Advanced Optimization
Lecture 1: Randomized Algorithms for Discrete Problems

November 22, 2016

Master AIC

Université Paris-Saclay, Orsay, France

Anne Auger

INRIA Saclay – Ile-de-France

Dimo Brockhoff

INRIA Saclay – Ile-de-France

Numerical Blackbox Optimization

Typical scenario in the continuous case:

Optimize $f: \Omega \subset \mathbb{R}^n \mapsto \mathbb{R}^k$

derivatives not available or not useful

Lecture Goals

As in introductory lecture: always examples and small exercises to learn “on-the-fly” the concepts and fundamentals

Overall goals:

1. give more details on a few important aspects of blackbox optimization
2. prepare you better for a potential Master's thesis (PhD thesis) on the topic

Hence, I will give later on some details on our available projects

Course Overview

	Date		Topic
1	Tue, 22.11.2016	Dimo	Randomized Algorithms for Discrete Problems
2	Tue, 29.11.2016	Dimo	Exercise: The Travelling Salesperson Problem
3	Tue, 6.12.2016	Anne	Continuous Optimization I
	vacation		
4	Tue, 3.1.2017	Anne	Continuous Optimization II
5	Tue, 10.1.2017	Anne	Continuous Optimization III
6	Tue, 17.1.2017	Dimo	Evolutionary Multiobjective Optimization I
7	Tue, 31.1.2017	Dimo	Evolutionary Multiobjective Optimization II
	???		oral presentations (individual time slots)

all from 14:00 till 17:15 in PUIO - E213

Remark: new lecture, hence not all prepared yet ☺

No Exam...

Since the idea is to prepare you for your Master's thesis:

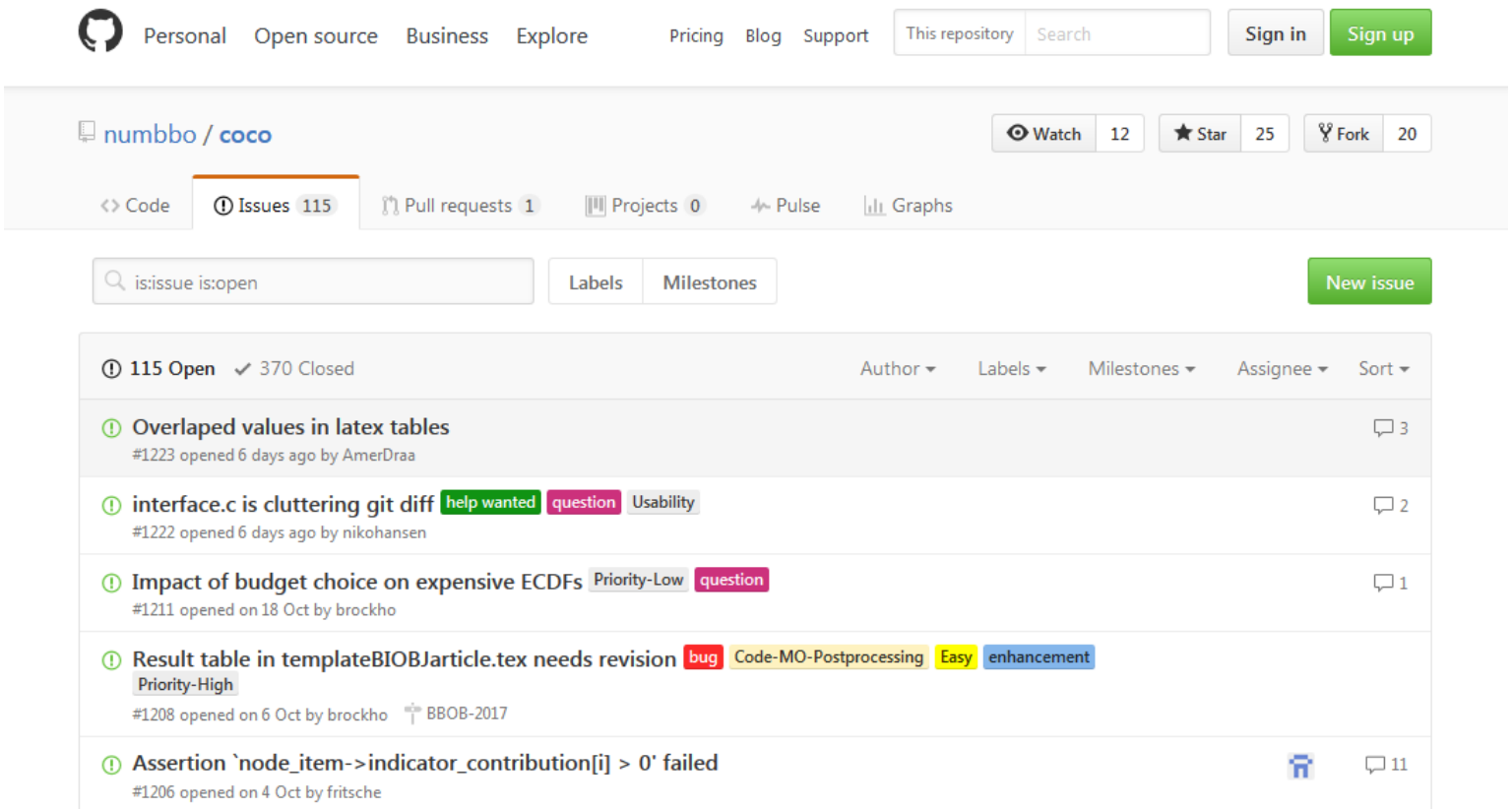
- we don't have a written exam ☺
- but instead work towards research:
  - each student is assigned a scientific paper (next week)
  - which is to be read, understood, critically questioned, and finally presented
  - summarize the paper in a short abstract in your own words
  - oral presentations in the end of the course
    (15min presentation + 15min oral "exam")
  - more in next week's lecture

