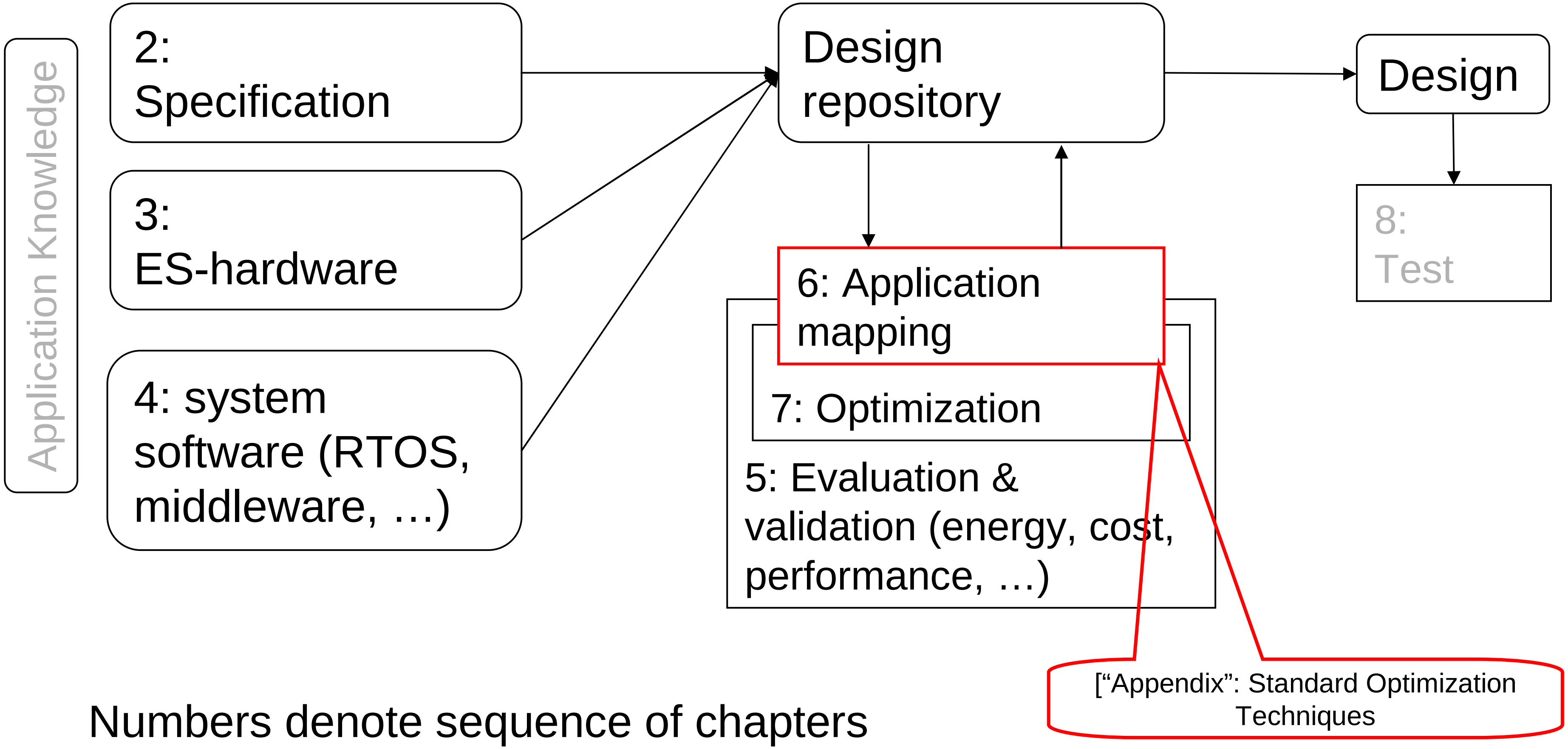
Standard Optimization Techniques

Peter Marwedel
TU Dortmund, Informatik 12
Germany

2010/12/20

Structure of this course



Numbers denote sequence of chapters

Integer (linear) programming models

Ingredients:

- Cost function
- Constraints

} Involving linear expressions of integer variables from a set X

Cost function $$C = \sum_{x_i \in X} a_i x_i \text{ with } a_i \in R, x_i \in \mathbb{N} \quad (1)$$

Constraints: $$\forall j \in J : \sum_{x_i \in X} b_{i,j}\, x_i \geq c_j \text{ with } b_{i,j}, c_j \in \mathbb{R} \quad (2)$$

Def.: The problem of minimizing (1) subject to the constraints (2) is called an **integer (linear) programming (ILP) problem**.

