Markov Nets

a theory of concurrent stochastic machines

Albert Benveniste, Eric Fabre, Stefan Haar, IRISA

February 2001

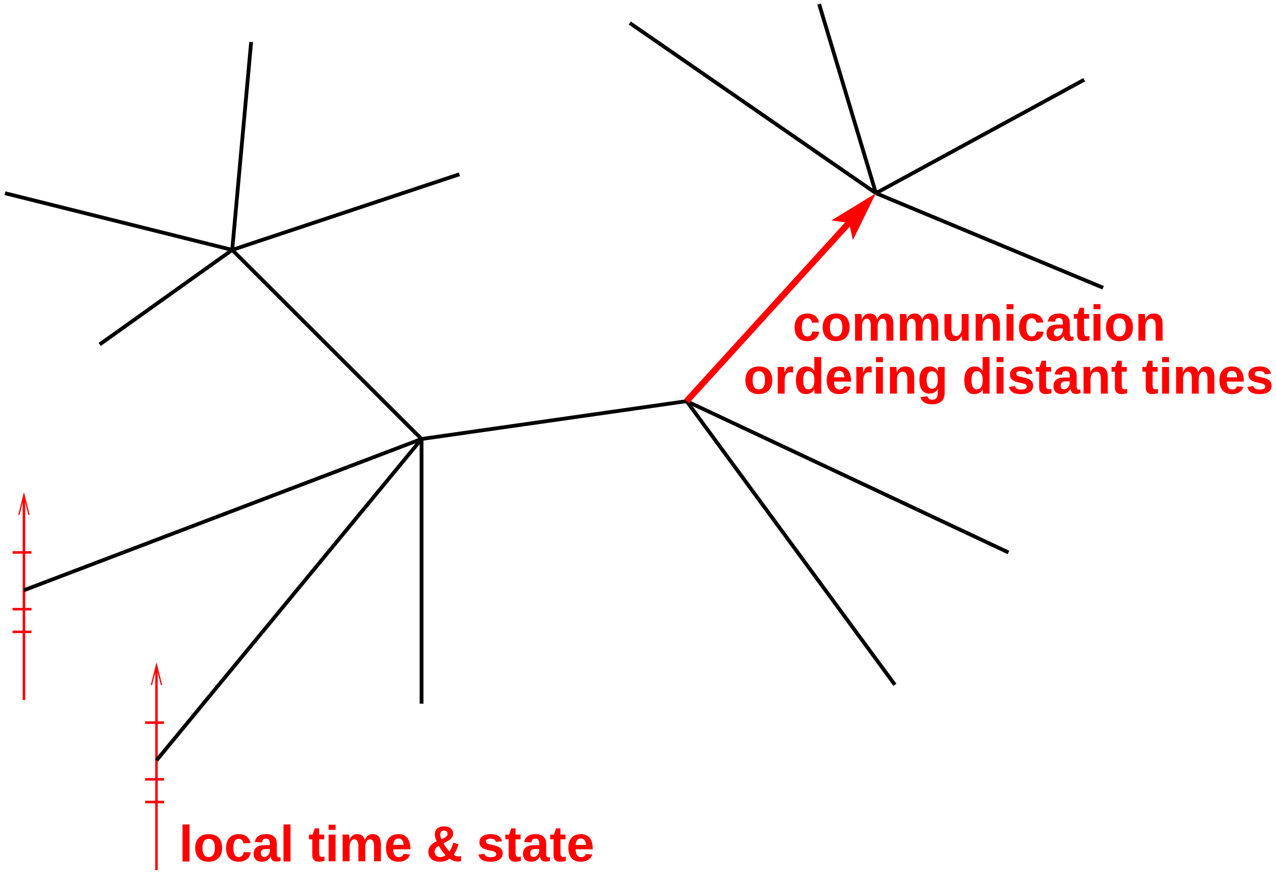
motivations & objectives

monitoring (or control) of large networked systems

architecture of distributed agents

seen as hidden state estimation from local observations

need for an HMM (Hidden Markov Model) theory for distributed and concurrent systems time & state in distributed systems. . .



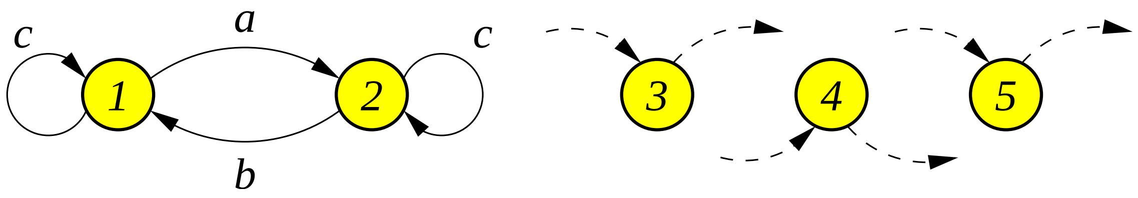
. . . they are *local*

automata & their products

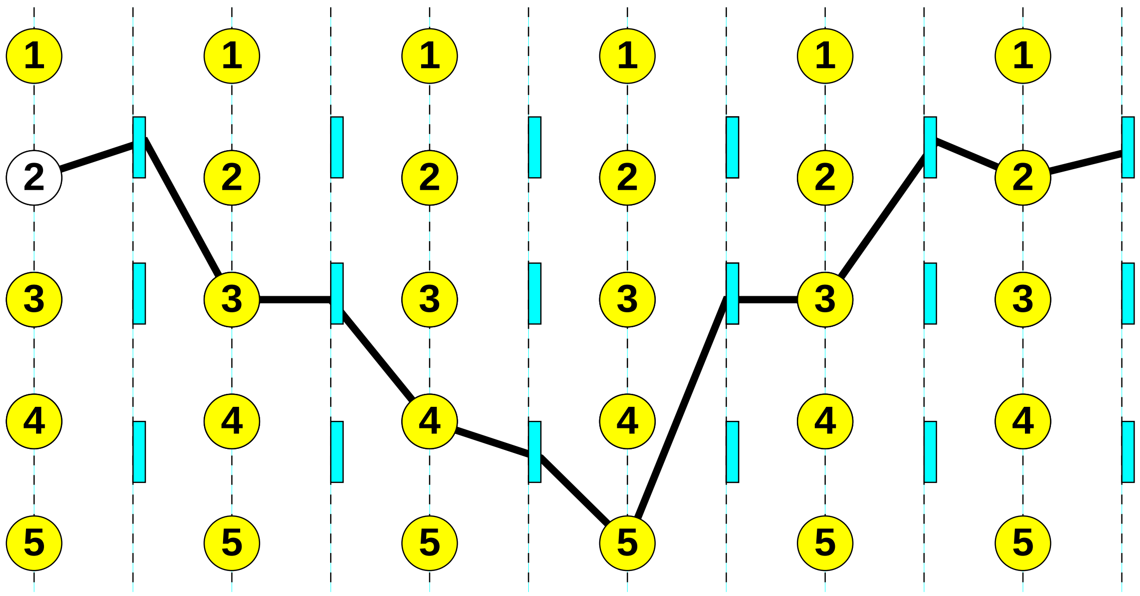
$\Downarrow$ *by randomizing*

HMM & their products

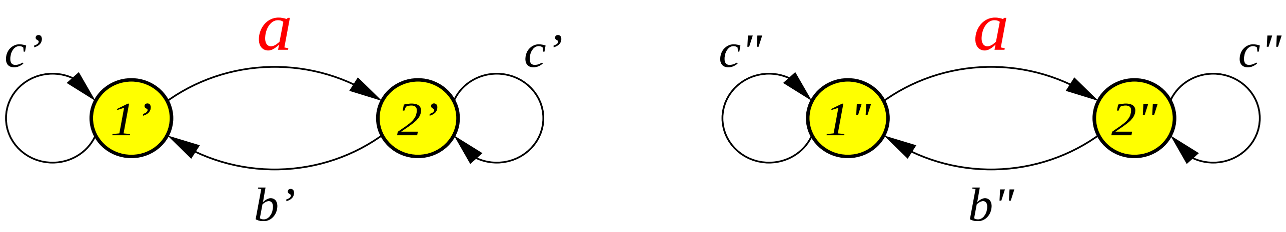
automata & HMM, and their runs

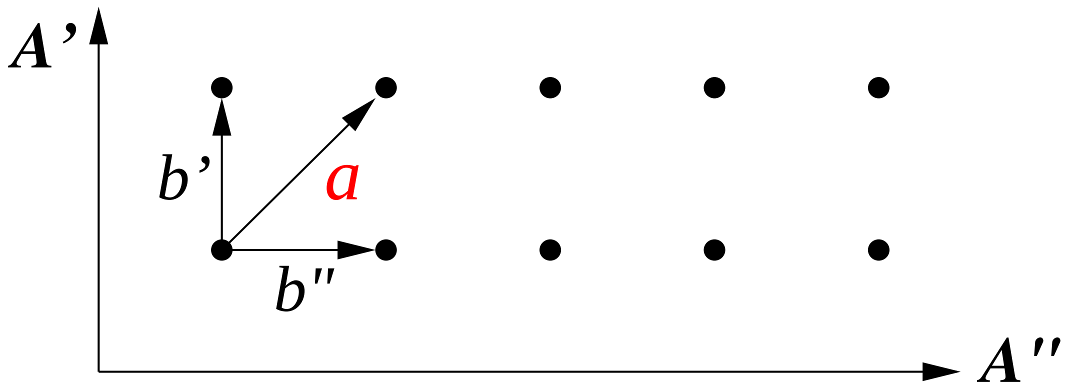


$$(\mu(x_0), \pi(x_-, e, x)) \longmapsto \left\{\Omega, \mathbf{P}^{(\mu,\pi)}\right\}$$



