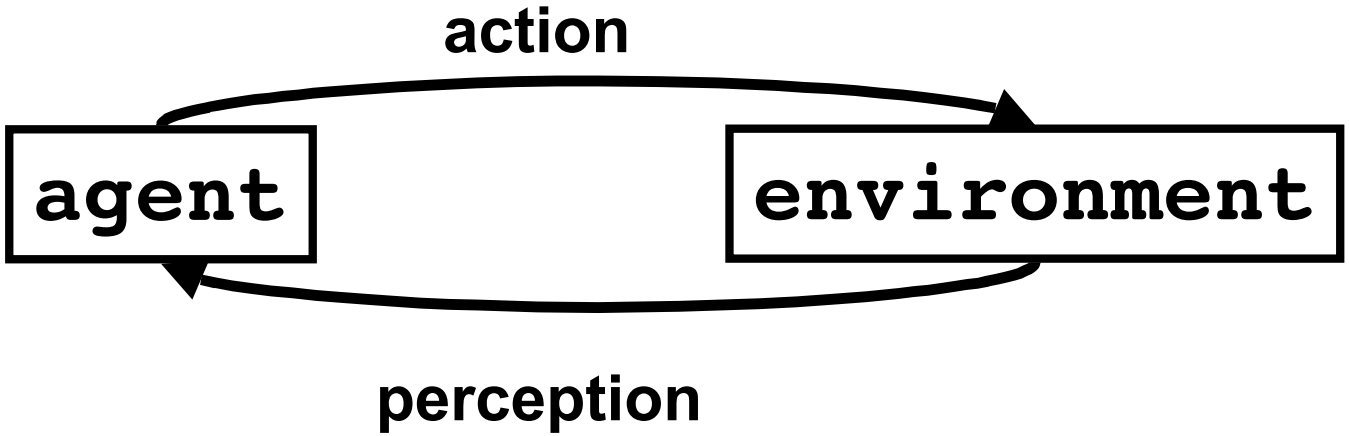
EARNING:
Architectural Framework for Evolutionary Computing & Reinforment LearNing Applied to Adaptive TradING

