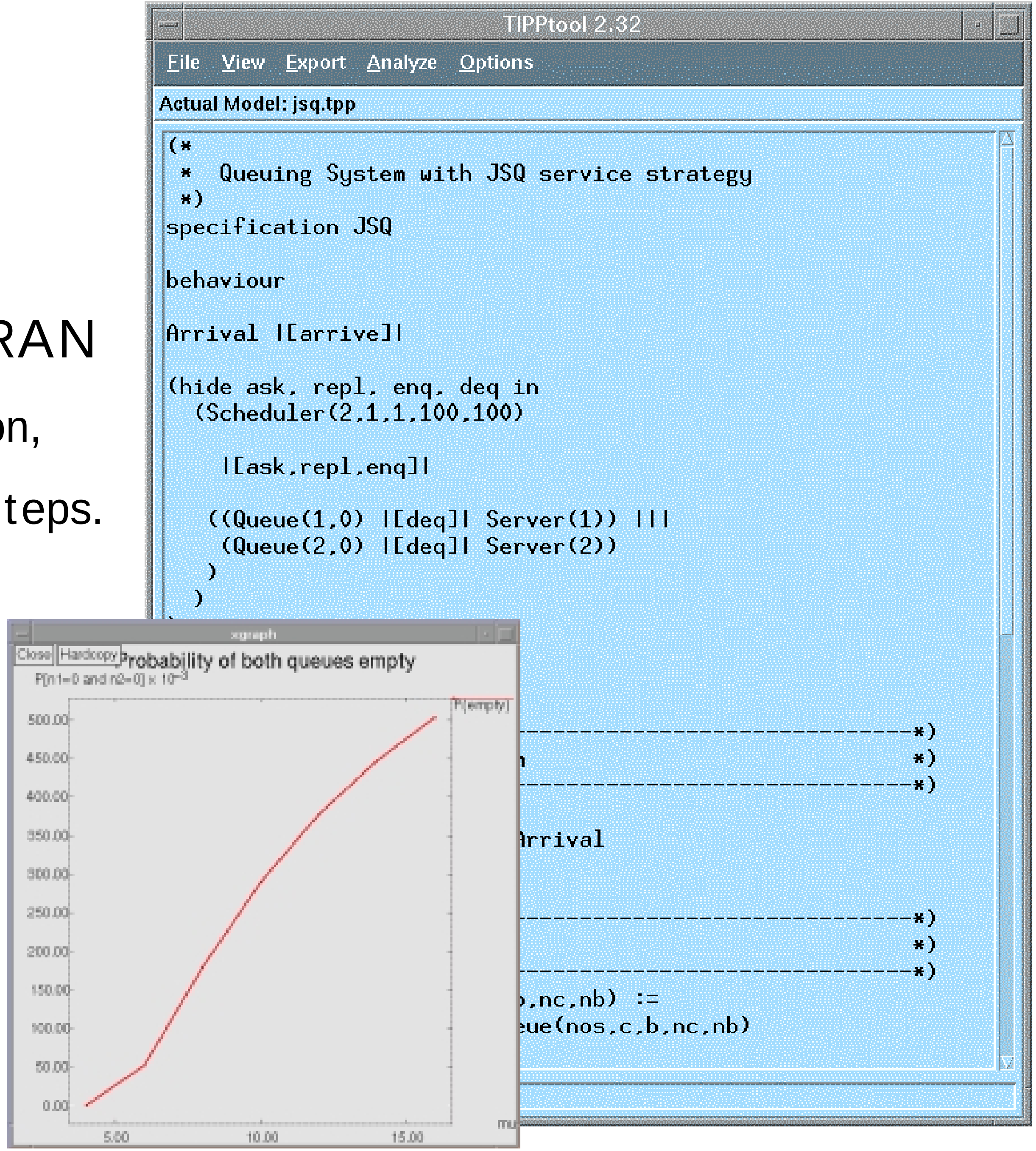
Analysis results

- 14 different time constraints incorporated.
- Compositional minimisation to avoid state space explosion.
- Here: two subscribers phoning each other.