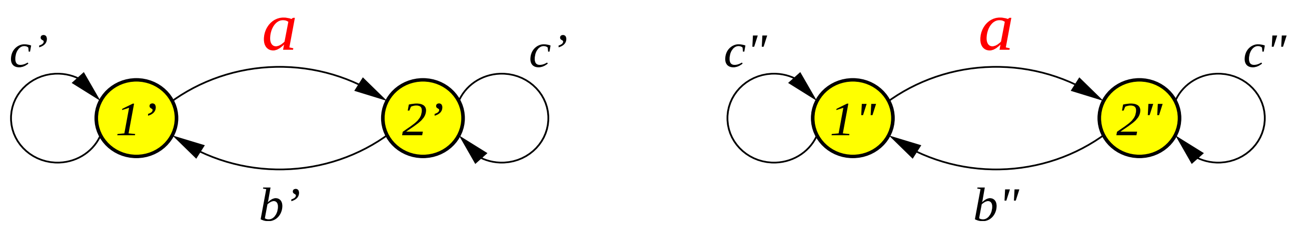
products of automata





components synchronize on their shared events

products of HMMs ?



randomize the following :

(1) to have A' making a private move

(2) to have A'' making a private move

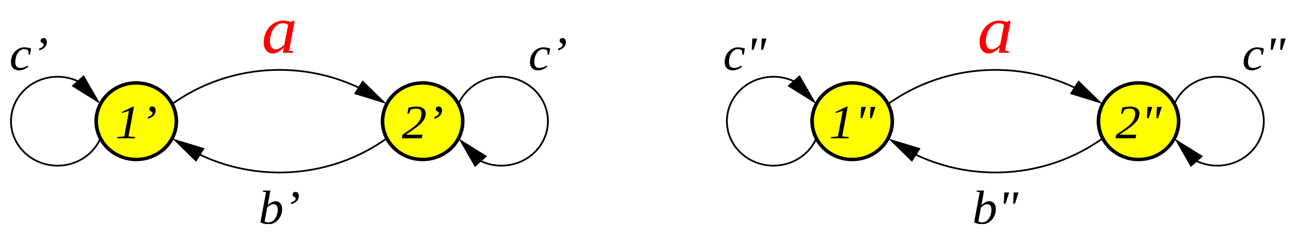
(3) to have A', A'' making a joint move.

Possible solution : a triple $(\alpha_p, X'_{n'}, X''_{n''})$, where

- $\alpha_p \in \{1, 2, 3\}$ is a randomized arbiter,
- $X'_{n'}, X''_{n''}$ move depending on the case $1, 2, 3$ selected by α_p

$\sim$ race policies in stochastic timed Petri nets ; gives different likelihoods to different interleavings ; independence does not match concurrency. We dislike !

$\boxed{\text{products of HMMs: a better idea}}$



(1) to have A' making a private move
(2) to have A'' making a private move
(3) to have A', A'' making a **joint** move.

a **triple** $(A_p, X'_{n'}, X''_{n''})$, where

- $A_p \in \{1, 2, 3\}$ is an arbiter (**not randomized** !),
- $X'_{n'}, X''_{n''}$ move depending on the case $1, 2, 3$ selected by α_p

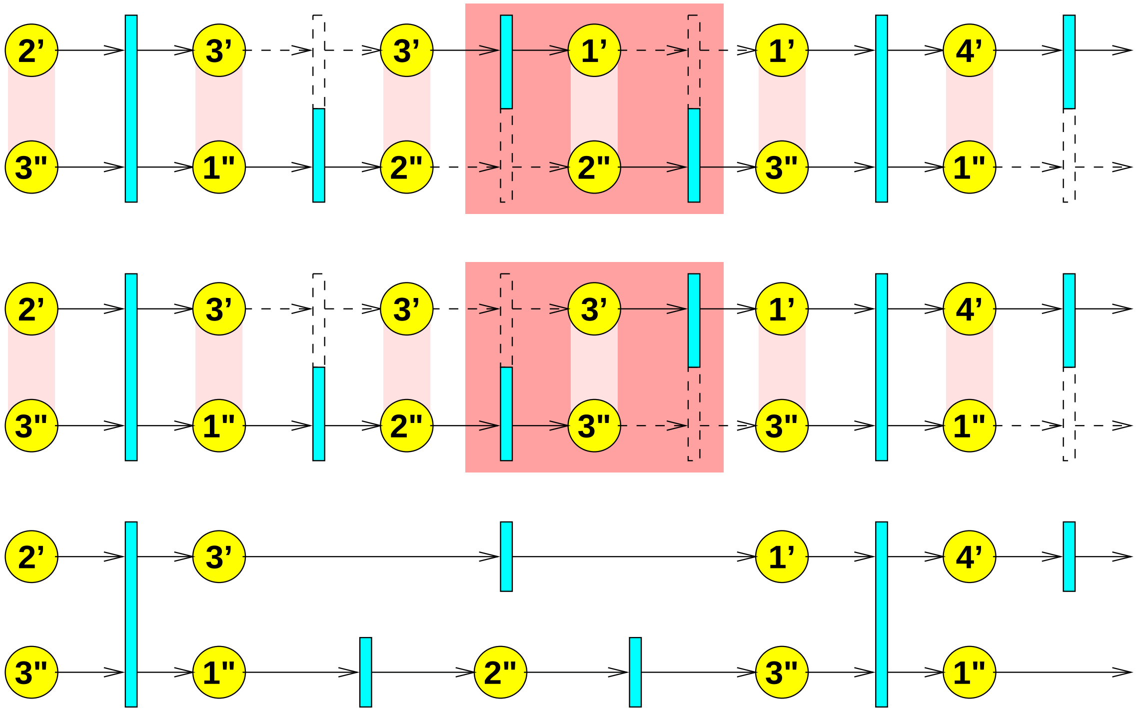
Leads to $\boxed{\text{partially}}$ **stochastic processes**, difficult

Petri nets & their products

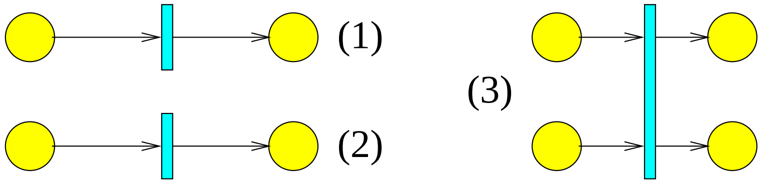
$\Downarrow$ *by randomizing*

Markov nets & their products

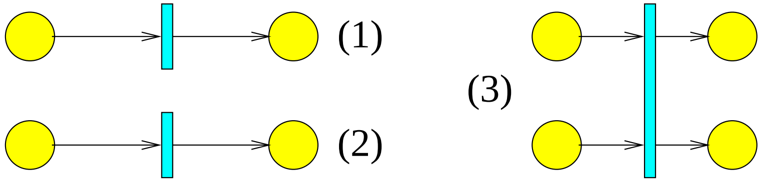
from interleaving to partial orders...

