Distributed Evolutionary Algorithms Inspired by Membranes in Solving Continuous Optimization Problems

Gabriel Ciobanu

Institute of Computer Science, Iasi, Romania

gabriel@iit.tuiasi.ro

Daniela Zaharie

West University of Timisoara, Romania

dzaharie@info.uvt.ro

Outline

- Motivation
- Panmictic Evolutionary Algorithms (EAs)
  - Evolutionary operators and evolution rules
  - A non-standard strategy for applying the evolutionary operators
- Distributed Evolutionary Algorithms (DEAs)
  - Communication topologies and membrane structures
  - A membrane inspired distributed evolutionary algorithm
- Numerical results for continuous optimization problems
- Conclusions and further work

Motivation

Nature inspired computation models

- Based on applying evolution(ary) rules to a multiset of objects
- Characterized by distributed features

Evolutionary computing

- Objects are binary strings, real vectors..
- Approximate computation based on evolving populations of candidate solutions
- Evolutionary operators inspired by biological species evolution

Membrane computing

- Objects are symbols
- Exact computation based on evolution rules
- Evolution and communication rules inspired by biological processes in cells

Motivation

- How similar are membrane computing and evolutionary computing ?
- Can ideas from membrane computing improve evolutionary computing or viceversa ?

Previous work [Nishida, 2004]: membrane inspired algorithm for solving combinatorial optimization problems

Aim of this work: to analyze deeper the relationship between membrane systems and distributed evolutionary algorithms

Evolutionary Algorithms

Optimization problem:

Find x^* such that

$$f(x^*) = \min_{x \in D} f(x)$$

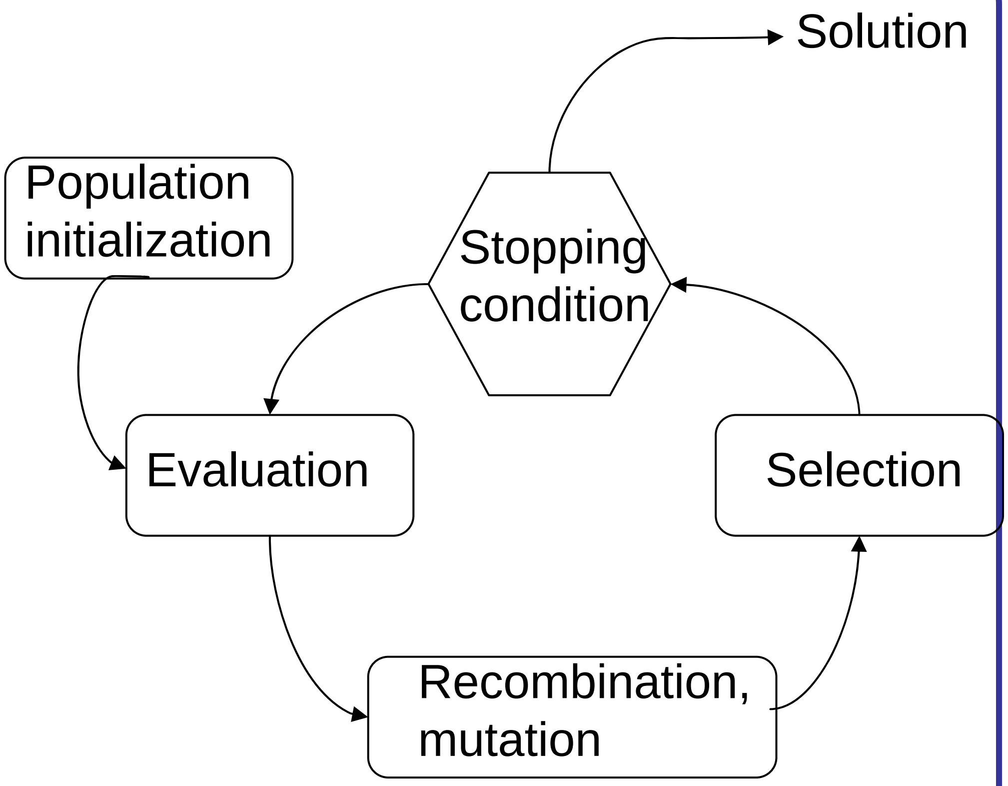
$$f : D^n \to R$$

Population:

$$(x_1, \ldots, x_m), \;\; x_i \in D^n$$

Evolutionary operators:

- Explorative: recombination and mutation
- Exploitative: selection



Evolutionary Operators and Evolution Rules

Recombination operator:

$$(x_1, \ldots, x_m)$$

$\downarrow$ parents selection

$$(x_{i_1}, \ldots, x_{i_r})$$

$\downarrow$ offsprings generation

$$(y_{i_1}, \ldots, y_{i_q})$$

Example: $q = 1,\ y = \sum_{j=1}^{r} w_j x_{i_j}$

Corresponding evolution rule:

$$x_{i_1} \ldots x_{i_r} \rightarrow x_{i_1} \ldots x_{i_r} y_{i_1} \ldots y_{i_q}$$

Recombination effect:

- Generates new elements

Parents selection:

- Randomly
- By favoring the best elements

Evolutionary Operators and Evolution Rules

Mutation operator:

$$y_i$$
$$\downarrow \quad \text{perturbation}$$
$$z_i$$

Example:

$$z_i^j = y_i^j + N(0, \sigma^j), \quad j = \overline{1, n}$$

Corresponding evolution rule:

$$y_i \rightarrow y_i z_i$$

Mutation effect:

- Generate a new element by perturbing an existing one

Perturbation:

- By adding a random term
- By randomly changing some components

Evolutionary Operators and Evolution Rules

Selection operator:

- It does not generate new configurations
- Change the distribution of the elements
  - Low quality elements are eliminated (do not survive in the next generation)
  - For good quality elements, multiple copies can be created

Corresponding evolution rules:

Selection by cloning

$$x_* \to x_* x_*$$

Selection by deletion

$$x_- \to \lambda$$

x_* - best element

x_- - worst element

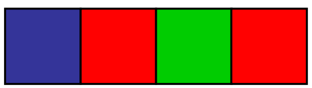
Evolutionary Operators and Evolution Rules

Strategies for applying the evolutionary operators:

- Generational (synchronous)
  - Starting from the current population a new one is generated by recombination and mutation
  - The population corresponding to the next generation is obtained by selection from these two populations
- Steady state (asynchronous)
  - When a new element is obtained it is assimilated into population if it is good enough (by replacing one of its parents or the worst element from the population)

Evolutionary Operators and Evolution Rules

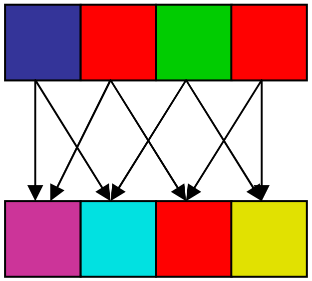
- Generational



Current population

Evolutionary Operators and Evolution Rules

- Generational



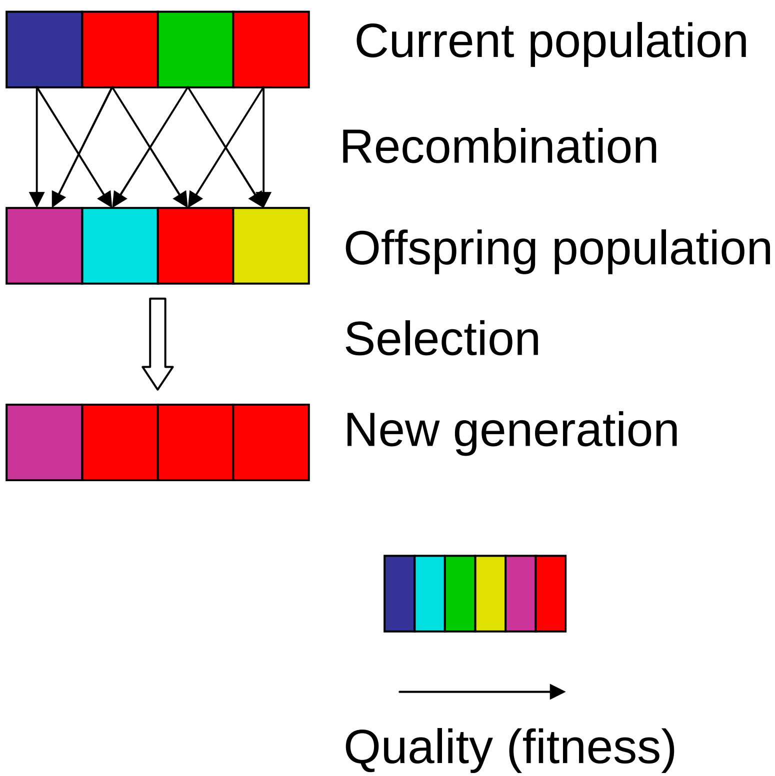
Current population

Recombination

Offspring population

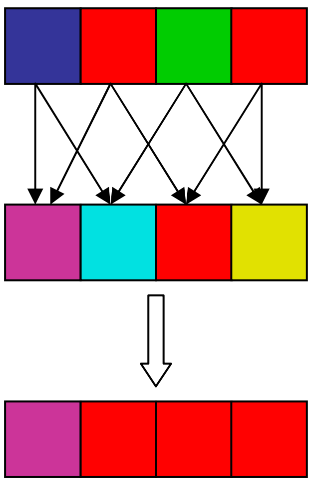
Evolutionary Operators and Evolution Rules

- Generational



Evolutionary Operators and Evolution Rules

- Generational



Current population

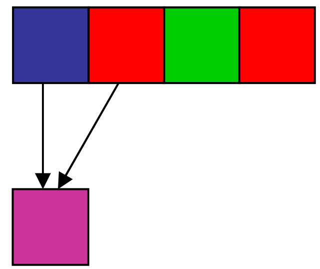
Recombination

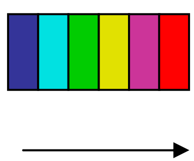
Offspring population

Selection

New generation

- Steady state

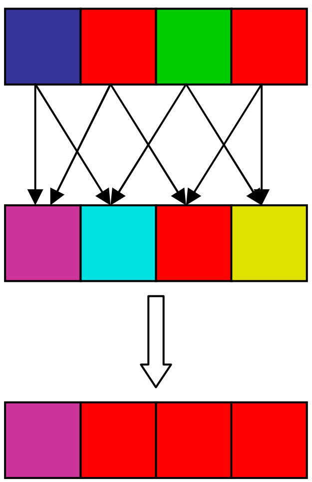




Quality (fitness)

