Genetické algoritmy

Jiří Vomlel

Laboratoř inteligentních systémů
Vysoká škola ekonomická Praha

Tato prezentace je k dispozici na:
http://www.utia.cas.cz/vomlel/

Motivace z Darwinovy teorie evoluce

Přírodní evoluce je úspěšná a robustní metoda adaptace v biologických systémech.

- děti dědí vlastnosti rodičů
- lepší jedinci lépe přežívají a mají tudíž více potomků

Evoluce v přírodě vyžaduje čas - ovšem pomocí počítačů můžeme vytvořit a ohodnotit tisíce umělých individuí během zlomku vteřiny.

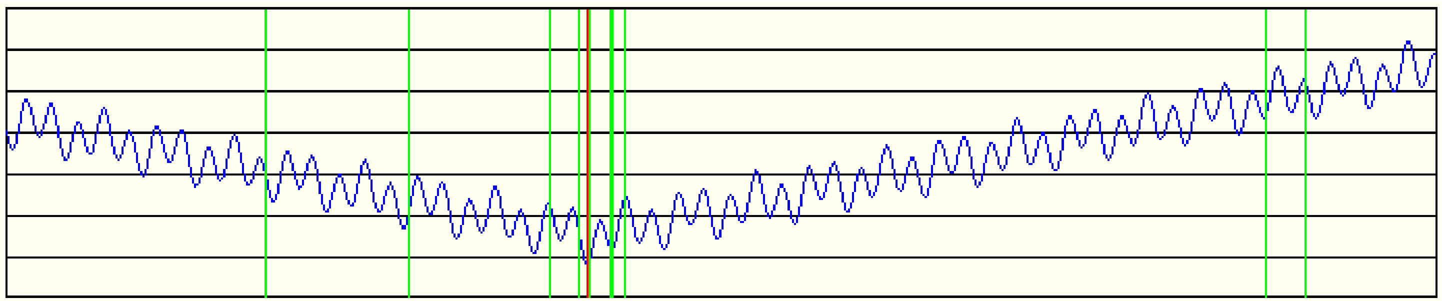
Korespondence s Darwinovou teorií evoluce

v přírodě	v genetických algoritmech
jedinec	řetězec symbolů, např. $h = (1001)$
přírodní výběr	výběr podle hodnotící funkce $f(h)$
křížení	kombinace dvou řetězců
	např. $(000 \mid 1) + (101 \mid 0) \longrightarrow (0000) + (1011)$
mutace	náhodná záměna 0 a 1 v řetězci
	např. $(0010) \longrightarrow (1010)$