N. GUELFI, Prof., O. SIGAUD, Prof., Project co-directors
S. Landau, PhD & R. Razavi, PhD, Team members

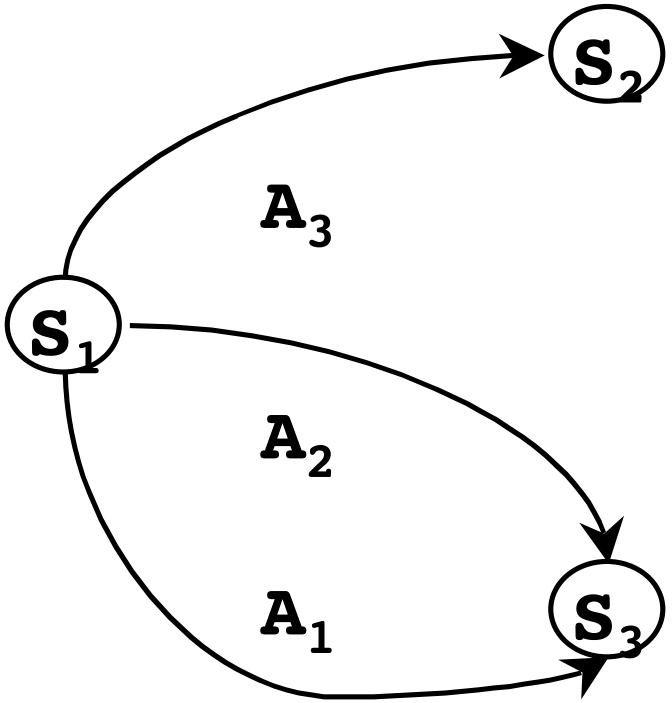
January 19, 2004

Objective



Learning to behave in an unknown environment

Markov Decision Process



- **States <S>**

$S_1, S_2, \ldots$

- **Actions <A>**

$A_1, A_2, \ldots$

- **Transitions T**

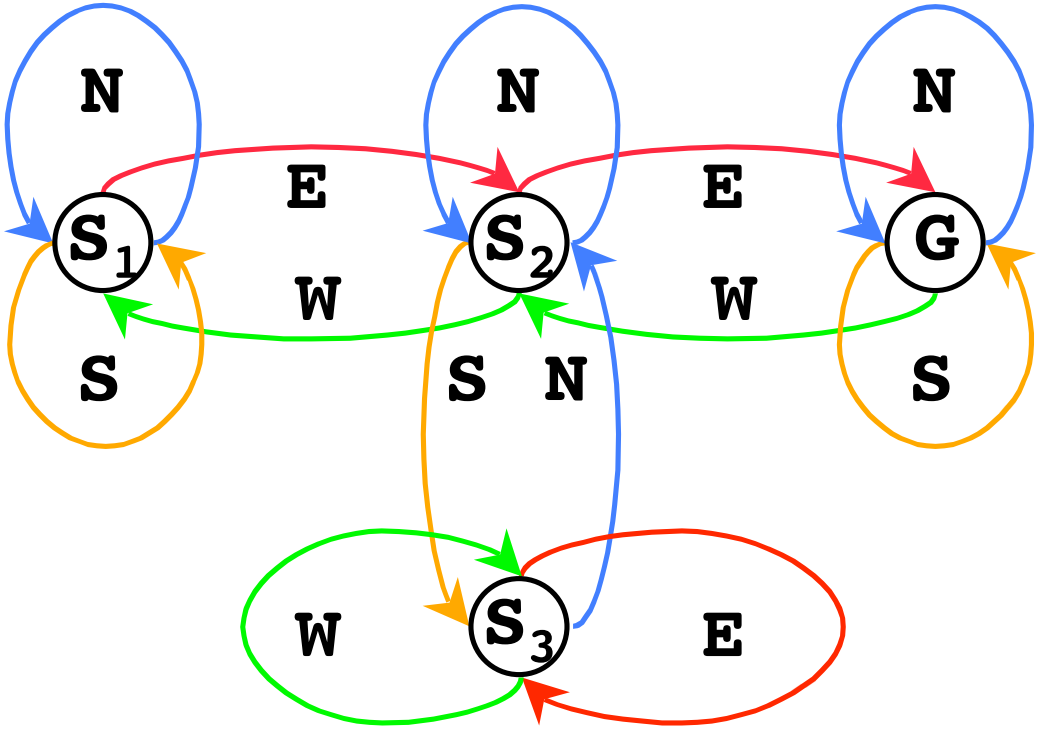
$T: SxAxS \rightarrow [0,1]$

- **Rewards R**

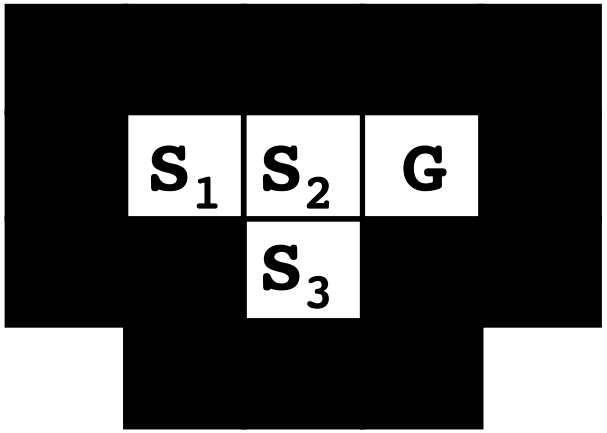
$R: SxA \rightarrow R$

$$\pi : S \rightarrow A$$

Example

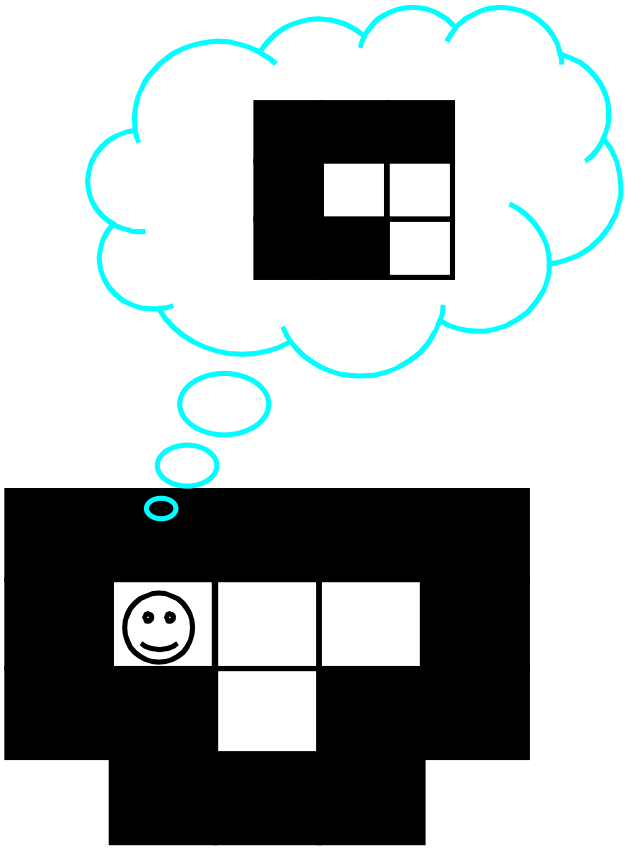


- <S> = {S_1,S_2, S_3, G}
- <A> = {N,S,E,W}
- T: 1 for all triplets represented by edges, `otherwise 0`
- R: (S_2,E) → `r>0,otherwise 0`



World model not known in advance

Observations during exploration



- Observations <O>
  $O_1, O_2, \ldots$
- Actions <A>
  $A_1, A_2, \ldots$
- Transitions T

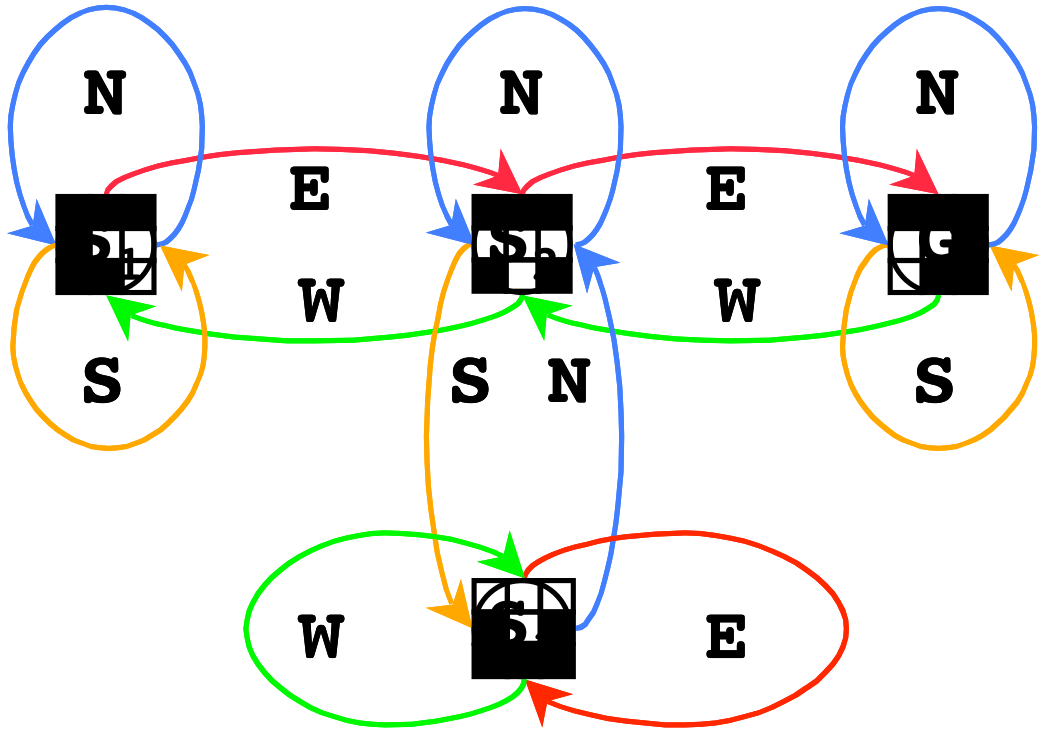
T: SxAxS → [0,1]

- Rewards R

R: SxA → R

$$\pi_O : O \rightarrow A$$

Markov environment



$$O \approx S \Rightarrow \pi_O \approx \pi$$

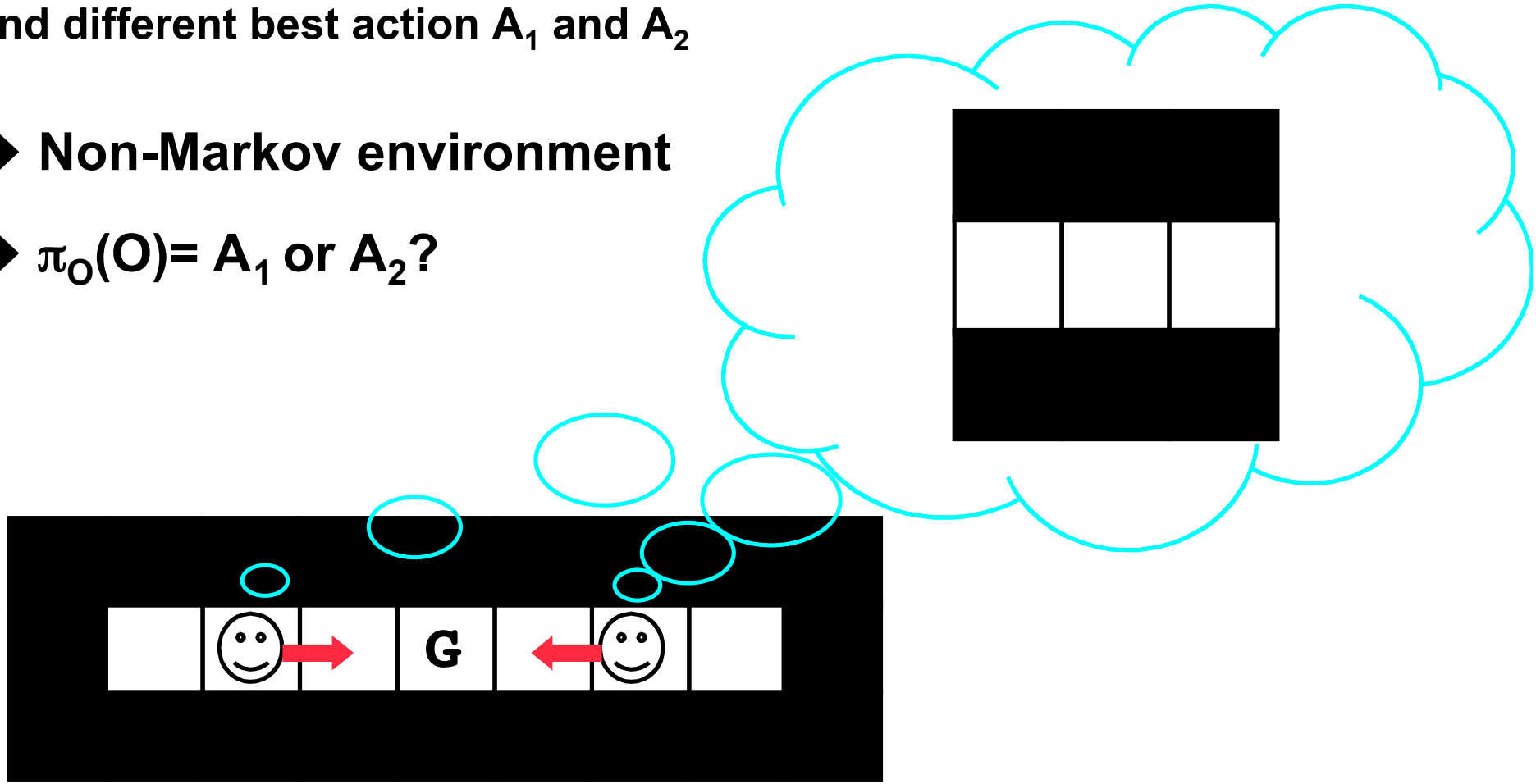
Partially Observable Markov Decision Process (POMDP)