If all x_i are constrained to be either 0 or 1, the IP problem said to be a **0/1 integer (linear) programming problem**.

Cost function $C = \sum_{x_i \in X} a_i x_i$ with $a_i \in R, x_i \in \mathbb{N}$ (1)

Constraints: $\forall j \in J : \sum_{x_i \in X} b_{i,j} x_i \geq c_j$ with $b_{i,j}, c_j \in \mathbb{R}$ (2)

Example

$$C = 5x_1 + 6x_2 + 4x_3$$

$$x_1 + x_2 + x_3 \geq 2$$

$$x_1, x_2, x_3 \in \{0,1\}$$

x_1	x_2	x_3	C
0	1	1	10
1	0	1	9
1	1	0	11
1	1	1	15

← Optimal (row: 1, 0, 1, 9)

Remarks on integer programming

- Maximizing the cost function: just set $C' = -C$
- Integer programming is NP-complete.
- Running times depend exponentially on problem size, but problems of >1000 vars solvable with good solver (depending on the size and structure of the problem)
- The case of $x_i \in \mathbb{R}$ is called *linear programming* (LP). Polynomial complexity, but most algorithms are exponential, in practice still faster than for ILP problems.
- The case of some $x_i \in \mathbb{R}$ and some $x_i \in \mathbb{N}$ is called *mixed integer-linear programming.*
- ILP/LP models good starting point for modeling, even if heuristics are used in the end.
- Solvers: lp_solve (public), CPLEX (commercial), …

Simulated Annealing

- General method for solving combinatorial optimization problems.
- Based the model of slowly cooling crystal liquids.
- Some configuration is subject to changes.
- Special property of Simulated annealing: Changes leading to a poorer configuration (with respect to some cost function) are accepted with a certain probability.
- This probability is controlled by a temperature parameter: the probability is smaller for smaller temperatures.

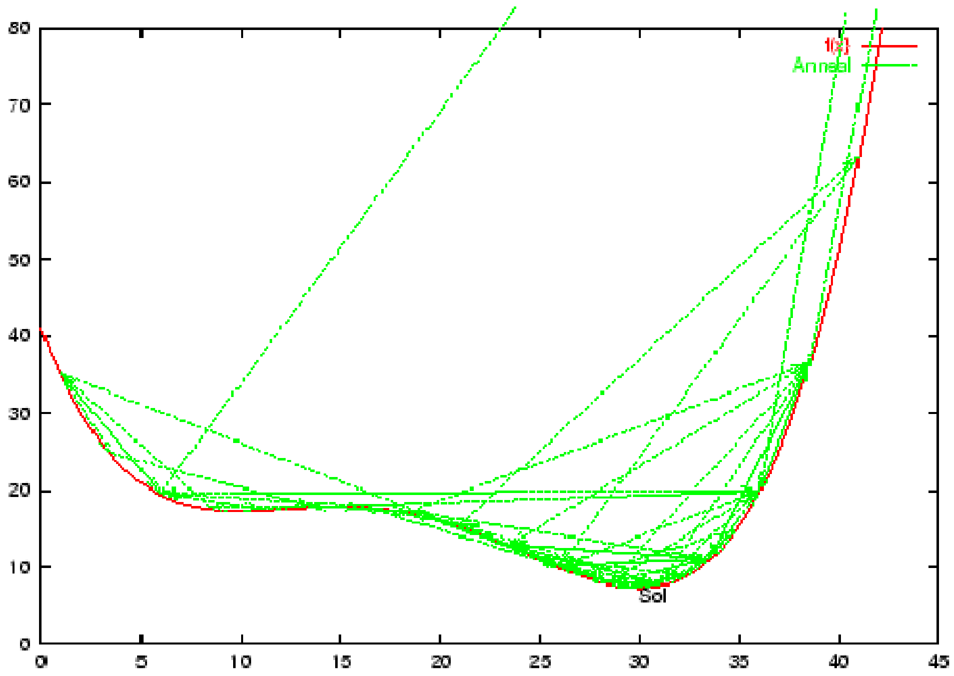
Simulated Annealing Algorithm

```
procedure SimulatedAnnealing;
var i, T: integer;
 begin
  i := 0; T := MaxT;
  configuration:= <some initial configuration>;
  while not terminate(i, T) do
   begin
    while InnerLoop do
      begin NewConfig := variation(configuration);
        delta := evaluation(NewConfig,configuration);
        if delta < 0
        then configuration := NewConfig;
        else if SmallEnough(delta, T, random(0,1))
          then configuration := Newconfiguration;
      end;
  T:= NewT(i,T); i:=i+1;
 end; end;
```

