

# Evoluční algoritmy

- třída optimalizačních metod inspirovaných principy biologické evoluce
- zahrnují mj. tzv. - genetické algoritmy
  - evoluční strategie
  - genetické programování
- particle swarm optimization
- ant colony optimization
- } zpravidla nejsou považovány za evoluční
- společně: použití evolučních principů (přírodní výběr, mutace) na populaci potenciálních řešení daného prostoru
- řadí se mezi tzv. metaheuristický (další jsou např. gradientní metody, simulované žití)
- obecné výpočetní metody pro hledání suboptimálního řešení kompromisů s numerickým řešením; zpravidla iterativní;  
(tedy ne v jednom kroku analyticky)
- pro řešení obtížných problémů, ke kterým neexistují rychlé spolehlivé metody
- není zaručeno nalezení uspokojivého řešení
- zpravidla forma guided random search (řízené náhodné průhledávání) v prostoru možných řešení

## Biologická evoluce

- Charles Darwin, *The Origin of Species* (1859) (jedinci)
- 3 základní principy: výhodné vlastnosti, které se šíří v populaci v rámci procesu přirozeného výběru
- nové, popř. modifikované, vlastnosti, jedinci se šíří různými mechanismy; zejm.:
  - mutace - zpravidla náhodné změny genů (např. radiace, mutageny, chyby při kopírování genů v intermice)
    - tj. změny v místěk me-  
sencích genů v intermice
  - Křížení (crossover) - výměna částí genů v intermice mezi dvěma systémy souběžných genů v intermice (dva chromozomy)
- přirozený výběr = mechanismus díky kterému přežívají jen ti nejschopnější (fittest); ostatní nepřežijí, protože jsou omezení

Pozn.: Mutace, křížení a přirozený výběr jsou 3 základní principy biologické evoluce. Existuje řada důvodů, které se v optimalizačních metodách využívají (viz Kruse et al., *Computational Intelligence*, 2016).

Umělé (simulační) evoluce

- principy biologické evoluce lze využít pro řešení optimalizačních problémů na základě představy, že udržujeme množinu (populaci) potenciálních řešení problému a tato množina prochází evolucí
- idea: je-li biologická evoluce možná, je možné, že tento postup vyprodukuje uspořádané řešení (kvalitní jedince)
- optimalizační problém:  $\langle \Omega, f \rangle$ , kde  $\Omega$  je množina možných řešení,  $f: \Omega \rightarrow \mathbb{R}$  máří kvalitu, tj.  $f(w) \in \mathbb{R}$  pro každé možné řešení; je možné, že hledáme max-řešení nebo min-řešení (max-řešení je  $w \in \Omega$  t.j.  $\forall w' \in \Omega$  je  $f(w') \leq f(w)$ ).  
stejně je možné omezení se na max (nebo na min)
- základní pojmy evolučních algoritme a jejich biologické vzory viz Kruse et al., str. 193 (kopie)

Základní schéma evolučního algoritmu

- pro specifikaci evol. algoritmu musíme určit:
- způsob kódování kandidátních řešení → vysoce specifické podle  
řešeného problému;  
základní důležitost  
(populace = množina kódů nějakého počtu kandidátních řešení)  
(např. číslo  $n \in \mathbb{N}$  kódujeme řetězcem bitů reprezentujícím bin. zápis)
  - jak vytvořit počáteční populaci (zpravidla náhodně, respektovat omezení plynoucí ze základní podmínky)
  - fitness funkci  $f$  pro ohodnocení vhodnosti kandid. řešení  
(zpravidla  $f$  je optimalizační funkce)
  - metodu výběru jedinců na základě fitness  $f$  (do další generace)  
(z aktuální populace do populace následující generace)  
(pro výběr rodičů potomků do další generace a pro výběr jedinců přímo postupující)
  - genetické operátory pro modifikaci chromozómů (tj. kódů kandidátních řešení) a způsob jejich použití; operátory mohou být generické, ale i vysoce problemově závislé; výsledek musí respektovat omezení problému; opět základní důležitost
  - ukončovací podmínku  
(pozn.: evol. algoritmus lze zastavit u každém okamžiku  
(např. na základě počtu generací nebo když je nalezeno uspokojivé řešení))
  - hodnoty příp. parametrů algoritmu (např. velikost populace, pravděpodobnost mutace, velikost části populace použité pro vytvoření potomků)