one observation O, two states S_1 and S_2

and different best action A_1 and A_2

➡ Non-Markov environment

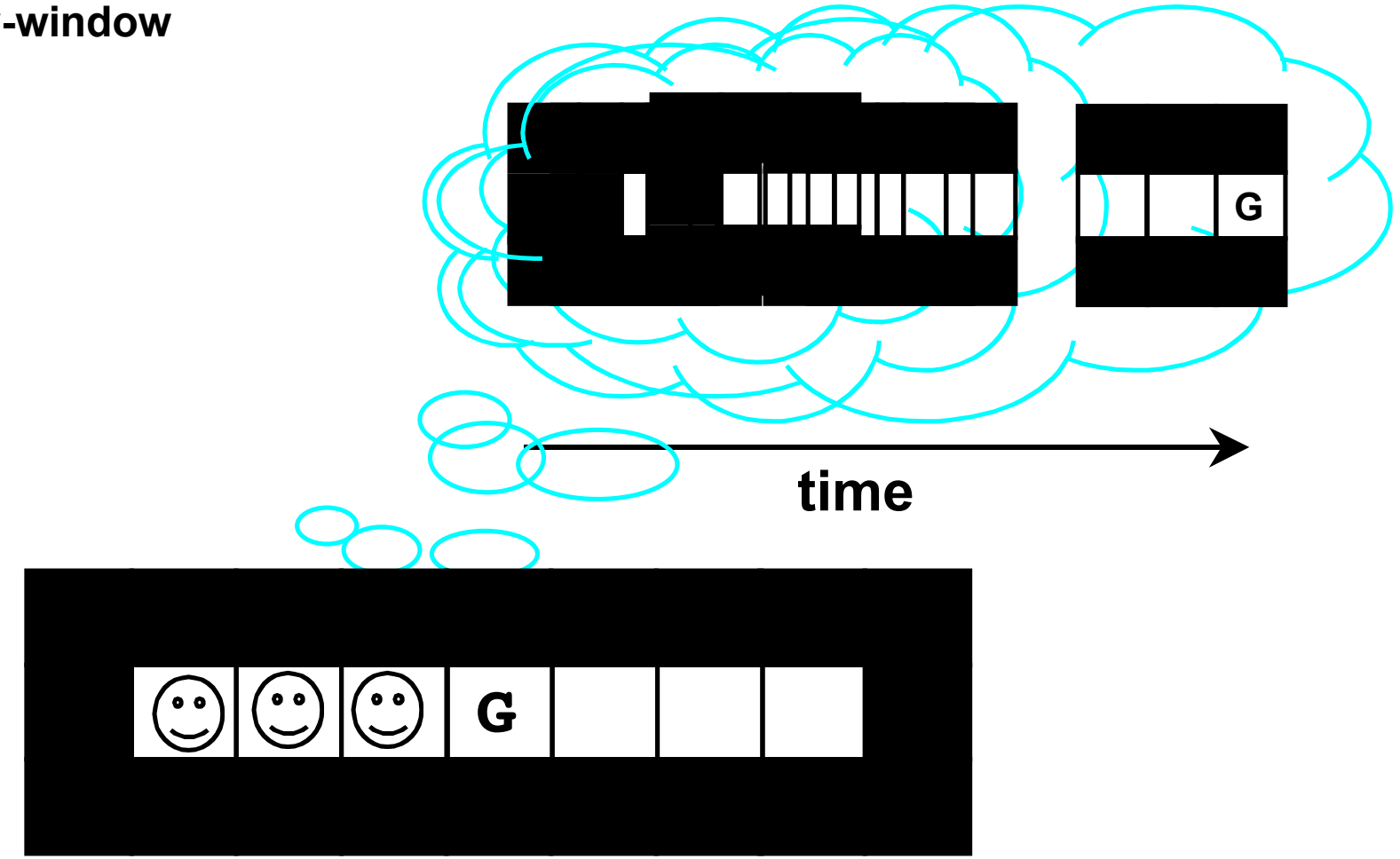
➡ $\pi_O(O) = A_1$ or A_2?