Tools used

- CAESAR/ALDEBARAN
  - original specification,
  - first minimisation steps.
- TIPPtool
  - time constraints,
  - final minimisations,
  - numerical analysis.



Contents

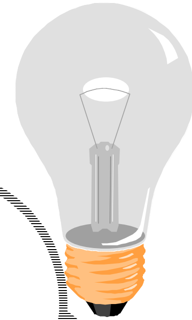
- Introduction to Stochastic Process Algebra
  motivation, concepts of PA & SPA
- Markovian Process Algebra
  Interactive Markov Chains.
- Non-Markovian Process Algebra
  GSMPs, Discrete Event Simulation.

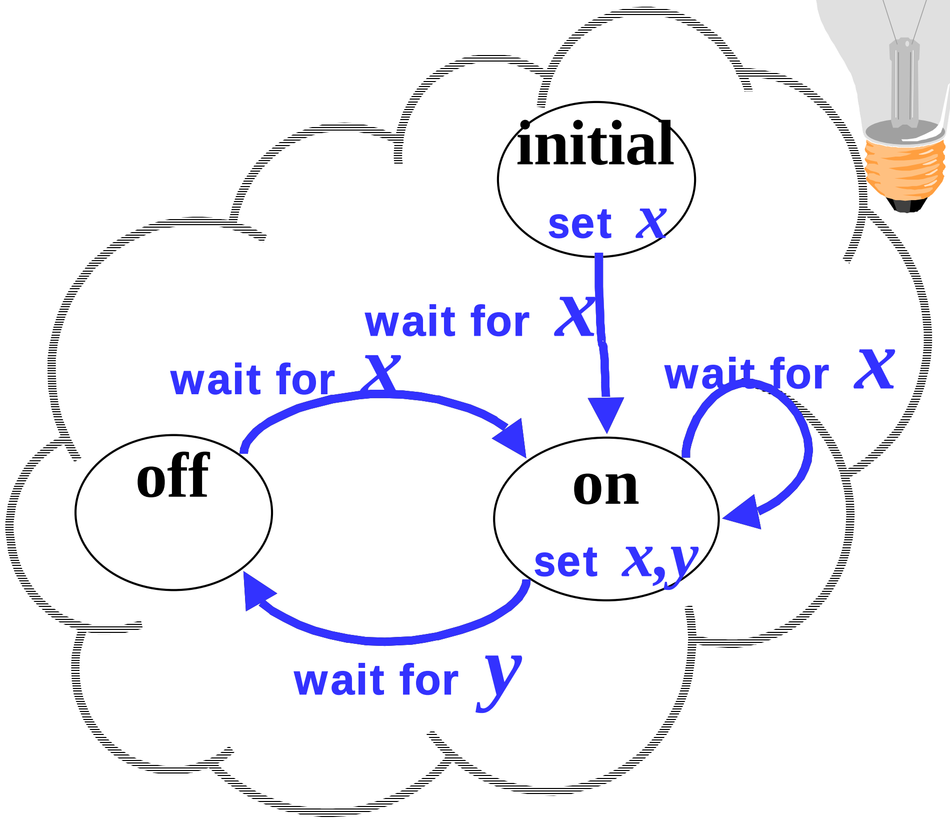
Non-Markovian approaches

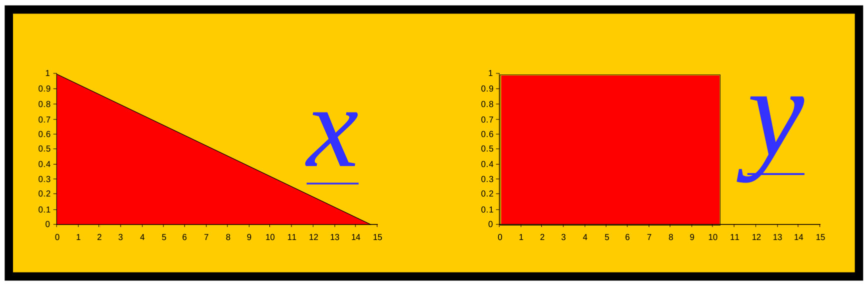
- **Traditional methods:**
  - **queueing networks**
  - **stochastic Petri nets (SPN)**
  - **generalized semi-Markov processes (GSMP)**
  - **no compositionality**
- **General SPAs: TIPP, GSPA, Sπ^+**
  - **compositionality**
  - **no expansion law**
  - **infinite semantic objects for recursion**