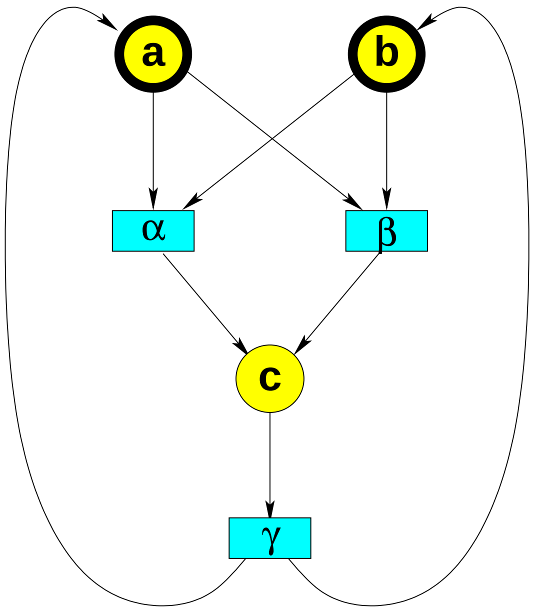


...and to Petri nets:

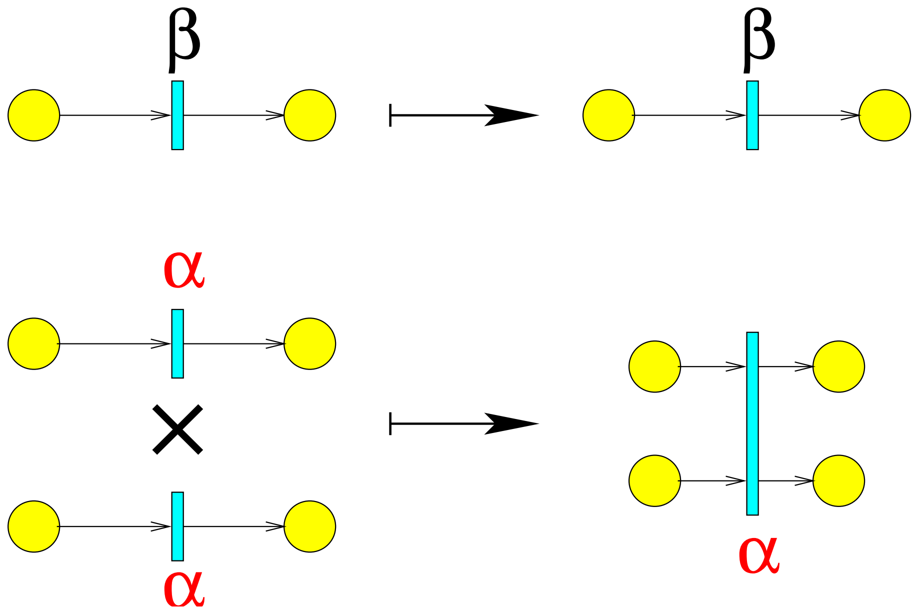


labelled Petri nets (of capacity one)





products of labelled Petri nets

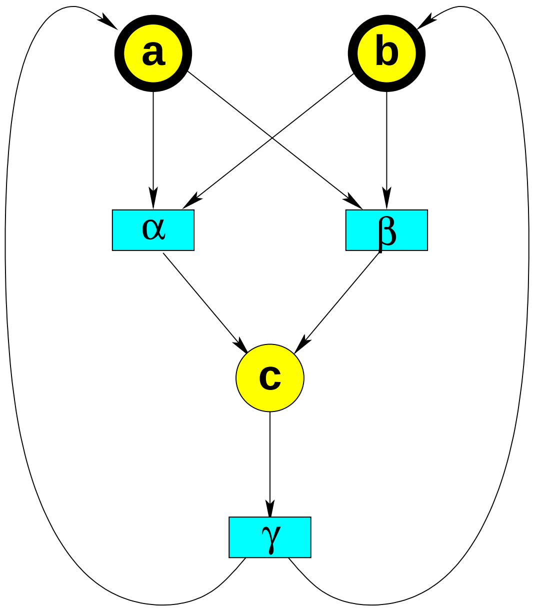


β : local label

α : shared label

the set of product transitions builds the product PN

labelled Petri nets



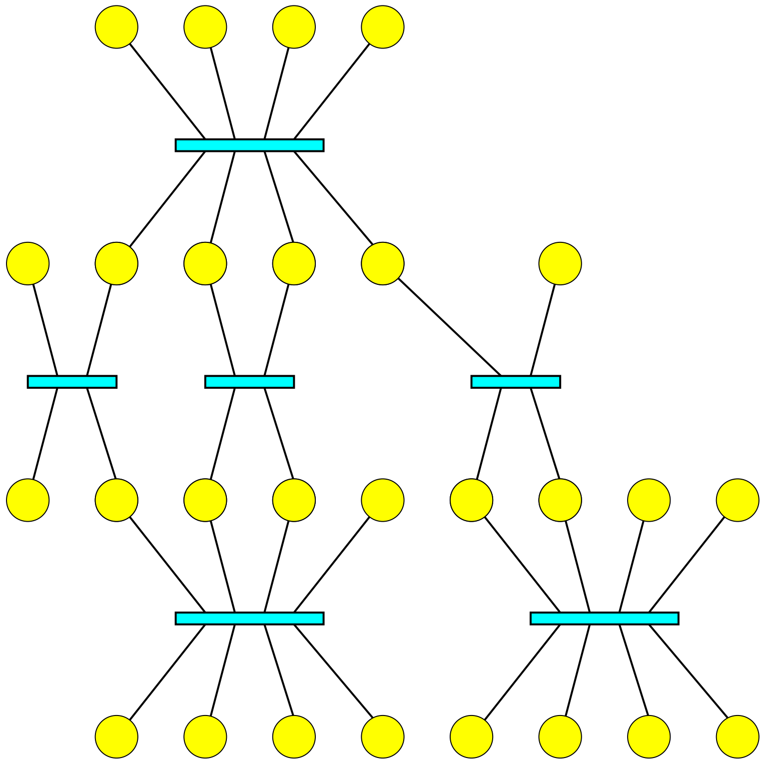
how to model their runs ?