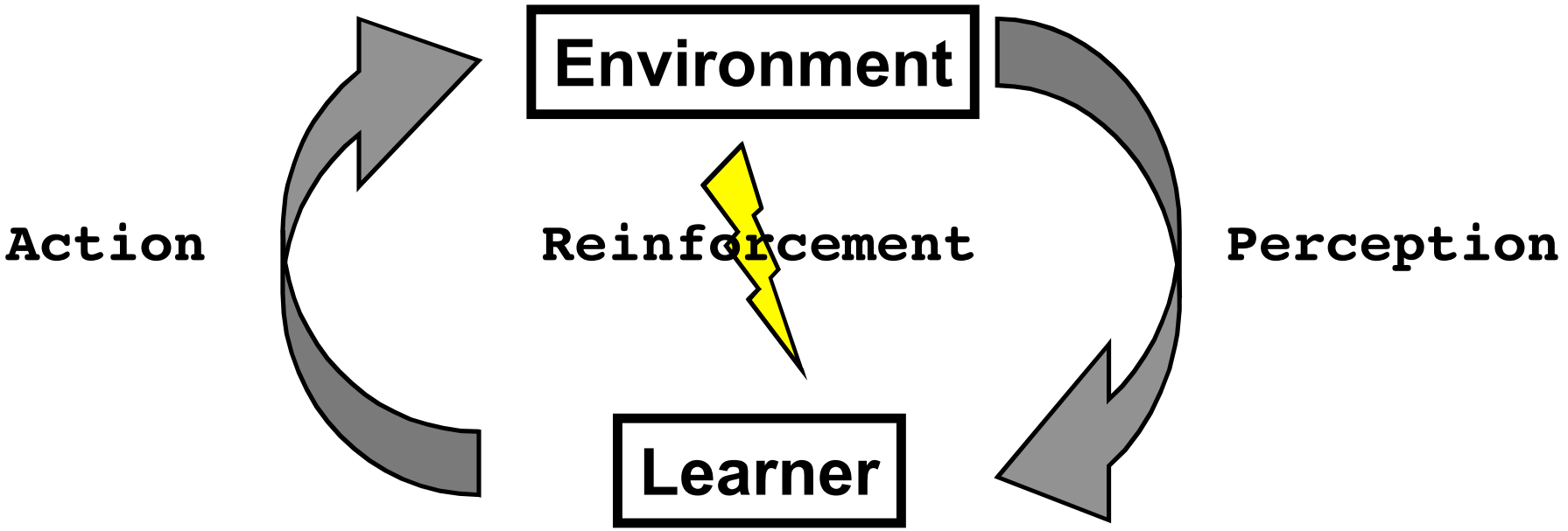
Disambiguation

Policies with internal states

e.g. history-window



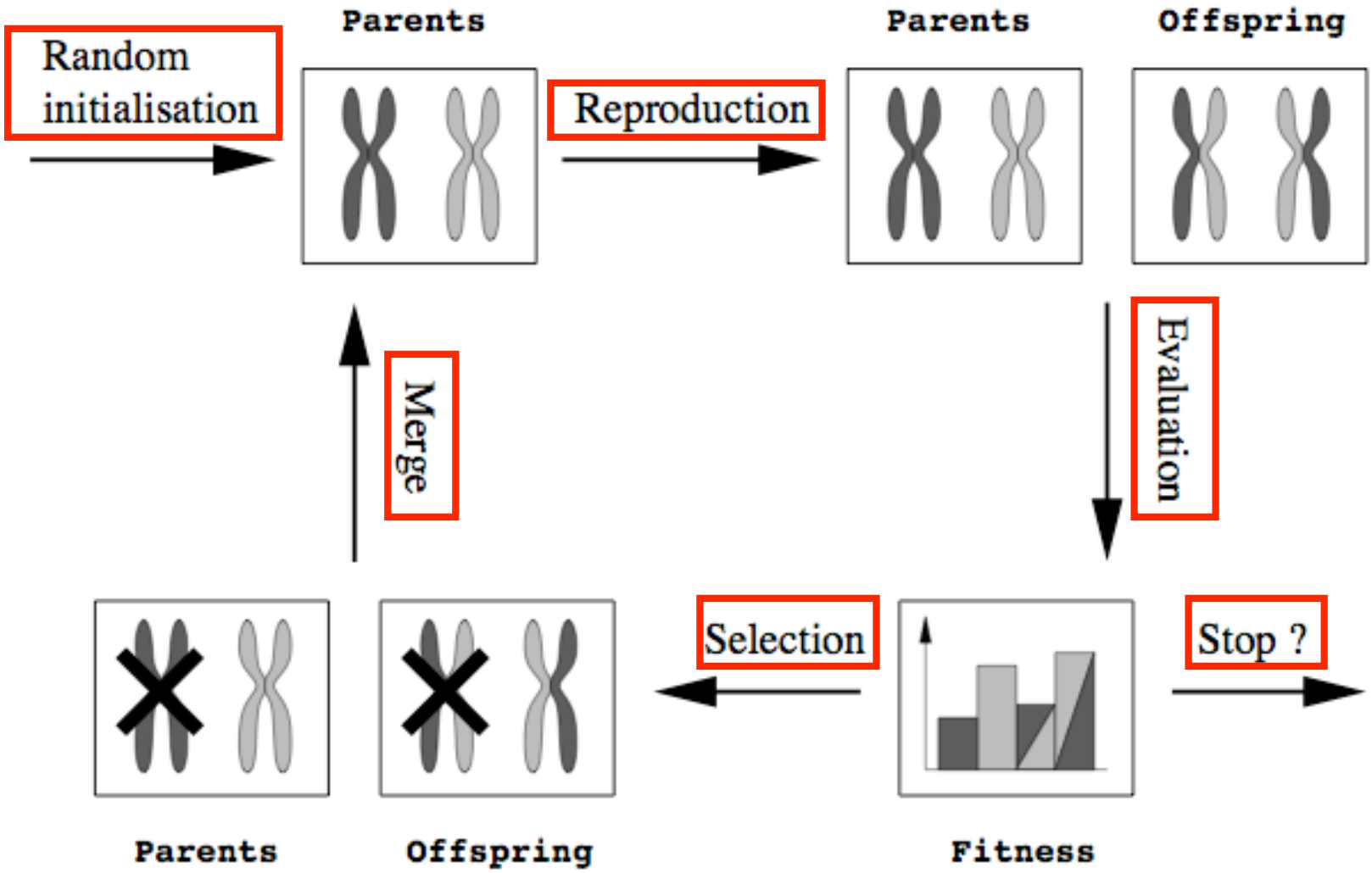
Addressing general MDP: State-of-the-art



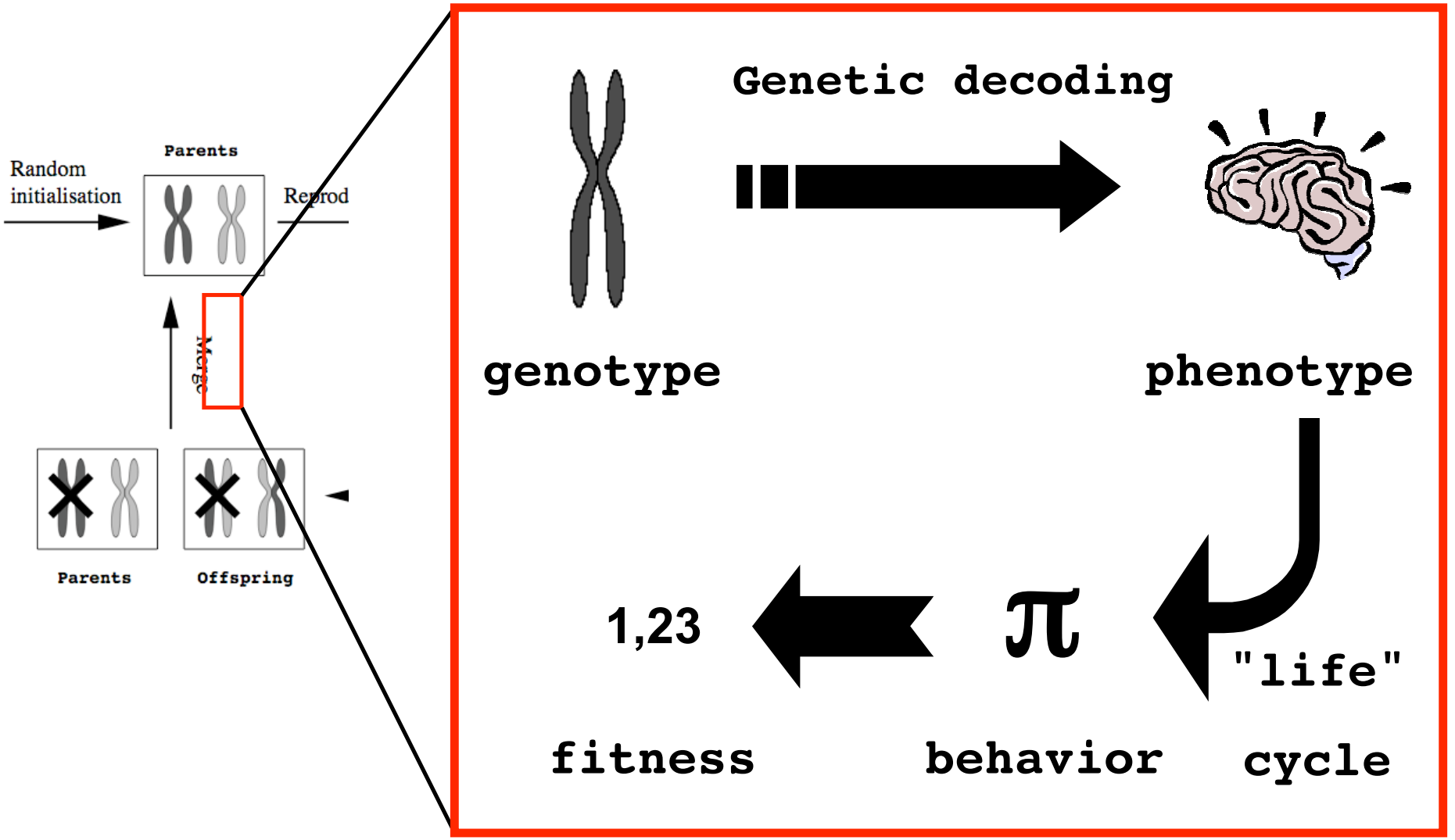
Solution techniques:

- **Evolutionary Algorithms**
- **Reinforcement Learning**

Evolutionary Algorithms



Application to the search of a policy

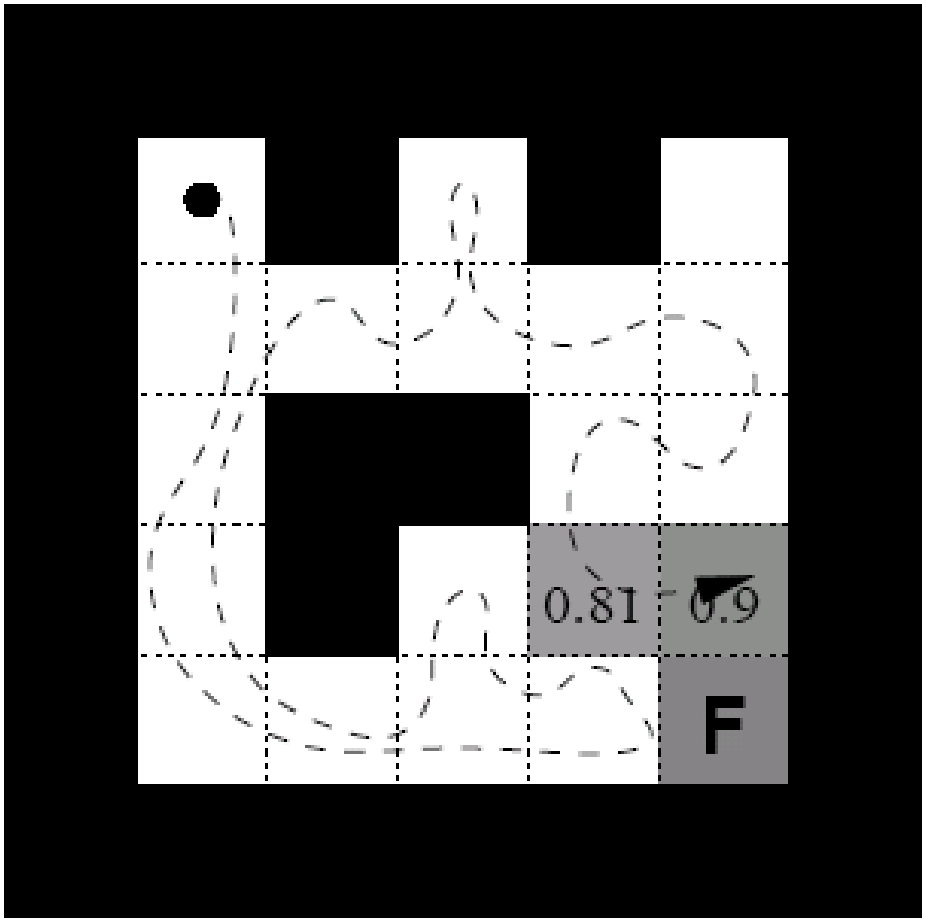


Reinforcement Learning

Q-Lear

$Q_{t+1}(s,a$

$\pi(s)=ar$



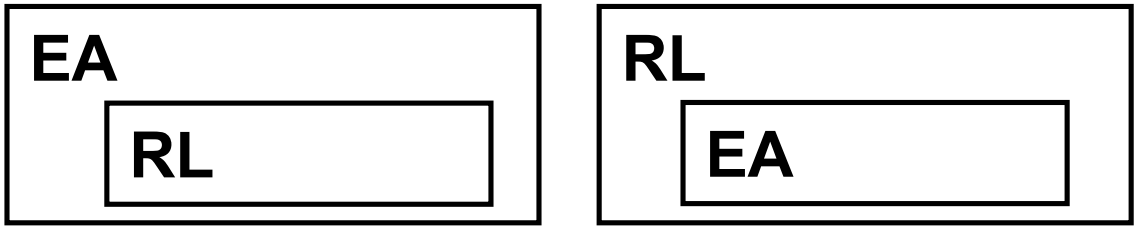
$\cdot Q_t(s,a)$

Summary of techniques properties

	Evolutionary Algorithms	Reinforcement Learning
Scalability for finding near-optimal solutions	Higher	Lower
CPU consumption (~number of samplings)	Higher	Lower
Memory consumption	Lower	Higher

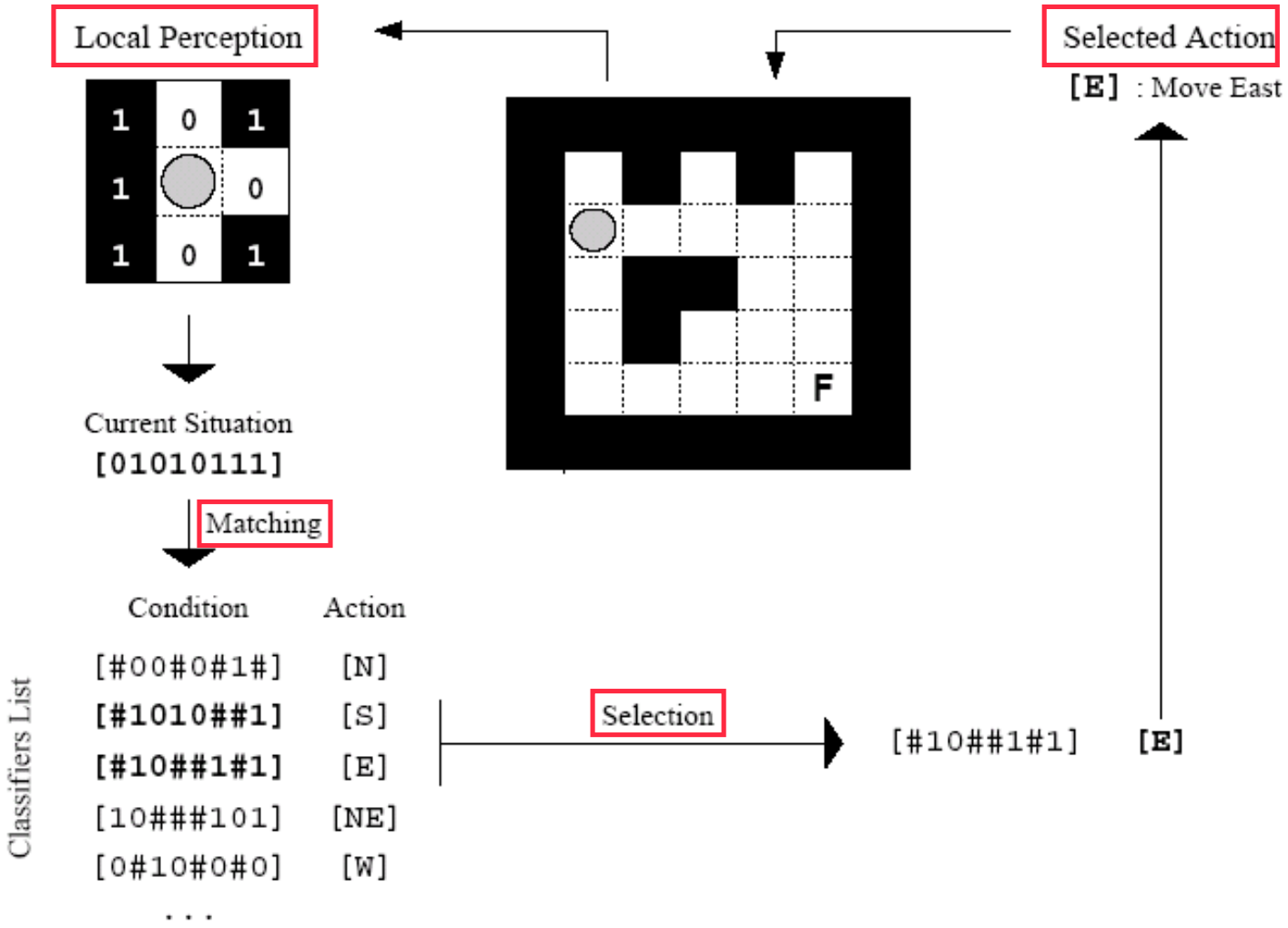
➔ **Towards hybrid solutions**

Hybrid solutions: accumulating shortcomes or advantages ?



