Computational Intelligence

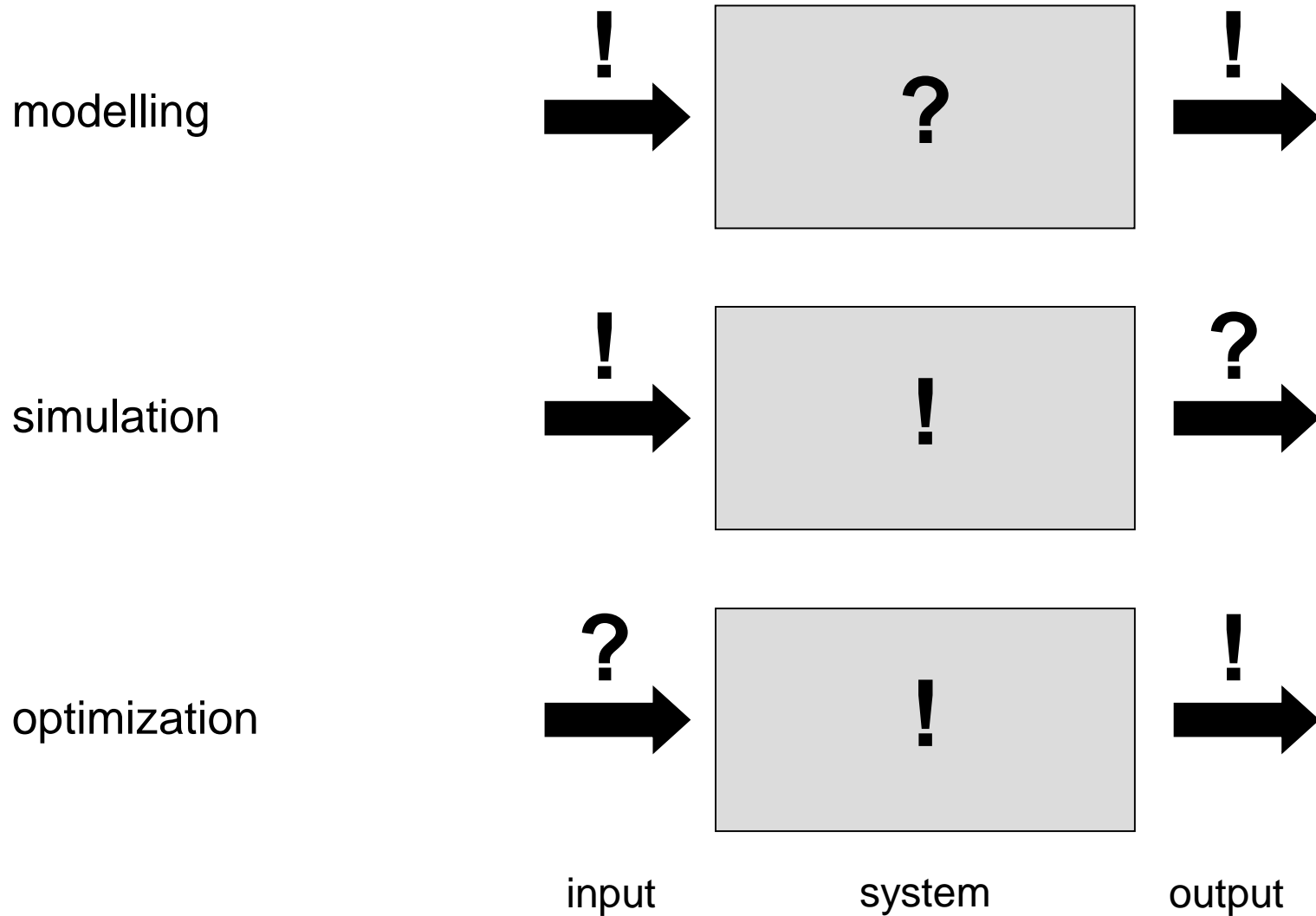**Winter Term 2019/20**

Prof. Dr. Günter Rudolph

Lehrstuhl für Algorithm Engineering (LS 11)

Fakultät für Informatik

TU Dortmund

- Evolutionary Algorithms (EA)

