Advanced Optimization
Lecture 1: Discrete Optimization

November 27, 2019

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	Wed, 27.11.2019	Dimo	Randomized Algorithms for Discrete Problems
2	Wed, 4.12.2019	Dimo	Exercise: The Travelling Salesperson Problem
3	Wed, 11.12.2019	Dimo	Evolutionary Multiobjective Optimization I
4	Mon, 16.12.2019	Dimo	Evolutionary Multiobjective Optimization II
5	Wed, 18.12.2019	Dimo	Looking at Data
	Vacation		
6	Wed, 8.1.2020 (morning!)	Anne	Continuous Optimization I
7	Wed, 22.1.2020 (morning!)	Anne	Continuous Optimization II
	Wed, 5.2.2020		oral presentations (individual time slots)

No Exam...

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

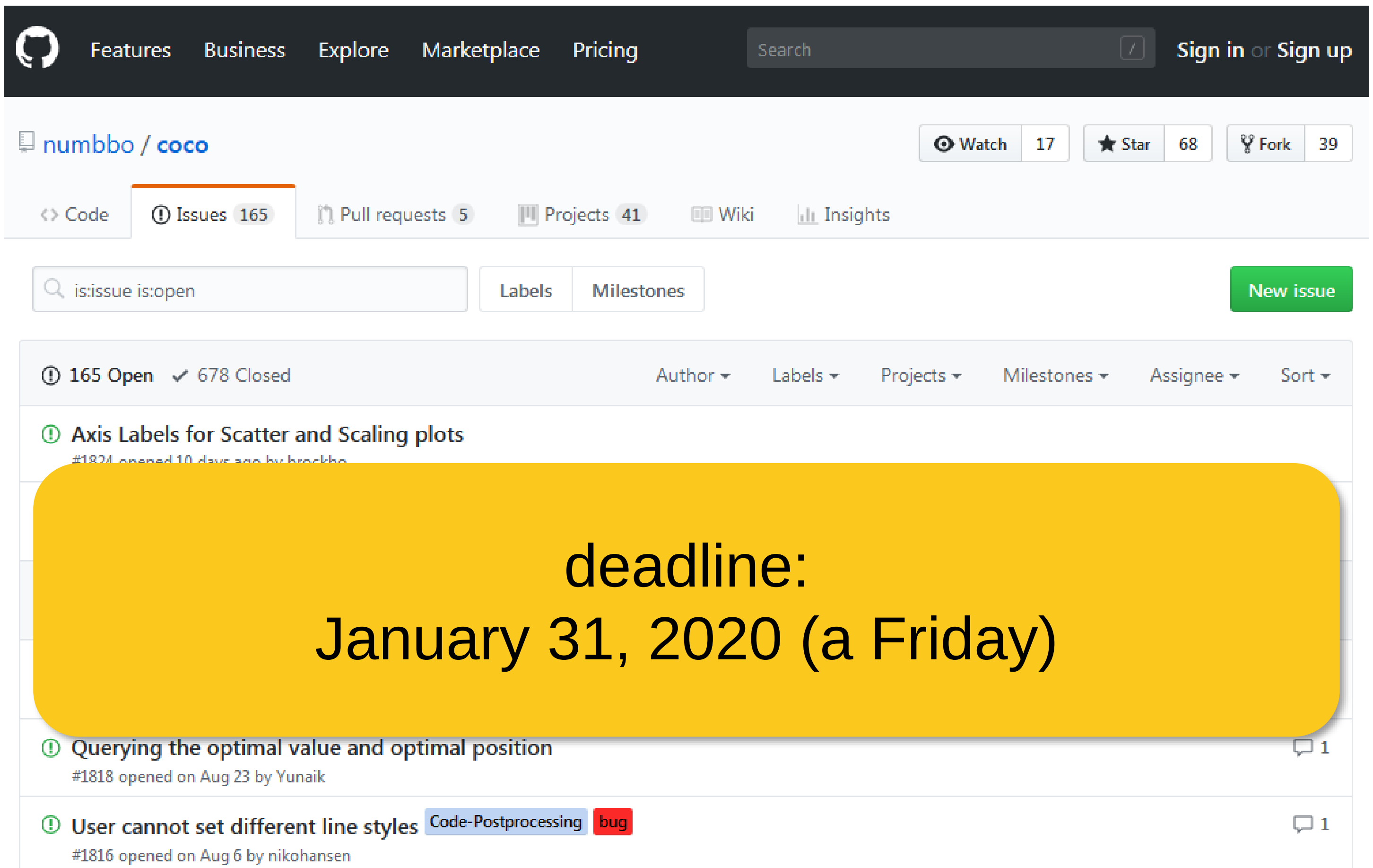
- we don't have a written exam ☺
- but instead work towards research:
  - each student is assigned a scientific paper
  - which is to be read, understood, critically questioned, and finally presented
  - summarize the paper in a short abstract in your own words
  - oral presentations in the end of the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