  - also the exercises are (closer to) research questions than before

Additional Offer: Solving COCO Issues

In addition, we plan to offer an upgrade of your grade (by 1 point max.) if you happen to solve an issue from the COCO issue tracker!

https://github.com/numbbo/coco/issues/



Today's Lecture

1. present open projects
2. randomized search heuristics in the discrete domain
3. exercise: Pure Random Search (PRS) and the (1+1)EA

❶ present open projects

Potential Research Topics for Master's/PhD Theses

http://randopt.gforge.inria.fr/thesisprojects/

Trace: • start

Logged in as: Dimo Brockhoff (brockho)

THESIS PROJECTS

[[start]]

Home

Welcome!

On this page, you will find various current technical and scientific projects in the field of stochastic blackbox optimization proposed by Anne Auger, Dimo Brockhoff, and Nikolaus Hansen at Inria. Depending on the subject, the projects can be Bachelor, Master's, or PhD theses, or related to internships and might be carried out in close relationship with external collaborators, including companies.

If you are interested in (stochastic) blackbox optimization but your favorite topic is not mentioned here, feel free to contact us personally. We might always have other topics in mind, which range from theoretical studies to algorithm design but which have not yet been formalized here.

Current Openings

- Stopping Criteria for Multiobjective Optimizers (Master's project)
- Various technical projects around the COCO platform (Internships/Bachelor)
- Large-scale Stochastic Black-box Optimization (Master's project)
- The Orbit Algorithm for Expensive Numerical Blackbox Problems (Bachelor/Master's project)
- Data Mining Performance Results of Numerical Optimizers (Master's project)
- General Constraint Handling in the Stochastic Numerical Optimization Algorithm CMA-ES (CIFRE PhD)
- Designing Variants of the Covariance Matrix Adaptation Evolution Strategy to Handle Multiobjective Blackbox Problems (CIFRE PhD)

start.txt · Last modified: 2016/11/15 23:11 by brockho

Log Out | Edit this page | more actions ... | Search | Search

Potential Research Topics for Master's/PhD Theses

More projects without the involvement of companies:

- stopping criteria in multiobjective optimization
- large-scale variants of CMA-ES
- algorithms for expensive optimization based on CMA-ES

all above: relatively flexible between theoretical (e.g. proofs of convergence) and practical projects

Coco-related:

- implementing and benchmarking algorithms for expensive opt.
- data mining performance results

not all subjects online yet:
please contact us if you are interested!

