# EVOLUTIONARY ALGORITHMS

# *The evolutionary algorithm is a search system*

❑ It handles several elements (individuals) of the problem space at each iteration and permanently modifies this population that becomes better and better until the solution (the goal, the best individual) appears.

❑ An individual is better than the other if it is somehow closer to the correct answer. The fitness function tries to measure this „distance" between an individual and the solution.

❑ First an initial population is selected mostly at random. In every turn the population is updated with the mutated offspring of the individuals that are selected for reproduction. This change of the population is an irreversible step, so the evolution is controlled by an irrevocable strategy.

# *Evolutionary operators and termination condition*

❑ *Selection*: better individuals are selected for reproduction

❑ *Recombination (crossover)*: pairs of the selected individuals (parents) are bred in order to create their offspring

❑ *Mutation*: offspring can be changed a bit

❑ *Replacement*: a new population is constructed from the elder one and the mutated offspring

❑ Termination condition:

- either a goal individual appears in the population

- or the overall fitness value of the population is not being changed

DATA *:= initial value*
**while** ¬ *termination condition*(DATA) **loop**
    SELECT R FROM *rules that can be applied*
    DATA *:=* R(DATA)
**endloop**

# *Evolutionary algorithm*

**Procedure** EA

$population := initial\ population$

**while** $not\ termination\ condition$ **loop**

    $parents\ \ := selection(population\ )$

    $offspring := recombination(parents\ )$

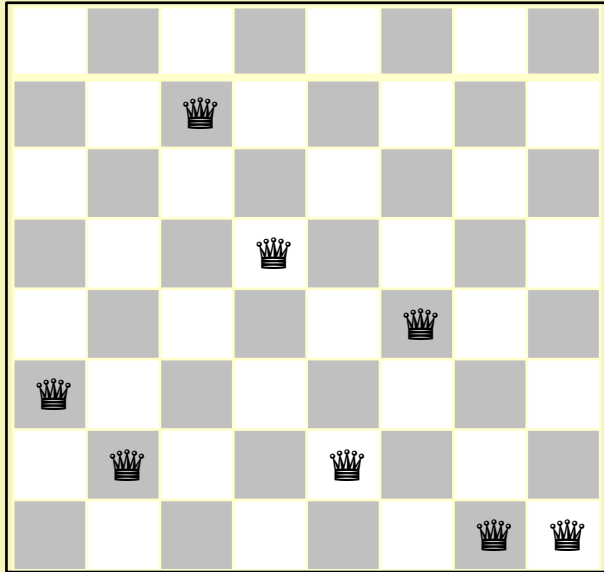    $offspring := mutation(offspring)$

    $population := replacement(population\ ,\ offspring)$

**endloop**

# *Playing Super Mario with neuroevolution*

- ❑ https://www.youtube.com/watch?v=qv6UVOQ0F44
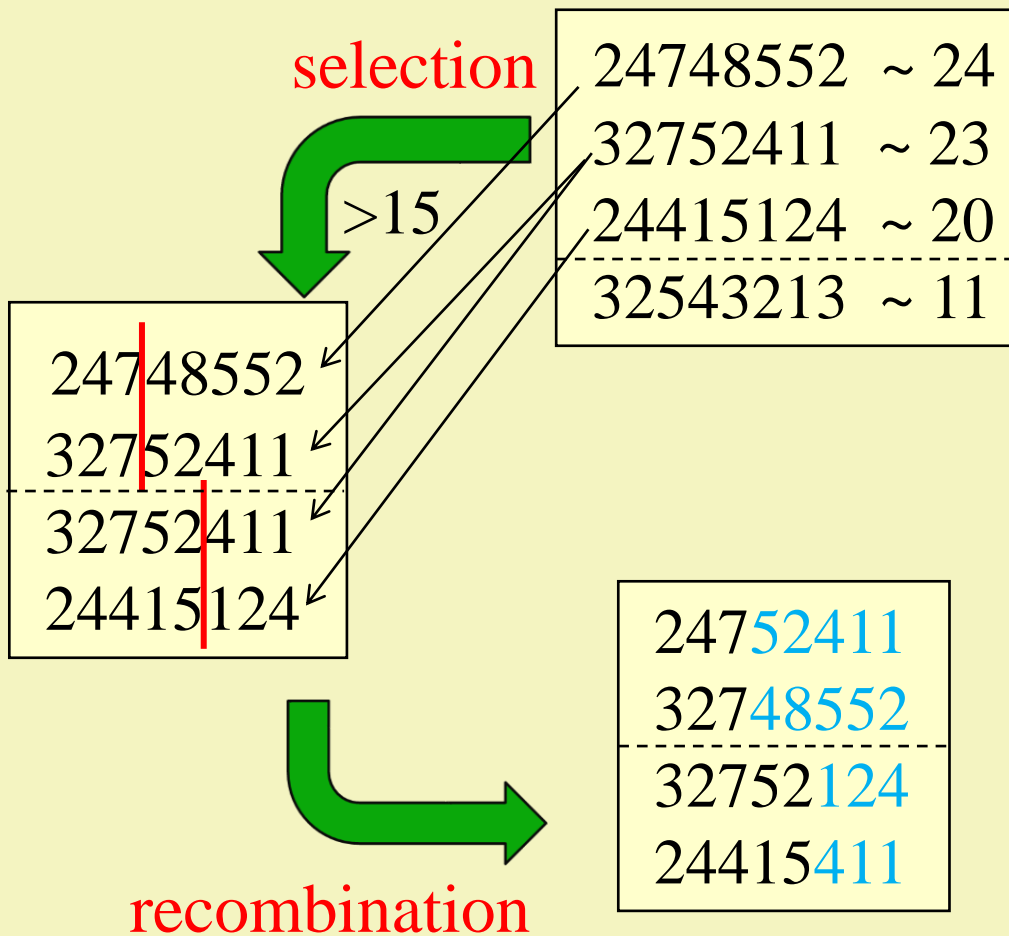- ❑ https://www.youtube.com/watch?v=iakFfOmanJU
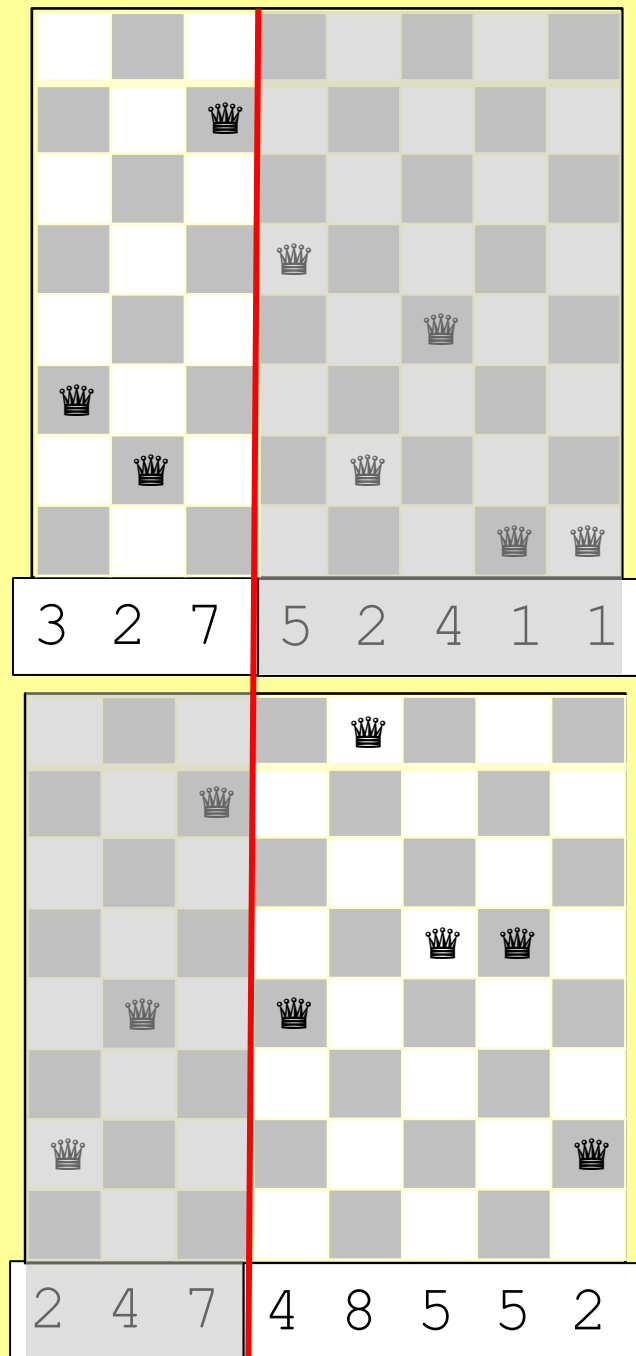
```
3  2  7  5  2  4  1  1
```

fitness value:  23

- ❑ <u>Individual</u>:  a possible placement of the queens where each column contains exactly one queen

- ❑ <u>Representation</u>:  the sequence of the row positions of the queens

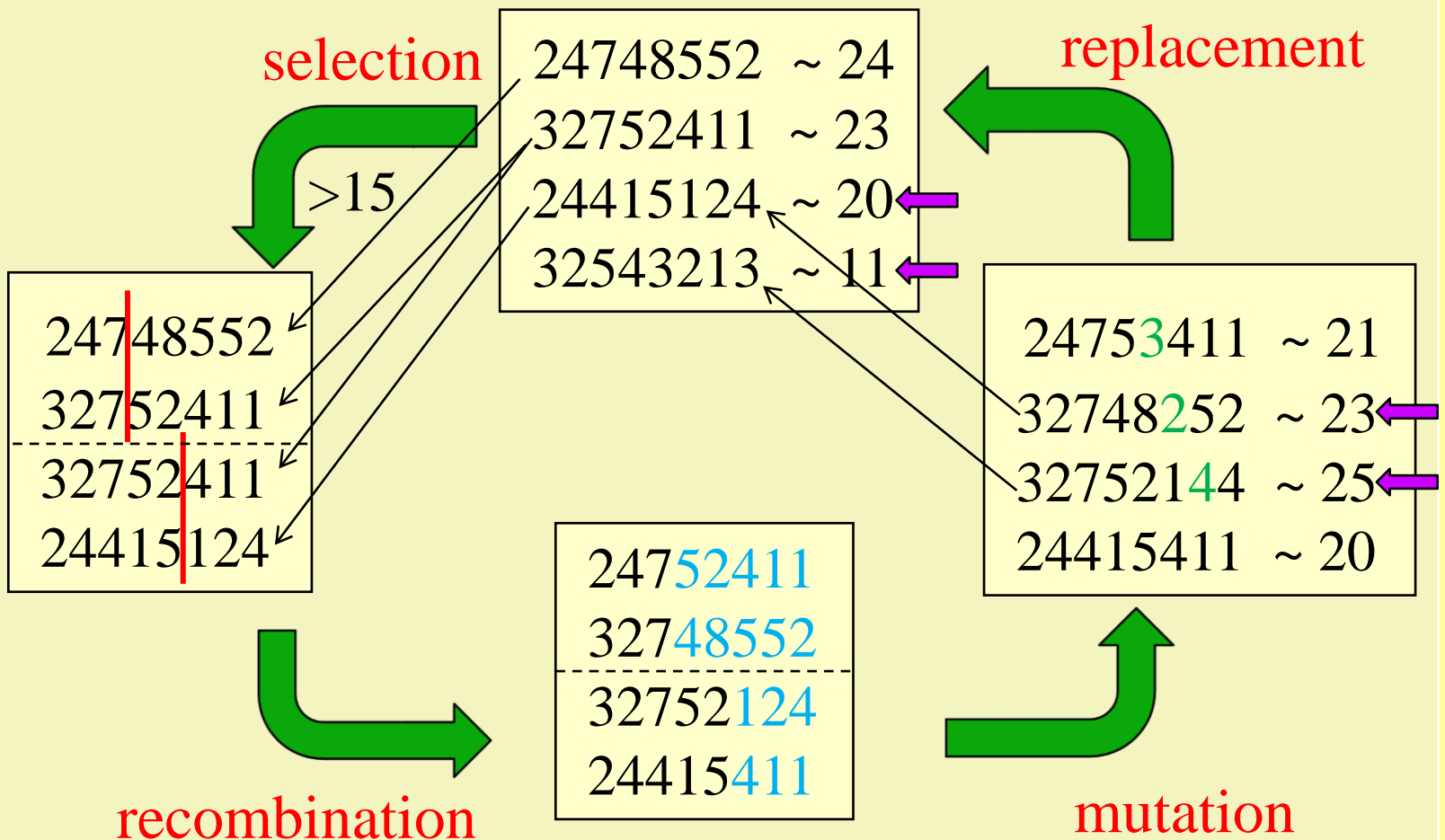- ❑ <u>Fitness function</u>:  number of nonattacking pairs of queens