  - 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/



deadline:
January 31, 2020 (a Friday)

Details on the Paper Project/Oral Presentation

- no written exam but instead each student is assigned a scientific paper (list online and on next slide)
  - to be read, understood, critically questioned, and presented
  - maximal 2 students per paper
  - decision made until *December 4, 2019 (next lecture)*
- summary of the paper in a short abstract in your own words
  - handed in via email until *January 15, 2020, 23h59*
  - 4000 characters max. (please check before submission)
- individual oral presentations at the end of the course
  - 15min presentation + 15min oral "exam"
  - on February 5, 2020, times to be decided with you towards the end of the lecture
  - slides to be sent by email to us until last lecture (*Jan. 22, 2020*)

The List of Papers

All papers are relevant to current research in Randopt.

1) Two-dimensional subset selection for hypervolume and epsilon-indicator

2) RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm.

3) A universal catalyst for first-order optimization.

4) Optimized Approximation Sets for Low-Dimensional Benchmark Pareto Fronts.

5) Efficient optimization of many objectives by approximation-guided evolution.

6) A Mean-Variance Optimization Algorithm.

7) Theoretical foundation for CMA-ES from information geometry perspective.

8) Population Size Adaptation for the CMA-ES Based on the Estimation Accuracy of the Natural Gradient.

9) CMA-ES with Optimal Covariance Update and Storage Complexity.

10) Challenges of Convex Quadratic Bi-objective Benchmark Problems

links to the papers available on the lecture webpage at

`http://www.cmap.polytechnique.fr/~dimo.brockhoff/advancedOptSaclay/2019/paperproject.php`

Today's Lecture

❶ present open projects

❷ randomized search heuristics in the discrete domain

❸ 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

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)

Previous Announcements

- Adaptive Stochastic Search Algorithms for Constrained Optimization (Master's thesis project)
- Data Mining Performance Results of Numerical Optimizers (Master's thesis 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

Potential Research Topics for Master's/PhD Theses

Projects without the involvement of companies:

- stopping criteria in multiobjective optimization
- mixed-integer CMA-ES

all above: relatively flexible between theoretical and practical projects

Coco-related:

- implementing and benchmarking existing algorithms
  [new test suites for constraints & mixed-integer available]
- recommendations for noisy optimization

not all subject ideas online:
please contact us if you are interested!

❷ randomized search heuristics in the discrete domain

[mainly what we couldn’t do in the introductory lecture]

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
  - dynamic programming
  - randomized search heuristics
  - branch and bound

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

Reminder: Stochastic Search Template

A stochastic blackbox search template to minimize $f: \Omega \to \mathbb{R}$

Initialize distribution parameters θ, set population size $\lambda \in \mathbb{N}$

While happy do:

- Sample distribution $P(\boldsymbol{x}|\theta) \to \boldsymbol{x}_1, \dots, \boldsymbol{x}_\lambda \in \Omega$
- Evaluate $\boldsymbol{x}_1, \dots, \boldsymbol{x}_\lambda$ on f
- Update parameters $\theta \leftarrow F_\theta(\theta, \boldsymbol{x}_1, \dots, \boldsymbol{x}_\lambda, f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_\lambda))$

Deterministic algorithms can be cast in this framework as well:

for example in $\mathbb{R}^n$: gradient descent

or local search in discrete Ω

well-known stochastic example:

Covariance Matrix Adaptation Evolution Strategy (CMA-ES):

sample distributions = multivariate Gaussian distributions

Lecture Outline Randomized Search Heuristics

Here, we touch algorithms for discrete Ω

❶ Randomized Local Search (RLS)

❷ Evolutionary Algorithms (EAs)

❸ Compact GA: an estimation of distribution algorithm on bitstrings

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

Mini-Exercise: Neighborhood Sizes

- What are the neighborhood sizes for a Hamming distance of k?
- or: how many solutions are a k-bit flip away?