$\boxed{\text{1 run} = \text{1 configuration}}$

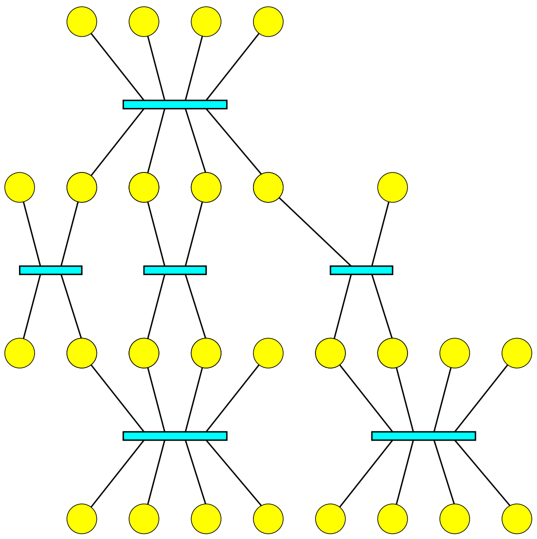
“puzzle” from
a set of
prototype tiles

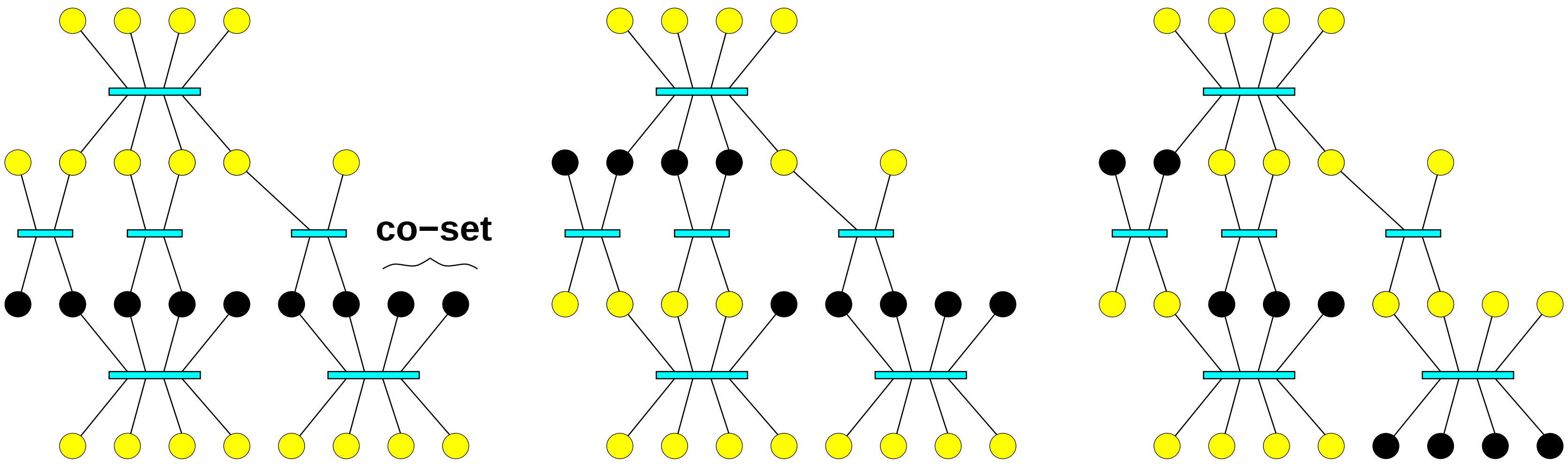
places exhibit
no choice

transitions
can be concurrent

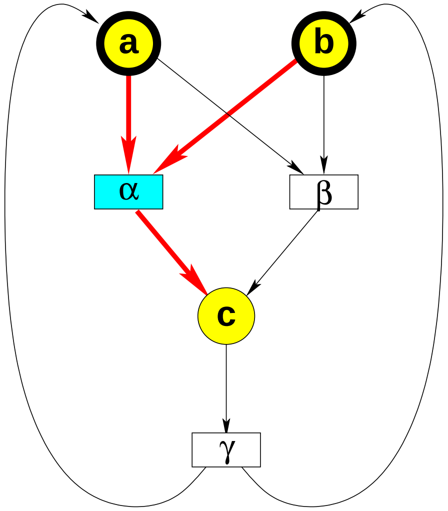


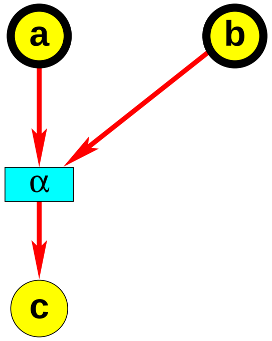
configurations, cuts (states), and co-sets (local states)



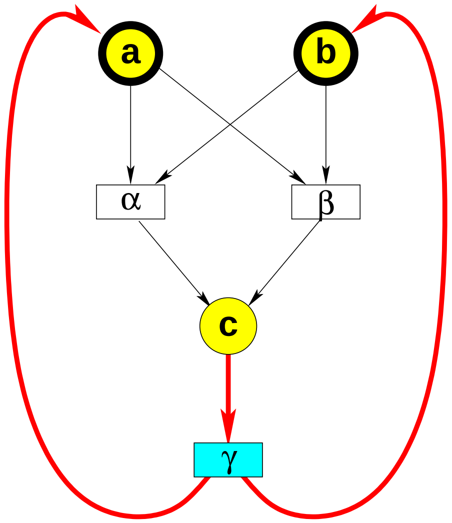


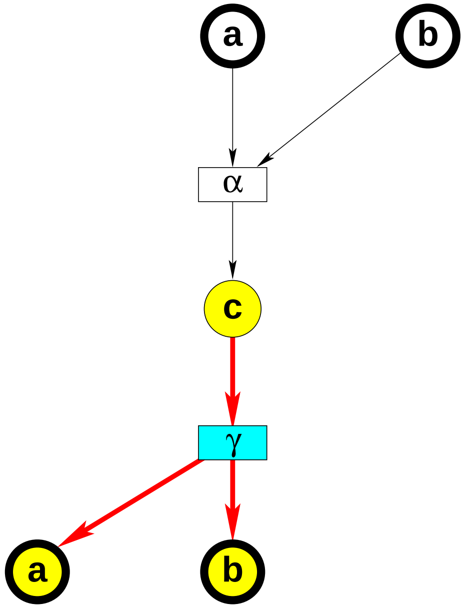
set of all runs of Petri nets : *unfoldings*