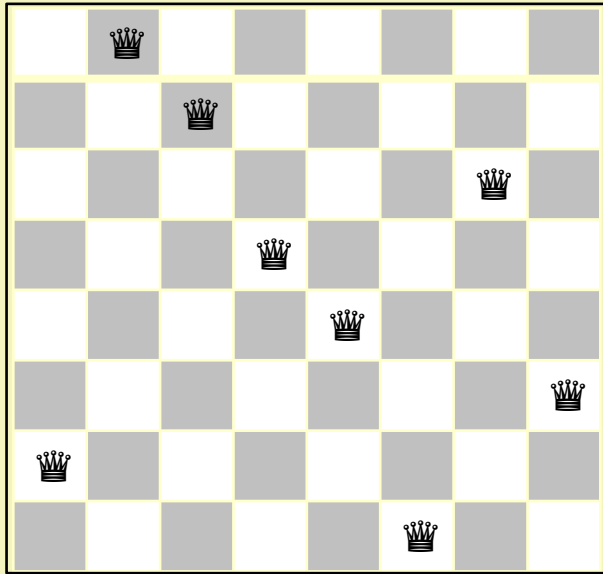# *Evolutionary cycle*

selection

>15

| | |
|---|---|
| 24748552 | ~ 24 |
| 32752411 | ~ 23 |
| 24415124 | ~ 20 |
| 32543213 | ~ 11 |

