Algoritmy pre AI robotiku, IV. Diel

projekty, Learning Classifier Systems

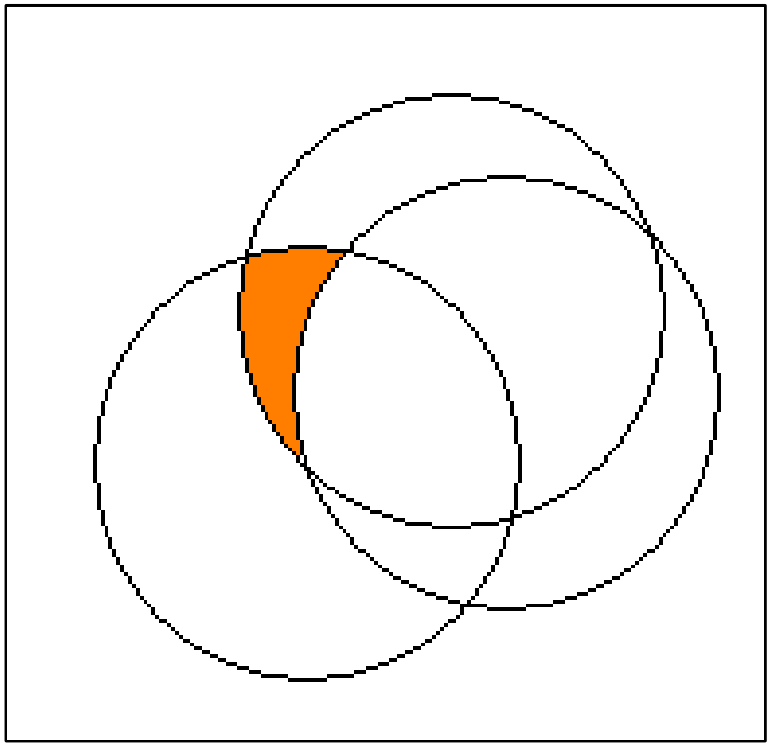
Pavel Petrovič
ppetrovic@acm.org

Projekty

- Visia na stránke predmetu,
  http://dai.fmph.uniba.sk/courses/airob/
  výber mailom

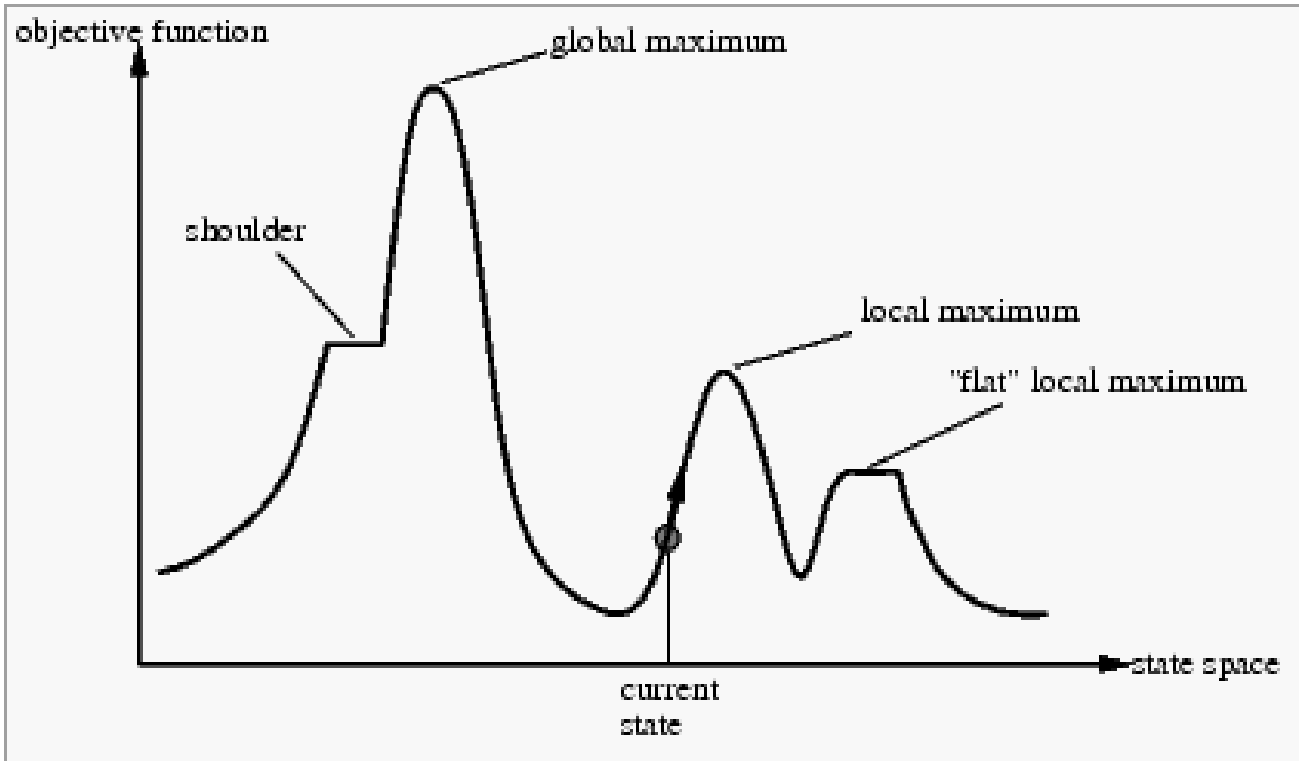
Stochastické metódy: Monte Carlo

- Zistiť obsah určitého tvaru:



Stochastické metódy: Simulované žíhanie

- Prehľadávací priestor skúmaný na základe lokálneho okolia:

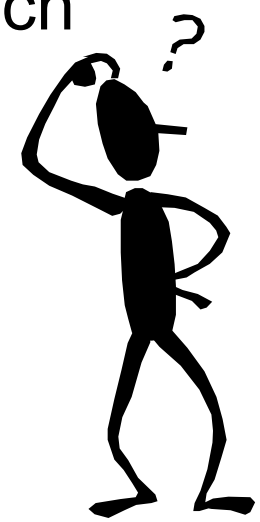


Princípy Prirodzenej Evolúcie

- Informácia o jedincoch je uložená v ich genotype, ktorý pozostáva z génov / aliel
- Úspešnejšie jedince majú väčšiu šancu prežiť a preto vyššiu pravdepodobnosť potomstva
- Populácia jedincov sa prispôsobuje meniacim sa podmienkam, keďže viac prispôsobené jedince budú prevládať v populácii
- Zmeny genotypu prichádzajú prostredníctvom mutácie a rekombinácie

Evolučné výpočty

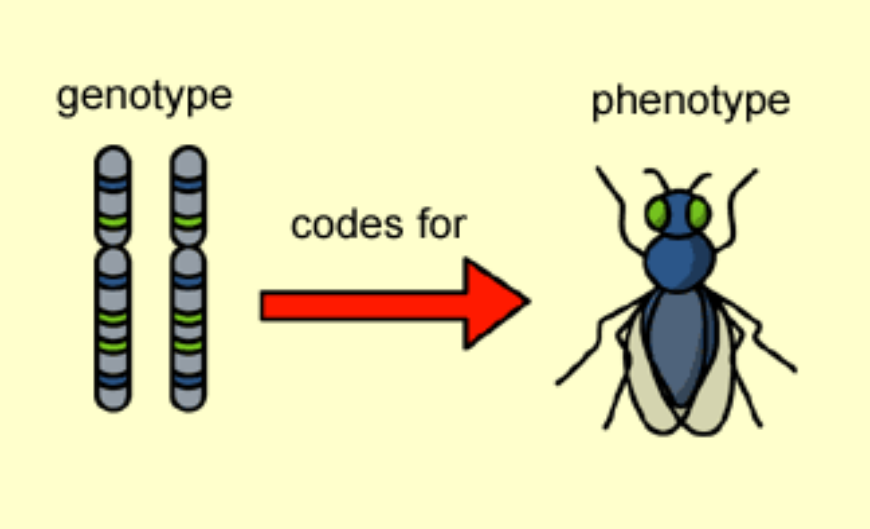
- Hľadáme riešenie nejakého problému
- Jednotné kódovanie riešení (reprezentácia)
- Fitness: účelová funkcia, ktorá číselne ohodnotí vhodnosť jedinca
- Populácia: množina náhodne vygenerovaných jedincov
- Princípy prirodzenej evolúcie:
  - výber, rekombinácia, mutácia
- Beží mnoho generácií



EA pojmy

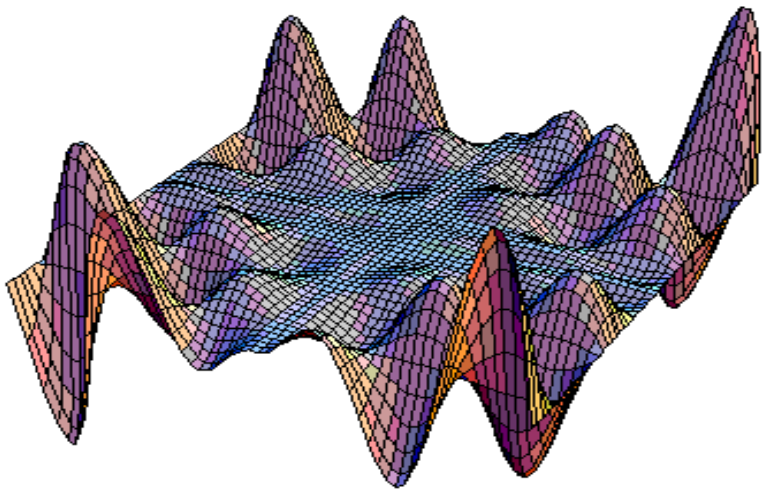
- genotype and phenotype
- fitness landscape
- diversity, genetic drift
- premature convergence
- exploration vs. exploitation
- selection methods: roulette wheel (fit.prop.), tournament, truncation, rank, elitist
- selection pressure
- direct vs. indirect representations
- fitness space

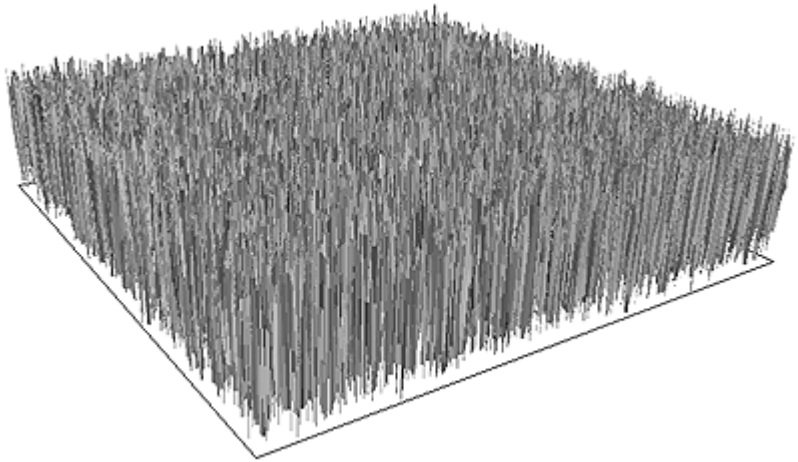
Genotype and Phenotype



- *Genotype* – all genetic material of a particular individual (genes)
- *Phenotype* – the real features of that individual