❷ randomized search heuristics in the discrete domain

Reminder: Discrete Optimization

Context discrete optimization:

- discrete variables
- or optimization over discrete structures (e.g. graphs)
- search space often finite, but typically too large for enumeration
- → need for smart algorithms

Algorithms for discrete problems:

- typically problem-specific
- but some general concepts are repeatedly used:
  - greedy algorithms
  - branch and bound
  - dynamic programming

  } seen in introductory lecture

  - randomized search heuristics } now

Remark: Coping with Difficult Problems

Exact

- brute-force often too slow
- better strategies such as dynamic programming & branch and bound
- still: often exponential runtime

Approximation Algorithms

- guarantee of low run time
- guarantee of high quality solution
- obstacle: difficult to prove these guarantees

Heuristics

- intuitive algorithms
- guarantee to run in short time
- often no guarantees on solution quality

Motivation General Search Heuristics

- often, problem complicated and not much time available to develop a problem-specific algorithm
- search heuristics are a good choice:
  - relatively easy to implement
  - easy to adapt/change/improve
    - e.g. when the problem formulation changes in an early product design phase
    - or when slightly different problems need to be solved over time
    - remember blackbox scenario
- search heuristics are also often "any-time", i.e. give a feasible solution early on which is then improved throughout the algorithm run → might be important in practice

Lecture Outline Randomized Search Heuristics

Which algorithms will we touch?

1. Randomized Local Search (RLS)
2. Variable Neighborhood Search (VNS)
3. Tabu Search (TS)
4. Evolutionary Algorithms (EAs)

Neighborhoods

For most (stochastic) search heuristics, we need to define a *neighborhood structure*

- which search points are close to each other?

Example: k-bit flip / Hamming distance k neighborhood

- search space: bitstrings of length n ($\Omega=\{0,1\}^n$)
- two search points are neighbors if their Hamming distance is k
- in other words: x and y are neighbors if we can flip exactly k bits in x to obtain y
- 0001001101 is neighbor of
  0001000101 for k=1
  0101000101 for k=2
  1101000101 for k=3

Neighborhoods II

Example: possible neighborhoods for the knapsack problem

- search space again bitstrings of length n ($\Omega=\{0,1\}^n$)
- Hamming distance 1 neighborhood:
  - add an item or remove it from the packing
- replacing 2 items neighborhood:
  - replace one chosen item with an unchosen one
  - makes only sense in combination with other neighborhoods because the number items stays constant
- Hamming distance 2 neighborhood on the contrary:
  - allows to change 2 arbitrary items, e.g.
    - add 2 new items
    - remove 2 chosen items
    - or replace one chosen item with an unchosen one

Randomized Local Search (RLS)

Idea behind (Randomized) Local Search:

- explore the local neighborhood of the current solution (randomly)

Pure Random Search:

- go to randomly chosen neighbor

First Improvement Local Search:

- go to first (randomly) chosen neighbor which is better

Best Improvement strategy:

- always go to the best neighbor
- not random anymore
- computationally expensive if neighborhood large

Variable Neighborhood Search

Main Idea: [Mladenovic and P. Hansen, 1997]

- change the neighborhood from time to time
  - local optima are not the same for different neighborhood operators
  - but often close to each other
  - global optimum is local optimum for all neighborhoods
- rather a framework than a concrete algorithm
  - e.g. deterministic and stochastic neighborhood changes
- typically combined with (i) first improvement, (ii) a random order in which the neighbors are visited and (iii) restarts

N. Mladenovic and P. Hansen (1997). "Variable neighborhood search". Computers and Operations Research 24 (11): 1097–1100.