  - Optimization Basics

  - EA Basics

modelling

simulation

optimization

input   system   output

technische universität
dortmund

given:

objective function f: $X \to \mathbb{R}$

feasible region X (= nonempty set)

objective: find solution with *minimal* or *maximal* value!

optimization problem:

find $x^* \in X$ such that $f(x^*) = \min\{ f(x) : x \in X \}$

x^*	**global solution**
$f(x^*)$	**global optimum**

note:

$\max\{ f(x) : x \in X \} = - \min\{ - f(x) : x \in X \}$

technische universität
dortmund

local solution $x^* \in X$:

$\forall x \in N(x^*): f(x^*) \leq f(x)$

if x^* local solution then

$f(x^*)$ **local optimum / minimum**

neighborhood of x^* =
bounded subset of X

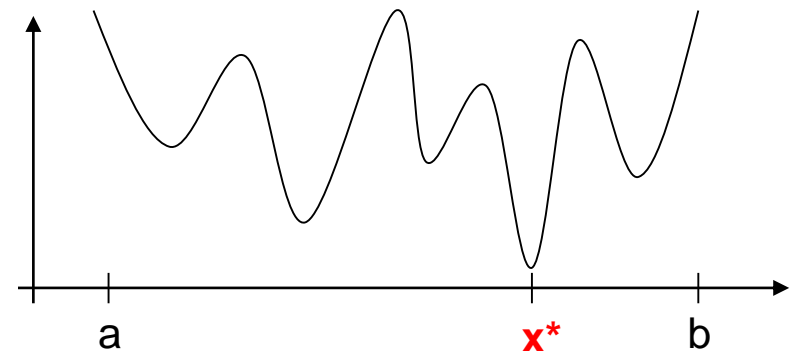example: $X = \mathbb{R}^n$, $N_\varepsilon(x^*) = \{ x \in X: \| x - x^* \|_2 \leq \varepsilon \}$

remark:

evidently, every global solution / optimum is also local solution / optimum;

the reverse is wrong in general!

example:

f: $[a,b] \to \mathbb{R}$, global solution at **x***

technische universität
dortmund

What makes optimization difficult?

<u>some causes:</u>

• local optima (is it a global optimum or not?)

• constraints (ill-shaped feasible region)

• non-smoothness (weak causality) $\longrightarrow$ strong causality needed!

• discontinuities ($\Rightarrow$ nondifferentiability, no gradients)

• lack of knowledge about problem ($\Rightarrow$ black / gray box optimization)

$f(x) = a_1 x_1 + ... + a_n x_n \rightarrow$ max! with $x_i \in \{0,1\}$, $a_i \in \mathbb{R}$

$\Rightarrow x_i^* = 1$ iff $a_i > 0$

add constaint $g(x) = b_1 x_1 + ... + b_n x_n \leq b$

$\Rightarrow$ NP-hard

add capacity constraint to TSP $\Rightarrow$ CVRP

$\Rightarrow$ still harder

technische universität
dortmund

When using which optimization method?

mathematical algorithms

- problem explicitly specified

- problem-specific solver available

- problem well understood

- ressources for designing algorithm affordable

- solution with proven quality required

⇒ **don't apply EAs**

randomized search heuristics

- problem given by black / gray box

- no problem-specific solver available

- problem poorly understood

- insufficient ressources for designing algorithm

- solution with satisfactory quality sufficient

⇒ **EAs worth a try**

<u>idea:</u> using **biological evolution** as **metaphor** and as **pool of inspiration**

$\Rightarrow$ <u>interpretation</u> of biological evolution as iterative method of improvement

feasible solution $x \in X = S_1 \times ... \times S_n$ = chromosome of **individual**