Neighborhoods II

Example: neighborhoods for permutation problems

- search space: all permutations of length n ($\Omega = S_n$)
- swap neighborhood:
  - swap two entries in the permutation
- equivalence to Hamming distance: swap distance
  - allow to swap k pairs
  - possible to sample in a given distance of k, but algorithm is not trivial
- more neighborhoods for permutations later

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:

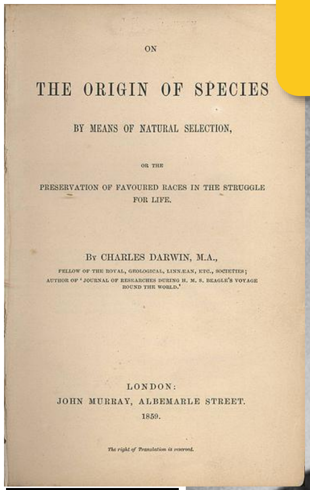
- go to first (randomly) chosen neighbor which is better

Best Improvement strategy:

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

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

1859


ON

THE ORIGIN OF SPECIES

BY MEANS OF NATURAL SELECTION,

OR THE

PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE.

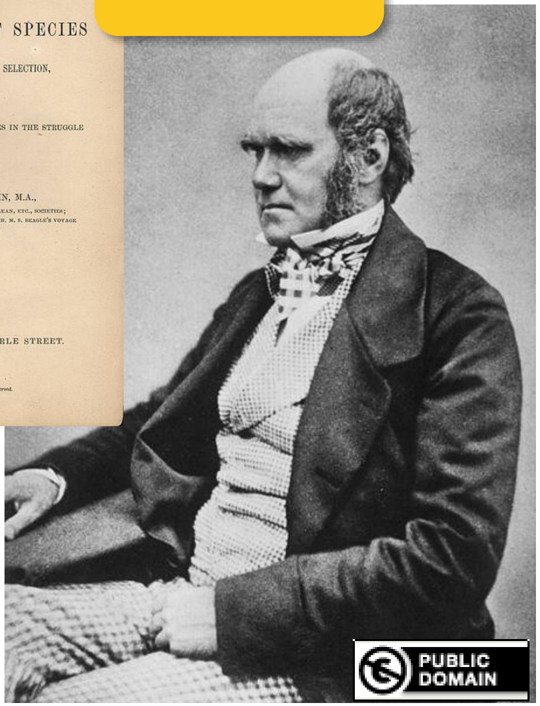
By CHARLES DARWIN, M.A.,

FELLOW OF THE ROYAL, GEOLOGICAL, LINNÆAN, ETC., SOCIETIES;
AUTHOR OF 'JOURNAL OF RESEARCHES DURING H. M. S. BEAGLE'S VOYAGE ROUND THE WORLD.'

LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.

The right of Translation is reserved.