Tabu Search

Disadvantages of local searches (with or without varying neighborhoods)

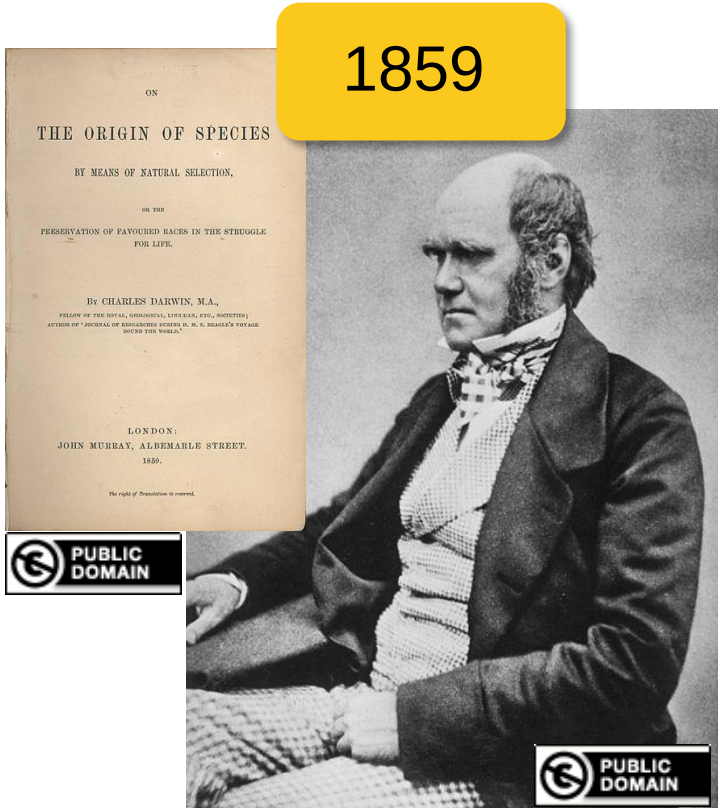
- they get stuck in local optima
- have problems to traverse large plateaus of equal objective function value ("random walk")

Tabu search addresses these by

- allowing worsening moves if all neighbors are explored
- introducing a tabu list of temporarily not allowed moves
- those restricted moves are
  - problem-specific and
  - can be specific solutions or not permitted "search directions" such as "don't include this edge anymore" or "do not flip this specific bit"
- the tabu list is typically restricted in size and after a while, restricted moves are permitted again

Stochastic Optimization Algorithms

One class of (bio-inspired) stochastic optimization algorithms: Evolutionary Algorithms (EAs)