multiset of feasible solutions   = **population**: multiset of individuals

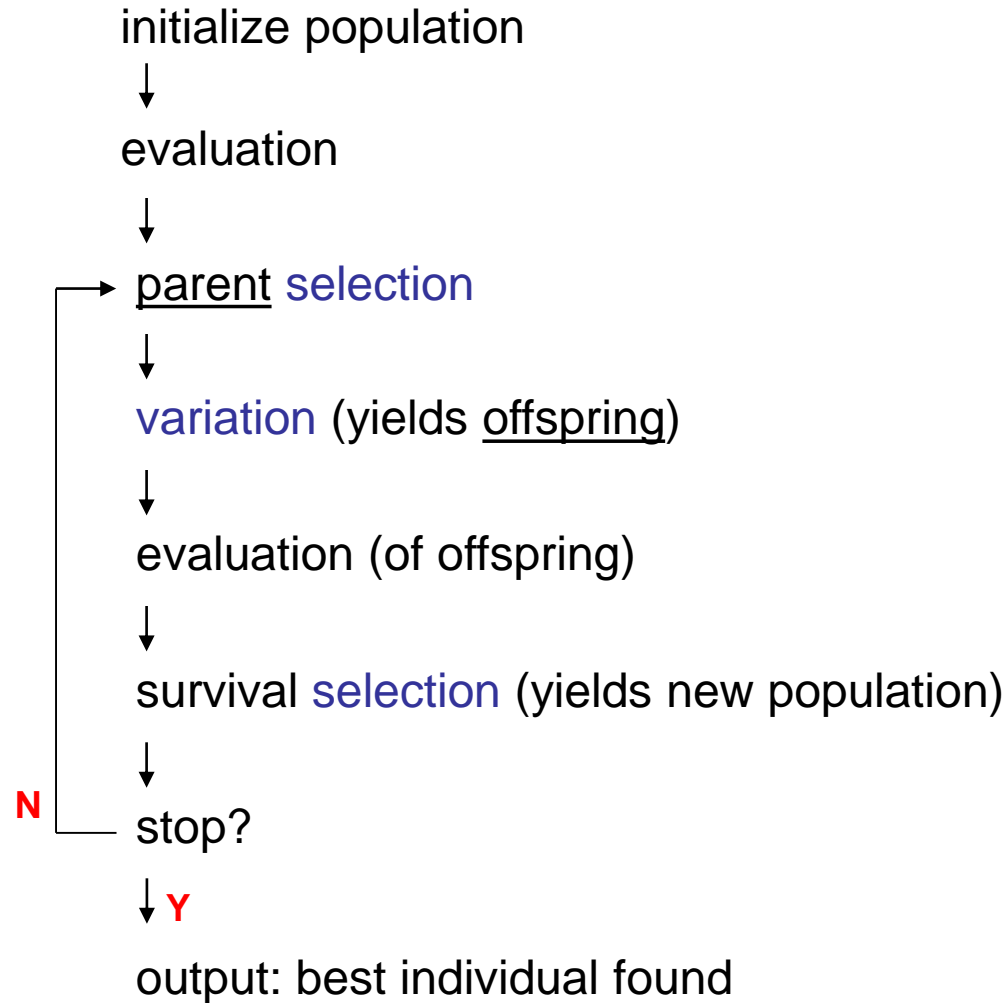**objective function** $f: X \rightarrow \mathbb{R}$ = **fitness function**

<u>often:</u> $X = \mathbb{R}^n$, $X = \mathbb{B}^n = \{0,1\}^n$, $X = \mathbb{P}_n = \{ \pi : \pi \text{ is permutation of } \{1,2,...,n\} \}$

<u>also :</u> combinations like $X = \mathbb{R}^n \times \mathbb{B}^p \times \mathbb{P}_q$ or non-cartesian sets

$\Rightarrow$ structure of feasible region / search space defines **representation** of individual

algorithmic
skeleton

initialize population
↓
evaluation
↓
<u>parent</u> selection
↓
variation (yields <u>offspring</u>)
↓
evaluation (of offspring)
↓
survival selection (yields new population)
↓
N stop?
↓ **Y**
output: best individual found

technische universität
dortmund

Specific example: (1+1)-EA in $\mathbb{B}^n$ for minimizing some f: $\mathbb{B}^n \rightarrow \mathbb{R}$

population size = 1, number of offspring = 1, selects best from 1+1 individuals

↑ ↑
parent offspring
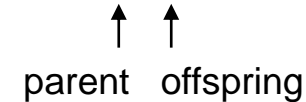
1. initialize $X^{(0)} \in \mathbb{B}^n$ uniformly at random, set t = 0