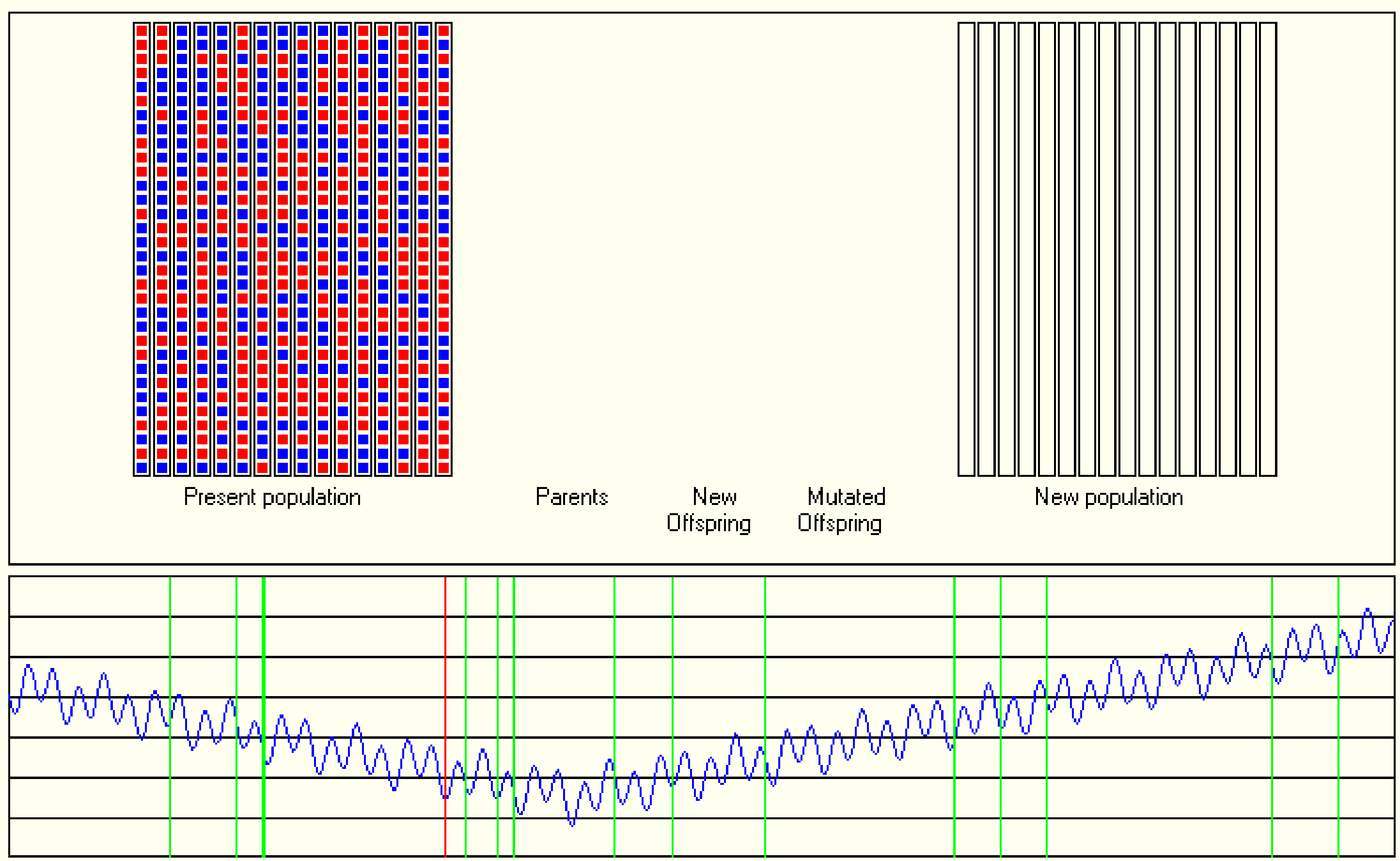
Příklad

převzatý z http://cs.felk.cvut.cz/~xobitko/ga/

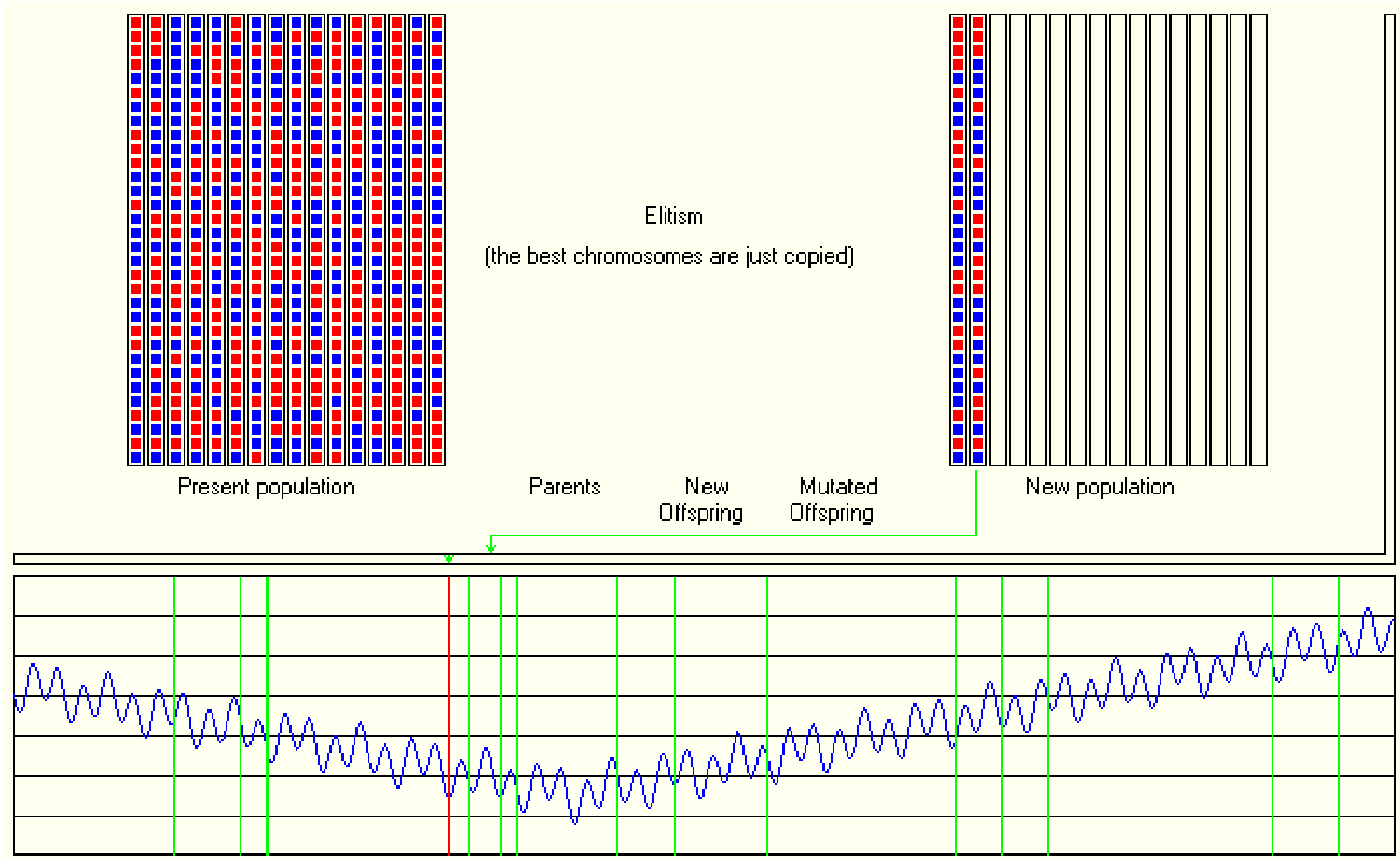
Je dána funkce $f : X \mapsto \mathbb{R}$ a cílem genetického algoritmu je najít $x \in X$ v němž funkce nabývá globálního minima.