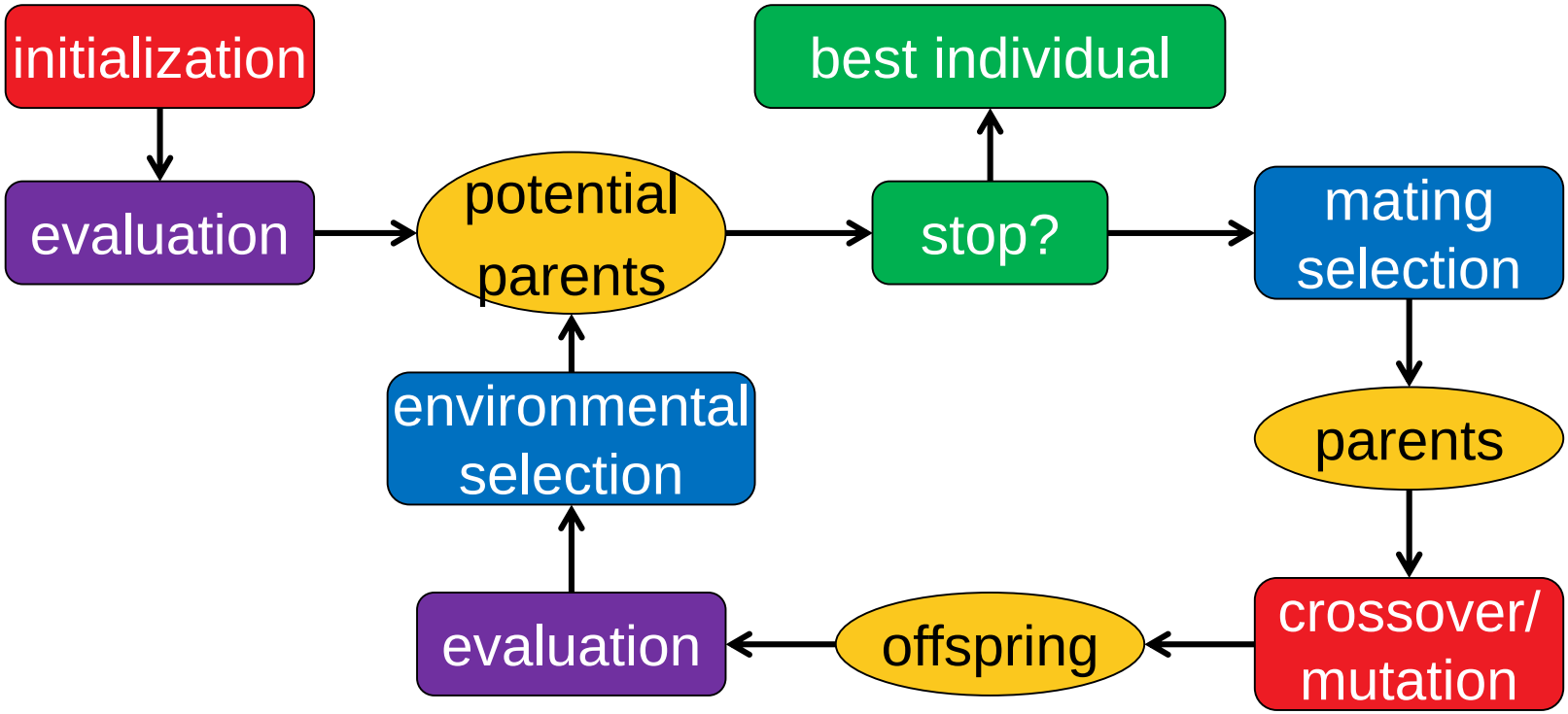


- Class of optimization algorithms originally inspired by the idea of biological evolution
- selection, mutation, recombination

Metaphors

Classical Optimization	Evolutionary Computation
variables or parameters	variables or chromosomes
candidate solution vector of decision variables / design variables / object variables	individual, offspring, parent
set of candidate solutions	population
objective function loss function cost function error function	fitness function
iteration	generation

Generic Framework of an EA



stochastic operators

"Darwinism"

stopping criteria

Important:
representation (search space)

The Historic Roots of EAs

Genetic Algorithms (GA)

J. Holland 1975 and D. Goldberg (USA)

$$\Omega = \{0,1\}^n$$

Evolution Strategies (ES)

I. Rechenberg and H.P. Schwefel, 1965 (Berlin)

$$\Omega = \mathbb{R}^n$$

Evolutionary Programming (EP)

L.J. Fogel 1966 (USA)

Genetic Programming (GP)

J. Koza 1990 (USA)

$$\Omega = \text{space of all programs}$$

nowadays one umbrella term: evolutionary algorithms

Genotype – Phenotype mapping

The genotype – phenotype mapping

- related to the question: how to come up with a fitness of each individual from the representation?
- related to DNA vs. actual animal (which then has a fitness)

fitness of an individual not always = f(x)

- include constraints
- include diversity
- others
- but needed: always a total order on the solutions

Handling Constraints

Several possible ways to handle constraints, e.g.:

- resampling until a new feasible point is found ("often bad idea")
- penalty function approach: add constraint violation term (potentially scaled)
- repair approach: after generation of a new point, repair it (e.g. with a heuristic) to become feasible again if infeasible
  - continue to use repaired solution in the population or
  - use repaired solution only for the evaluation?
- multiobjective approach: keep objective function and constraint functions separate and try to optimize all of them in parallel
- many more...

Examples for some EA parts

Selection

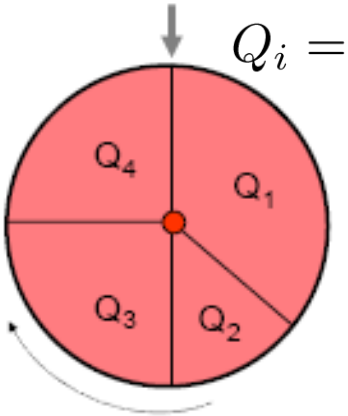
Selection is the major determinant for specifying the trade-off between exploitation and exploration