schéma: Evolutionary-algorithm

```

t ← 0; // inic. číslo generací
initialize population(t);
evaluate population(t);
while not termination-criterion do
  t ← t + 1;
  select pop.(t) from population(t-1);
  genetic-alter population(t);
  evaluate population(t);
end

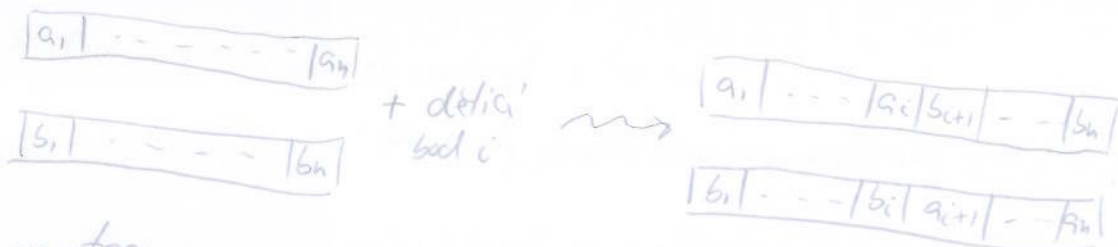
```



základní metoda kódování (přel kouzíme o genetických algoritmech):  
kandidátní řešení  $\rightarrow$  řetězec bitů

základní křížení (crossover):

pro 2 zvolené chromosomy (kódy kandid. řešení) se určí dělicí bod  $s$  proběhne přelitění:



základní mutace:

každý prvek (pozice, bit) v chromosomu je změněn s danou velmi malou pravděpodobností

Příklad 1: Maximalizace kvadratické funkce.

$$f(n) = -n^2 + 2n + 2, \quad n \in \Omega = \{0, \dots, 2^k - 1\}$$

číslo  $n$  reprezentujeme bitovými řetězcí délky  $k$

Příklad 2: Problém rozmístění  $n$ -dem na šachovnici  $n \times n$   
možné řešení reprezentujeme vektorem  $q \in \{0, \dots, n-1\}^n$

$q_i = k \Leftrightarrow$  v řádku  $i$  je dáms na sloupci  $k$

viz Kruse et al., str. 197 ff (lit kopie)

## Genetické algoritmy a věta o schématech (schema theorem)


- GA: chromozomy = řetězce bitů

základní schéma genetického algoritmu - Kruse et al., str. 246 (vše kopie)

k tomu: - roulette-wheel selection

- po každé iteraci <sup>pop</sup> máme populaci  $f(s)$  a  $prob(s) = \frac{f(s)}{\sum f(s')}$

- do další populace vyber z dané populace stejný počet prvků jako její počet prvků

-  $\approx$    $s_i$ , ne stejný už zraje kolečka; opakuje se, než vybereme potřebný počet prvků

- one-point crossover (crossover 1 point)

Pro dané řetězce  $s_1, s_2 \in \{0,1\}^k$  zvolíme náhodně jejich libovolný bod  $i \in \{1, \dots, k-1\}$  a vytvoříme křížením nové  $s_1', s_2'$ :

$$s_1' = [s_{1,1} \dots s_{1,i} | s_{2,i+1} \dots s_{2,k}]$$

$$s_2' = [s_{2,1} \dots s_{2,i} | s_{1,i+1} \dots s_{1,k}]$$

Věta o schématech - viz přednáška a Kruse et al., str. 247 ff.