24748552
32752411
32752411
24415124

recombination

247**52411**
327**48552**
32752**124**
24415**411**

*Crossover*

# *Evolutionary cycle*

selection

24748552 ~ 24
32752411 ~ 23
24415124 ~ 20
32543213 ~ 11

>15

replacement

24748552
32752411
32752411
24415124

24753411 ~ 21
32748252 ~ 23
32752144 ~ 25
24415411 ~ 20

recombination

24752411
32748552
32752124
24415411

mutation

# *n-queens problem 2*



```
2 8 7 5 4 1 6 3
```
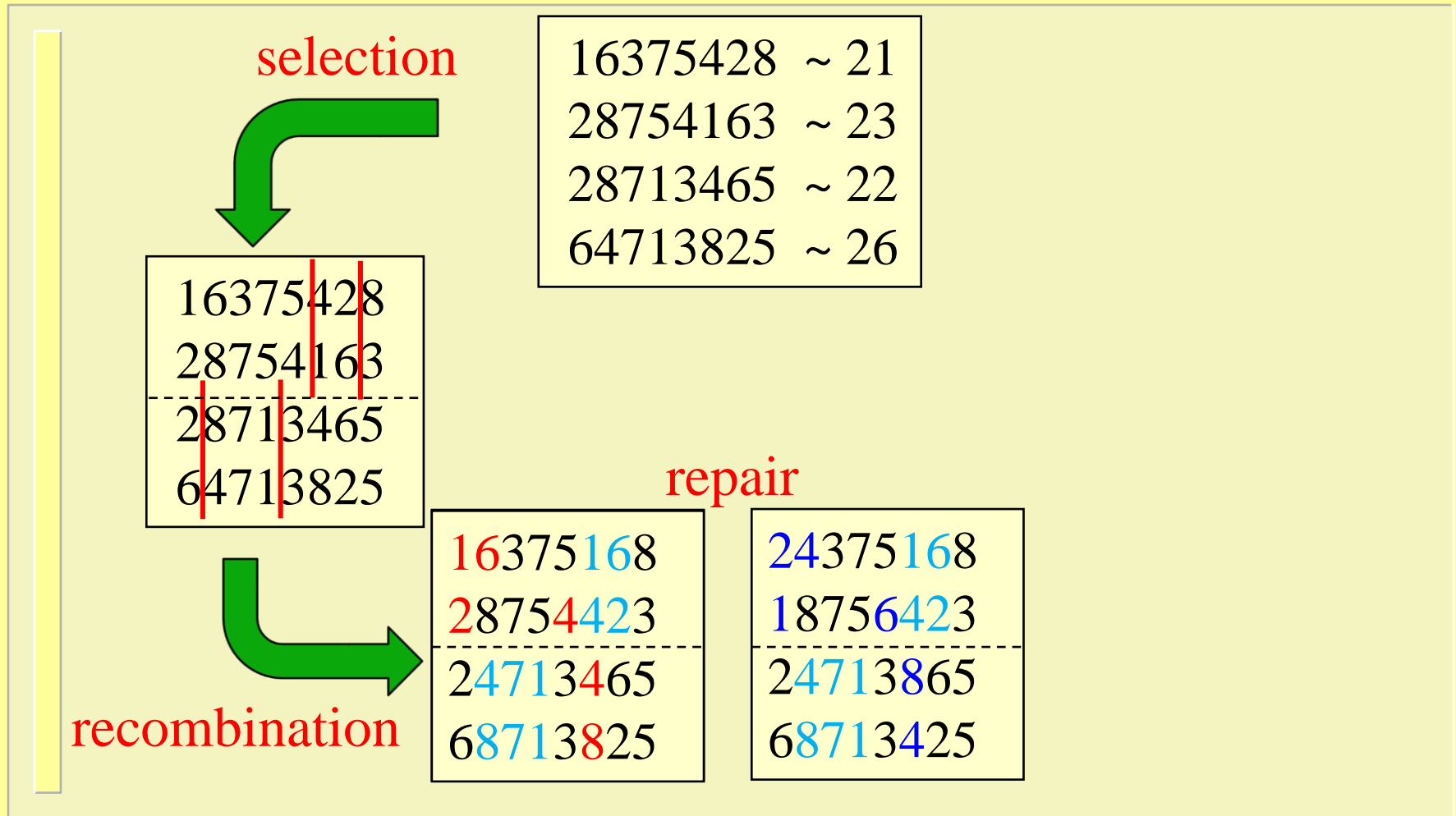
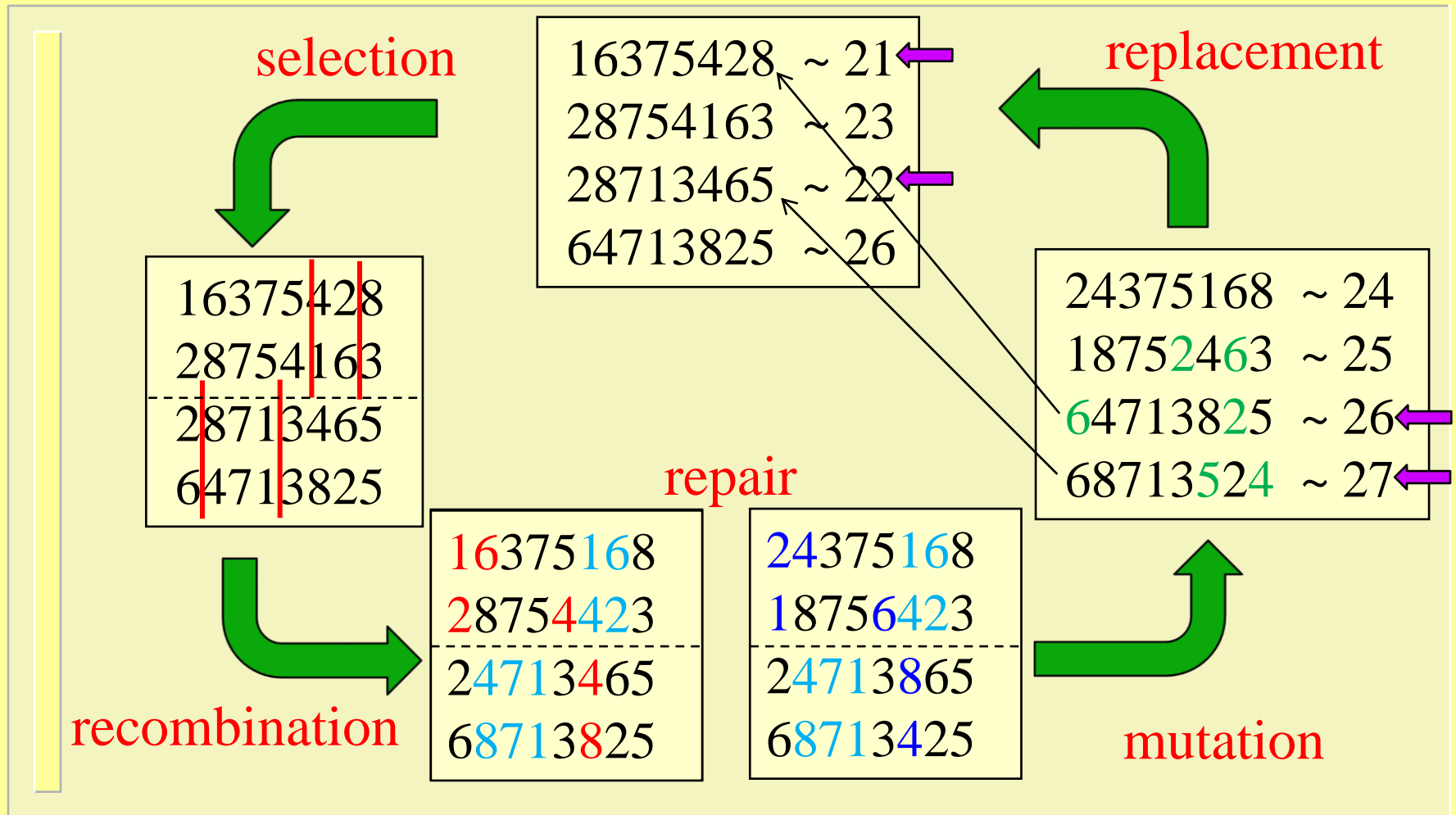fitness value:  23

❑ <u>Individual</u>: a possible placement of the queens where each column and each row contains exactly one queen

❑ <u>Representation</u>: permutation of the row positions: unique (non-repeating) numbers