2. evaluate $f(X^{(t)})$

3. select parent: $Y = X^{(t)}$ ————————————————→   no choice, here

4. variation: flip each bit of Y independently with probability $p_m = 1/n$

5. evaluate $f(Y)$

6. selection: if $f(Y) \leq f(X^{(t)})$ then $X^{(t+1)} = Y$ else $X^{(t+1)} = X^{(t)}$

7. if not stopping then t = t+1, continue at (3)

Specific example: (1+1)-EA in $\mathbb{R}^n$ for minimizing some f: $\mathbb{R}^n \to \mathbb{R}$

population size = 1, number of offspring = 1, selects best from 1+1 individuals

↑   ↑
parent   offspring
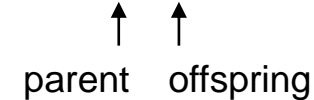
compact set = closed & bounded

1. initialize $X^{(0)} \in C \subset \mathbb{R}^n$ uniformly at random, set t = 0

2. evaluate $f(X^{(t)})$

3. select parent: $Y = X^{(t)}$ ————————————————→ no choice, here

4. variation = add random vector: Y = Y + Z,   e.g. $Z \sim N(0, I_n)$

5. evaluate f(Y)

6. selection: if $f(Y) \leq f(X^{(t)})$ then $X^{(t+1)} = Y$ else $X^{(t+1)} = X^{(t)}$

7. if not stopping then t = t+1, continue at (3)

technische universität
dortmund

Selection

(a) select parents that generate offspring → selection for **reproduction**

(b) select individuals that proceed to next generation → selection for **survival**

necessary requirements:

- selection steps must not favor worse individuals

- one selection step may be neutral (e.g. select uniformly at random)

- at least one selection step must favor better individuals

typically : selection only based on fitness values f(x) of individuals

seldom : additionally based on individuals' chromosomes x (→ maintain diversity)

Selection methods

population P = $(x_1, x_2, ..., x_\mu)$ with μ individuals

two approaches:

1. repeatedly select individuals from population with replacement

2. rank individuals somehow and choose those with best ranks (no replacement)

- ***uniform / neutral selection***
  choose index i with probability $1/\mu$

- ***fitness-proportional selection***
  choose index i with probability $s_i = \dfrac{f(x_i)}{\sum\limits_{x \in P} f(x)}$

  problems: $f(x) > 0$ for all $x \in X$ required $\Rightarrow$ $g(x) = \exp(f(x)) > 0$

  don't use!

  but already sensitive to additive shifts $g(x) = f(x) + c$

  almost deterministic if large differences, almost uniform if small differences

Selection methods

population $P = (x_1, x_2, ..., x_\mu)$ with μ individuals

- ***rank-proportional selection***
  order individuals according to their fitness values
  assign ranks
  fitness-proportional selection based on ranks

  $\Rightarrow$ avoids all problems of fitness-proportional selection
  but: best individual has only small selection advantage (can be lost!)

outdated!

- ***k-ary tournament selection***
  draw k individuals uniformly at random (typically with replacement) from P
  choose individual with best fitness (break ties at random)

  $\Rightarrow$ has all advantages of rank-based selection and
  probability that best individual does not survive: $\left(1 - \dfrac{1}{\mu}\right)^{k\,\mu} \begin{array}{l} < \ e^{-k} \\ \geq \ 4^{-k} \end{array}$

Selection methods without replacement

population $P = (x_1, x_2, ..., x_\mu)$ with μ parents and

population $Q = (y_1, y_2, ..., y_\lambda)$ with λ offspring

- **(μ, λ)-*selection*** or ***truncation selection on offspring*** or ***comma-selection***
  rank λ <u>offspring</u> according to their fitness
  select μ offspring with best ranks

$\Rightarrow$ best individual may get lost, $\lambda \geq \mu$ required

- **(μ+λ)-*selection*** or ***truncation selection on parents + offspring*** or ***plus-selection***
  merge λ <u>offspring</u> and μ <u>parents</u>
  rank them according to their fitness
  select μ individuals with best ranks