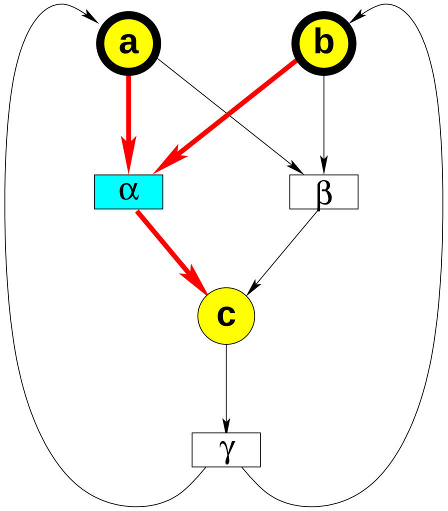


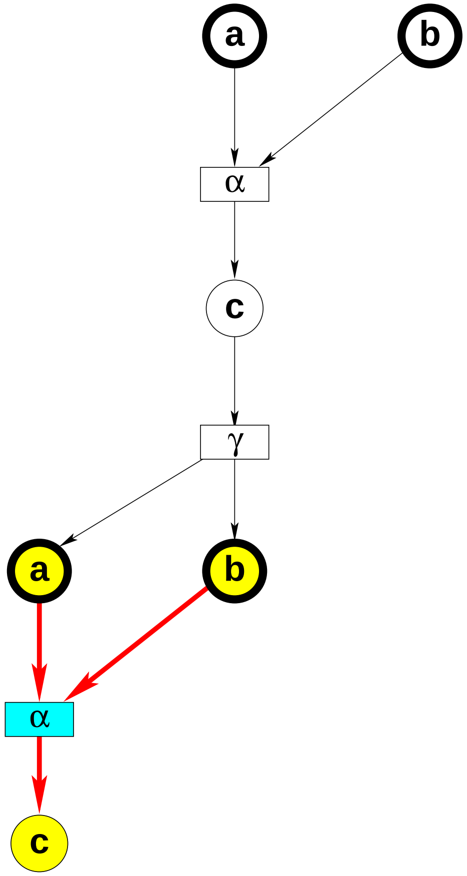
runs of Petri nets : *unfoldings*



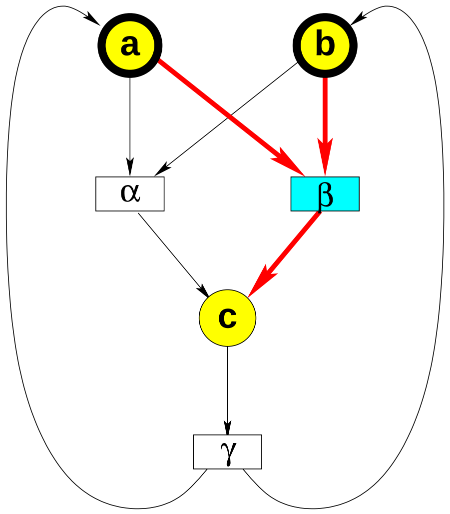


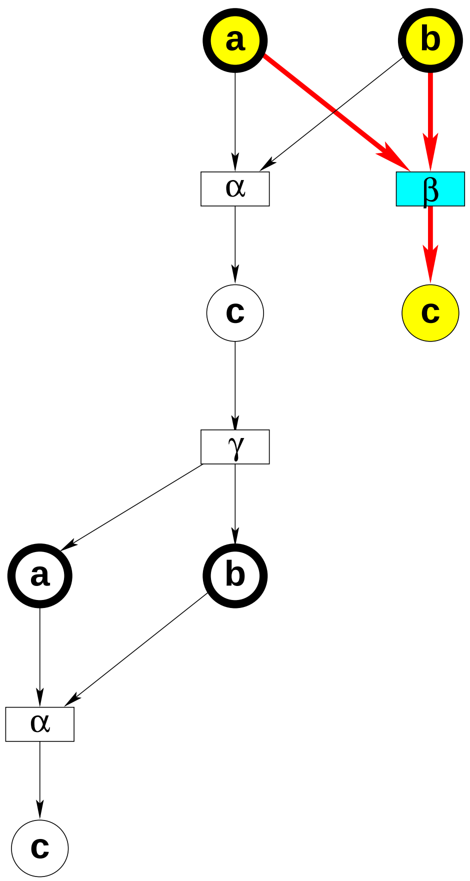
runs of Petri nets : *unfoldings*



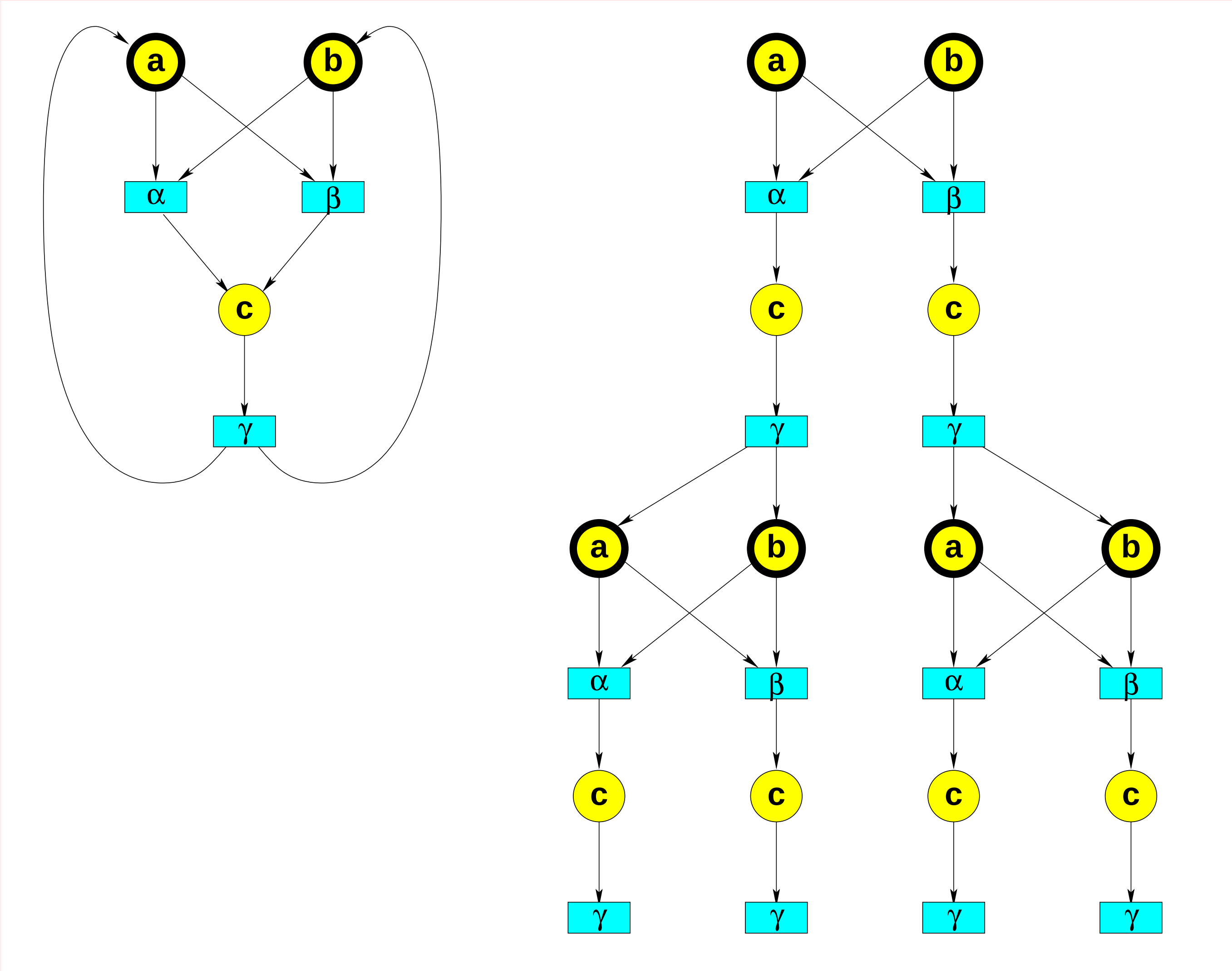


runs of Petri nets : *unfoldings*

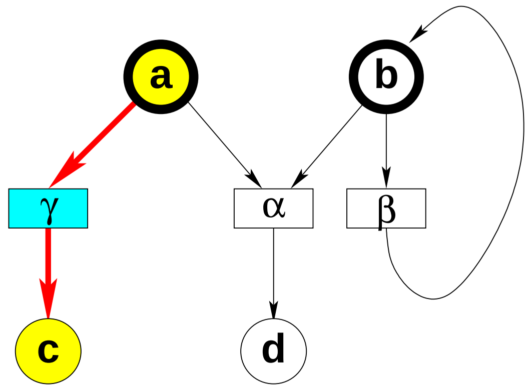


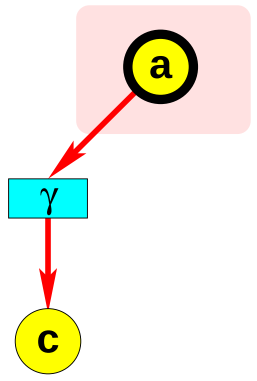


runs of Petri nets : *unfoldings*