Evolutionary Operators and Evolution Rules

- Generational



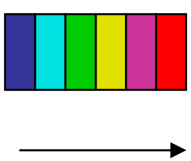
Current population

Recombination

Offspring population

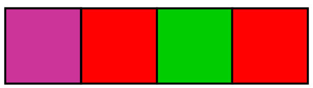
Selection

New generation



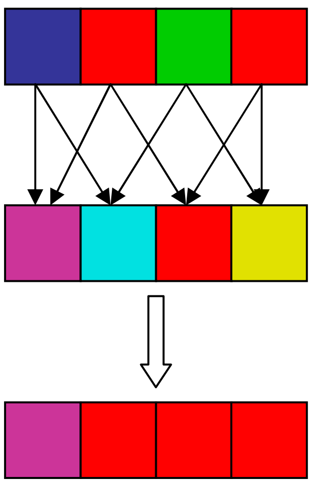
Quality (fitness)

- Steady state



Evolutionary Operators and Evolution Rules

- Generational



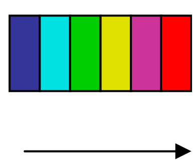
Current population

Recombination

Offspring population

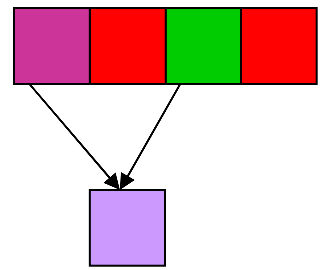
Selection

New generation



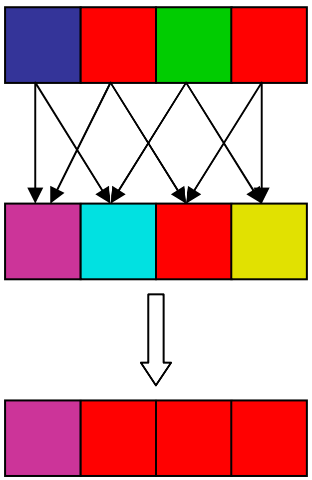
Quality (fitness)

- Steady state



Evolutionary Operators and Evolution Rules

- Generational



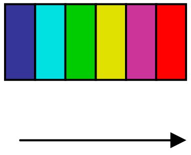
Current population

Recombination

Offspring population

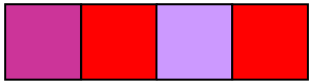
Selection

New generation



Quality (fitness)

- Steady state



- Different strategies lead to different resulting populations
- What about other strategies ?

Evolutionary Operators and Evolution Rules

Describing an EA as a simple membrane system:

- One population = one membrane
- Individuals=objects
- Evolutionary operators=evolution rules
- Classical EA: the operators are applied in a predefined order
- A different approach: at each step any rule can be applied

Objects:

$(x_1, x_2, \ldots, x_m), x_i \in D^n$

Rules:

$R_1 : x_{i_1} \ldots x_{i_r} \rightarrow x_{i_1} \ldots x_{i_r} y_{i_1} \ldots y_{i_q}$

$R_2 : x_i \rightarrow x_i z_i$

$R_3^c : x_* \rightarrow x_* x_*$

$R_3^d : x_- \rightarrow \lambda$

A more flexible strategy

- Apply the evolutionary operators in an arbitrary manner, based on some probabilities:
  - Recombination: p_r
  - Mutation: p_m
  - Selection: $1-(p_m+p_r)$
- Population of variable size
- The recombination and mutation operators are inhibited when $m(t) > 2m(0)$
- The selection by deletion is inhibited when $m(t) < m(0)/2$

Rules application:

```
u:=rand(0,1)
if u<p_r and m(t)<2m(0)
then apply Recombination
else
   if u<p_r+p_m and m(t)<2m(0)
   then apply Mutation
    else
     if m(t)>m(0)/2
     then apply Selection by
                     deletion
```

Distributed Evolutionary Algorithms

- Basic idea:
  - Divide the population in communicating subpopulations
  - On each subpopulation is applied an EA
  - Periodically some elements are transferred between subpopulations
- Motivation:
  - Stimulates the population diversity in order to avoid premature convergence
  - Allows efficient parallel implementations
- Main elements:
  - Communication topology: which subpopulations can communicate ?
  - Communication policy: how communicate the subpopulations ?

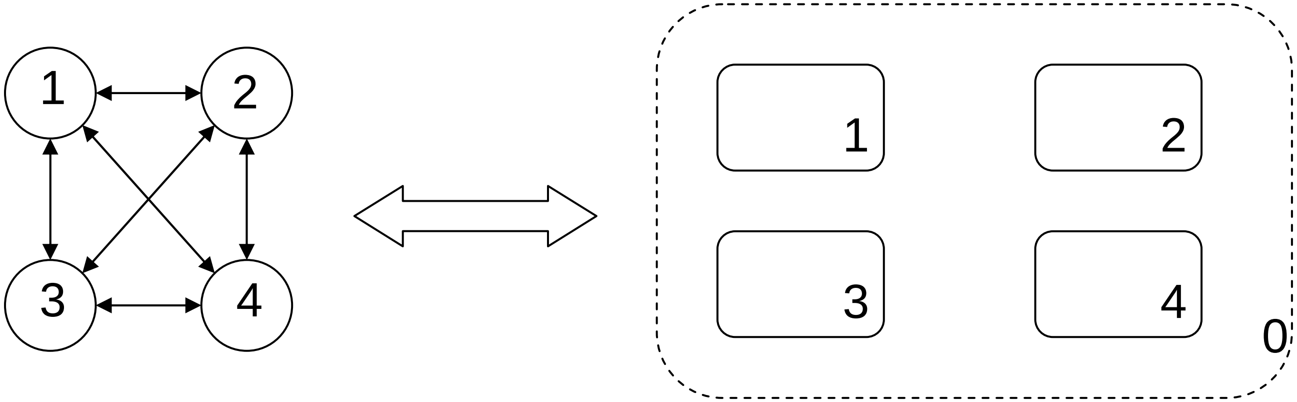
Distributed EAs and Membrane Systems

Distributed EA	Membrane System
Subpopulation	Region
Communication topology	Membrane structure
Communication policy	Communication rules