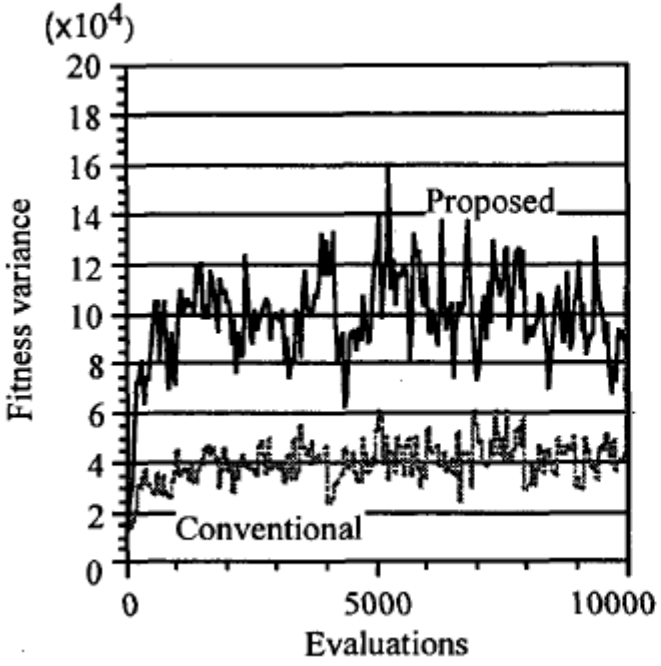
Fitness landscape

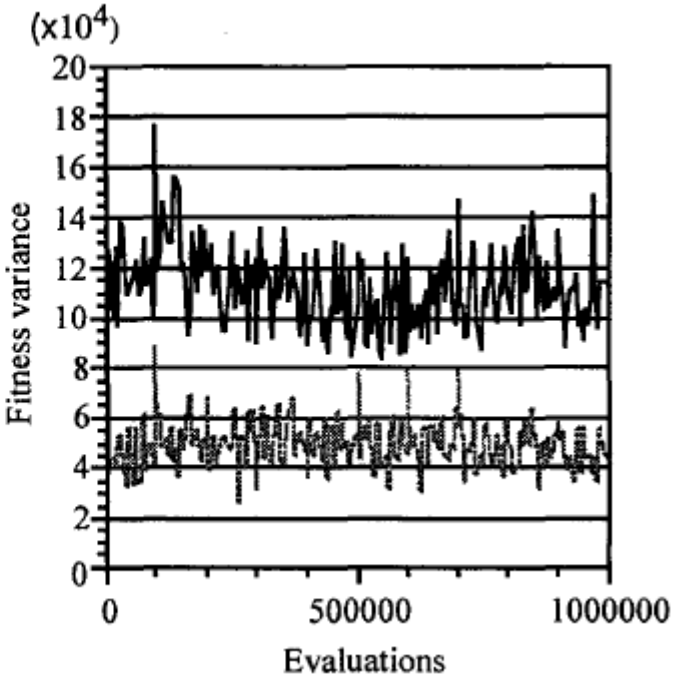




- Genotype space – difficulty of the problem – shape of fitness landscape, neighborhood function