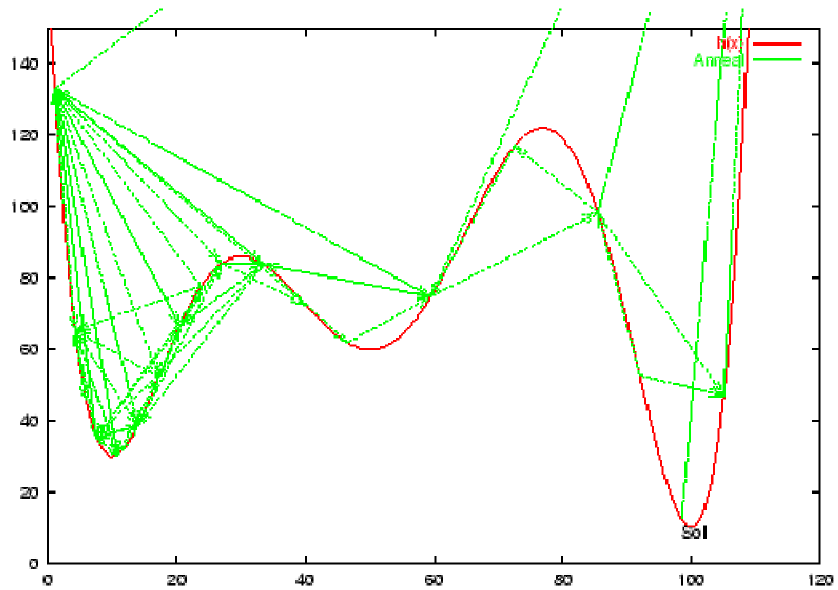
Explanation

- Initially, some random initial configuration is created.
- Current temperature is set to a large value.
- Outer loop:
  - Temperature is reduced for each iteration
  - Terminated if (temperature ≤ lower limit) or (number of iterations ≥ upper limit).
- Inner loop: For each iteration:
  - New configuration generated from current configuration
  - Accepted if (new cost ≤ cost of current configuration)
  - Accepted with temperature-dependent probability if (cost of new config. > cost of current configuration).

Behavior for actual functions



130 steps



200 steps

[people.equars.com/~marco/poli/phd/node57.html]

http://foghorn.cadlab.lafayette.edu/cadapplets/fp/fpIntro.html

Performance

- This class of algorithms has been shown to outperform others in certain cases [Wegener, 2005].
- Demonstrated its excellent results in the TimberWolf layout generation package [Sechen]
- Many other applications …

Evolutionary Algorithms (1)

- ***Evolutionary Algorithms*** *are based on the collective learning process within a population of individuals, each of which represents a search point in the space of potential solutions to a given problem.*
- *The population is arbitrarily initialized, and it evolves towards better and better regions of the search space by means of randomized processes of*
  - ***selection*** *(which is deterministic in some algorithms),*
  - ***mutation****, and*
  - ***recombination*** *(which is completely omitted in some algorithmic realizations).*

[Bäck, Schwefel, 1993]

Evolutionary Algorithms (2)

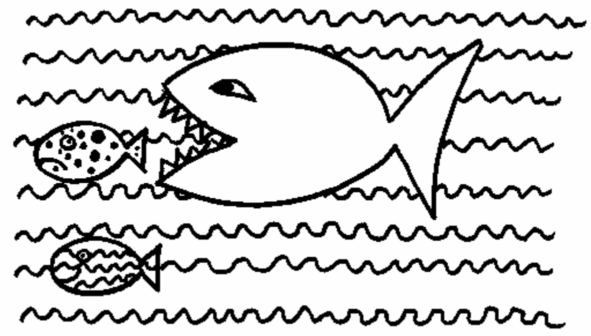
- *The environment (given aim of the search) delivers a quality information (**fitness value**) of the search points, and the selection process favours those individuals of higher fitness to reproduce more often than worse individuals.*
- *The recombination mechanism allows the mixing of parental information while passing it to their descendants, and mutation introduces innovation into the population*

[Bäck, Schwefel, 1993]