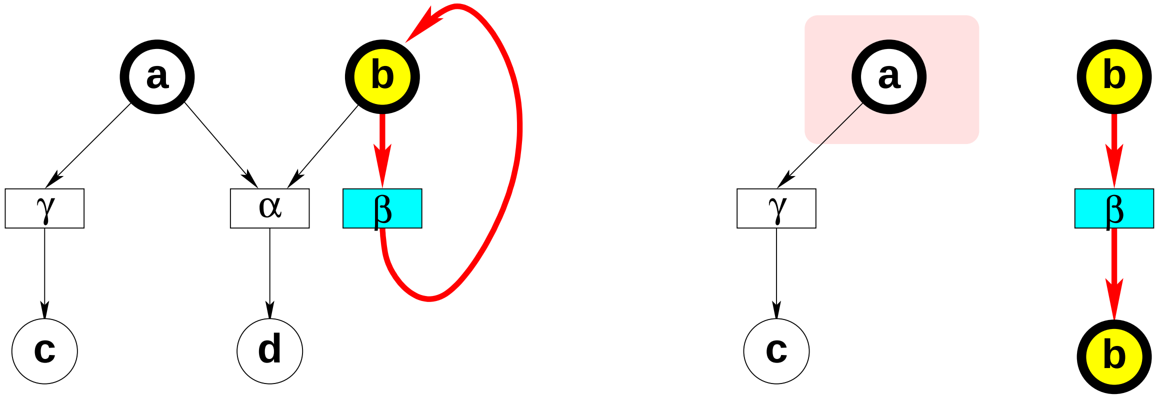


Petri nets and their unfoldings, cont'd

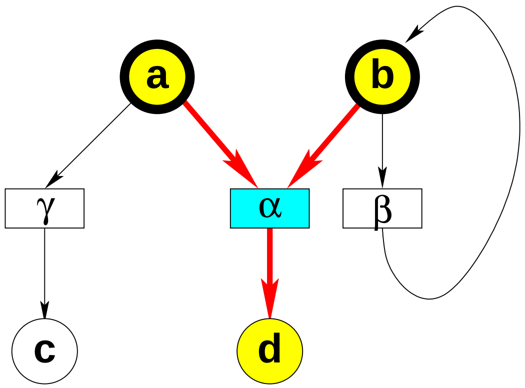


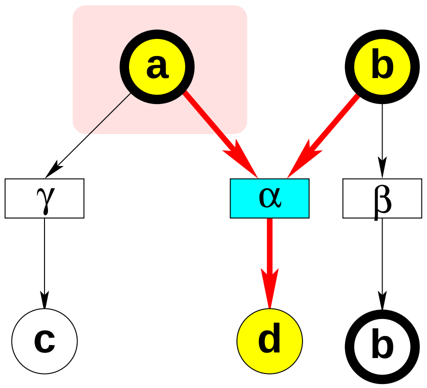


Petri nets and their unfoldings, cont’d

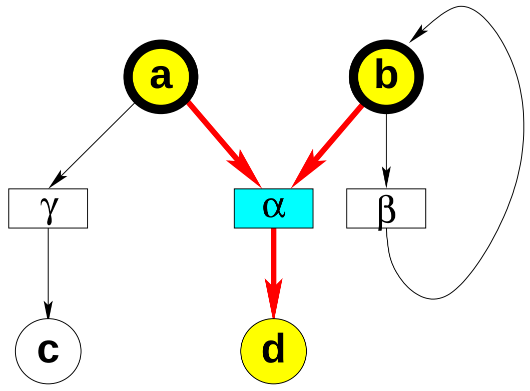


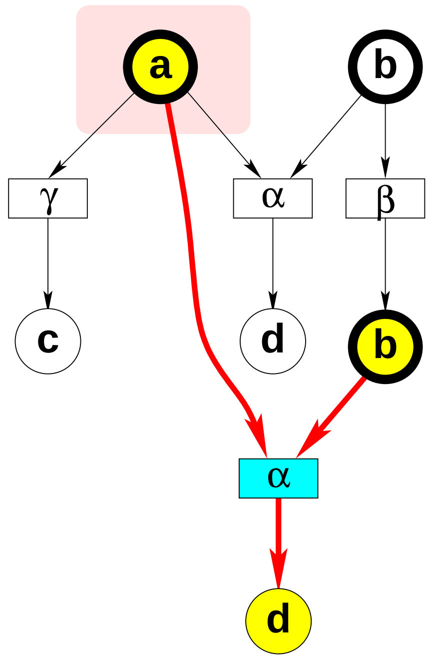
Petri nets and their unfoldings, cont'd



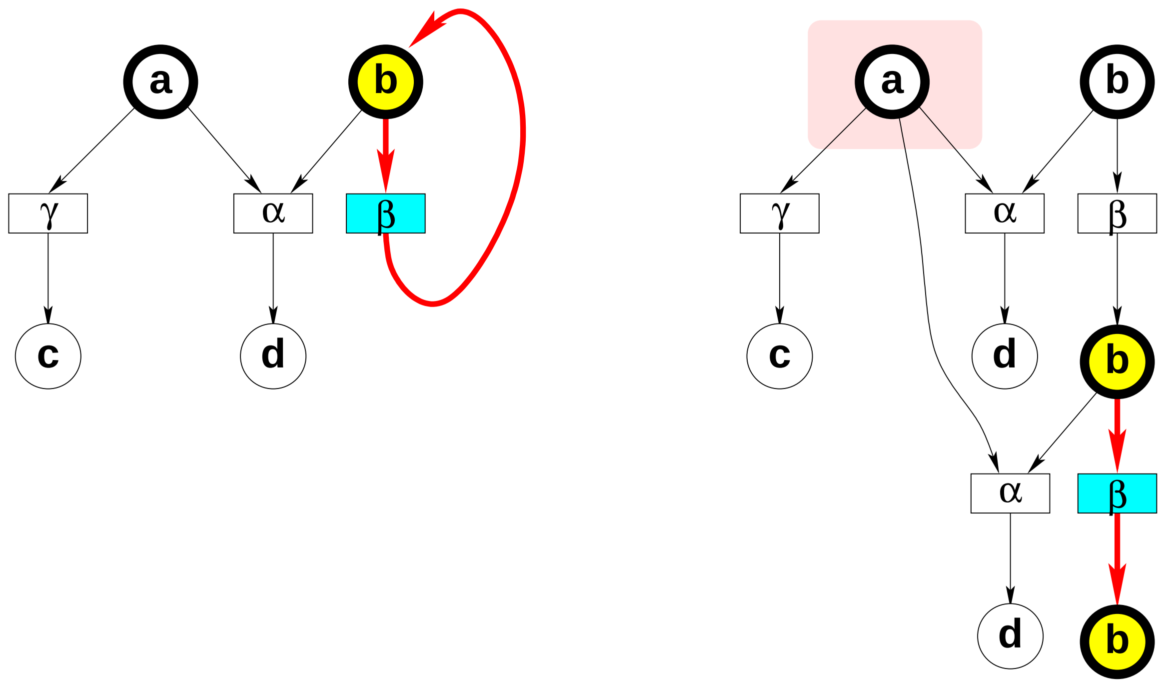


Petri nets and their unfoldings, cont'd

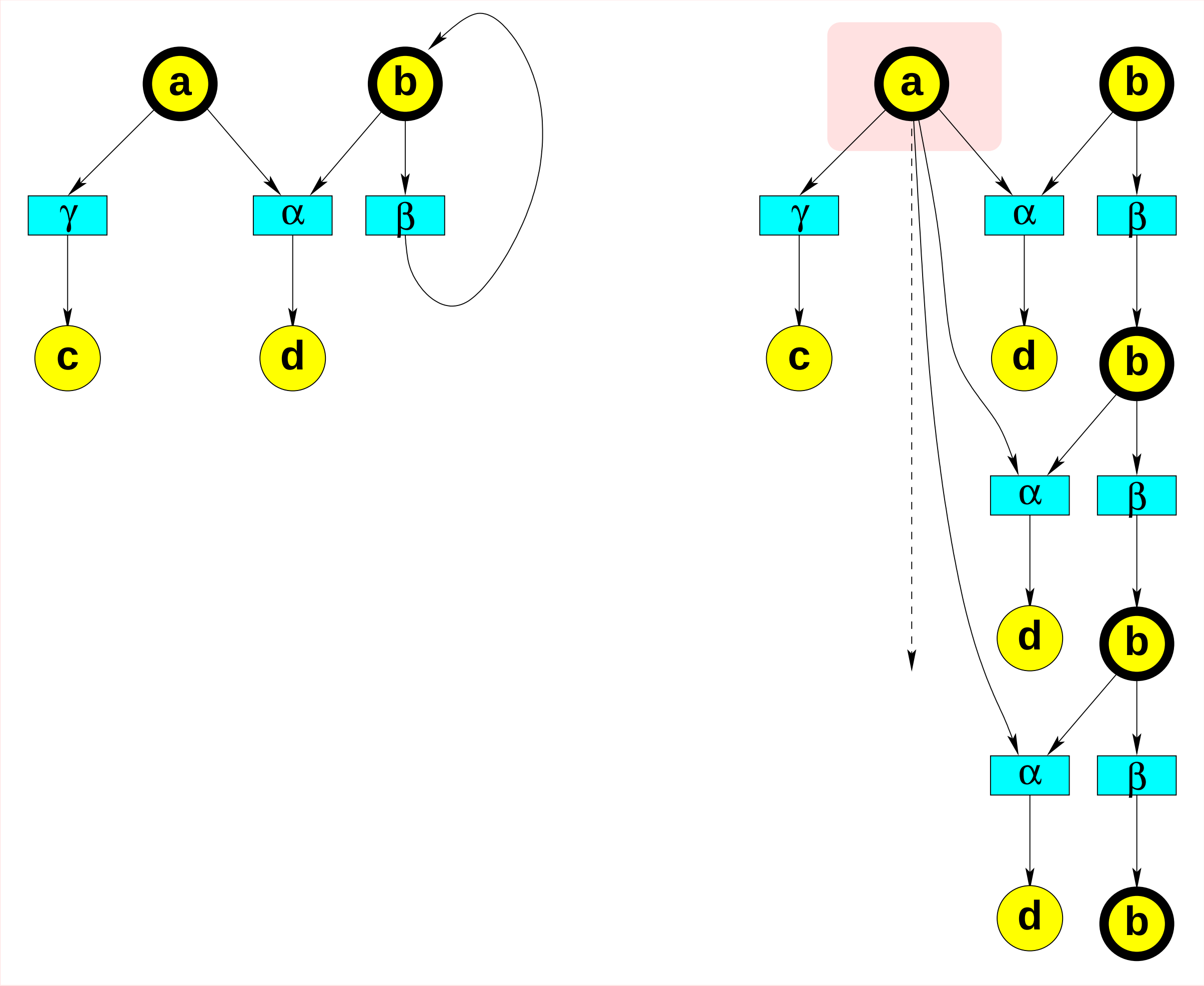