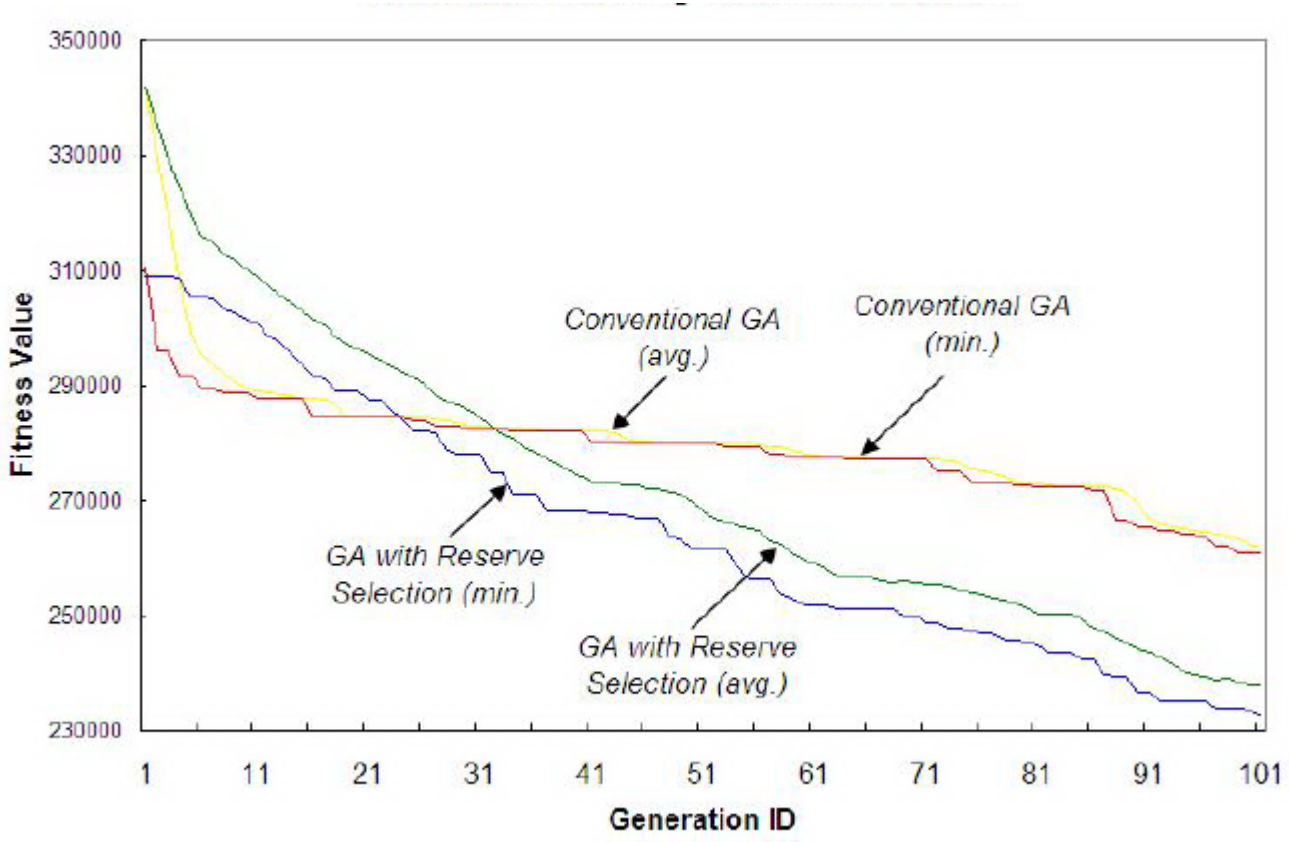
Population diversity





- Must be kept high for the evolution to advance

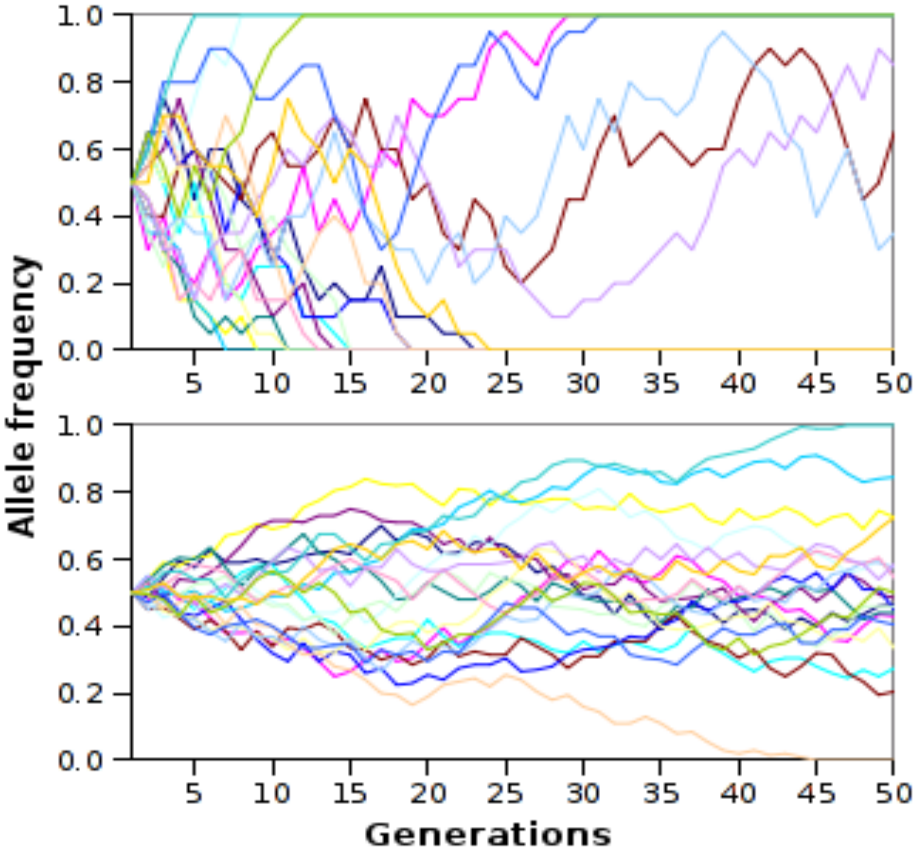
Premature convergence



- important building blocks are lost early in the evolutionary run

Genetic drift

- Loosing the population distribution due to the sampling error



Exploration vs. Exploitation

- Exploration phase: localize promising areas
- Exploitation phase: fine-tune the solution

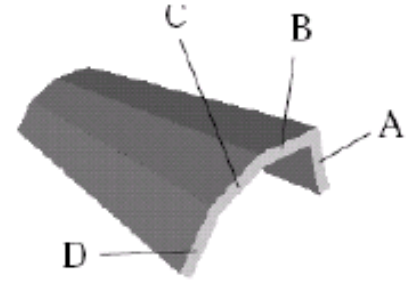
Selection methods

- roulette wheel (fitness proportionate selection),
- tournament selection
- truncation selection
- rank selection
- elitist strategies