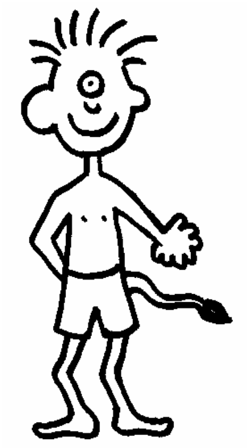
Evolutionary Algorithms

Principles of Evolution

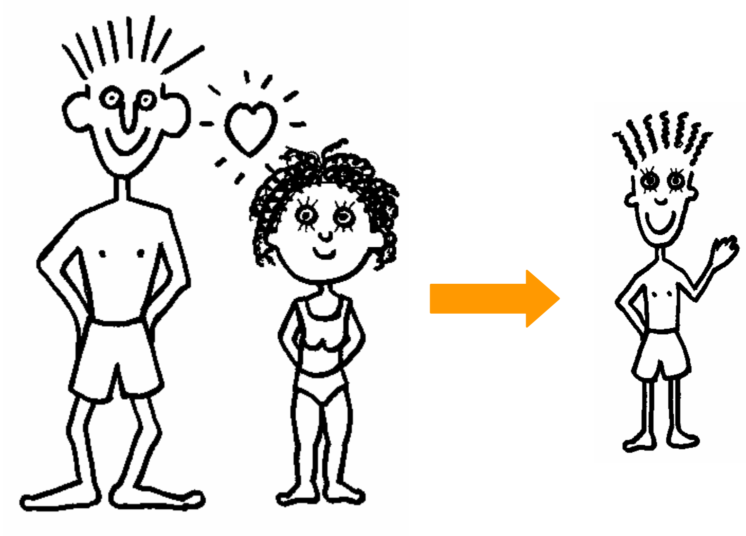
❶ Selection



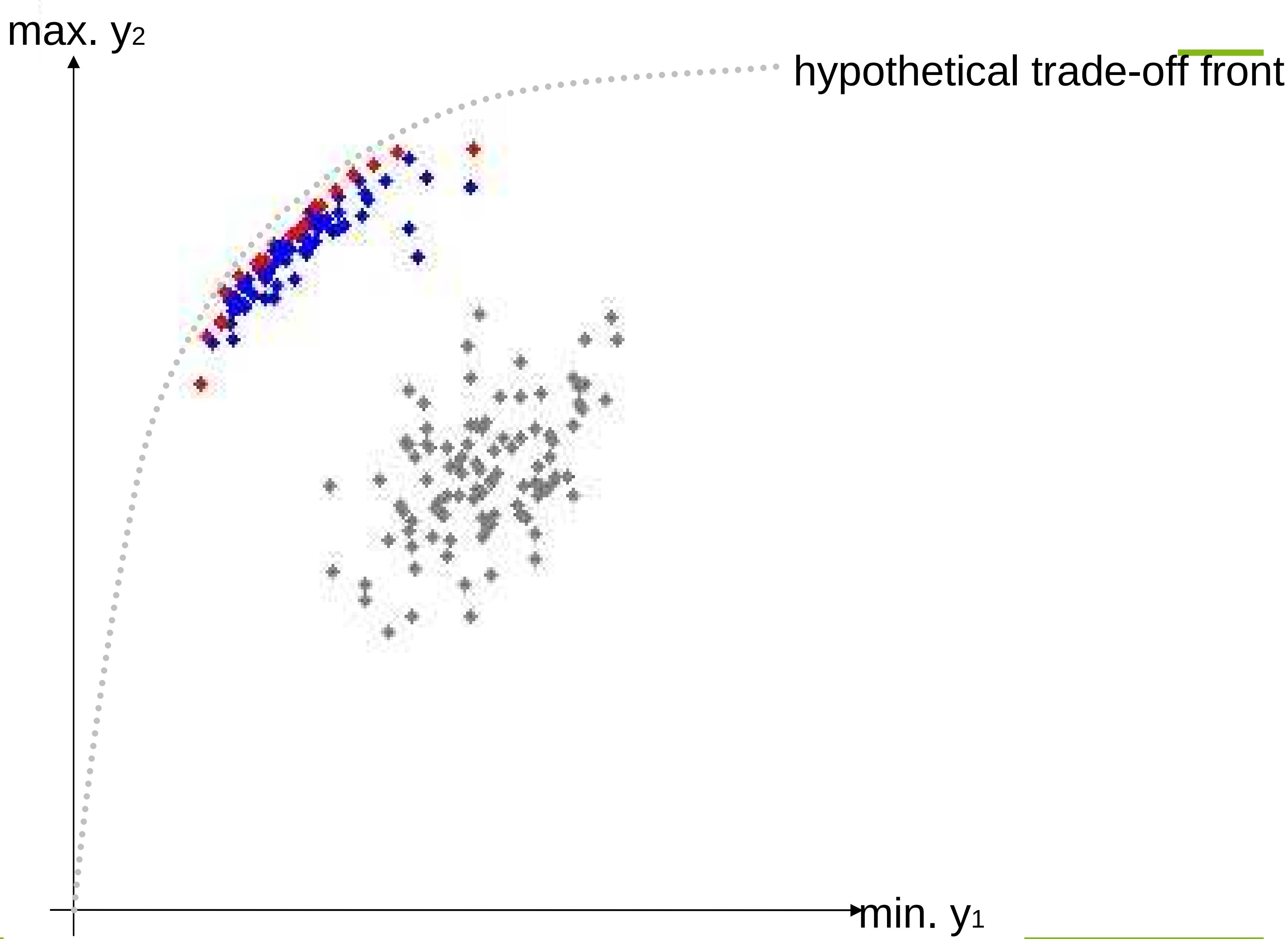
❷ Mutation