A light controller

- The light is turned **on** if someone enters the stairway.
- It goes **off** after 10.3 minutes exactly.
- People arrive randomly, at least every 15 minutes, with **uniform** probability.







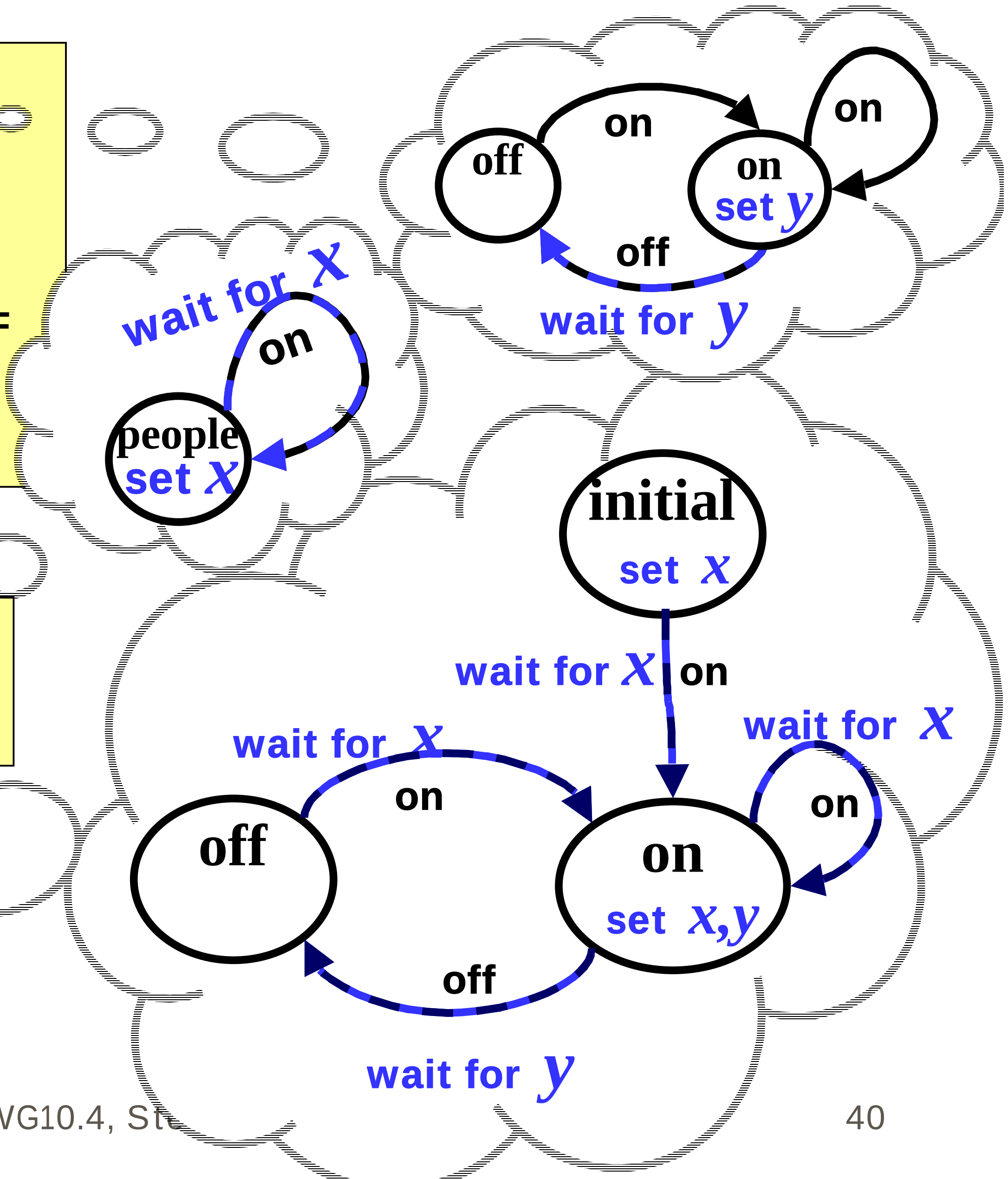
A light controller

```
process LIGHT_OFF :=
   on ; LIGHT_ON
endproc

process LIGHT_ON :=
   {|y|}{y} -> off; LIGHT_OFF
   + on ; LIGHT_ON
endproc
```

```
process PEOPLE :=
   {|x|}{x} -> on; PEOPLE
endproc
```