Our aim was to analyze in more details the similarities between the corresponding elements of distributed EAs (DEAs) and Membrane Systems

Communication topologies and membrane structures

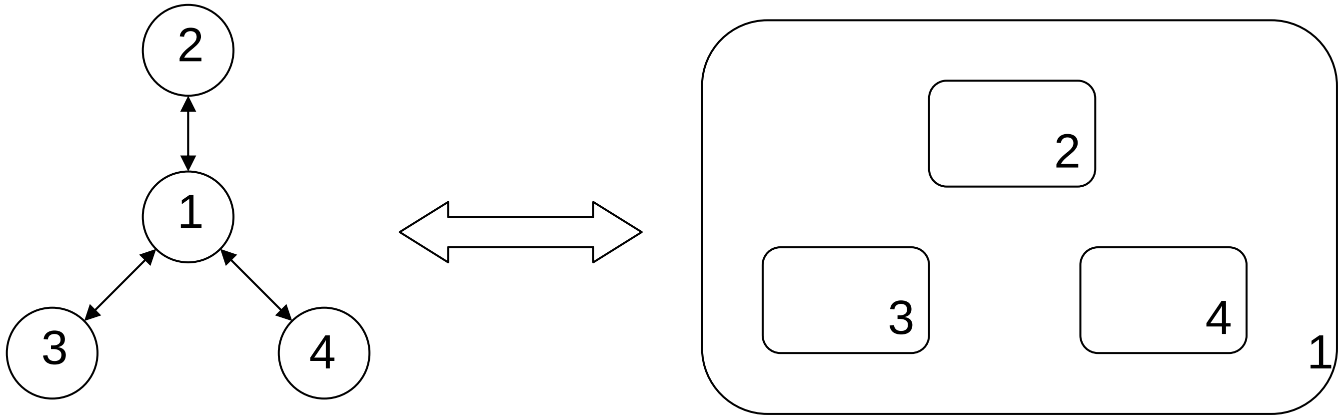
- Fully connected topology: any two subpopulations can communicate



- The skin membrane plays the role of a communication environment; it contains only communication rules
- A more natural correspondence is between fully connected topology and the structure of tissue P systems

Communication topologies and membrane structures

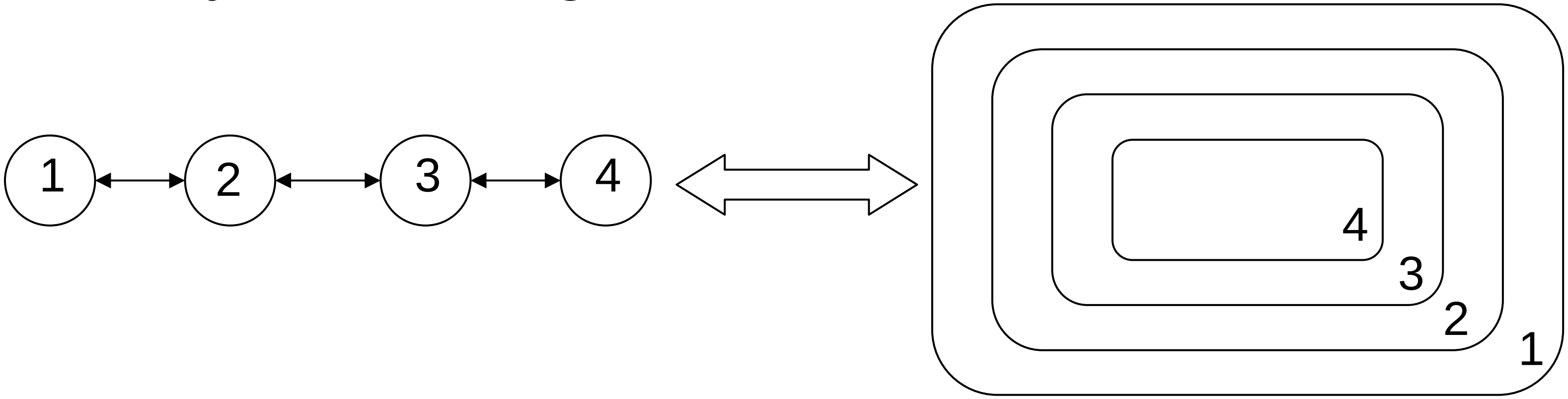
- Star topology: the subpopulations communicate through a kernel subpopulation



- The skin membrane corresponds to the kernel subpopulations; it contains objects, evolution rules and communication rules
- The star topology is not as frequently used in DEA as other topologies