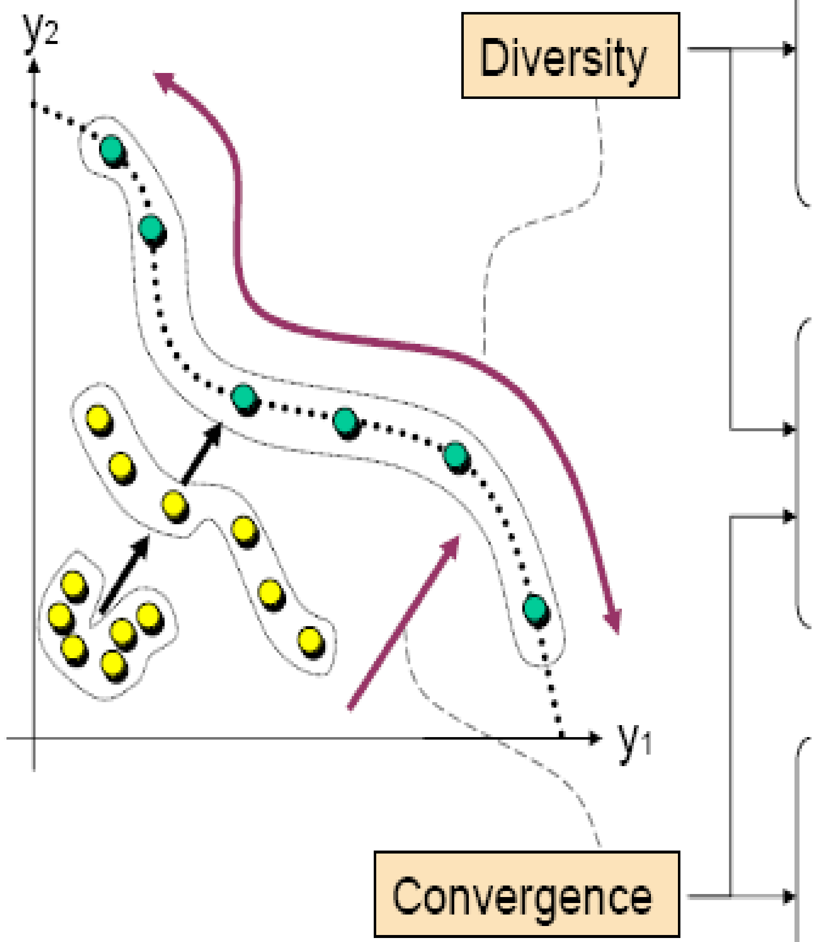
❸ Cross-over



An Evolutionary Algorithm in Action



Issues in Multi-Objective Optimization



- How to maintain a diverse Pareto set approximation?