Selection pressure

- Influenced by the problem
- Relates to evolutionary operators

Direct vs. Indirect Representations

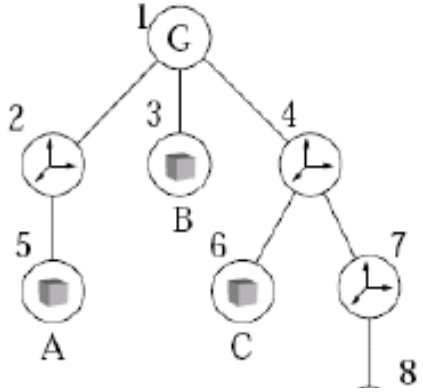


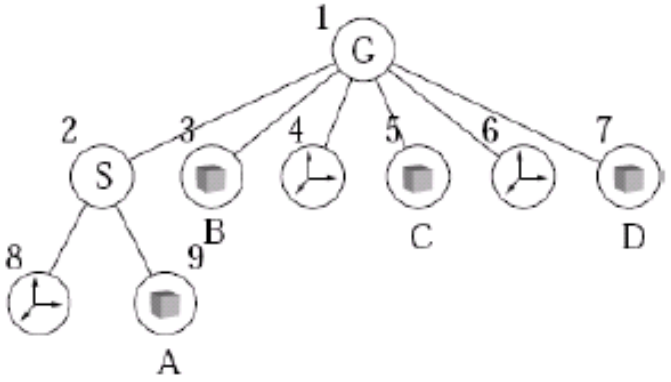
G tree root

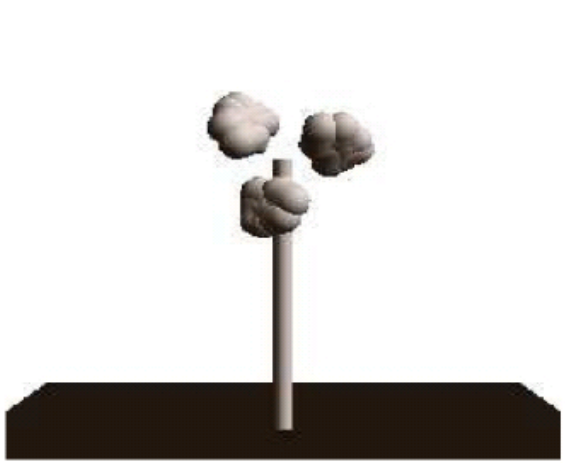
S separator node

transformation node

leaf - prism







fitness=0.105

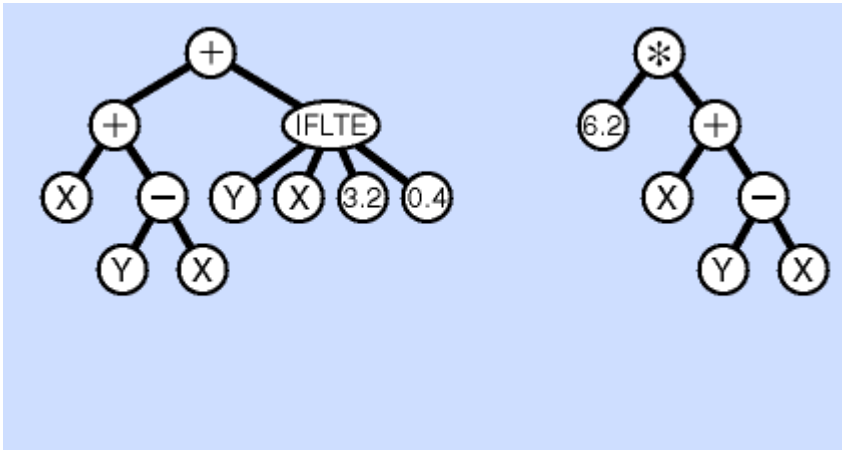
fitness=0.652

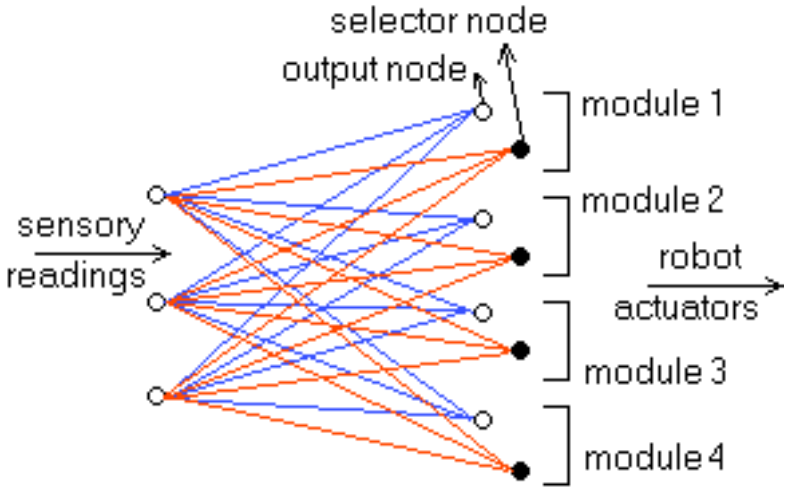
Fitness Space (Floreano)