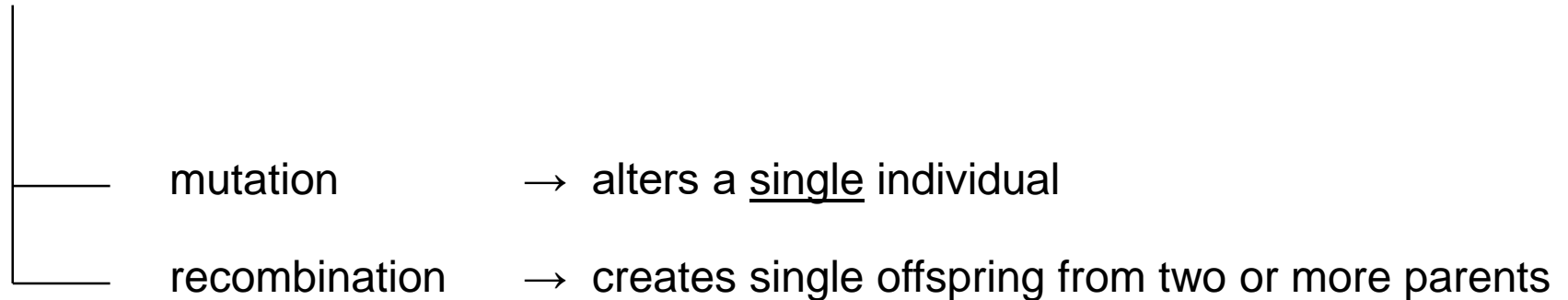$\Rightarrow$ best individual survives for sure

Selection methods: Elitism

Elitist selection: best parent is not replaced by worse individual.

- *Intrinsic elitism*: method selects from parent and offspring,
  best survives with probability 1

- *Forced elitism*: if best individual has not survived then re-injection into population,
  i.e., replace worst selected individual by previously best parent

method	P{ select best }	from parents & offspring	intrinsic elitism
neutral	< 1	no	no
fitness proportionate	< 1	no	no
rank proportionate	< 1	no	no
k-ary tournament	< 1	no	no
$(\mu + \lambda)$	= 1	yes	yes
(μ , λ)	= 1	no	no

technische universität
dortmund

Variation operators: depend on representation

          mutation   → alters a <u>single</u> individual

          recombination   → creates single offspring from two or more parents
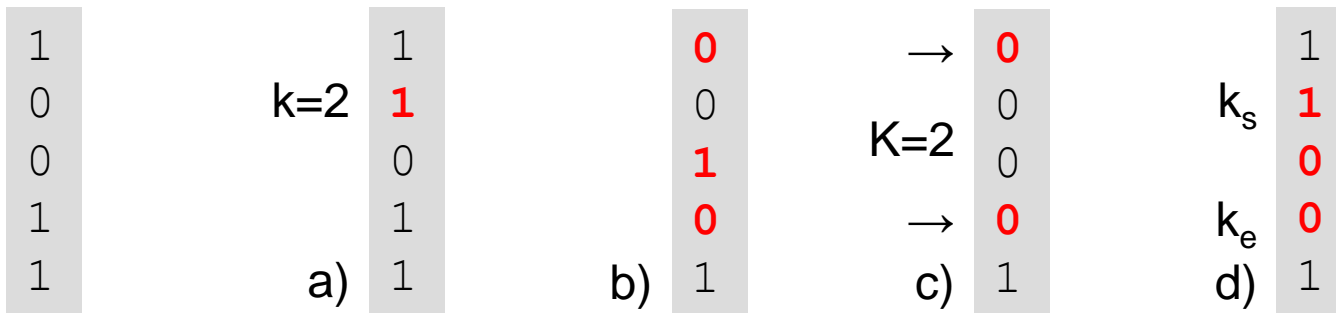
may be applied

- exclusively (either recombination or mutation) chosen in advance

- exclusively (either recombination or mutation) in probabilistic manner

- sequentially (typically, recombination before mutation); for each offspring

- sequentially (typically, recombination before mutation) with some probability

Variation in $\mathbb{B}^n$

Individuals $\in \{ 0, 1 \}^n$

● Mutation

a) local $\quad\rightarrow$ choose index $k \in \{ 1, …, n \}$ uniformly at random,
flip bit k, i.e., $x_k = 1 - x_k$

b) global $\quad\rightarrow$ for each index $k \in \{ 1, …, n \}$: flip bit k with probability $p_m \in (0,1)$