Communication topologies and membrane structures

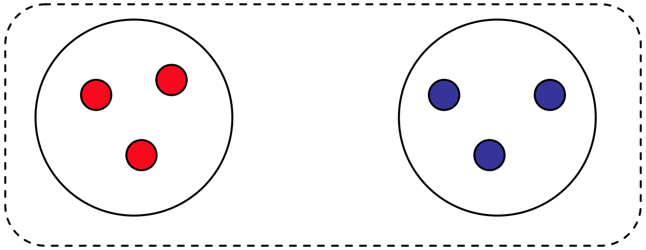
- **Linear topology**: the subpopulations are organized as a linear structure and each one can communicate only with its neighbors



- This structure is similar with that used in the membrane algorithm by [Nishida, 2004]
- Other topologies (e.g. ring and other cyclic graphs based) can be easily put in correspondence with tissue P systems

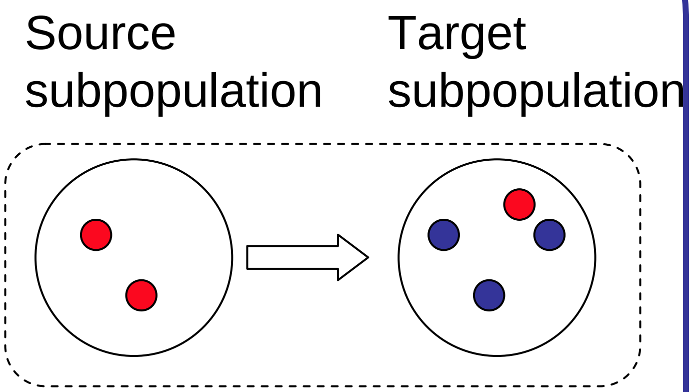
Communication policies and communication rules

- Migration:
  - move an element from the source subpopulation to the target one



Communication policies and communication rules

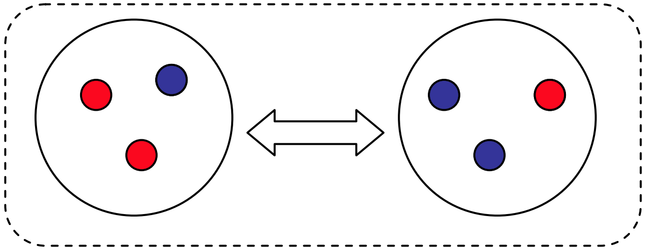
- Migration:
  - move an element from the source subpopulation to the target one



Communication policies and communication rules

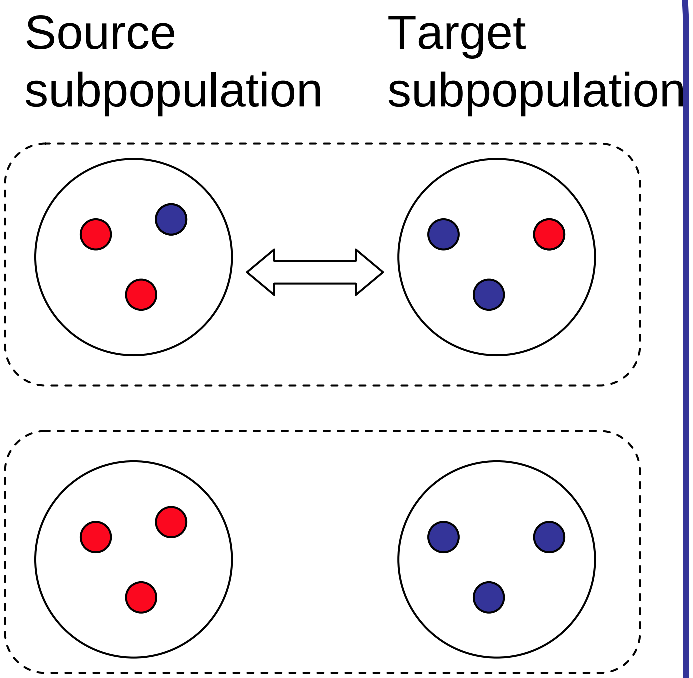
- Migration:
  - move an element from the source subpopulation to the target one

Source subpopulation Target subpopulation



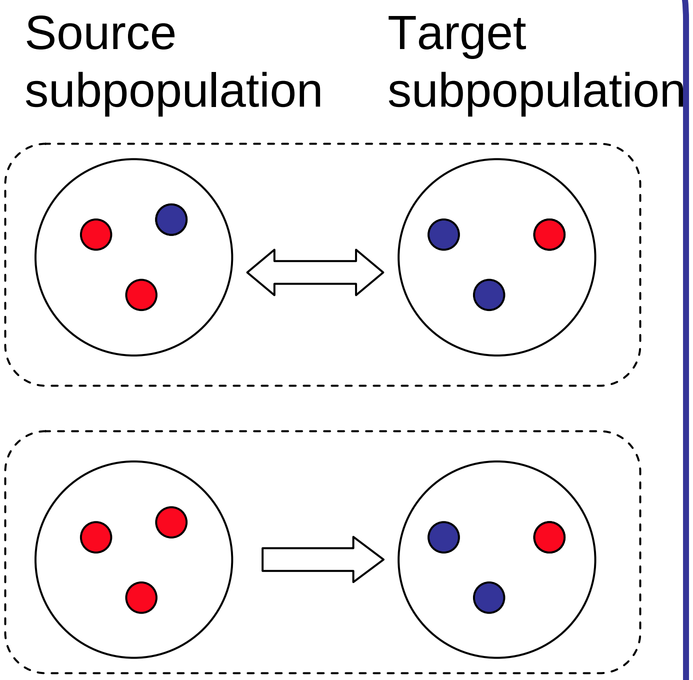
Communication policies and communication rules