PUBLIC DOMAIN

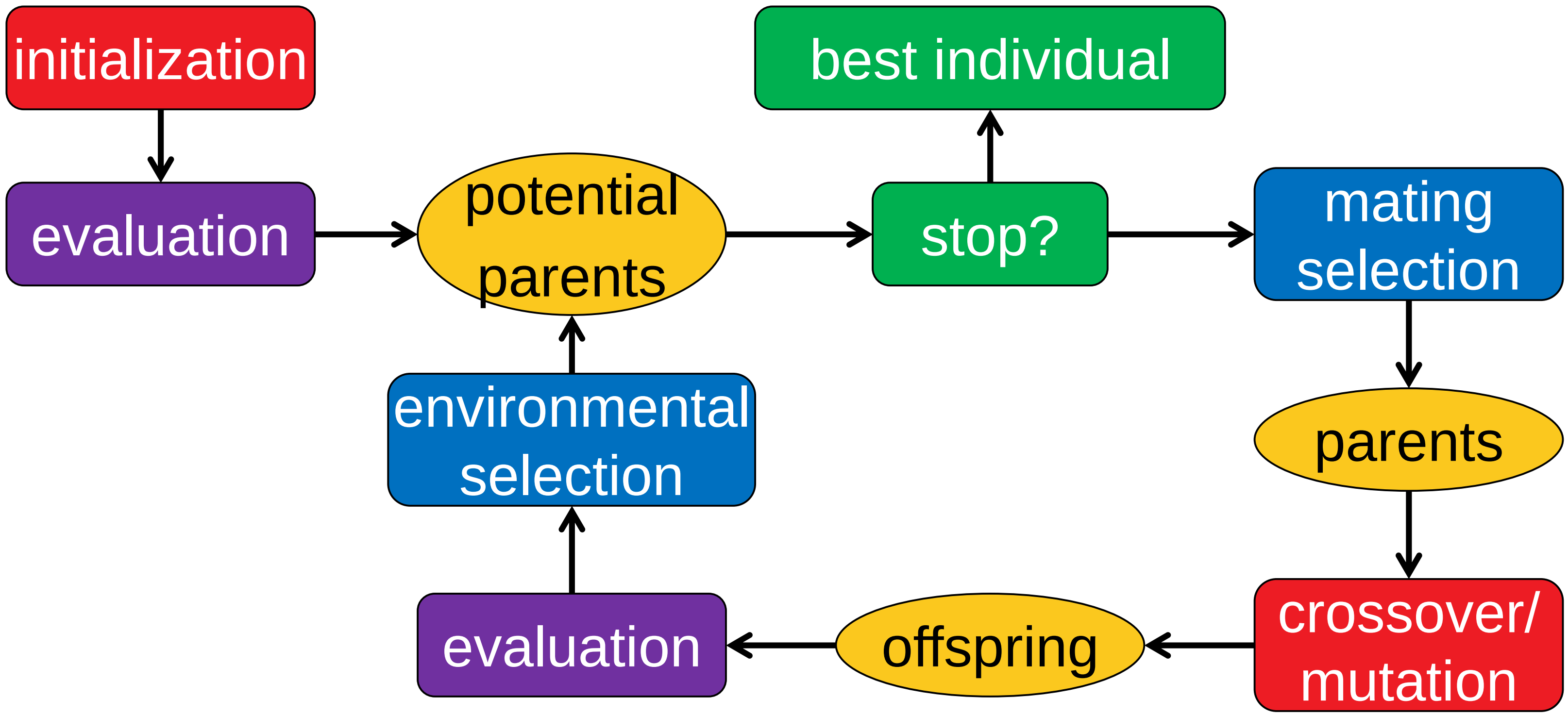


- Class of optimization algorithms originally inspired by the idea of biological evolution
- selection, mutation, recombination

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 ("quality") 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

Examples for some EA parts

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

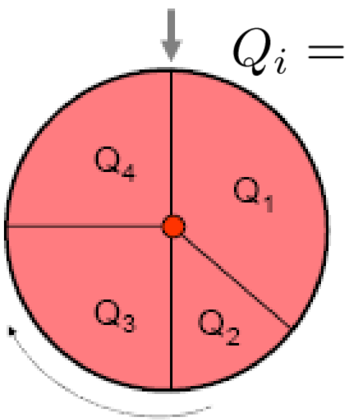
Selection is either

stochastic or deterministic

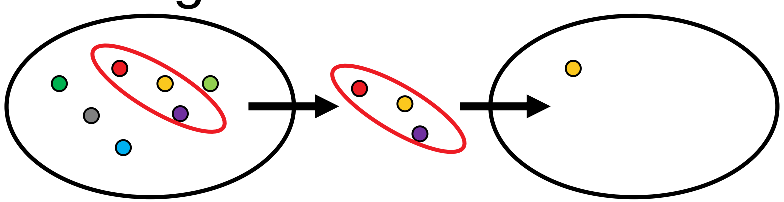
e.g. fitness proportional

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

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.