❑ <u>Fitness function</u>: number of nonattacking pairs of queens

# *Evolutionary cycle*

selection

| | |
|---|---|
| 16375428 | ~ 21 |
| 28754163 | ~ 23 |
| 28713465 | ~ 22 |
| 64713825 | ~ 26 |

16375428
28754163
28713465
64713825

repair

16375168
28754423
24713465
68713825

24375168
18756423
24713865
68713425

recombination

# *Evolutionary cycle*



selection

replacement

16375428 ~ 21
28754163 ~ 23
28713465 ~ 22
64713825 ~ 26

16375428
28754163
28713465
64713825

24375168 ~ 24
18752463 ~ 25
64713825 ~ 26
68713524 ~ 27

repair

16375168
28754423
24713465
68713825

24375168
18756423
24713865
68713425

recombination

mutation

# Satisfiability problem (SAT)

*There is a Boolean statement given in CNF with n Boolean variables. Find a truth assignment for the variables so that the formula is true.*
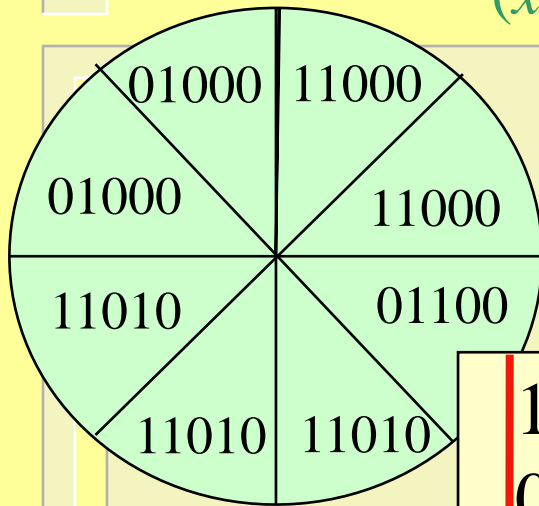
*E.g.:* $(x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_5)$

one solution: $x_1 = false, x_2 = true, x_3 = false, x_4 = true, x_5 = true$

❑ <u>Individual</u> :          a possible truth assignment

❑ <u>Representation</u>:     a sequence of bits

❑ <u>Fitness function</u>:    the number of the clauses of the formula that has true value in the given truth assignment
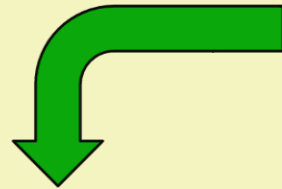
# *Evolutionary cycle*

$(x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_5)$
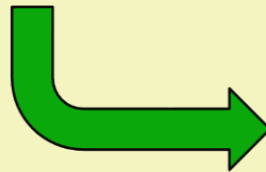


01000   11000

01000        11000

11010        01100

11010   11010

roulette wheel

**selection**

| | |
|---|---|
| 01000 | ~ 2 |
| 11010 | ~ 3 |
| 01100 | ~ 1 |
| 11000 | ~ 2 |

11010
01000
11010
11000

01010
11000
11000
11010
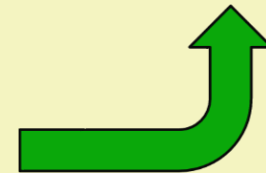
| | |
|---|---|
| 01011 | ~ 4 |
| 11110 | ~ 3 |
| 11010 | ~ 3 |
| 11001 | ~ 3 |

**recombination**

**mutation**

Gregorics Tibor                    Artificial intelligence
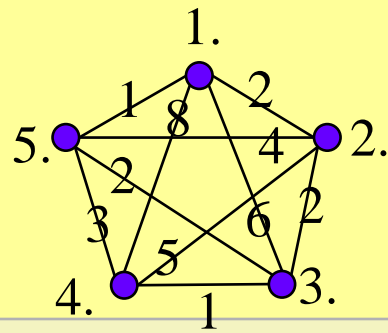
# *Traveling salesman problem*