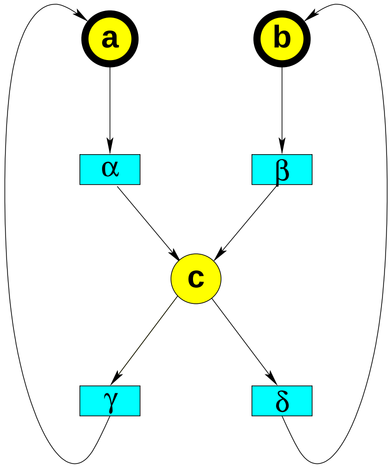
Petri nets and their unfoldings, cont'd

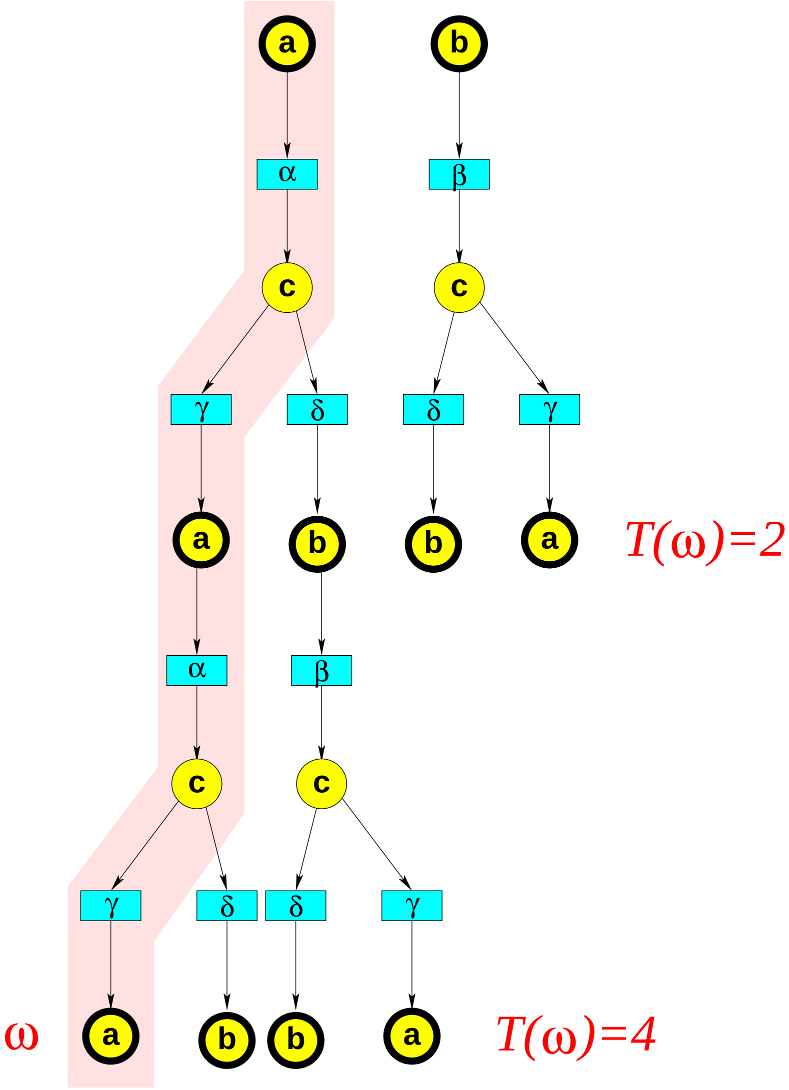


Petri nets and their unfoldings, cont'd

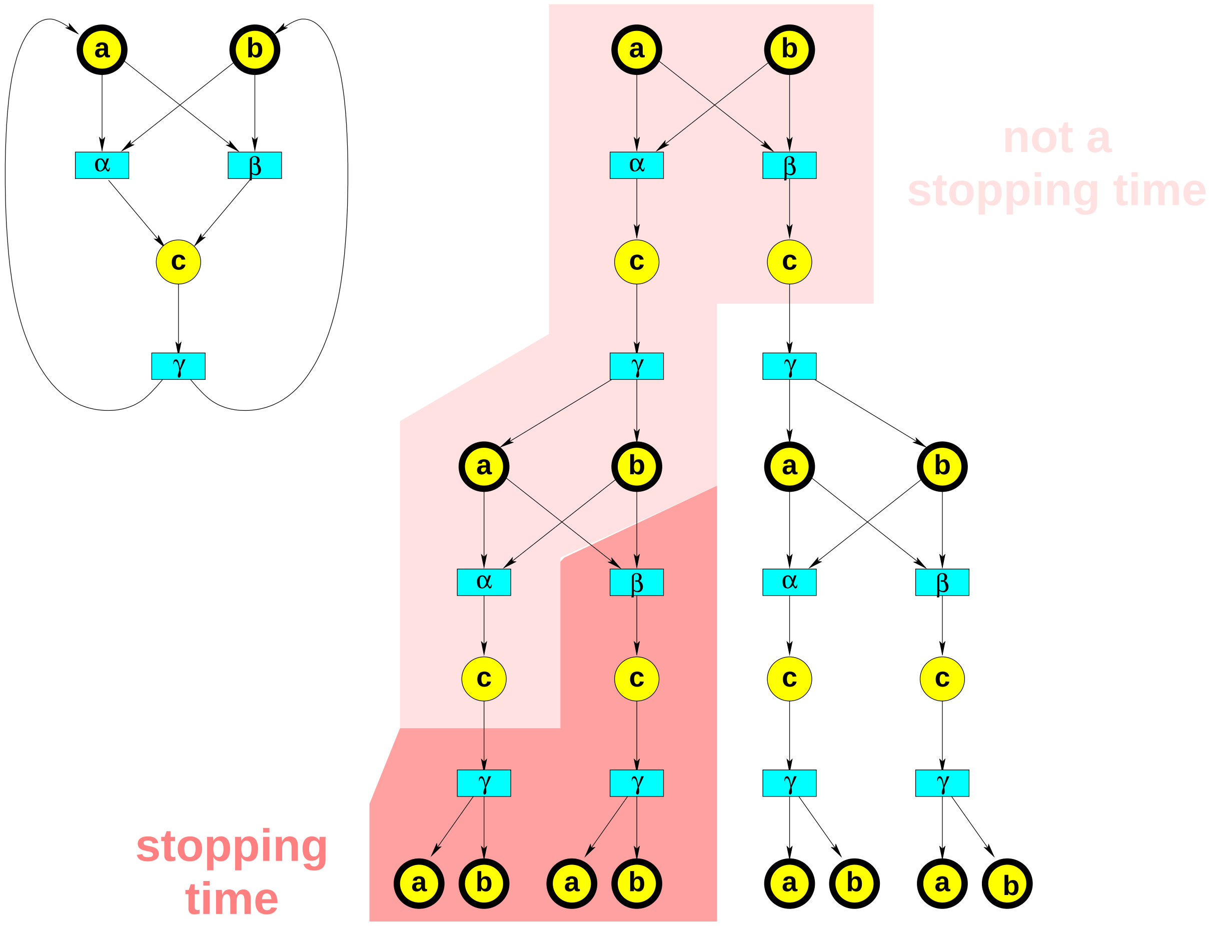


Unfolding automata, stopping times

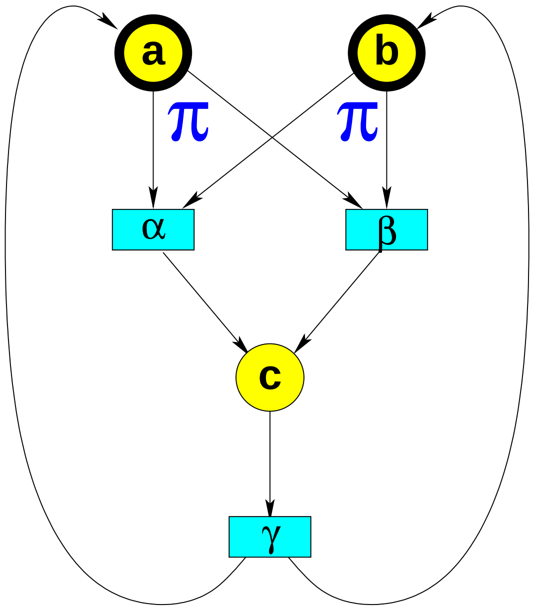




Petri net unfoldings : stopping times



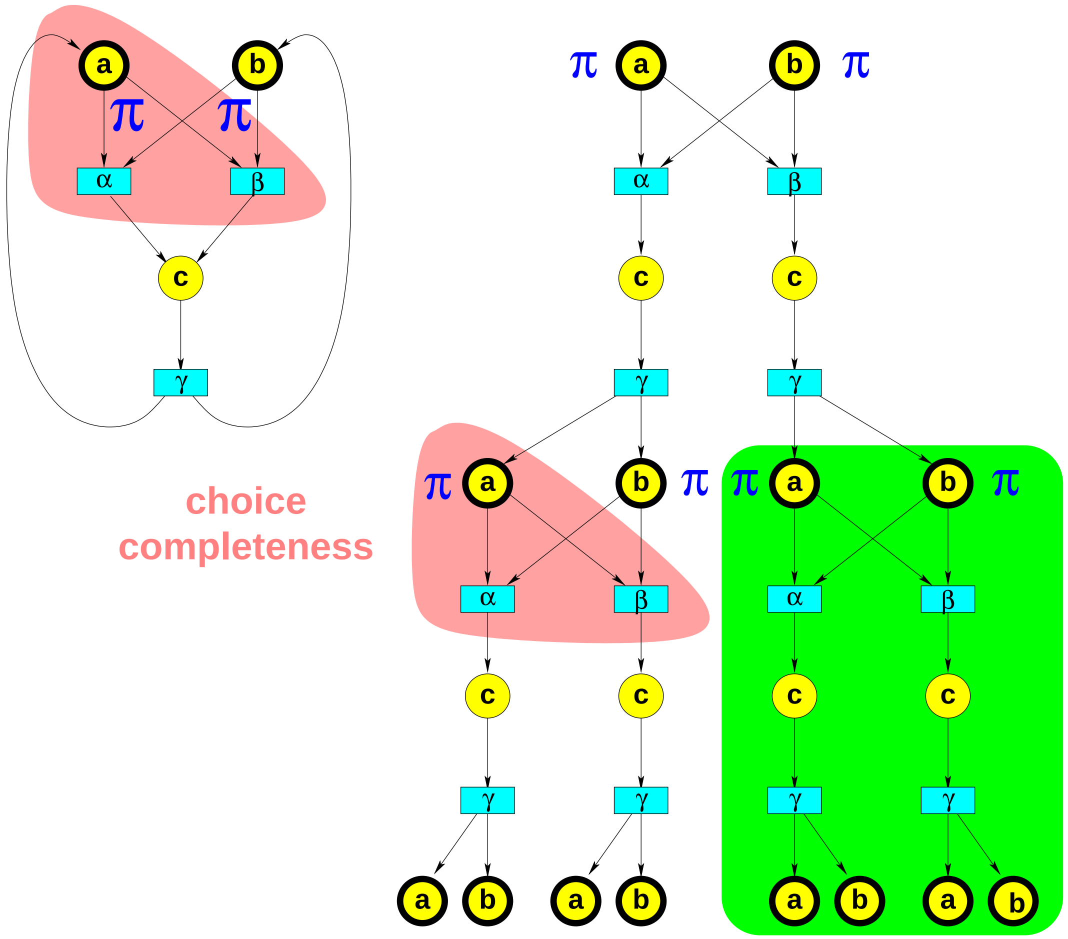
Markov net : randomizing Petri nets



since places are not shared, product is immediate !