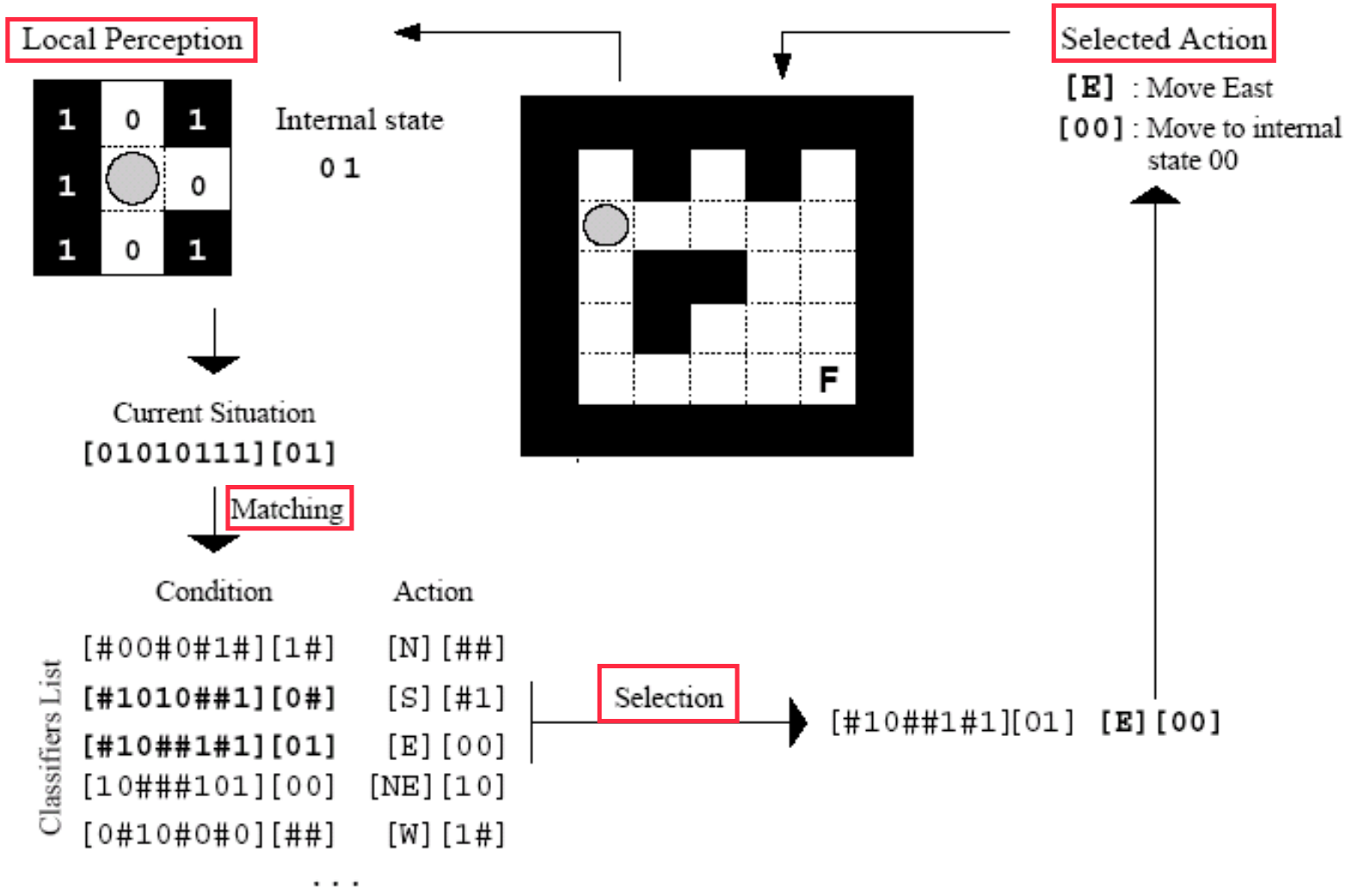
	EA/RL	RL/EA
Scalability for finding near-optimal solutions	Lower	Higher
CPU consumption (~number of samplings)	Higher	Lower
Memory consumption	Equivalent	Equivalent

RL/EA: Learning Classifier Systems (LCS)



LCS with additionnal internal states

e.g. XCSM

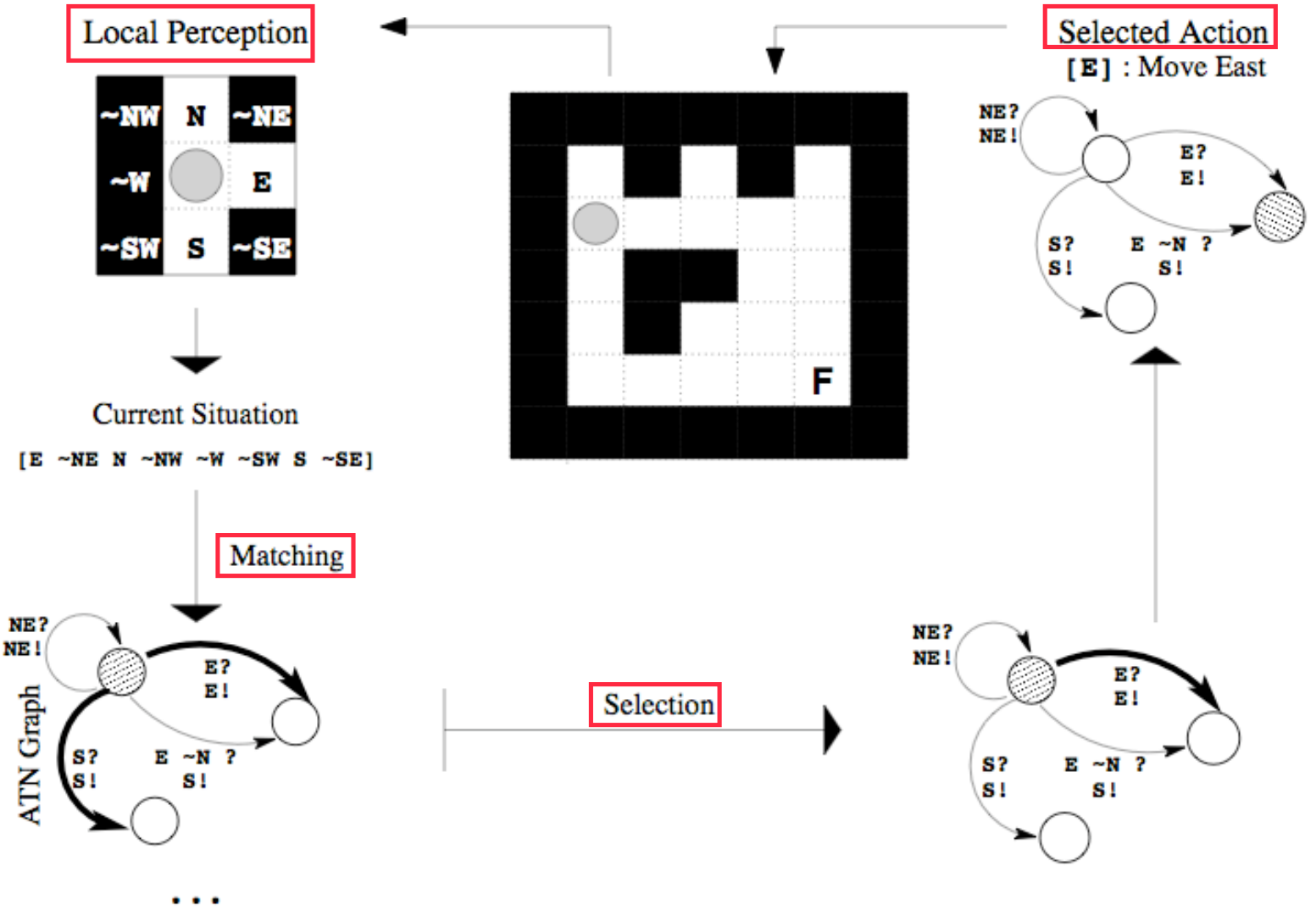


Summary: LCSs applied to POMDP

- mainly XCSM and ACS
- fixed number of internal states
- far from optimal

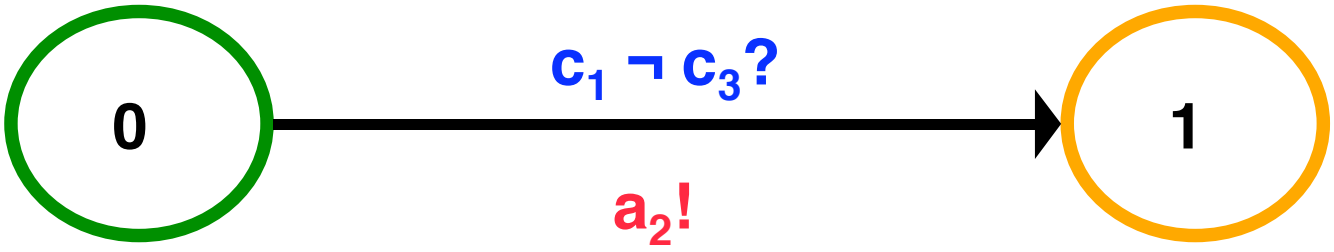
outperformed by a formally equivalent evolved executable structure

Augmented Transition Network (ATN)



ATN / LCS Equivalence

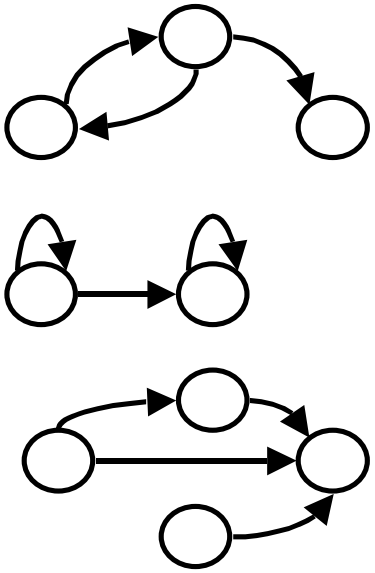
ATN transition



conditions			actions	
external		internal	external	internal
c_1	c_2 c_3	s_1	a_1 a_2	g_1
1	# 0	0	0 1	1

