Probabilistic Methods in Concurrency

Lecture 7

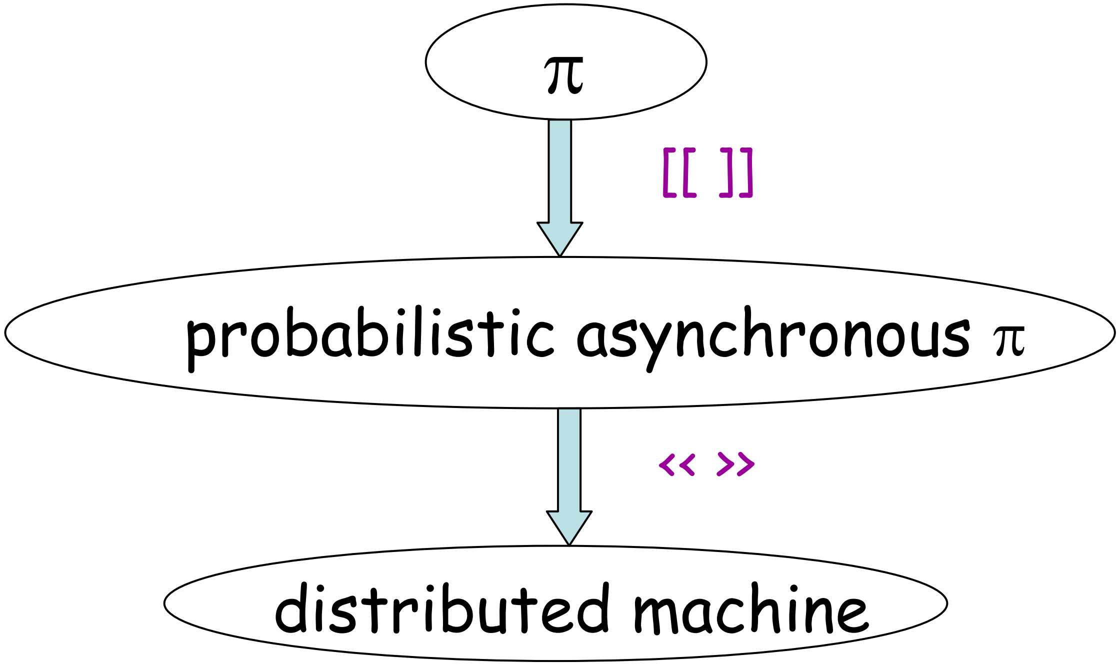
The probabilistic asynchronous π-calculus

Catuscia Palamidessi
catuscia@lix.polytechnique.fr
www.lix.polytechnique.fr/~catuscia

Page of the course:
www.lix.polytechnique.fr/~catuscia/teaching/Pisa/

The probabilistic asynchronous π-calculus

- Originally developed as an intermediate language for the fully distributed implementation of the π-calculus (Herescu and Palamidessi)
- The results of Lecture 4 show that a fully distributed implementation of π must necessarily be randomized
- *A two-steps approach:*



Advantages:
the correctness proof is easier since [[ ]] (which is the difficult part of the implementation) is between two similar languages