- Functional vs. behavioral
- Explicit vs. implicit
- External vs. internal

Evolutionary Robotics

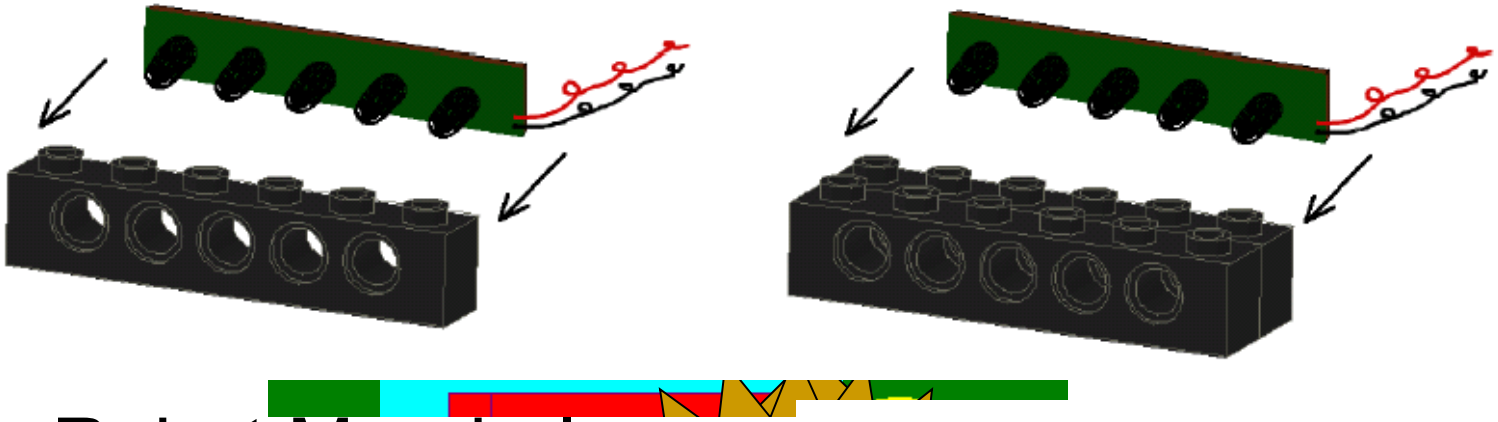
- Solution: Robot's controller



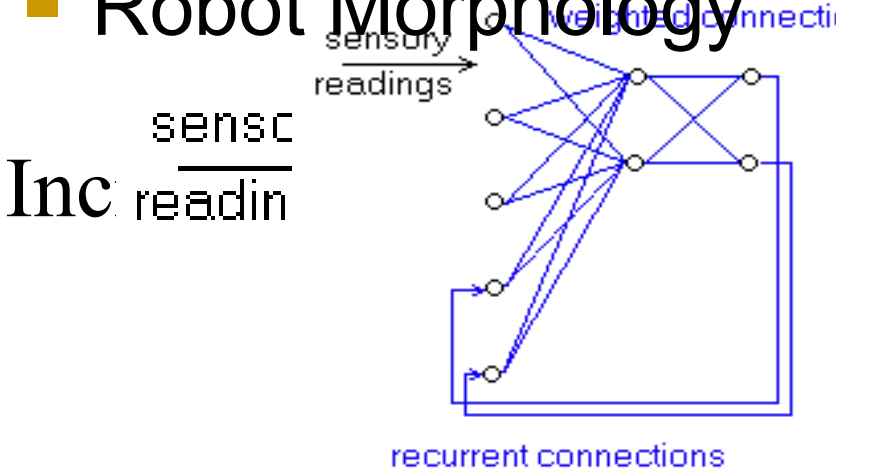


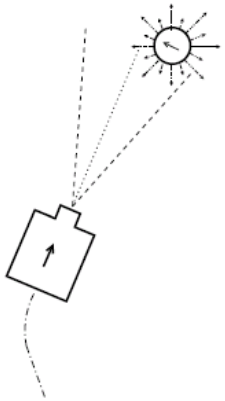
- Fitness: how well the robot performs
- Simulation or real robot

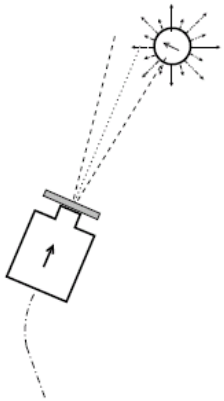
Fitness Influenced by



- Robot Morphology







Evolvable Tasks

- Wall following
- Obstacle avoidance
- Docking and recharging
- Artificial ant following
- Box pushing
- Lawn mowing
- Legged walking
- T-maze navigation
- Foraging strategies
- Trash collection
- Vision discrimination and classification tasks
- Target tracking and navigation
- Pursuit-evasion behaviors
- Soccer playing
- Navigation tasks

MDP

2.2.1 Markov Decision Processes A MDP is defined as the collection of the following elements:

- a finite set S of discrete states s of an agent ;
- a finite set A of discrete actions a ;
- a transition function $P : S \times A \rightarrow \Pi(S)$ where $\Pi(S)$ is the set of probability distributions over S. A particular probability distribution $Pr(s_{t+1}|s_t, a_t)$ indicates the probabilities that the agent reaches the different s_{t+1} possible states when he performs action a_t in state s_t ;
- a reward function $R : S \times A \rightarrow \mathbb{R}$ which gives for each (s_t, a_t) pair the scalar reward signal that the agent receives when he performs action a_t in state s_t .

The MDP formalism describes the stochastic structure of a problem faced by an agent, and does not tell anything about the behavior of this agent in its environment. It only tells what, depending on its current state and action, will be its future situation and reward.

MDP

The above definition of the transition function implies a specific assumption about the nature of the state of the agent. This assumption, known as the *Markov property*, stipulates that the probability distribution specifying the s_{t+1} state only depends on s_t and a_t, but not on the past of the agent. Thus $P(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_0, a_0)$. This means that, when the Markov property holds, a knowledge of the past of the agent does not bring any further information on its next state.

The behavior of the agent is described by a policy π giving for each state the probability distribution of the choice of all possible actions.

MDP

In order to define the accumulated reward, we introduce the *discount factor* $\gamma \in [0, 1]$. This factor defines how much the future rewards are taken into account in the computation of the accumulated reward at time t as follows:

$$Rc_\pi(t) = \sum_{k=t}^{T_{max}} \gamma^{(k-t)} r_\pi(k)$$

where T_{max} can be finite or infinite and $r_\pi(k)$ represents the immediate reward received at time k if the agent follows policy π.

DP methods introduce a *value function* V^π where $V^\pi(s)$ represents for each state s the accumulated reward that the agent can expect if it follows policy π from state s. If the Markov property holds, V^π is solution of the Bellman equation (Bertsekas, 1995):

$$\forall s \in S, V^\pi(s) = \sum_a \pi(s_t, a_t)[R(s_t, a_t) + \gamma \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) V^\pi(s_{t+1})] \quad (1)$$

MDP

Rather than the value function V^π, it is often useful to introduce an action-value function Q^π where $Q^\pi(s, a)$ represents the accumulated reward that the agent can expect if it follows policy π after having done action a in state s. Everything that was said of V^π directly applies to Q^π, given that $V^\pi(s) = \max_a Q^\pi(s, a)$. The corresponding optimal functions are independent of the policy of the agent; they are denoted V^* and Q^*.