c) "nonlocal" $\quad\rightarrow$ choose K indices at random and flip bits with these indices

d) inversion $\quad\rightarrow$ choose start index k_s and end index k_e at random
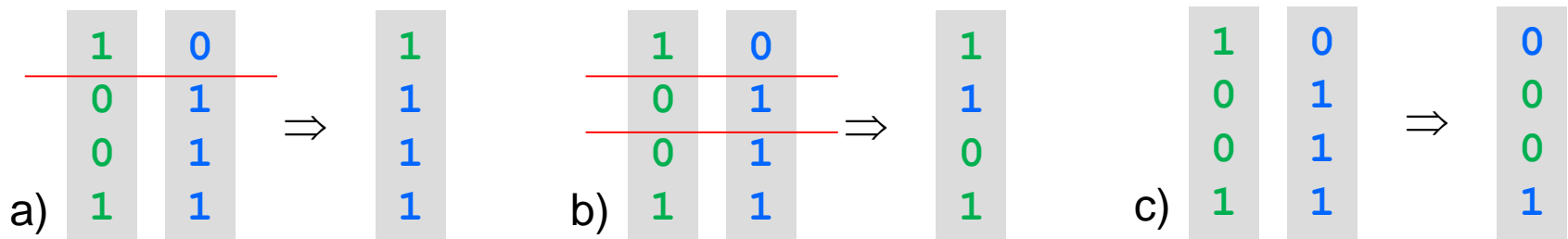invert order of bits between start and end index

| | | | | | | | | | |
|1| |1| |**0**|→|**0**| |1|
|0|k=2|**1**| |0| |0|k_s|**1**|
|0| |0| |**1**|K=2|0| |**0**|
|1| |1| |**0**|→|**0**|k_e|**0**|
|1|a)|1|b)|1|c)|1|d)|1|

Evolutionary Algorithm Basics | **Lecture 06**

Variation in $\mathbb{B}^n$

Individuals $\in \{ 0, 1 \}^n$

● Recombination (two parents)

a) 1-point crossover → draw cut-point $k \in \{1,\dots,n-1\}$ uniformly at random;
choose first k bits from 1st parent,
choose last $n-k$ bits from 2nd parent

b) K-point crossover → draw K distinct cut-points uniformly at random;
choose bits 1 to k_1 from 1st parent,
choose bits k_1+1 to k_2 from 2nd parent,
choose bits k_2+1 to k_3 from 1st parent, and so forth …

c) uniform crossover → for each index i: choose bit i with equal probability
from 1st or 2nd parent

G. Rudolph: Computational Intelligence ▪ Winter Term 2019/20

19

technische universität
dortmund

Variation in $\mathbb{B}^n$

Individuals $\in \{ 0, 1 \}^n$

● Recombination (multiparent: ρ = #parents)

a) diagonal crossover ($2 < \rho < n$)

→ choose $\rho - 1$ distinct cut points, select chunks from diagonals

AAAAAAAAAA   ABBBCCDDDD
BBBBBBBBBB   BCCCDDAAAA
CCCCCCCCCC   CDDDAABBBB
DDDDDDDDDD   DAAABBCCCC

can generate ρ offspring;
otherwise choose initial chunk
at random for single offspring

b) gene pool crossover ($\rho > 2$)

→ for each gene: choose donating parent uniformly at random

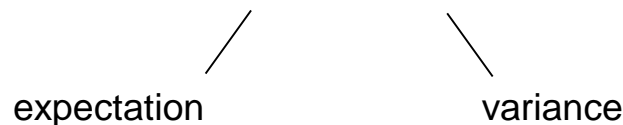**Variation** in $\mathbb{P}_n$ Individuals $\in$ X = π(1, …, n)

● Mutation

  a) local   → 2-swap   /   1-translocation

              5 3 2 4 1   5 3 2 4 1

              5 4 2 3 1   5 2 4 3 1

  b) global   → draw number K of 2-swaps, apply 2-swaps K times

              K is positive random variable;
              its distribution may be uniform, binomial, geometrical, …;
   E[K] and V[K] may control mutation strength

        expectation   variance

Variation in $\mathbb{P}_n$

Individuals $\in X = \pi(1, \ldots, n)$

- Recombination (two parents)

    a) order-based crossover (OBX)