Selection is either

stochastic or deterministic

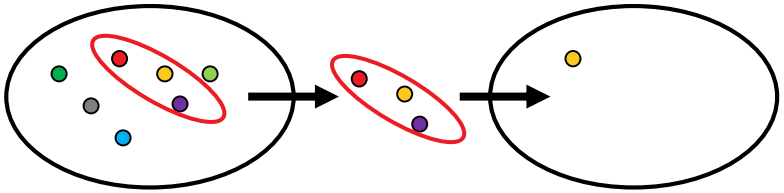
e.g. fitness proportional

$$Q_i = \frac{f(x_i)}{\sum_{j=1}^{\mu} f(x_j)}$$



Disadvantage: depends on scaling of f

e.g. via a tournament



e.g. (μ+λ), (μ,λ)

best μ from offspring *and* parents

best μ from offspring only

Mating selection (selection for variation): usually stochastic

Environmental selection (selection for survival): often deterministic

Variation Operators

Variation aims at generating new individuals on the basis of those individuals selected for mating

Variation = Mutation and Recombination/Crossover

mutation: *mut:* $\Omega \rightarrow \Omega$

recombination: *recomb:* $\Omega^r \rightarrow \Omega^s$ where $r \geq 2$ and $s \geq 1$

- choice always depends on the problem and the chosen representation
- however, there are some operators that are applicable to a wide range of problems and tailored to standard representations such as vectors, permutations, trees, etc.

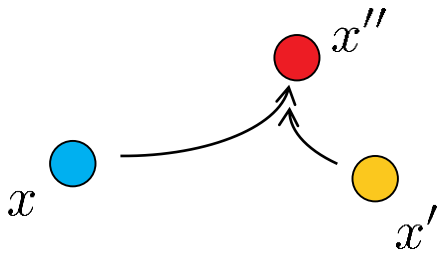
Variation Operators: Guidelines

Two desirable properties for mutation operators:

- every solution can be generation from every other with a probability greater than 0 (“exhaustiveness”)
- $d(x, x') < d(x, x'') => Prob(\mathrm{mut}(x) = x') > Prob(\mathrm{mut}(x) = x'')$ (“locality”)

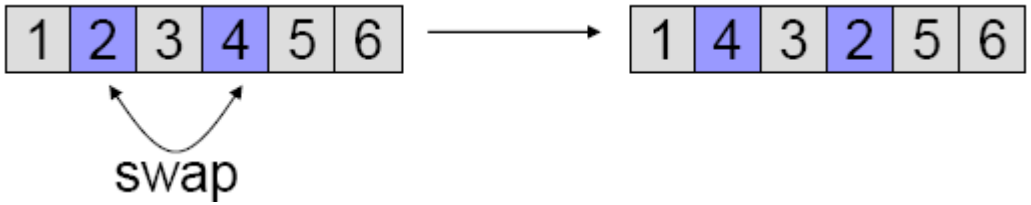
Desirable property of recombination operators (“in-between-ness”):

$$x'' = \mathrm{recomb}(x, x') \Rightarrow d(x'', x) \leq d(x, x') \wedge d(x'', x') \leq d(x, x')$$

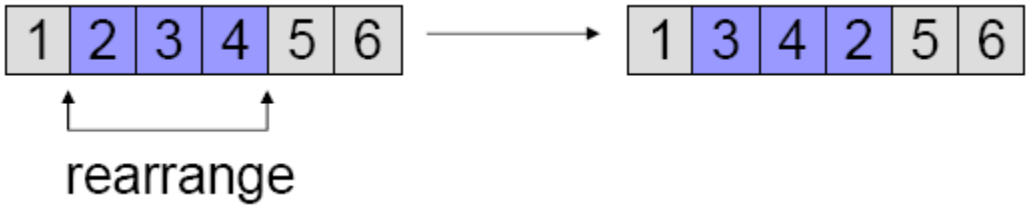


Examples of Mutation Operators on Permutations

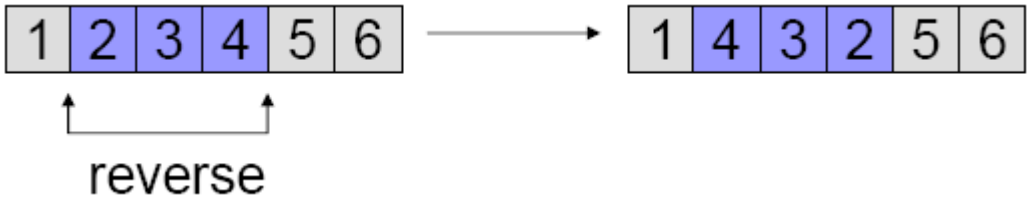
Swap:



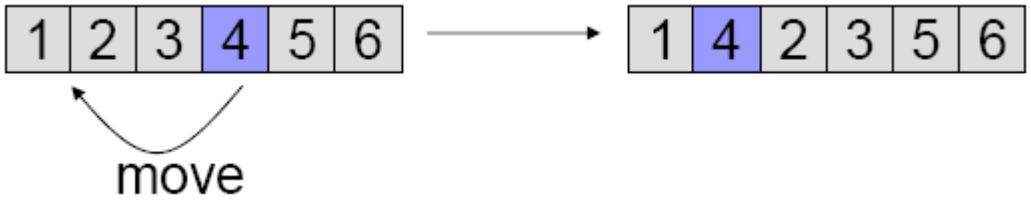
Scramble:



Invert:

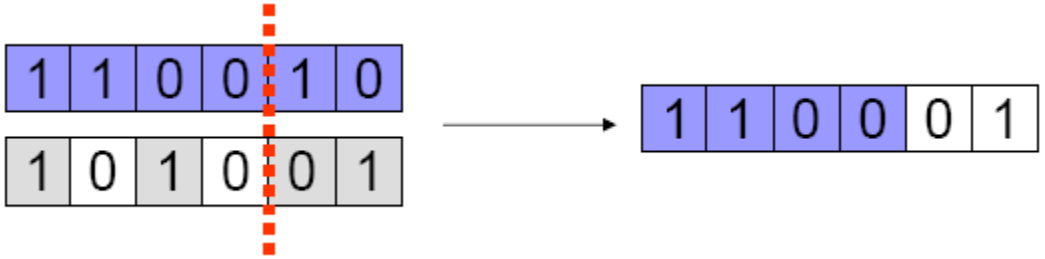


Insert:

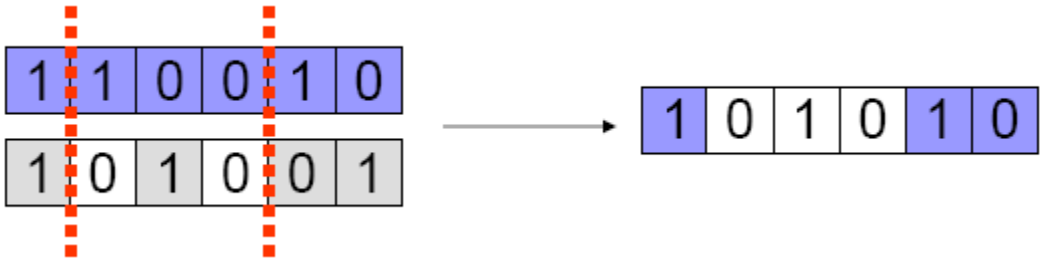


Examples of Recombination Operators: $\{0,1\}^n$

1-point crossover



n-point crossover



uniform crossover

choose each bit independently from one parent or another

A Canonical Genetic Algorithm

- binary search space, maximization
- uniform initialization
- generational cycle: of the population
  - evaluation of solutions
  - mating selection (e.g. roulette wheel)
  - crossover (e.g. 1-point)
  - environmental selection (e.g. plus-selection)

Conclusions

- EAs are generic algorithms (randomized search heuristics, meta-heuristics, ...) for black box optimization

  no or almost no assumptions on the objective function

- They are typically less efficient than problem-specific (exact) algorithms (in terms of #funevals)

  not the case in the continuous case (we will see later)

- Allow for an easy and rapid implementation and therefore to find good solutions fast

  easy to incorporate (and recommended!) to incorporate problem-specific knowledge to improve the algorithm

❸ exercise: Pure Random Search (PRS) and the (1+1)EA

Exercise:
Pure Random Search
and the (1+1)EA

`http://researchers.lille.inria.fr/~brockhof/advancedOptSaclay/2016/`

Conclusions

I hope it became clear...

...that heuristics is what we typically can afford in practice (no guarantees and no proofs)

...what are the main ideas behind evolutionary algorithms

...and that evolutionary algorithms and genetic algorithms are no synonyms