LCS rule

Proposal: new LCS extension



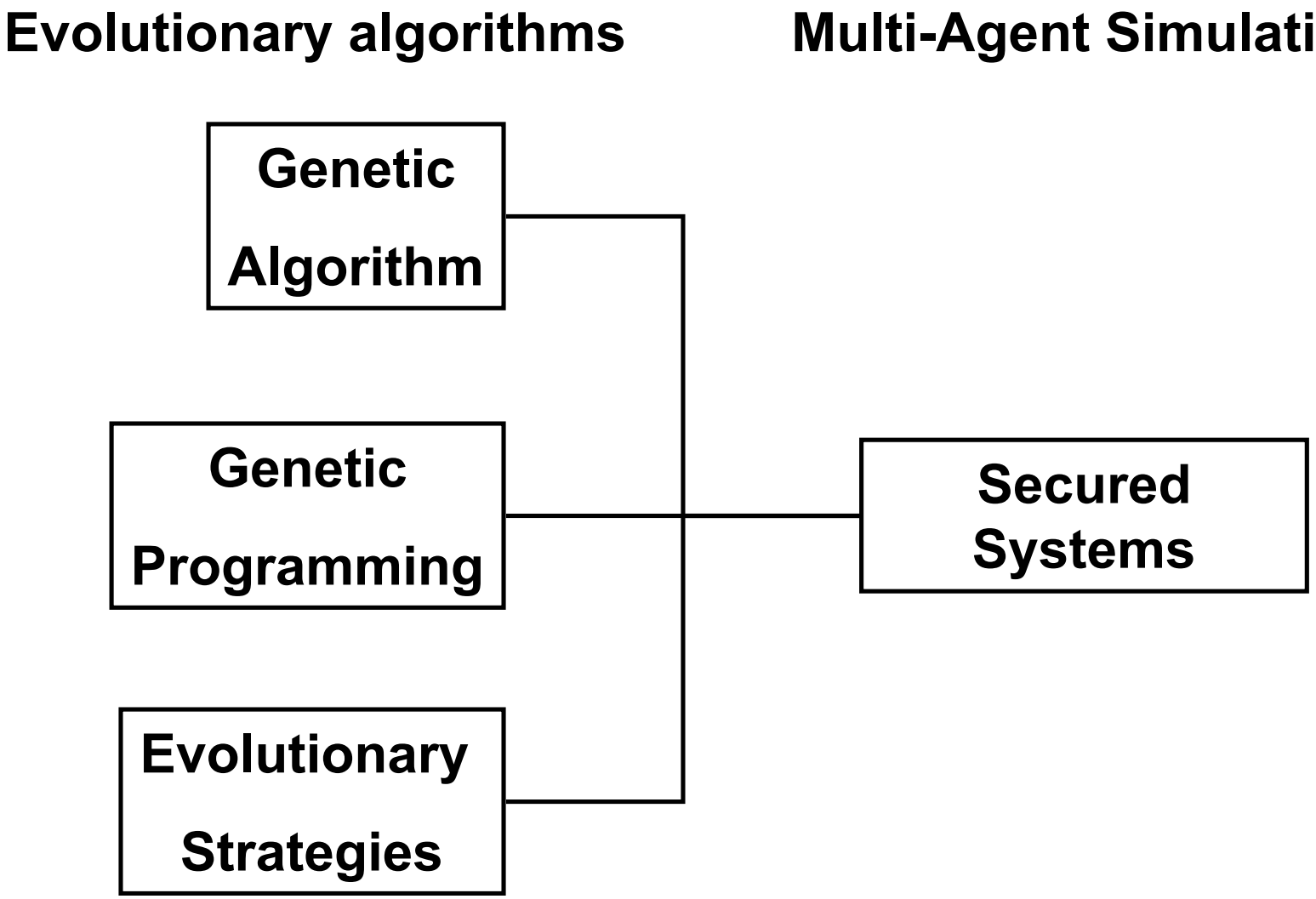
ATN List

- **an LCS**
- **number of internal states unspecified**
- **better support of POMDPs ?**

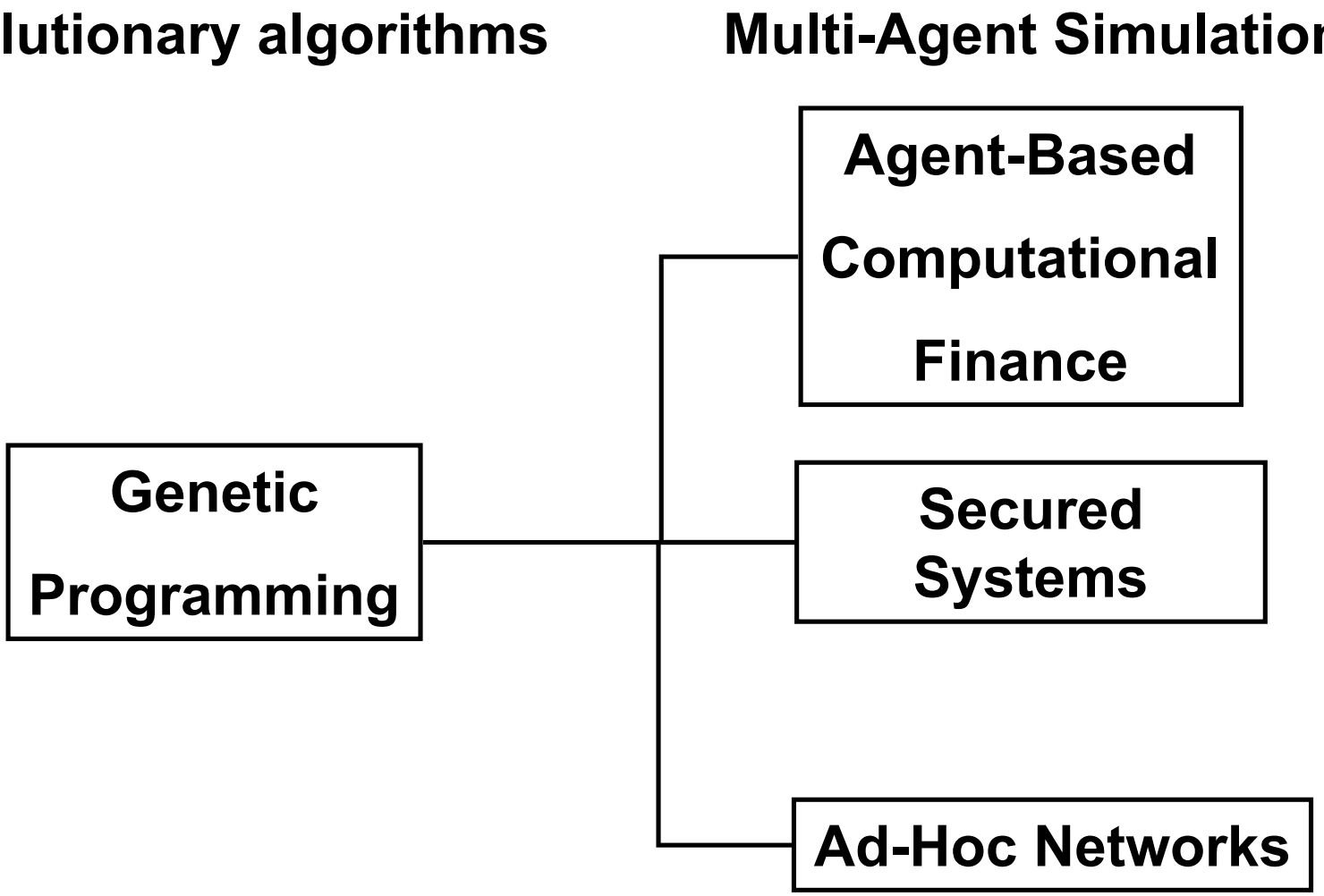
Implementation framework (1)