- Migration:
  - move an element from the source subpopulation to the target one
- Pollination:
  - send a copy of an element from the source subpopulation to the target one



Communication policies and communication rules

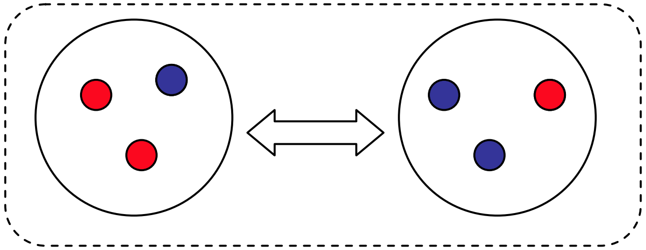
- Migration:
  - move an element from the source subpopulation to the target one
- Pollination:
  - send a copy of an element from the source subpopulation to the target one



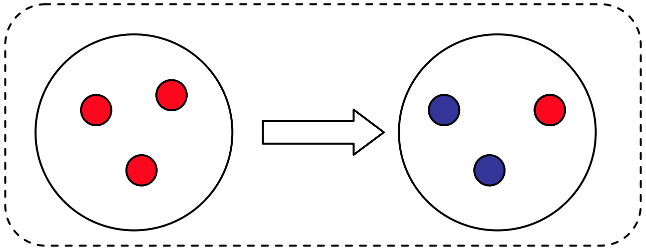
Communication policies and communication rules

Source subpopulation   Target subpopulation

- Migration:
  - move an element from the source subpopulation to the target one



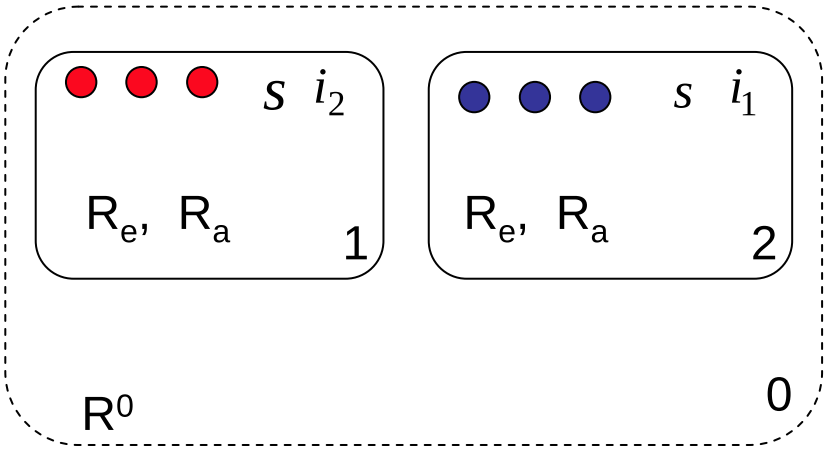
- Pollination:
  - send a copy of an element from the source subpopulation to the target one



- Selection of the elements to be transferred:
  - At random: each element has the same probability to be selected
  - In an elitist manner: copies of best elements are sent to replace worst elements in the target subpopulation

Communication policies and communication rules

- Example: random pollination in a fully connected topology



- Communication rules in elementary membranes:

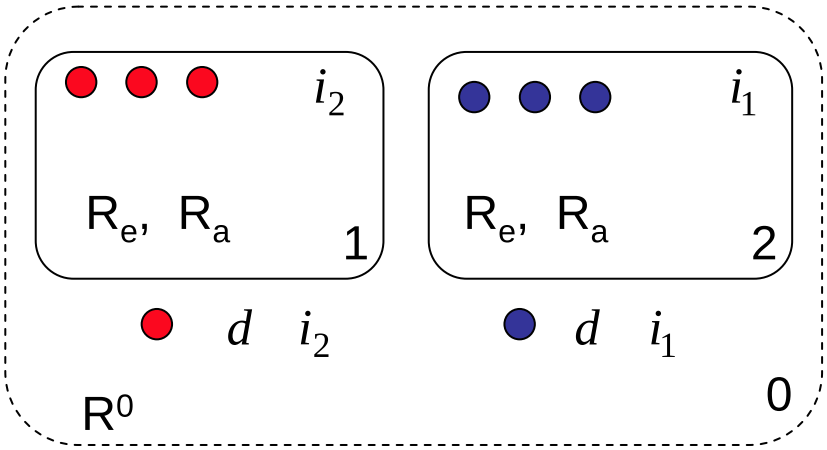
$$R_e : sxi_{id} \rightarrow (xi_{id}, here)(xi_{id}d, out)$$
$$R_a : dx \rightarrow \lambda$$

- Communication rules in the skin membrane:

$$R^0 : \{dxi_1 \rightarrow (dx, in_1), dxi_2 \rightarrow (dx, in_2)\}$$

Communication policies and communication rules

- Example: random pollination in a fully connected topology



- Communication rules in elementary membranes:

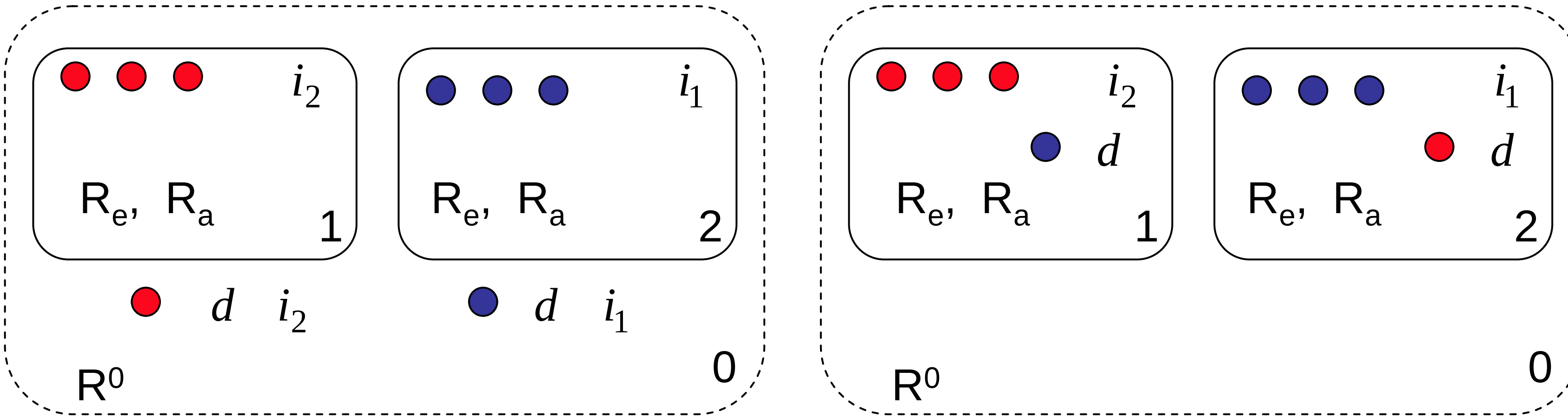
$$\boxed{R_e : sxi_{id} \to (xi_{id}, here)(xi_{id}d, out)}$$
$$R_a : dx \to \lambda$$

- Communication rules in the skin membrane:

$$R^0 : \{dxi_1 \to (dx, in_1), dxi_2 \to (dx, in_2)\}$$

Communication policies and communication rules

- **Example:** random pollination in a fully connected topology



- Communication rules in elementary membranes:

$$R_e : sxi_{id} \rightarrow (xi_{id}, here)(xi_{id}\, d, out)$$

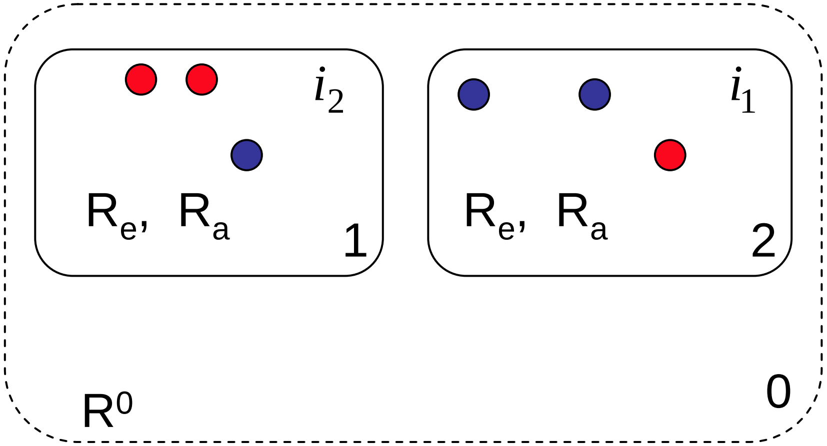
$$R_a : dx \rightarrow \lambda$$

- Communication rules in the skin membrane:

$$\boxed{R^0 : \{dxi_1 \rightarrow (dx, in_1), dxi_2 \rightarrow (dx, in_2)\}}$$

Communication policies and communication rules

- Example: random pollination in a fully connected topology



- Communication rules in elementary membranes:

$$R_e : sxi_{id} \rightarrow (xi_{id}, here)(xi_{id}d, out)$$

$$\boxed{R_a : dx \rightarrow \lambda}$$

- Communication rules in the skin membrane:

$$R^0 : \{dxi_1 \rightarrow (dx, in_1), dxi_2 \rightarrow (dx, in_2)\}$$

A membrane inspired DEA

❑ DEA structure

Initialization

Apply EA to all subpopulations / local search in the skin

Replace the worst element in subpopulations with an element from the skin

Send copies of the best elements from the subpopulations to the skin

❑ Membrane structure

EA + $x_* \to (x_*, here)(x_*, out)$
$dx\_ \to \lambda$
1

…

EA + $x_* \to (x_*, here)(x_*, out)$
$dx\_ \to \lambda$
t

Local search
Random immigrants
Communication rules:

$$x \to (x, here)(xd, in_1)$$

...

$$x \to (x, here)(xd, in_t)$$

0

A membrane inspired DEA

- Details on the local EA
  - Population elements: vectors of real values
  - Variation operator: recombination and mutation - typical for Differential Evolution algorithms [Storn&Price -1995]

$$y_i^j = \begin{cases} \gamma x_*^j + (1-\gamma)(x_{r_1}^j - x_*^j) + F \cdot (x_{r_2}^j - x_{r_3}^j) N(0,1) & \text{with probability } p \\ x_i^j \text{ or } rand(D) & \text{with probability } 1-p \end{cases}$$

$$\gamma \in [0,1],\ F \in (0,2),\ r_1, r_2, r_3 \text{ - random indices}$$

- Variants:
  - Generational strategy (constant population size)
  - Steady state strategy (constant population size)
  - Flexible strategy (variation operator + selection by deletion, variable population size)