MDP

Let us consider the case of the average immediate reward. Its exact value after k iterations is

$$E_k(s) = (r_1 + r_2 + \cdots + r_k)/k$$

Furthermore,

$$E_{k+1}(s) = (r_1 + r_2 + \cdots + r_k + r_{k+1})/(k+1)$$

thus

$$E_{k+1}(s) = k/(k+1)E_k(s) + r_{k+1}/(k+1)$$

which can be rewritten:

$$E_{k+1}(s) = (k+1)/(k+1)E_k(s) - E_k(s)/(k+1) + r_{k+1}/(k+1)$$

or

$$E_{k+1}(s) = E_k(s) + 1/(k+1)[r_{k+1} - E_k(s)]$$

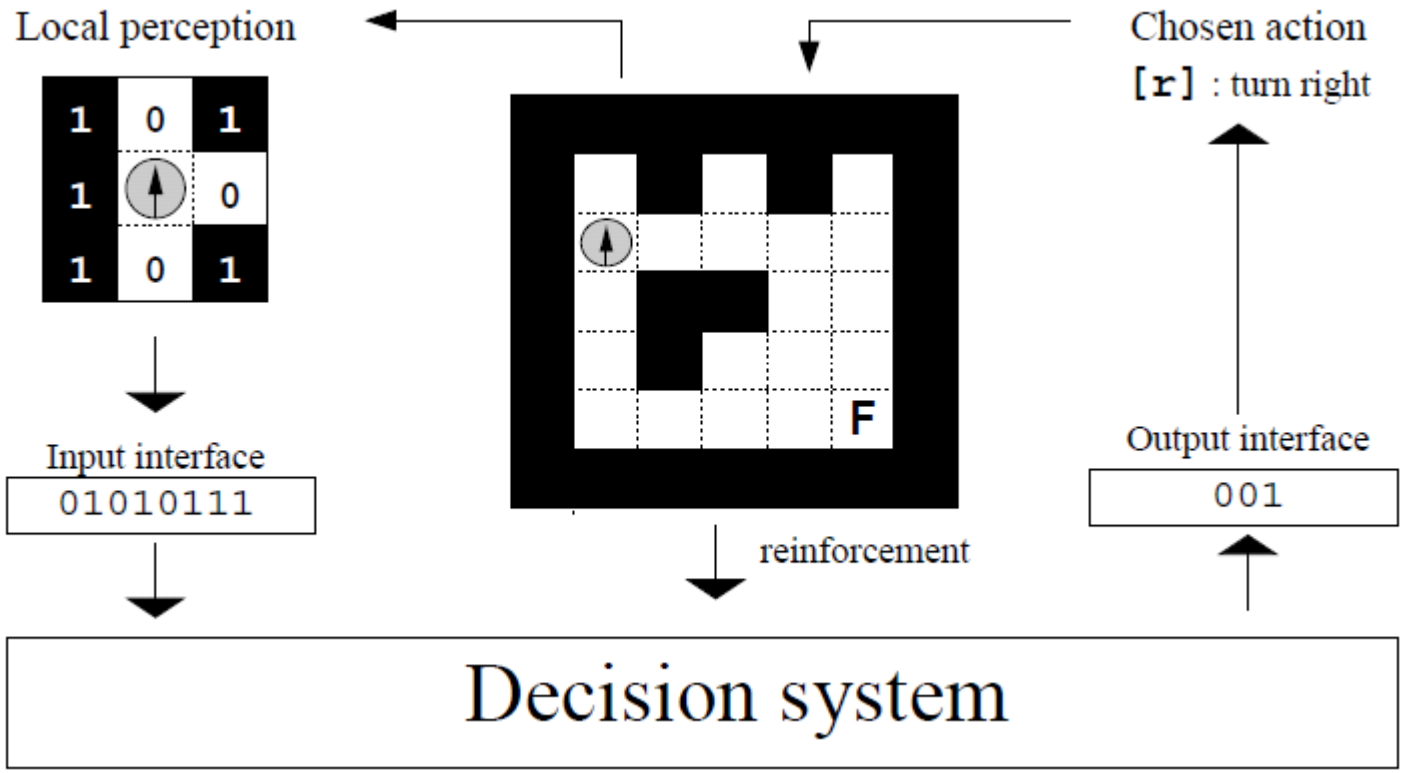
MDP

$$E_{k+1}(s) = E_k(s) + \alpha[r_{k+1} - E_k(s)]$$

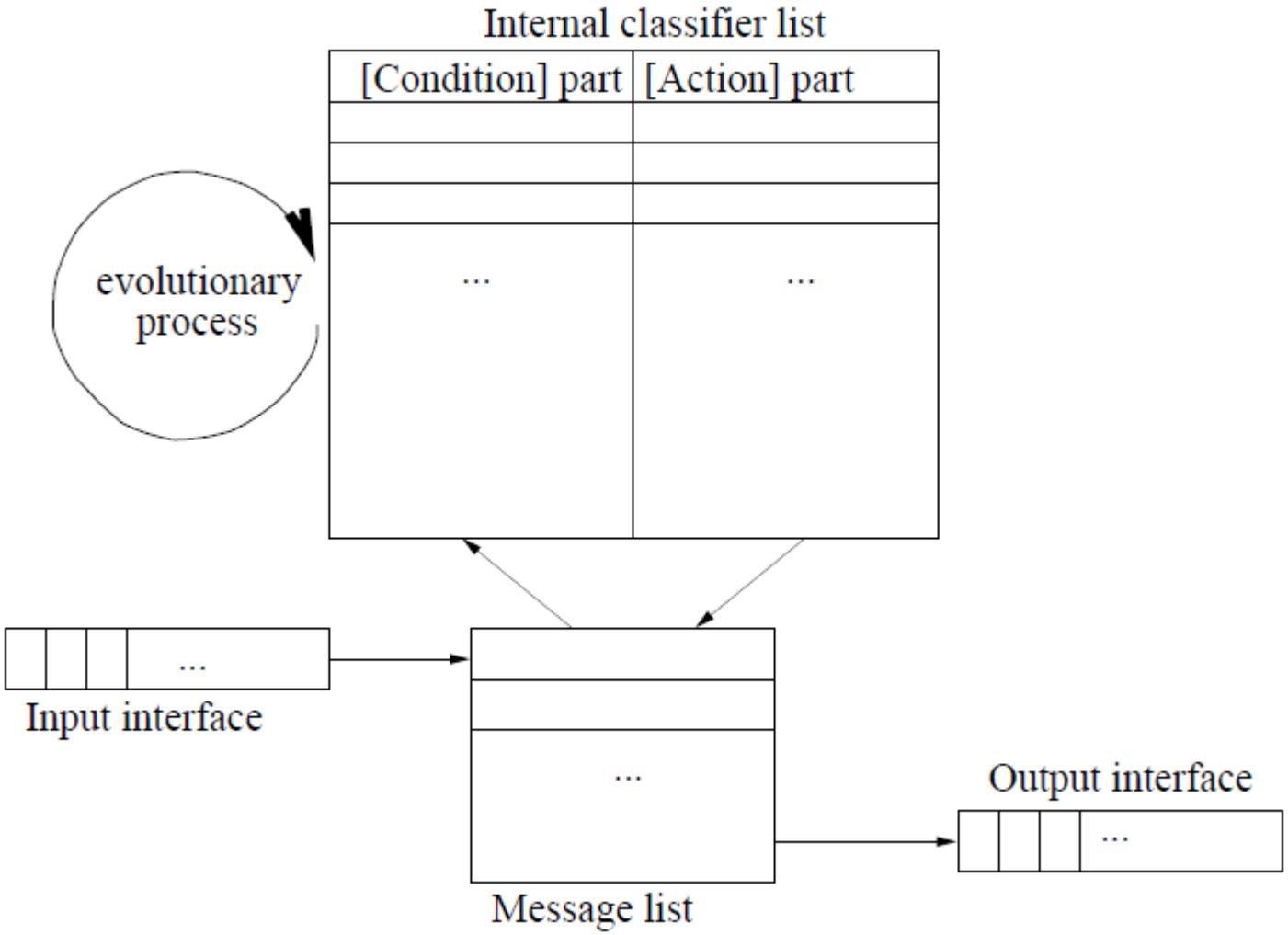
The parameter α, called *learning rate*, must be tuned adequately because it influences the speed of convergence towards the exact average.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