    - select two indices k_1 and k_2 with $k_1 \leq k_2$ uniformly at random
    - copy genes k_1 to k_2 from 1st parent to offspring (keep positions)
    - copy genes from left to right from 2nd parent,
      starting after position k_2

2	3	5	7	1	6	4
6	4	5	3	7	2	1

x	x	x	7	1	6	x

5	3	2	7	1	6	4
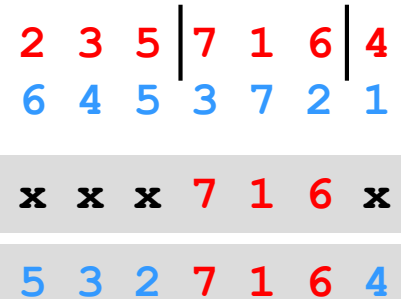
    b) partially mapped crossover (PMX) [a version of]

    - select two indices k_1 and k_2 with $k_1 \leq k_2$ uniformly at random
    - copy genes k_1 to k_2 from 1st parent to offspring (keep positions)
    - copy all genes not already contained in offspring from 2nd parent
      (keep positions)
    - from left to right: fill in remaining genes from 2nd parent

2	3	5	7	1	6	4
6	4	5	3	7	2	1

x	x	x	7	1	6	x

x	4	5	7	1	6	x

3	4	5	7	1	6	2

Variation in $\mathbb{P}_n$

Individuals $\in X = \pi(1, \ldots, n)$

● Recombination (two parents)

c) partially mapped crossover (PMX) [Grefenstette et al. 1985]

→ consider array as ring!

- given: 2 permutations **a** and **b** of length n
- select 2 indices k_1 and k_2 uniformly at random
- copy **b** to **c**
- procedure =

```
i = k1
repeat
  j = findIndex(a[i], c)
  swap(c[i], c[j])
  i = (i + 1) mod n
until i == k2
```

2	3	5	7	1	6	4
6	4	5	3	7	2	1

6	4	5	3	7	2	1

6	4	5	7	3	2	1

6	4	5	7	1	2	3

2	4	5	7	1	6	3

Variation in $\mathbb{R}^n$

Individuals $X \in \mathbb{R}^n$

● Mutation

additive:　　　　　Y = X + Z　　　　(Z: n-dimensional random vector)

offspring = parent + mutation

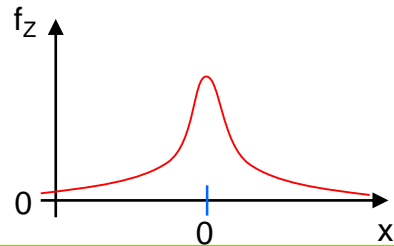a) local　　　　→ Z with bounded support
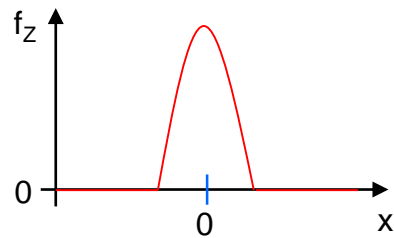
$$f_Z(x) = \frac{4}{3}\left(1 - x^2\right) \cdot 1_{[-1,1]}(x)$$

Definition
Let $f_Z : \mathbb{R}^n \to \mathbb{R}^+$ be p.d.f. of r.v. Z.
The set $\{\, x \in \mathbb{R}^n : f_Z(x) > 0 \,\}$ is termed the *support* of Z.

b) nonlocal　　　→ Z with unbounded support

$$f_Z(x) = \frac{1}{\sqrt{2\pi}}\,\exp\left(-\frac{x^2}{2}\right)$$

most frequently used!