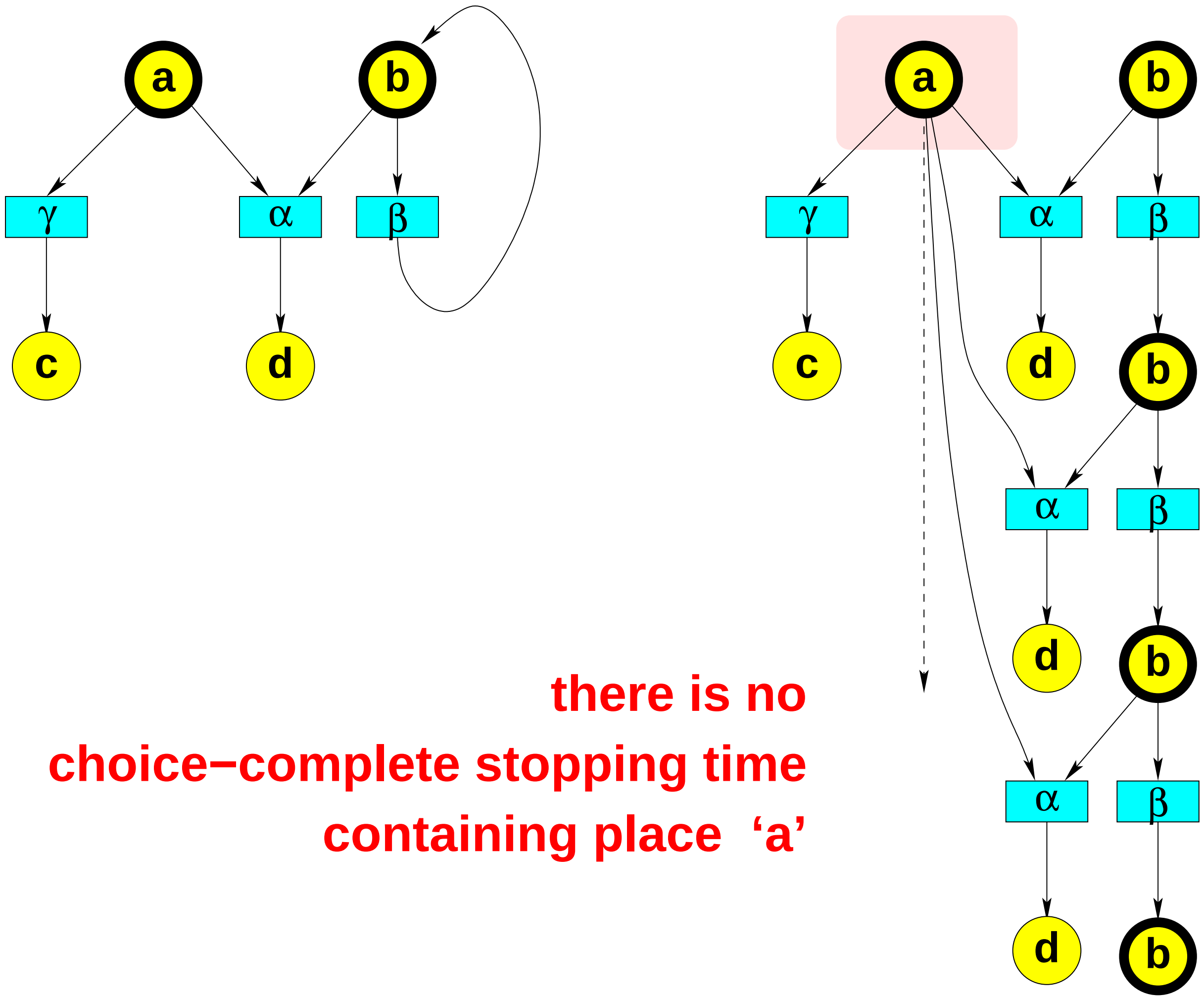
how to model their runs ?

Markov nets : *choice-complete* unfoldings



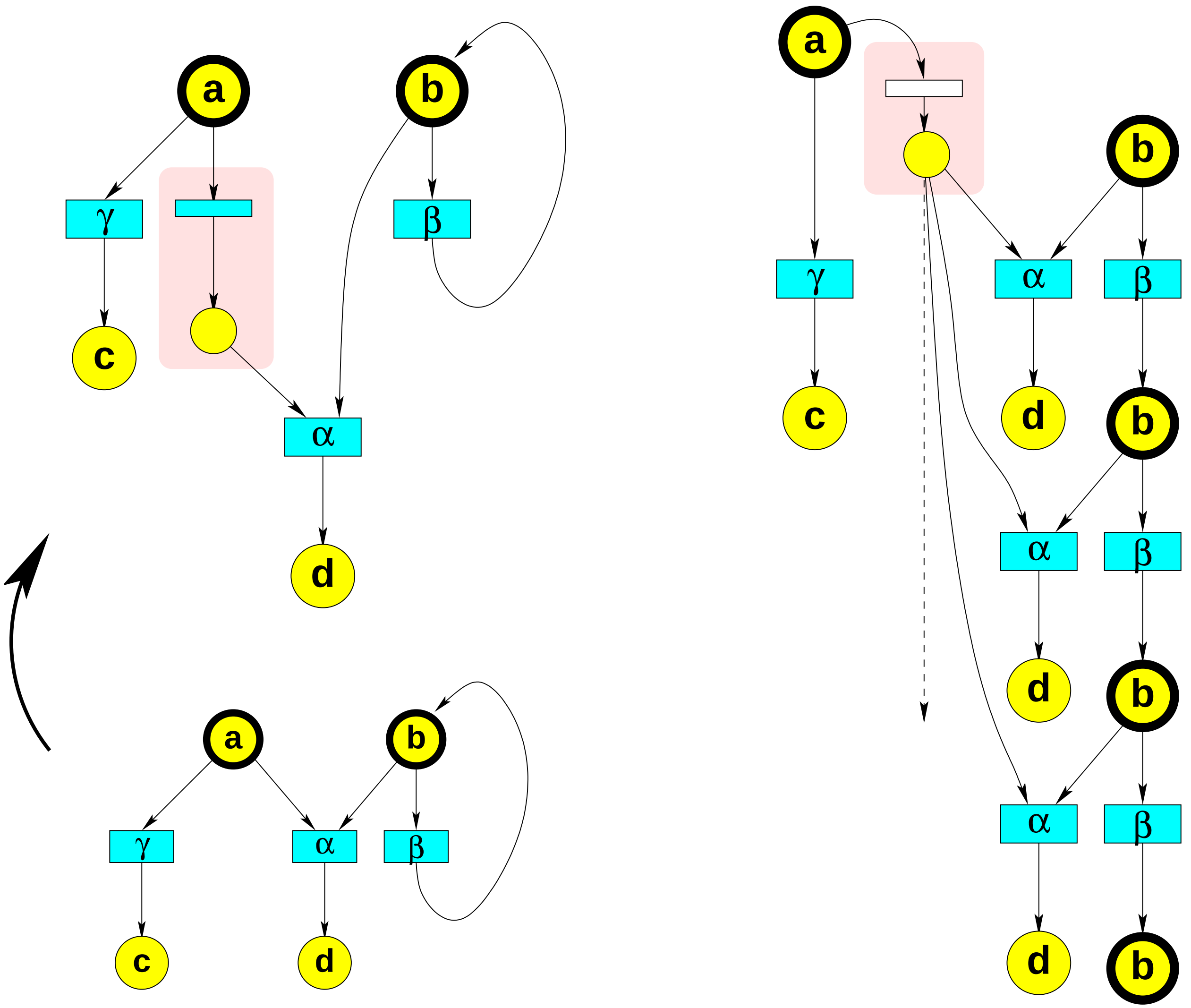
$$P^{\pi} \;=\; \prod_{\text{clusters}} P^{\pi}(\text{cluster})$$

randomizing Markov nets : can be problematic. . .



there is no
choice−complete stopping time
containing place ‘a’

. . . the solution : preselection



Markov nets : summary

Markov nets can be composed, their product is a straightforward extension of the product of Petri nets, in which the specified randomization is local to places exhibiting choice.

Time is partially ordered, stopping times are represented by unfoldings.

Kolmogorov extension theorem yields a randomized unfolding, its distribution randomizes partial orders, not interleavings. Independence matches concurrency.

Markov nets obey the strong Markov property.

distributed **ML state estimation :**

$\exists$ **prototype, application to telecom network diagnosis**

Markov net *distributed* **identification ?**