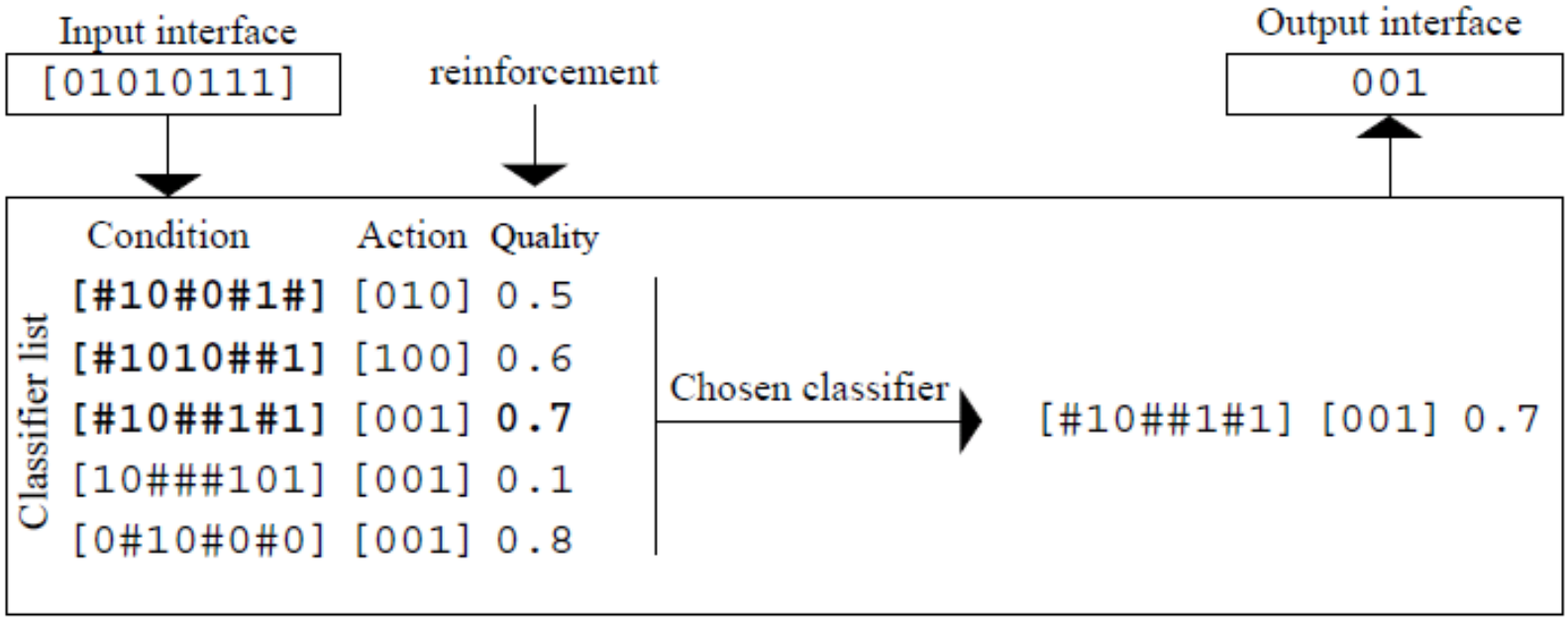
RL + GA = LCS


Local perception
1 0 1
1 0
1 0 1
Input interface
01010111
F
reinforcement
Chosen action
[r] : turn right
Output interface
001
Decision system

LCS


Internal classifier list
[Condition] part
[Action] part
...
...
evolutionary process
...
Input interface
...
Message list
Output interface
...

LCS


Input interface
[01010111]
reinforcement
Output interface
001
Condition
Action
Quality
Classifier list
[#10#0#1#] [010] 0.5
[#1010##1] [100] 0.6
[#10##1#1] [001] 0.7
[10###101] [001] 0.1
[0#10#0#0] [001] 0.8
Chosen classifier
[#10##1#1] [001] 0.7