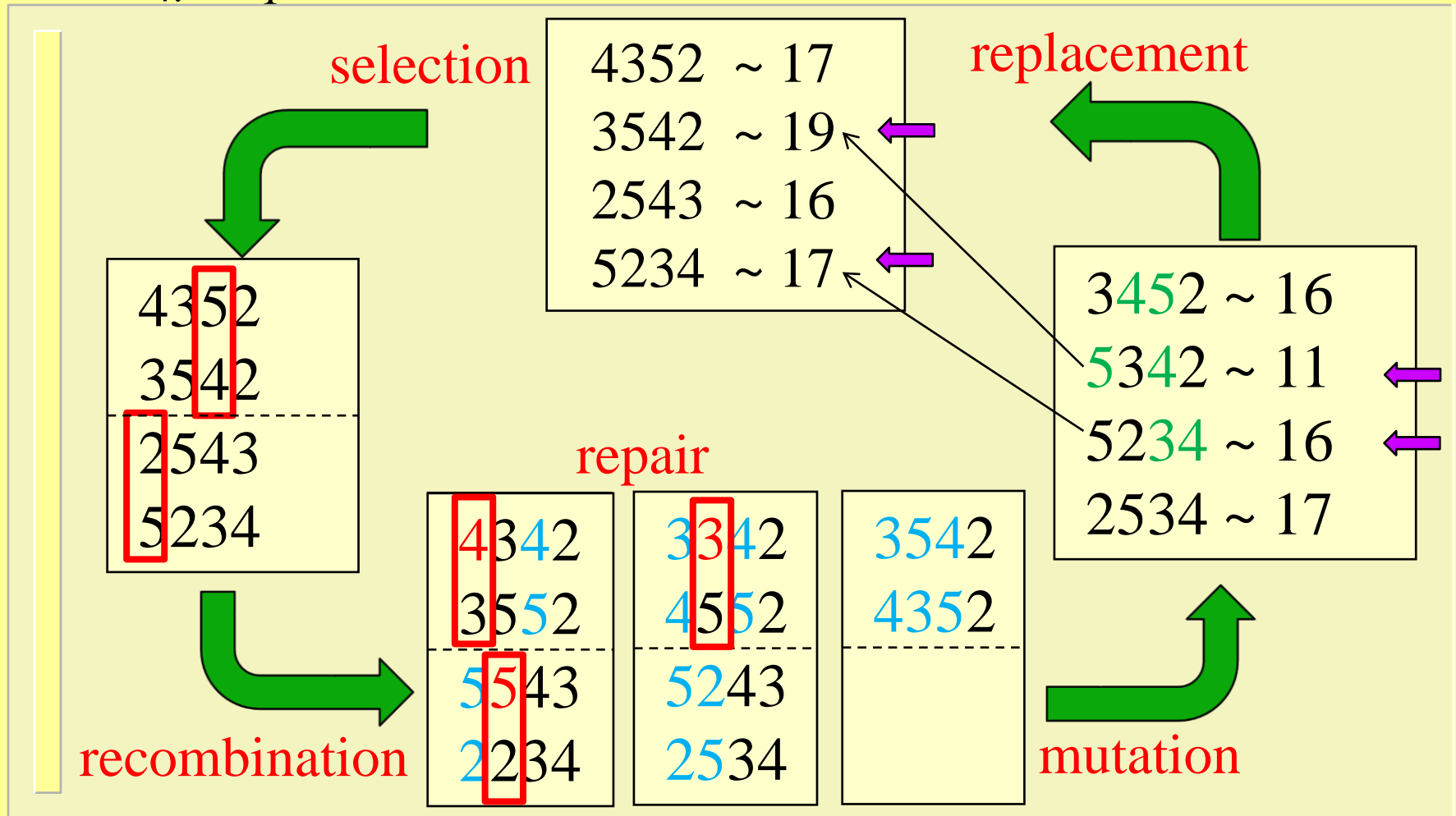*The traveling salesman must visit every city in his territory exactly once and then return to home city covering the shortest distance. (All distances between cities are known)*

- ❑ <u>Individual</u> :            one tour
- ❑ <u>Representation</u>:     permutation of the cities (numbers) without the home city
- ❑ <u>Fitness function</u>:    cost of the tour

*Evolutionary cycle*

selection

4352 ~ 17
3542 ~ 19
2543 ~ 16
5234 ~ 17

replacement

4352
3542
2543
5234

3452 ~ 16
5342 ~ 11
5234 ~ 16
2534 ~ 17

repair

4342
3552
5543
2234

3342
4552
5243
2534

3542
4352
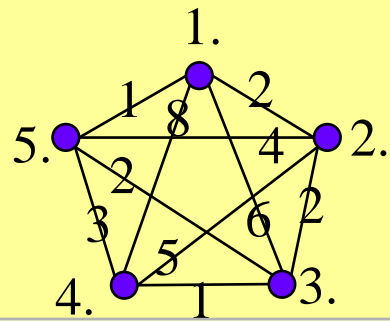
recombination

mutation

# *Elements of the evolutionary algorithm*

❑ Model: individuals and their representation

❑ Fitness function (heuristics)

❑ Evolutionary operators

– selection, recombination, mutation, replacement

❑ Initial population, termination condition (goal)

❑ Settings of the strategy parameters

– size of the population, probability of mutation, rate of the offspring, rate of the replacement

# *Representation*

❑ An individual is represented by a code (chromosome) that is most commonly a sequence of signals.

❑ One signal or one section of signals with its position in the code (gene) can describe one property of the individual.

  – Thus the structure of the code can be dismembered property by property. If one signal is changed, then one property of the individual is changed a bit, too.

  – Frequent coding methods:

  – Array: fixed length sequence of real numbers or integers

  – Binary code: fixed length sequence of bits

  – Permutation of the elements of a finite set

# *Representations and fitness functions of graph coloring problem*

*Color the vertices of a simple finite undirected graph with four colors so that no two adjacent vertices share the same color*
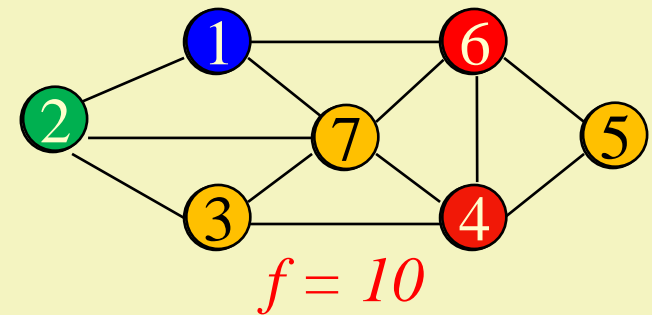
**Direct encoding**

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

$x[i]$ is the color of the $i^{th}$ vertex.

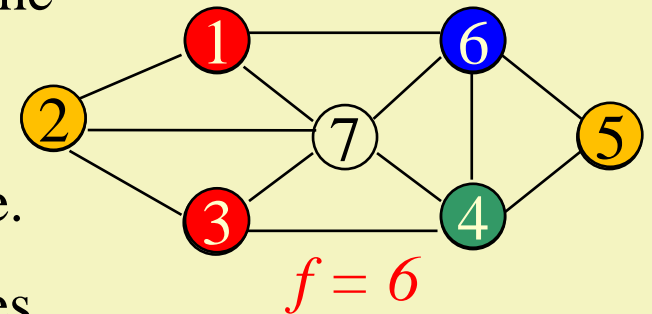$f$ is the number of the correct edges.

$f = 10$

**Indirect encoding**

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

In the $i^{th}$ step of the coloring process the $x[i]^{th}$ vertex must be colored with the possible lightest color regarding the colors of neighbor vertices, if possible.

$f$ is the number of the colored vertices.