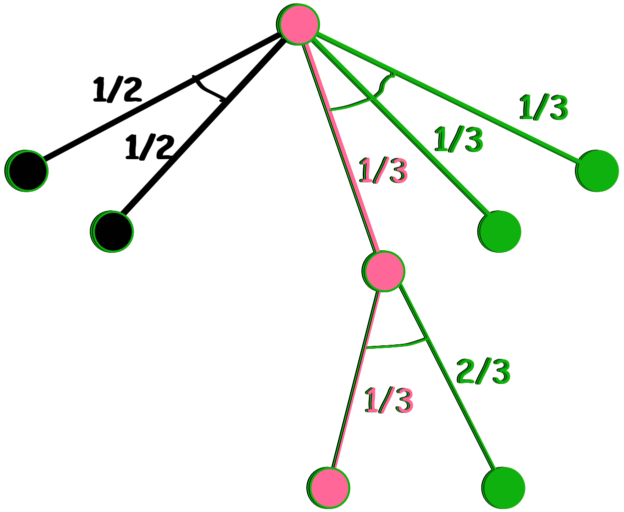
π_{pa}: the Probabilistic Asynchonous π

Syntax: **based on the asynchronous π of Amadio, Castellani, Sangiorgi**

g	$::=$	$x(y) \mid \tau$	prefixes
P	$::=$	$\Sigma_i\, p_i\, g_i\, .\, P_i$	pr. inp. guard. choice $\Sigma_i\, p_i = 1$
	$\mid$	$x \hat{} y$	output action
	$\mid$	$P \mid P$	parallel
	$\mid$	$(x)\, P$	new name
	$\mid$	$rec_A\, P$	recursion
	$\mid$	A	procedure name

The operational semantics of π_{pa}

- Based on the **Probabilistic Automata** of Segala and Lynch
  - Distinction between
    - nondeterministic behavior (choice of the scheduler) and
    - probabilistic behavior (choice of the process)



Scheduling Policy:

The scheduler chooses the group of transitions

Execution:

The process chooses probabilistically the transition within the group

The operational semantics of π_{pa}

- Representation of a group of transition

$$P \{ \text{--}g_i\text{->}_{p_i} P_i \}_i$$

- Rules

Choice $\Sigma_i\, p_i\, g_i \,.\, P_i \{\text{--}g_i\text{->}_{p_i} P_i \}_i$

Par $$\frac{P \{\text{--}g_i\text{->}_{p_i} P_i \}_i}{Q \mid P \{\text{--}g_i\text{->}_{p_i} Q \mid P_i \}_i}$$

The operational semantics of π_{pa}

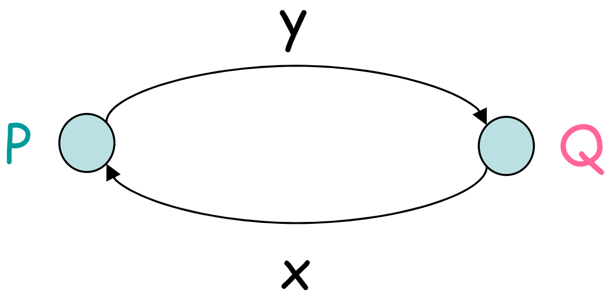
- Rules (continued)

Com
$$\frac{P\,\{\text{--}x_i(y_i)\text{->}_{p_i} P_i\,\}_i \qquad Q\,\{\text{--}x\hat{}z\text{->}_1\, Q'\,\}_i}{P \mid Q\ \ \{\text{--}\tau\text{->}_{p_i} P_i[z/y_i] \mid Q'\,\}_{xi=x} \cup \{\ \text{--}x_i(y_i)\text{->}_{p_i} P_i \mid Q\,\}_{xi=/=x}}$$

Res
$$\frac{P\,\{\text{--}x_i(y_i)\text{->}_{p_i} P_i\,\}_i}{(x)\ P\ \{\ \text{--}x_i(y_i)\text{->}_{q_i}\ (x)\ P_i\ \ \}_{xi\ =/=\ x}} \qquad q_i \text{ renormalized}$$

The expressive power of π

- Example of distributed agreement: the leader election problem



Implementation of π_{pa}

- **Compilation in Java** **<< >> : π_{pa} → Java**
  - **Distributed**
    « P | Q » = « P ».start(); « Q ».start();
  - **Compositional**
    « P op Q » = « P » jop « Q » for all op
  - Channels are one-position buffers with test-and-set (synchronized) methods for input and output
  - The probabilistic input guarded construct is implemented as a while loop in which channels to be tried are selected according to their probability. The loop repeats until an input is successful