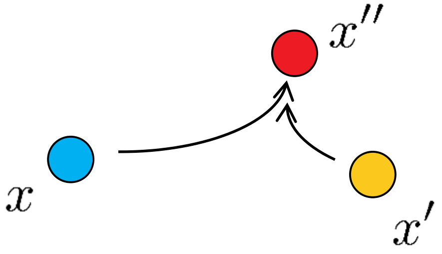
Variation Operators: Guidelines

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 $\{0,1\}^n$

1-bit flip mutation

- flip a randomly chosen bit (from 1 to 0 or vice versa)

k-bit flip mutation

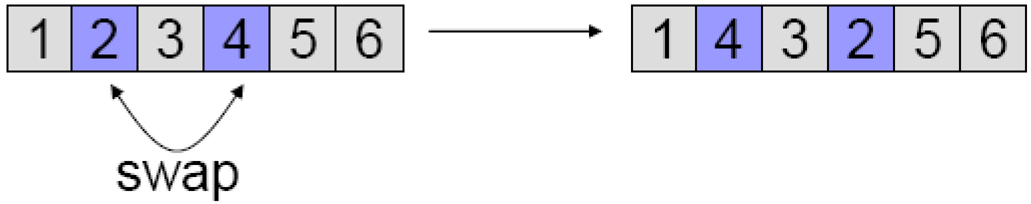
- choose k (different) bits uniformly at random
- flip each of those bits (from 1 to 0 or vice versa)

Standard bitflip mutation

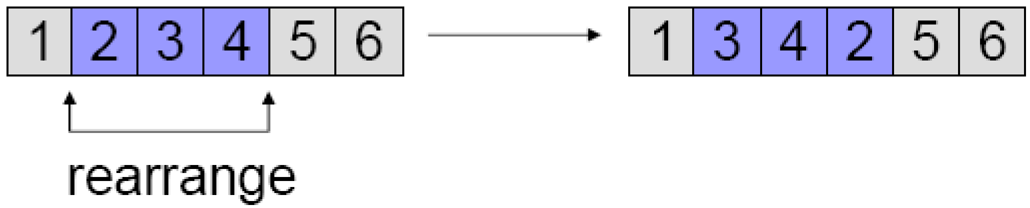
- flip each bit independently with probability 1/n
- expected number of bits changed: 1
- but also: $\lim_{n \to \pm\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0.367879$ i.e. no bit flipped with constant probability

Examples of Mutation Operators on Permutations

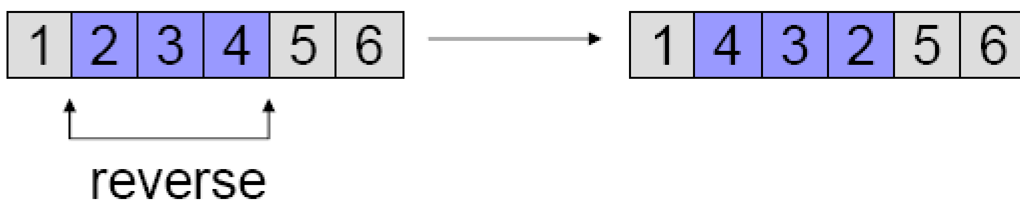
Swap:



Scramble:

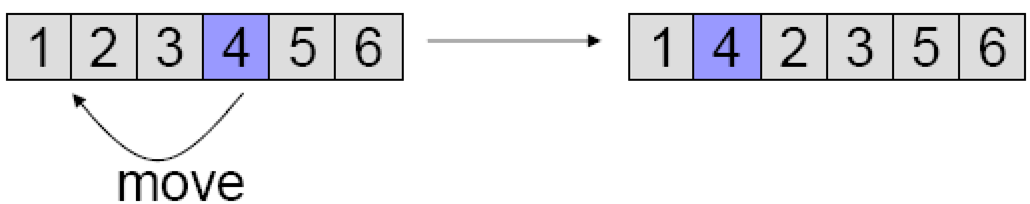


Invert:



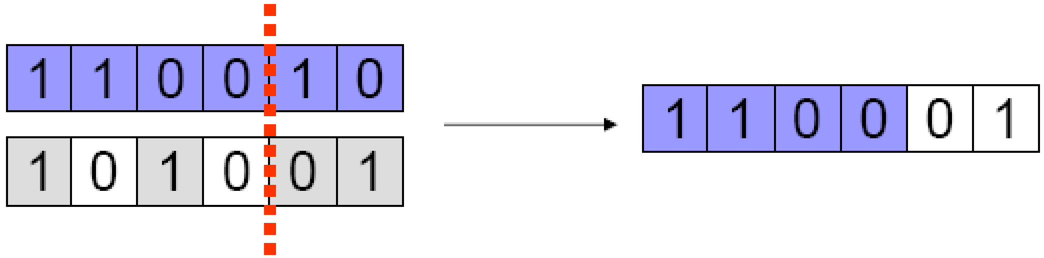
also known as 2-opt

Insert:

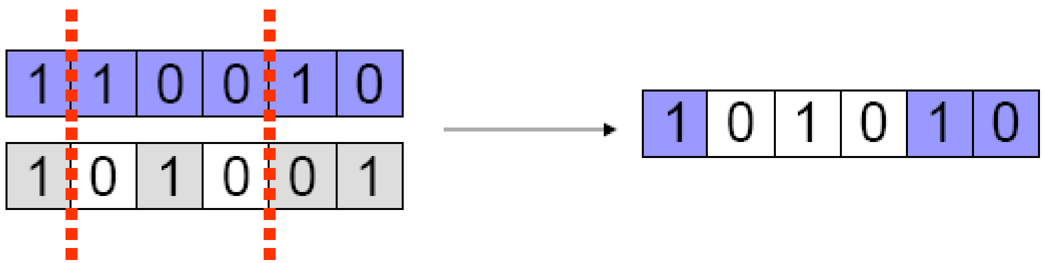


Examples of Recombination Operators on $\{0,1\}^n$

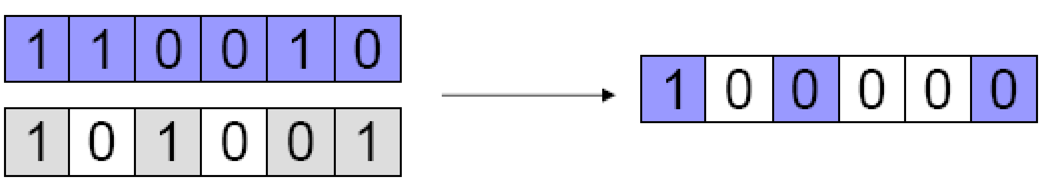
1-point crossover



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:
  - evaluation of solutions
  - mating selection (e.g. roulette wheel)
  - crossover (e.g. 1-point)
  - environmental selection (e.g. plus-selection)

You may ask: how does this fit
into the stochastic search template?
it does: population contained in state θ,
but update function difficult to write down

Estimation of Distribution Algorithms

- Estimation of Distribution Algorithms (EDAs) fit more obviously into the search template
- here, example of the compact Genetic Algorithm (cGA)
  - search space: $\Omega = \{0,1\}^n$
  - probability distribution: Bernoulli
    - store for each bit a probability p_i to sample a 1
    - sample bit i with probability p_i to 1 and with $(1 - p_i)$ to 0

The Compact GA

Parameters: number of variables n, learning rate K (typically $= n$)

Init:

$p = \left(\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2}\right) \in [0,1]^n$ # probabilities to sample new solutions

While happy:

create $S = (s_1, \dots, s_n)$ by sampling each s_i with probability p_i

create $S' = (s'_1, \dots, s'_n)$ by sampling each s'_i with probability p_i

evaluate S and S' on f

if $f(S) > f(S')$: # make sure that S is the better solution

$S, S' \leftarrow S', S$

update p parameter:

for $i \in \{1, \dots, n\}$:

$p_i \leftarrow \min\{\max\{p_i + (s_i - s'_i)/K, 1/n\}, 1 - 1/n\}$

return S

Conclusions

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

  no or almost no assumptions on the objective function

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

  we will see this next week

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

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

The Simple(st) Evolutionary Algorithm(s)

Assumptions:

- search space Ω = set of all bitstrings of length n ($\Omega = \{0,1\}^n$)
- minimization of objective function $f: \{0,1\}^n \to \mathbb{R}$

Algorithm:

- init: sample a point x uniformly at random in Ω
- while not happy:
  - $y \leftarrow \text{mutate}(x)$
  - if $f(y) \leq f(x)$:
    - $x \leftarrow y$

Variants:

- Randomized Local Search (RLS): mutate = sample a Hamming neighbor uniformly at random (= flip exactly one bit)
- (1+1)-EA: mutate = flip each bit with probability $1/n$

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

http://www.cmap.polytechnique.fr/~dimo.brockhoff/advancedOptSaclay/2019/exercises.php

FrEAK

If you want to play around a bit with these algorithms:

- https://sourceforge.net/projects/freak427/