A membrane inspired DEA

- Particularities of the implementation:
  - The (sub)populations are evolved in parallel for a given number of generations
  - The communications stage is based on a sequential application of the following steps:
    - Copies of the best element from each subpopulation are sent to the skin
    - Random elements are injected into the skin
    - The worst element in each subpopulation is replaced with an arbitrary element of the skin
- Key points:
  - Probabilistic applications of rules
  - Injection of random elements

Membrane inspired DEA vs. Nishida membrane algorithm

Membrane inspired DEA

- Star topology
- Medium sized population, small number of membranes
- Not very frequent communications (e.g. at 100 steps)
- Continuous optimization

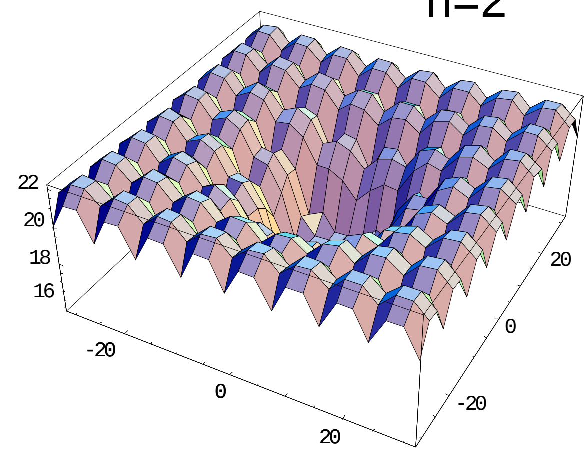
Nishida membrane algorithm

- Linear topology
- Micro-populations, larger number of membranes
- Frequent communications (at each step)
- Combinatorial optimization

Numerical results for continuous optimization problems

- Test problems
  - Multimodal minimization problem (n=30)
  - Minimum: (0,0…,0)

n=2

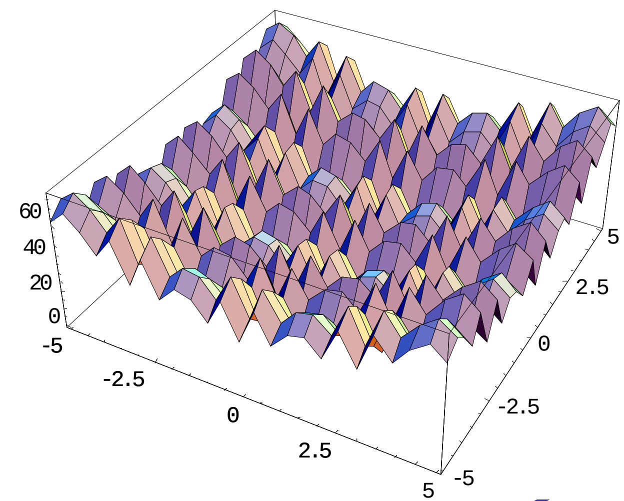


Ackley's function

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right) - \exp\left(\frac{\sqrt{\sum_{i=1}^{n}\cos(2\pi x_i)}}{n}\right) + 20 + e$$

Rastrigin's function

$$f(x) = \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$$



Numerical results for continuous optimization problems

- First set of experiments: one-population EA
  - Parameters: $m(0)=50$, $p=F=0.5$, $p_R=0.5$, $f_*=10^{-5}$
  - Measures:
    - Effectiveness: successful runs/total runs
    - Efficiency: number of function evaluations

	Generational		Steady state		Flexible strategy	
Test function	Success ratio	<Nfe> (std)	Success ratio	<Nfe> (std)	Success ratio	<Nfe> (std)
Ackley	30/30	37490 (653)	30/30	34785 (495)	30/30	47297 (5950)
Rastrigin	21/30	423364 (69057)	12/30	437212 (39375)	30/30	158794 (16534)

Premature convergence

Numerical results for continuous optimization problems

- Second set of experiments:
  - 5 populations
  - Parameters: m(0)=10, p=F=0.5, p_R=0.5, $f_*=10^{-5}$, communication at each 100 generations
  - Local search based on Nelder Mead heuristics

Test function	Generational+ Random migration: Success ratio	Generational+ Random migration: <Nfe> (std)	Membrane inspired DEA: Success ratio	Membrane inspired DEA: <Nfe> (std)
Ackley	30/30	38075 (1375)	30/30	74571 (11858)
Rastrigin	5/30	148380 (70126)	30/30	154939 (49979)

Conclusions

- There are clear conceptual similarities between membrane systems and distributed evolutionary systems
- For difficult optimization problems characterized by small basins of attraction of the global optimum (e.g. Rastrigin) the membrane inspired DEA avoids premature convergence and provides good results
- The degree of randomness used by the membrane inspired DEA slows down the convergence, and in this way, for simpler optimization problems characterized by larger basins of attraction of the global optimum (e.g. Ackley) the classical DEA provides better results

Further work

- Testing the proposed DEA for other classes of optimization problems
- Developing a strategy for adapting the probabilities used in applying the evolutionary operators
- Exploring the possibility of using the membrane systems formalism in order to understand the behavior of distributed evolutionary algorithms

THANK YOU !