PEOPLE |[on]| *LIGHT_OFF*



Stochastic automata (SA)

- model inspired by Timed Automata [Alur&Dill]
- close link to GSMPs [Whitt,Glynn]
- based on a notion clocks
- compositional
- operational model of a process algebra ♠
- expansion laws and finite objects

Ingredients of an SA

$$(S, s_0, C, A, \rightarrow, K, F)$$

- **control states** or **locations** S
- **initial state** s_0
- finite set of **clocks** C
- **actions** A
- **transition relation** $\rightarrow$
- **clock assignment** K
- **distribution assignment** F

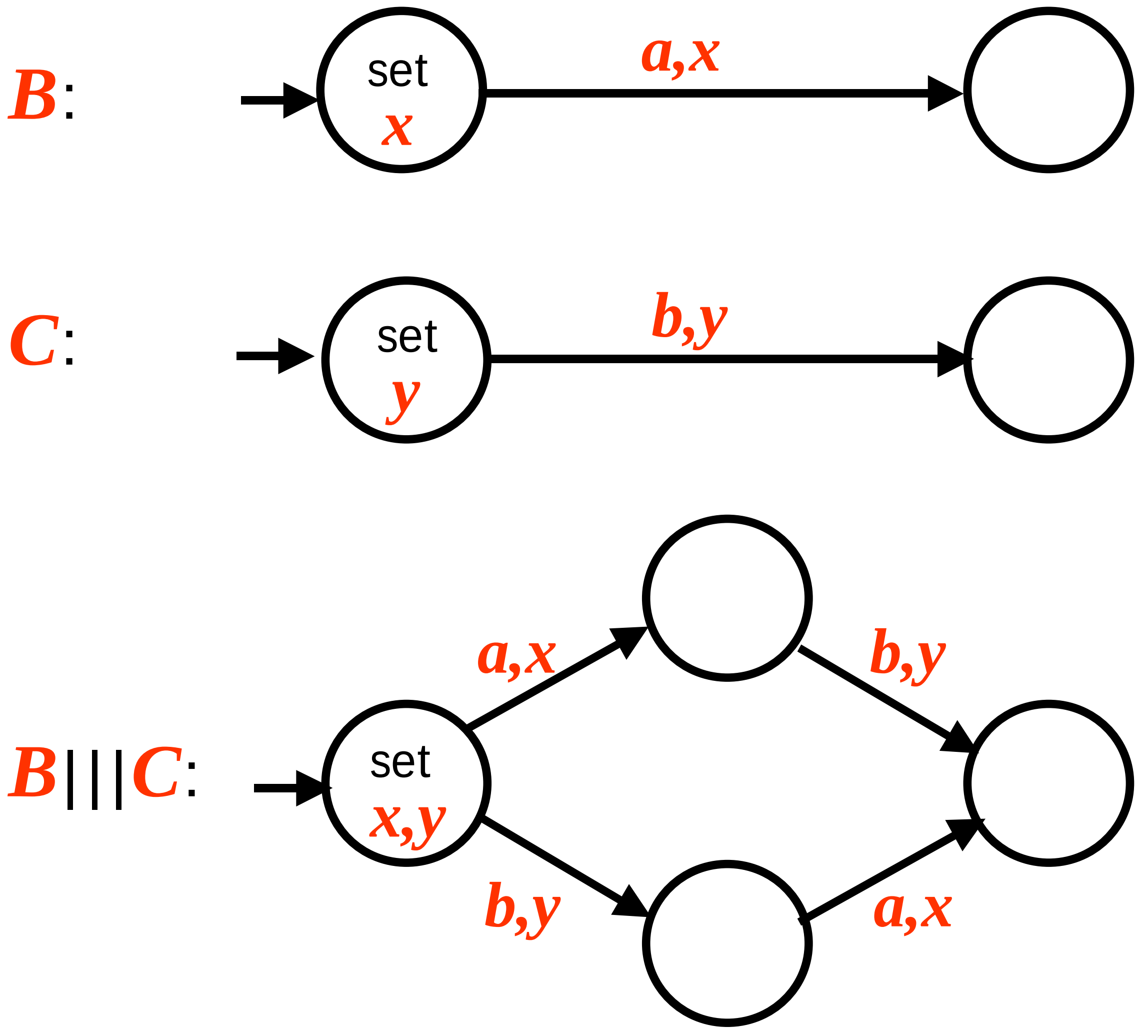
The algebra ♠

- signature of ordinary PA

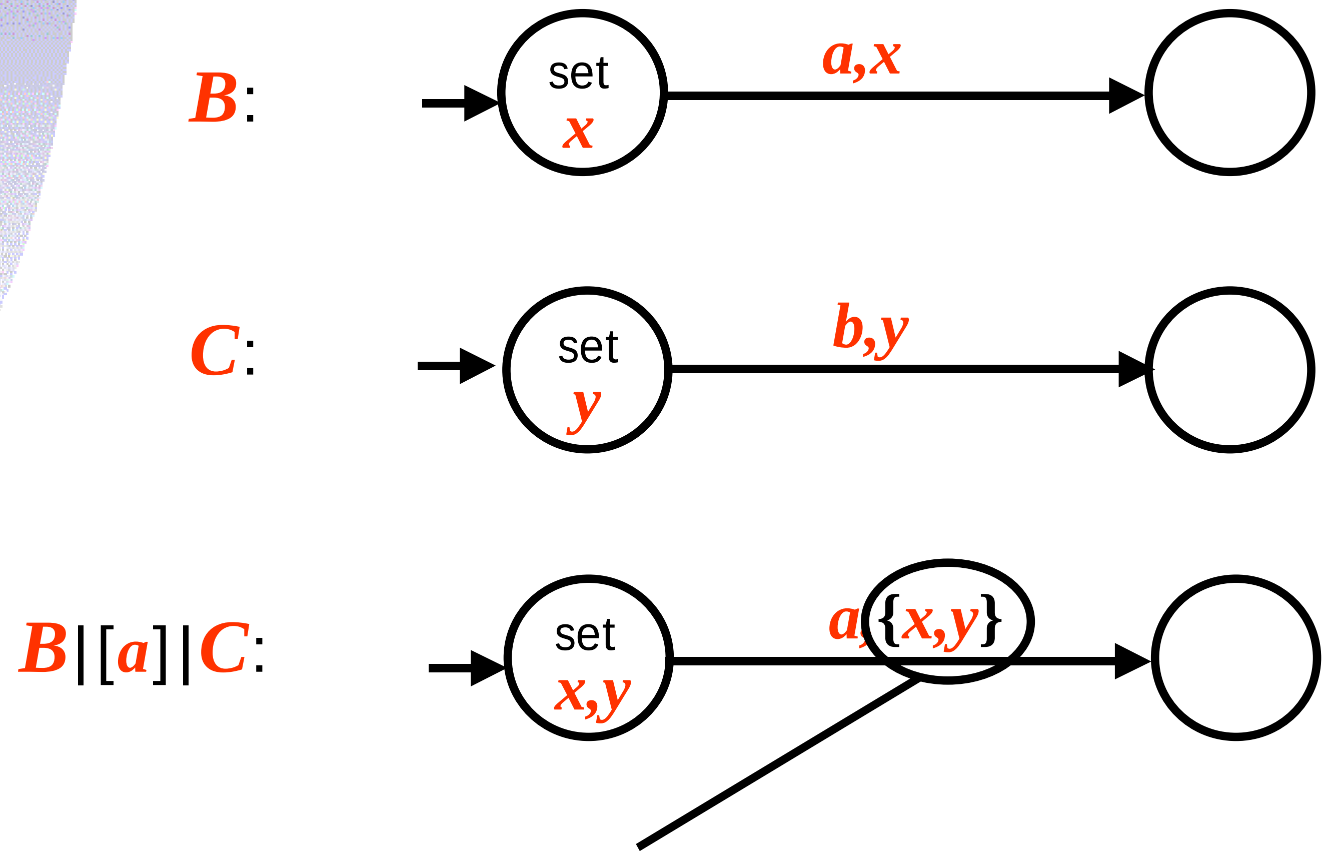
  a;B , B + C, B $||_A$ C, B \ A, etc.