- Evolutionary algorithm and multi-agent simulation framework
  - abstraction of evolutionary techniques concepts
  - abstraction of the evolution of distributed systems
- Implemented in C++ on Linux/BSD
- Already used and reused

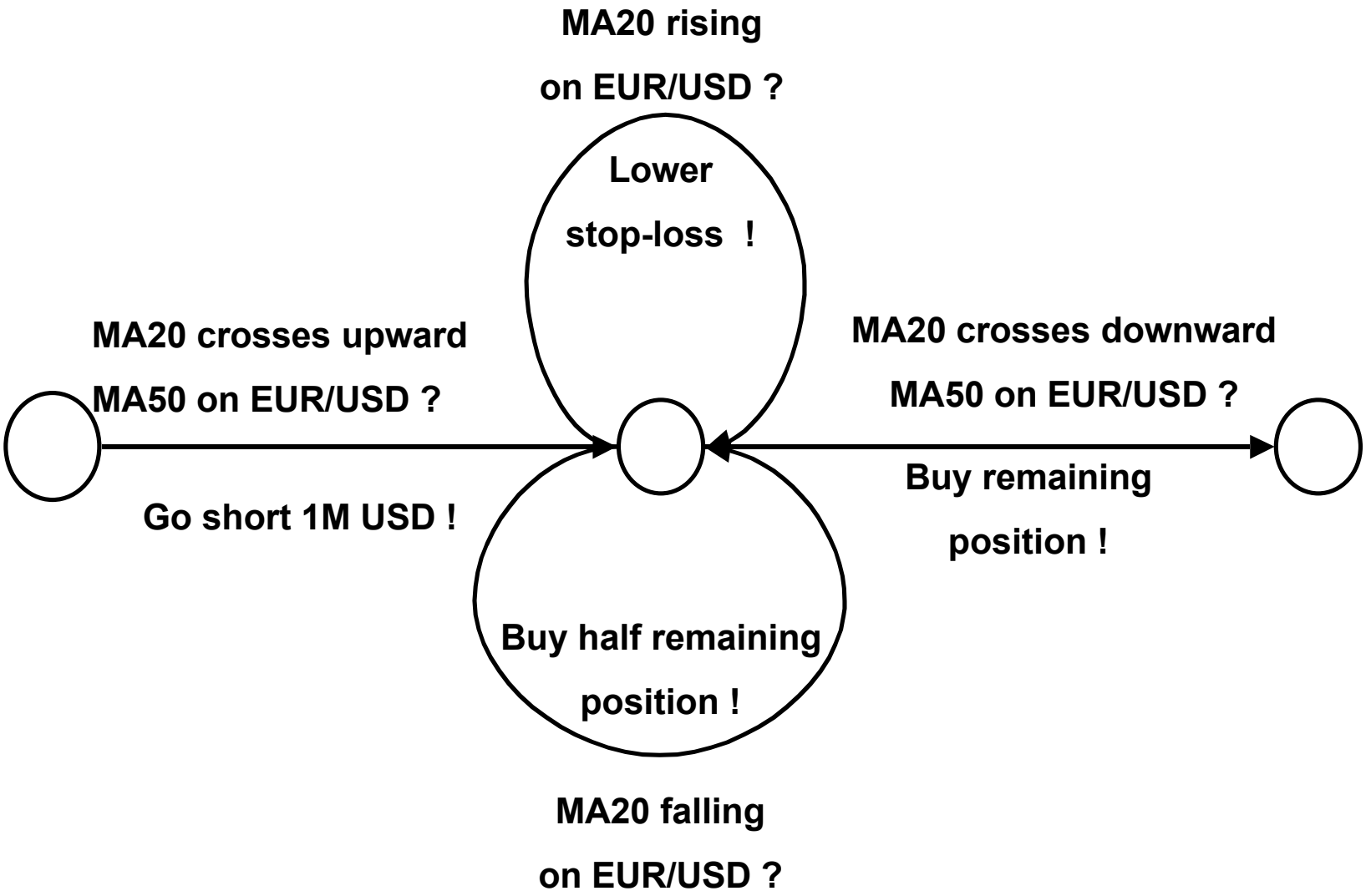
Implementation framework (2)



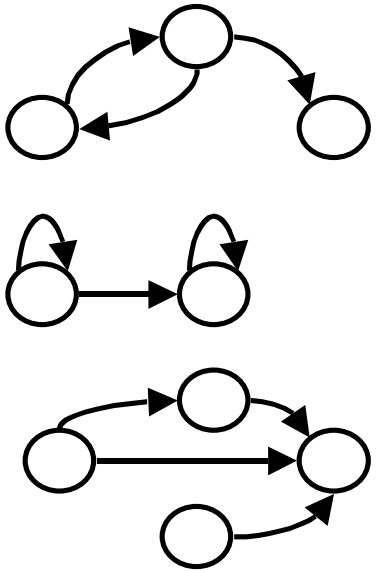
Implementation framework (3)



Application to the generation of trading plans (1)



Application to the generation of trading plans (2)



- length of plans not fixed in advance
- apprehensible representation

Competing trading plans

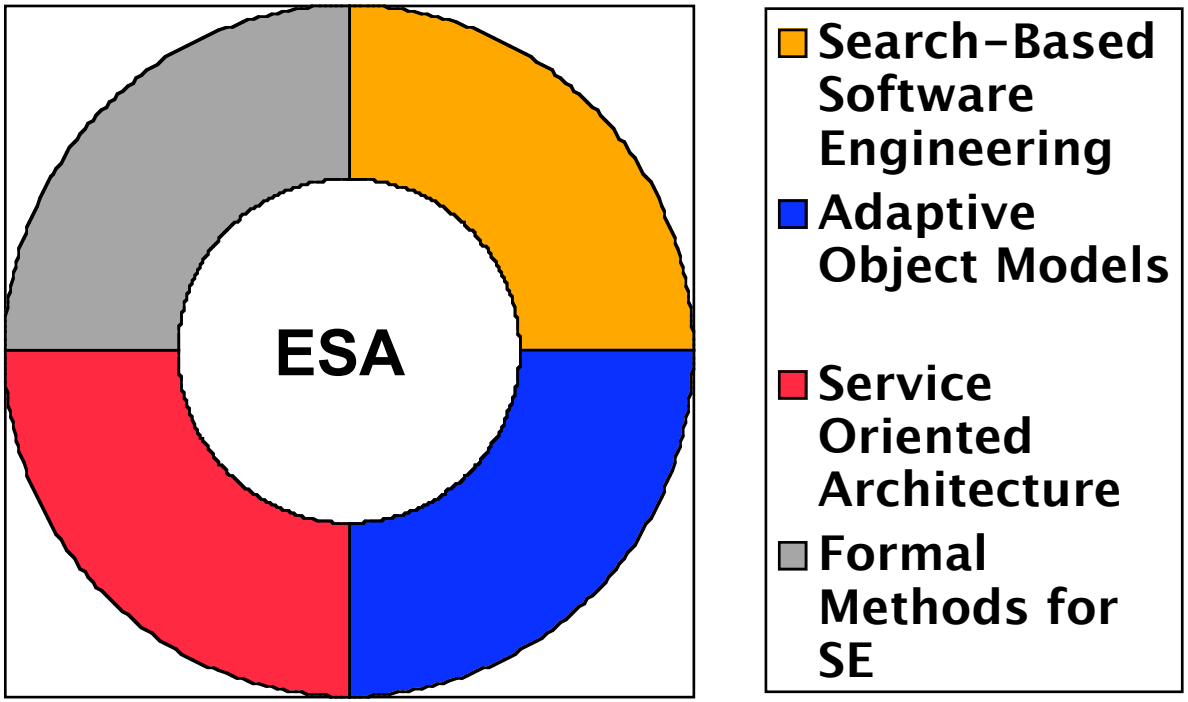
Conclusion

- **Insufficiency of the two main current families of techniques for general MDP**

- **Some hybrid solution already accumulates advantage sol hybrides, but can be improved**

- **Proposal of an extension to the best solution available with the hope to improve it**

Perspectives



ESA: architectural framework for Embedded, Secured, Adaptive systems