$f = 6$

# *Representation and fitness functions of problem of rock-paper-scissors*

*Find a good strategy for the rock-paper-scissors world championship!*

❑ A strategy should give our next step based on a few (say two) previous games.

- For example:

  *Antecedent*:   *Me*:   P R      *Suggestion*:   S

  *Op*:   S P

- This is only one slice of the strategy. We need a suggestion for each possible antecedent.

There are $3^{81}$ different strategies (individuals) and their codes are the elements of $\{0,1,2\}^{0..80}$

| Signals | Antecedents (MeOpMeOp) | Suggestions |
|---|---|---|
| R ~ 0 | RRRR ~ 0000 ~ 0 | P ~ 1 |
| P ~ 1 | RRRP ~ 0001 ~ 1 | S ~ 2 |
| S ~ 2 | RRRS ~ 0002 ~ 2 | R ~ 0 |
|  | RRPR ~ 0010 ~ 3 | P ~ 1 |
|  | ... | ... |
|  | SSSP ~ 2221 ~ 79 | S ~ 2 |
|  | SSSS ~ 2222 ~ 80 | R ~ 0 |

*Code of the strategy:*   1201 ... 20

*Strategy:* 1 2 0 1 ... 2 0

*Signal*
```
R ~ 0
P ~ 1
S ~ 2
```

*Specimen battle:*

*Player:* 00022212220011000

*Op:* 01021122201011011

| Case | → | Suggestion | Opponent | Value | |
|------|---|-----------|----------|-------|---|
| 0001 | → | 2 | 0 | defeat | -1 |
| 0001 | → | 2 | 1 | victory | +1 |
| 0100 | → | 1 | 1 | draw | 0 |
| | ... | | | | |
| 2221 | → | 2 | 1 | victory | +1 |
| 2222 | → | 0 | 0 | draw | 0 |

# *Selection*

❑ The better individuals must be selected but the worse ones must be given a chance to be chosen. (stochastic method)

  – **Roulette wheel** : the probability of selection of an individual is proportional to its fitness function value

  – **Ranking**: the probability of selection of an individual is proportional to its ranking based on the fitness function values

  – **Tournament**: the selected individuals are the best individuals of randomly selected groups of the population

  – **Culling**: all individuals below a given threshold are discarded and then the individuals are selected randomly from the remaining individuals

# *Recombination*

- ❑ Offspring must be created from the pairs of parents so that the offspring inherit the properties of their parents.
  - – **Crossover**: signals of the parent codes are switched at the positions chosen randomly
  - – **Recombination**: the corresponding signals of the parent codes are combined

- ❑ If the code has got some extra invariant condition (permutation), it must be preserved.

❑ **Single or multiple pointed crossover**

–  Some crossover points are chosen at random and the signals are exchanged in every second section.

| 0 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 |

➡

| 0 | 1 | 1 | 0 | 0 |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 |
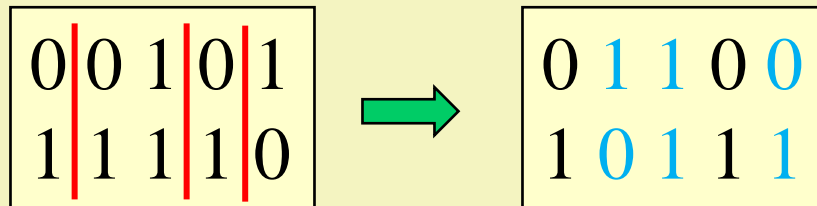
❑ **Uniform crossover**

–  Some positions are chosen at random and only the signals on these positions are exchanged.

| 0 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 |

➡

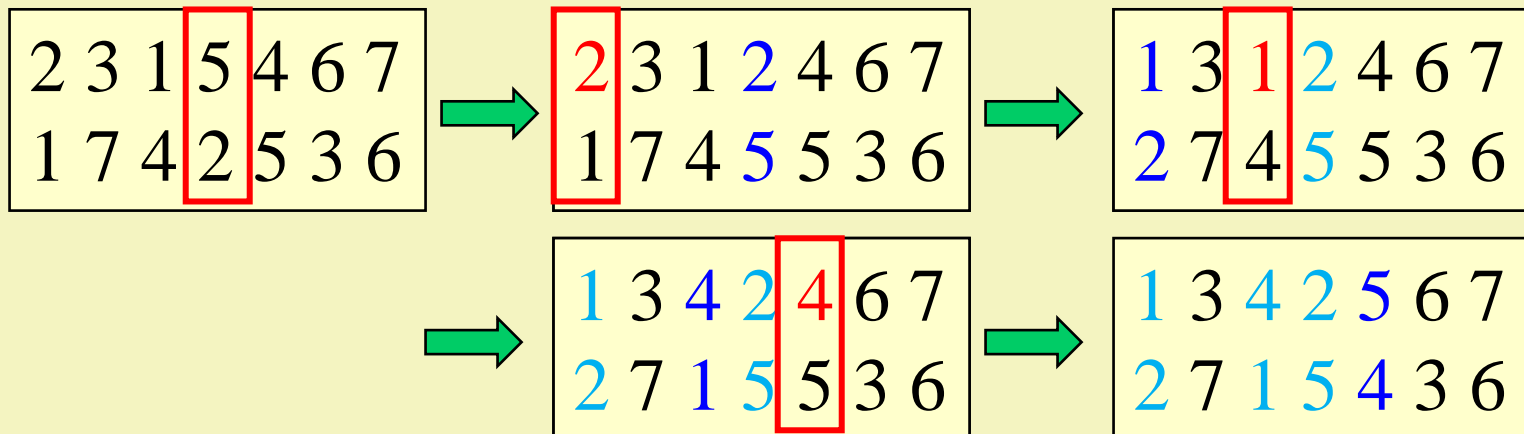| 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 |
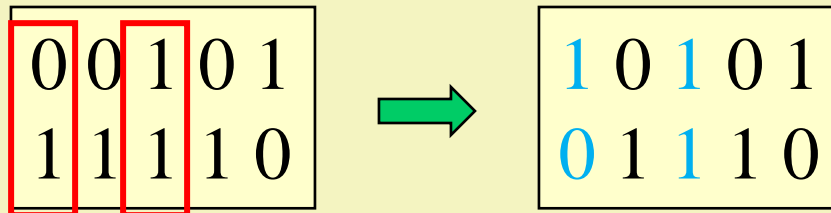
❑ **Partial crossover**