  ❷ density estimation

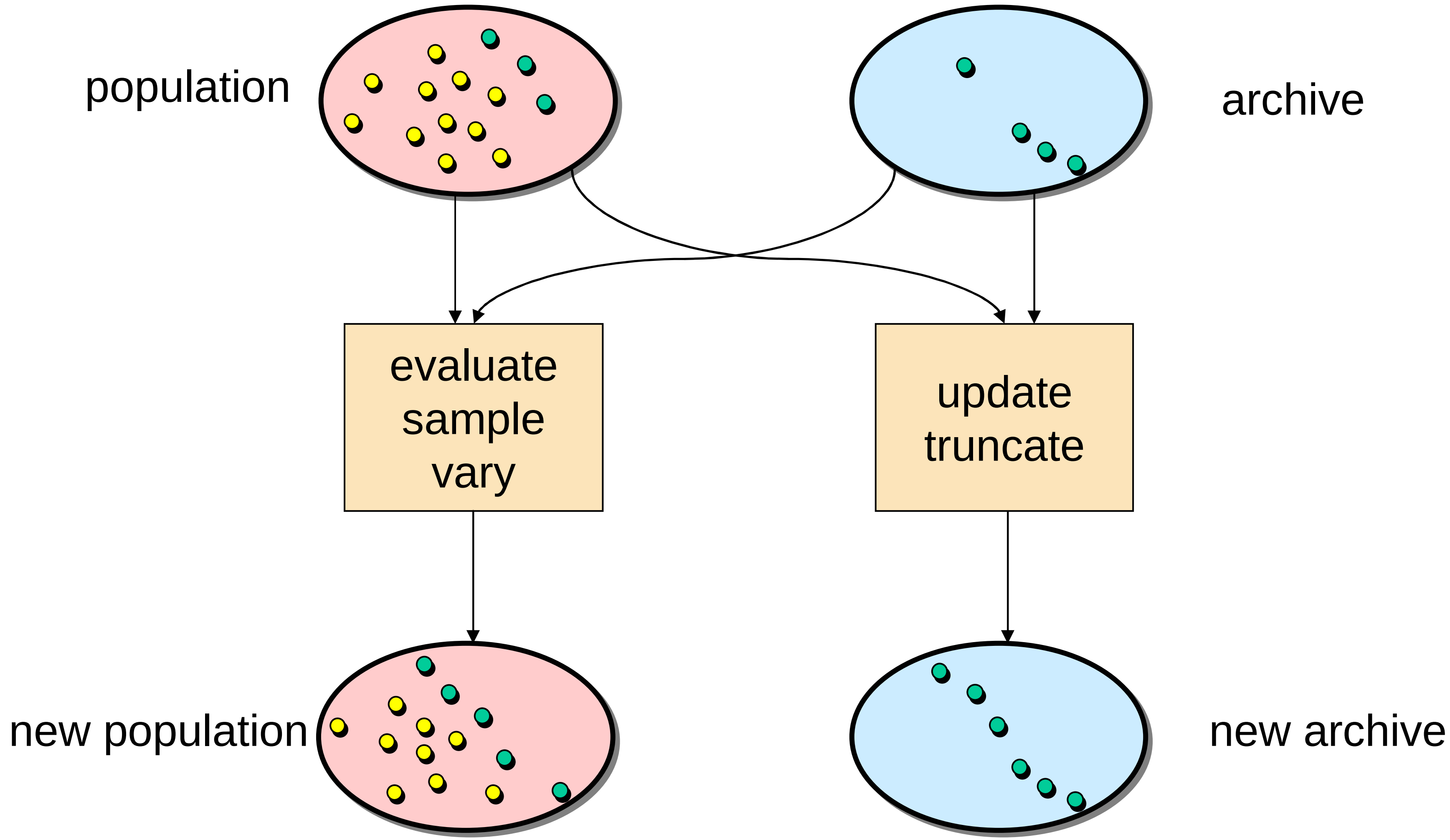
- How to prevent nondominated solutions from being lost?

  ❸ environmental selection

- How to guide the population towards the Pareto set?

  ❶ fitness assignment

A Generic Multiobjective EA



Example: SPEA2 Algorithm

Step 1:	Generate initial population P0 and empty archive (external set) A_0. Set t = 0.
Step 2:	Calculate fitness values of individuals in P_t and A_t.
Step 3:	A_{t+1} = nondominated individuals in P_t and A_t. If size of $A_{t+1} > N$ then reduce A_{t+1}, else if size of $A_{t+1} < N$ then fill A_{t+1} with dominated individuals in P_t and A_t.
Step 4:	If $t > T$ then output the nondominated set of A_{t+1}. Stop.
Step 5:	Fill mating pool by binary tournament selection.
Step 6:	Apply recombination and mutation operators to the mating pool and set P_{t+1} to the resulting population. Set $t = t + 1$ and go to Step 2.

Summary

Integer (linear) programming

- Integer programming is NP-complete
- Linear programming is faster
- Good starting point even if solutions are generated with different techniques

Simulated annealing

- Modeled after cooling of liquids
- Overcomes local minima

Evolutionary algorithms

- Maintain set of solutions
- Include selection, mutation and recombination