- clock related operators
  - **clock setting:** **{|C|}** ***B***
  - **guarding:** **C** $\rightarrow$ ***B***

Parallel composition



Synchronization



Synchronization by union of guards = maximum of distributions

Expansion law

Let $B = \{|C\ |\}\ B'$ **and** $D = \{|C'|\}\ D'$

with $B' = \Sigma_k\ C_k \rightarrow a_k\ ;\ B_k$ and $D' = \Sigma_l\ C_l \rightarrow c_l\ ;\ D_l$

then

$B\ ||_A\ C =$

$\{|C \cup C'\ |\}$

$(\ \Sigma\ \{C_k \rightarrow a_k\ ;(B_k\ ||_A\ C\ )\ |\ a_k \notin\ A\ \}\ +$

$\Sigma\ \{C_l\ \rightarrow c_l\ \ ;(B\ ||_A\ C_l\ \ )\ |\ c_l \notin\ A\ \}\ \ +$

$\Sigma\ \{(C\ \cup C'\ )\ \rightarrow d\ \ ;(B_k\ ||_A\ C_l\ )\ |\ d = a_k = c_l \in\ A\ \}\ )$

An application

A multiprocessor mainframe

[Herzog & Mertsiotakis]

- different programming jobs
- different user transactions
- maintenance database
- occurrence of software failures

A specification

System := Load $||_L$ (Mainframe $||_F$ Maintain)

Load := PL_1 $||_C$ UL_1 $||_C$ FL_1 $||_C$ ChangePhase

ChangePhase := change($x_{W(v,w)}$) ; ChangePhase

UL_1 := nextUserJob($xu_{exp(\mu_1)}$) ;
(userJob ; UL_1 + reject ; UL_1)
+ change ; UL_2

UL_2 := ... UL_3 := ...

Mainframe := Queues $||_{G \cup F}$ (P_1 $||_F$ P_2 $||_F$ ... $||_F$ P_m)

Maintain := fail ; repair($z_{\gamma\,(c,c')}$) ; Maintain

a(z) ; P is shorthand for {|z|}{z}$\rightarrow$ a; P

Simulation

- using variable time advance procedure
- relevant history of system stored in finite expressions in ♠
- calculate relevant parts of the SA on-the-fly using expansion theorem

Conclusion

It is possible to model and analyse both qualitative and quantitative aspects of reactive systems in one (family of) formalism(s)

Markov chains ⇔ Markovian PA & TIPPtool

analytic techniques & numerical algorithms

GSMPs ⇔ stochastic automata & ♠

discrete event simulation

+ qualitative analysis & nondeterminism

Current developments

- modelling language & toolset **MoDeST**
  - data structures
  - real time & stochastic time
  - open tool architecture
- model checking on CTMCs: **E$\tau\theta$MC^2**
  - specification logic for performance measures
  - automated property-driven CTMC simplification & analysis