    – Signals are exchanged in one section chosen at random and then the duplicated signals of the first and the second parent besides the crossover section are paired and switched.
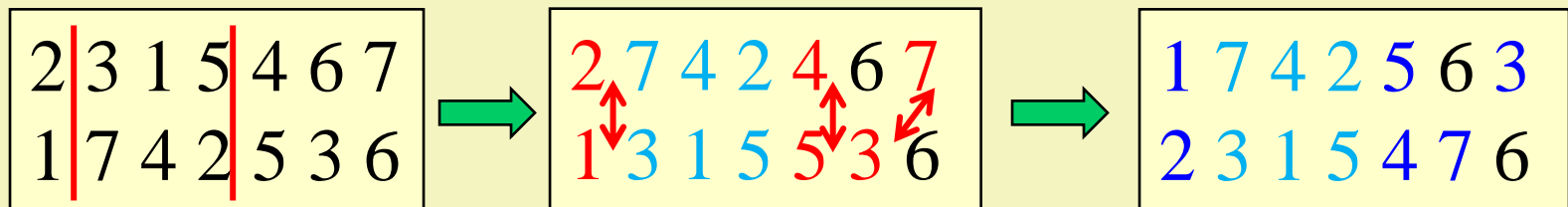
❑ **Cycle crossover**

1. Select $i \in [1..length]$ at random
2. Change $x_i \leftrightarrow y_i$
3. Find $j \in [1..length]$ $(j{\neq}i)$ where $x_j{=}x_i$,
4. If it is not found then over else $i{:}{=}j$
5. goto 2.

$$\begin{array}{cccccccc} 2 & 3 & 1 & 5 & 4 & 6 & 7 \\ 1 & 7 & 4 & 2 & 5 & 3 & 6 \end{array}$$

$$\begin{array}{cccccccc} 2 & 3 & 1 & 2 & 4 & 6 & 7 \\ 1 & 7 & 4 & 5 & 5 & 3 & 6 \end{array}$$

$$\begin{array}{cccccccc} 1 & 3 & 1 & 2 & 4 & 6 & 7 \\ 2 & 7 & 4 & 5 & 5 & 3 & 6 \end{array}$$

$$\begin{array}{cccccccc} 1 & 3 & 4 & 2 & 4 & 6 & 7 \\ 2 & 7 & 1 & 5 & 5 & 3 & 6 \end{array}$$

$$\begin{array}{cccccccc} 1 & 3 & 4 & 2 & 5 & 6 & 7 \\ 2 & 7 & 1 & 5 & 4 & 3 & 6 \end{array}$$
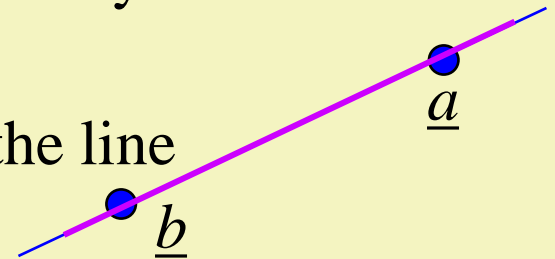
# *Recombinations of real vectors*

❑ **General recombination**

  – The offspring ($\underline{u}$) is somewhere around or in the hyper cube spanned by the parents (a, b)

  – $\forall i \in 1 \ldots n : u_i = c_i \cdot a_i + (1 - c_i) \cdot b_i$

     where $c_i \in [-h, 1+h]$ selected randomly
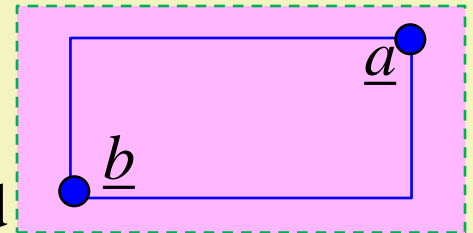
❑ **Linear recombination**

  – The offspring ($\underline{u}$) is somewhere on the line matching the parents (a, b)

  – $\forall i \in 1 \ldots n : u_i = c \cdot a_i + (1 - c) \cdot b_i$

     where $c \in [-h, 1+h]$ selected randomly

# *Mutation*

❏ Each position of the code is subject to random change with a small independent probability ($p$).

❏ When the code is an array including real numbers:

  – $\forall i = 1 \ldots n : z_i = x_i \pm domain_i \cdot p$

❏ When the code is a binary code:

  – $\forall i = 1 \ldots n : z_i = 1 - x_i$    if $random[0..1] < p$

❏ When the code is a permutation:

  – swapping signal-pairs

  – shifting the signals in a section cyclically

  – reversing or reordering the signals in a section

# *Replacement*

❑ Creation of the new population from the elder one and the mutated offspring: the selected individuals of the population are substituted with the selected offspring. two selections are needed

$$\text{offspring rate(u)} = \frac{\text{number of the offspring to be put in}}{\text{size of the population}}$$

$$\text{replacement rate(v)} = \frac{\text{number of the individuals to be removed}}{\text{size of the population}}$$

– u=v : total exchange

additional selection is needed

– u<v : some selected offspring must be used several times

additional selection is needed

– u>v : some selected offspring must be eliminated