Variation in $\mathbb{R}^n$ Individuals $X \in \mathbb{R}^n$

● Recombination (two parents)

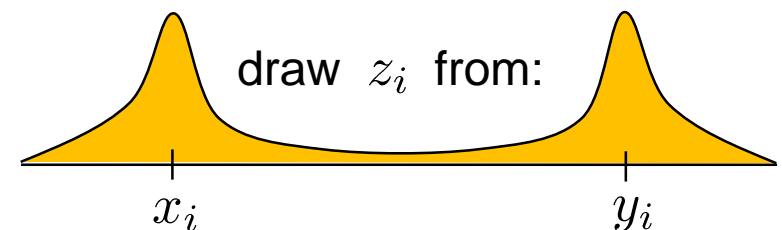a) all crossover variants adapted from $\mathbb{B}^n$

b) intermediate $z = \xi \cdot x + (1 - \xi) \cdot y$ with $\xi \in [0, 1]$

c) intermediate (per dimension) $\forall i : z_i = \xi_i \cdot x_i + (1 - \xi_i) \cdot y_i$ with $\xi_i \in [0, 1]$

d) discrete $\forall i : z_i = B_i \cdot x_i + (1 - B_i) \cdot y_i$ with $B_i \sim B(1, \frac{1}{2})$

e) simulated binary crossover (SBX)

$\rightarrow$ for each dimension with probability p_c draw z_i from:

x_i y_i

Variation in $\mathbb{R}^n$ Individuals $X \in \mathbb{R}^n$

● Recombination (multiparent), $\rho \geq 3$ parents

a) intermediate $z = \sum\limits_{k=1}^{\rho} \xi^{(k)} x_i^{(k)}$ where $\sum\limits_{k=1}^{\rho} \xi^{(k)} = 1$ and $\xi^{(k)} \geq 0$

      (all points in convex hull)

b) intermediate (per dimension) $\forall i : z_i = \sum\limits_{k=1}^{\rho} \xi_i^{(k)} x_i^{(k)}$

$$\forall i : z_i \in \left[\min_k \{ x_i^{(k)} \}, \max_k \{ x_i^{(k)} \} \right]$$

technische universität
dortmund

Theorem
Let f:$\mathbb{R}^n \to \mathbb{R}$ be a strictly quasiconvex function. If f(x) = f(y) for some x ≠ y then every offspring generated by intermediate recombination is better than its parents.

Proof:

f strictly quasiconvex $\Rightarrow f(\xi \cdot x + (1-\xi) \cdot y) < \max\{ f(x), f(y) \}$ for $0 < \xi < 1$

since $f(x) = f(y) \quad \Rightarrow \max\{ f(x), f(y) \} = \min\{ f(x), f(y) \}$

$\Rightarrow f(\xi \cdot x + (1-\xi) \cdot y) < \min\{ f(x), f(y) \}$ for $0 < \xi < 1$
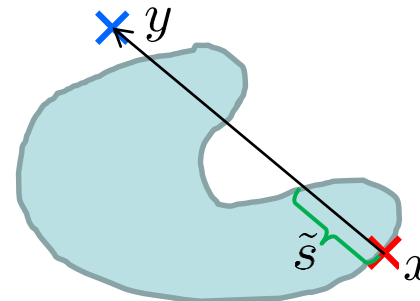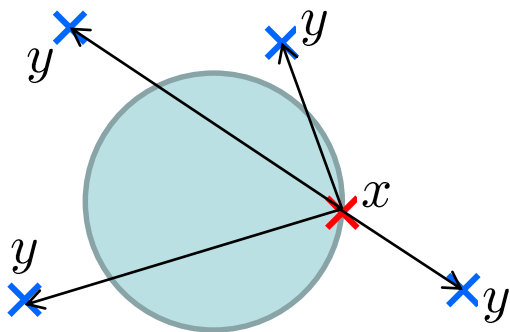
∎

Theorem
Let f:$\mathbb{R}^n \to \mathbb{R}$ be a differentiable function and f(x) < f(y) for some x ≠ y.
If (y − x)' ∇f(x) < 0 then there is a positive probability that an offspring
generated by intermediate recombination is better than both parents.

Proof:

If $d' \nabla f(x) < 0$ then $d \in \mathbb{R}^n$ is a direction of descent, i.e.

$$\exists \tilde{s} > 0 : \forall s \in (0, \tilde{s}] : f(x + s \cdot d) < f(x).$$

Here: $d = y - x$ such that $P\{f(\xi\, x + (1 - \xi)\, y) < f(x)\} \geq \dfrac{\tilde{s}}{\|d\|} > 0.$ ∎

sublevel set $S_\alpha = \{x \in \mathbb{R}^n : f(x) < \alpha\}$