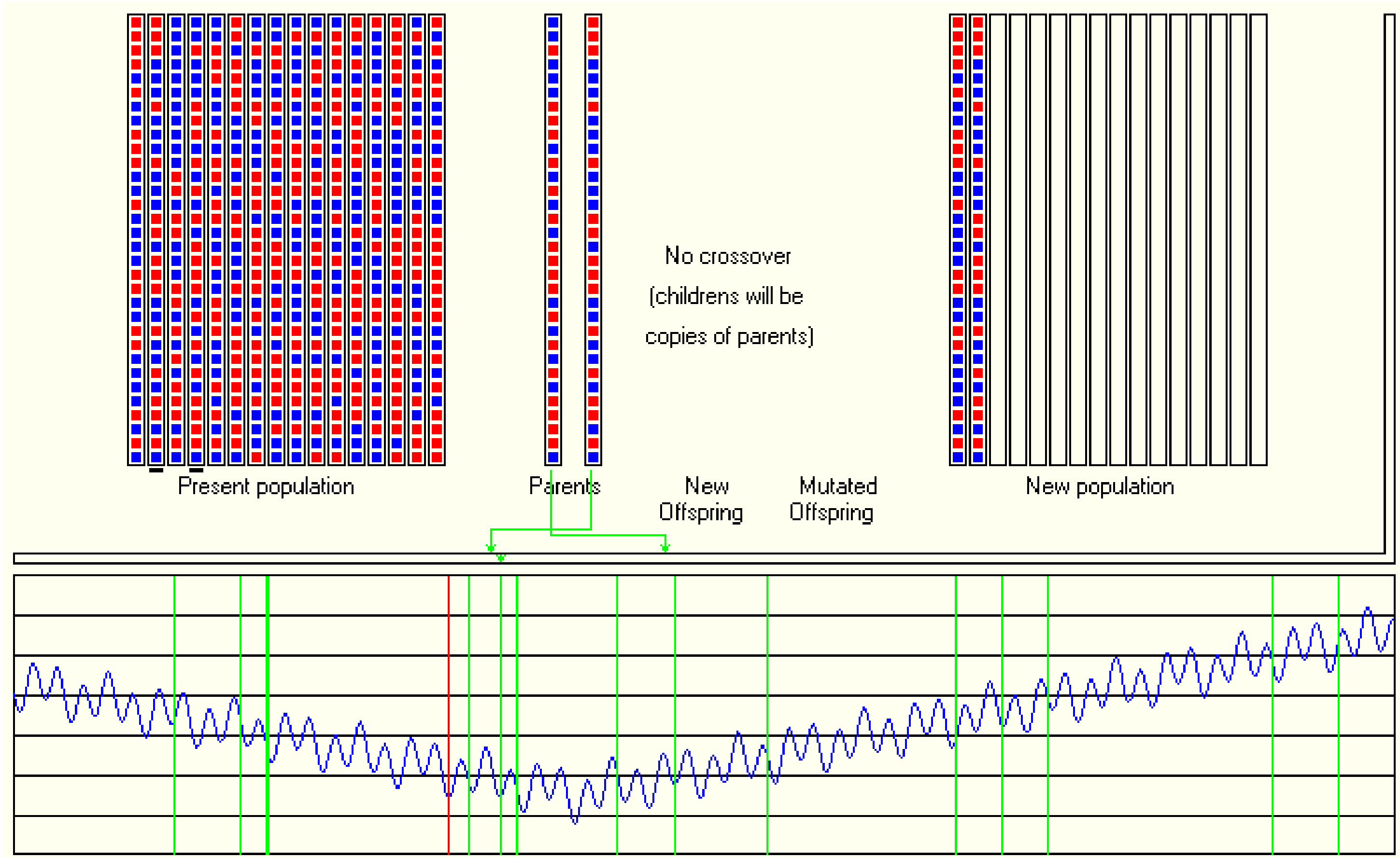
počáteční generace



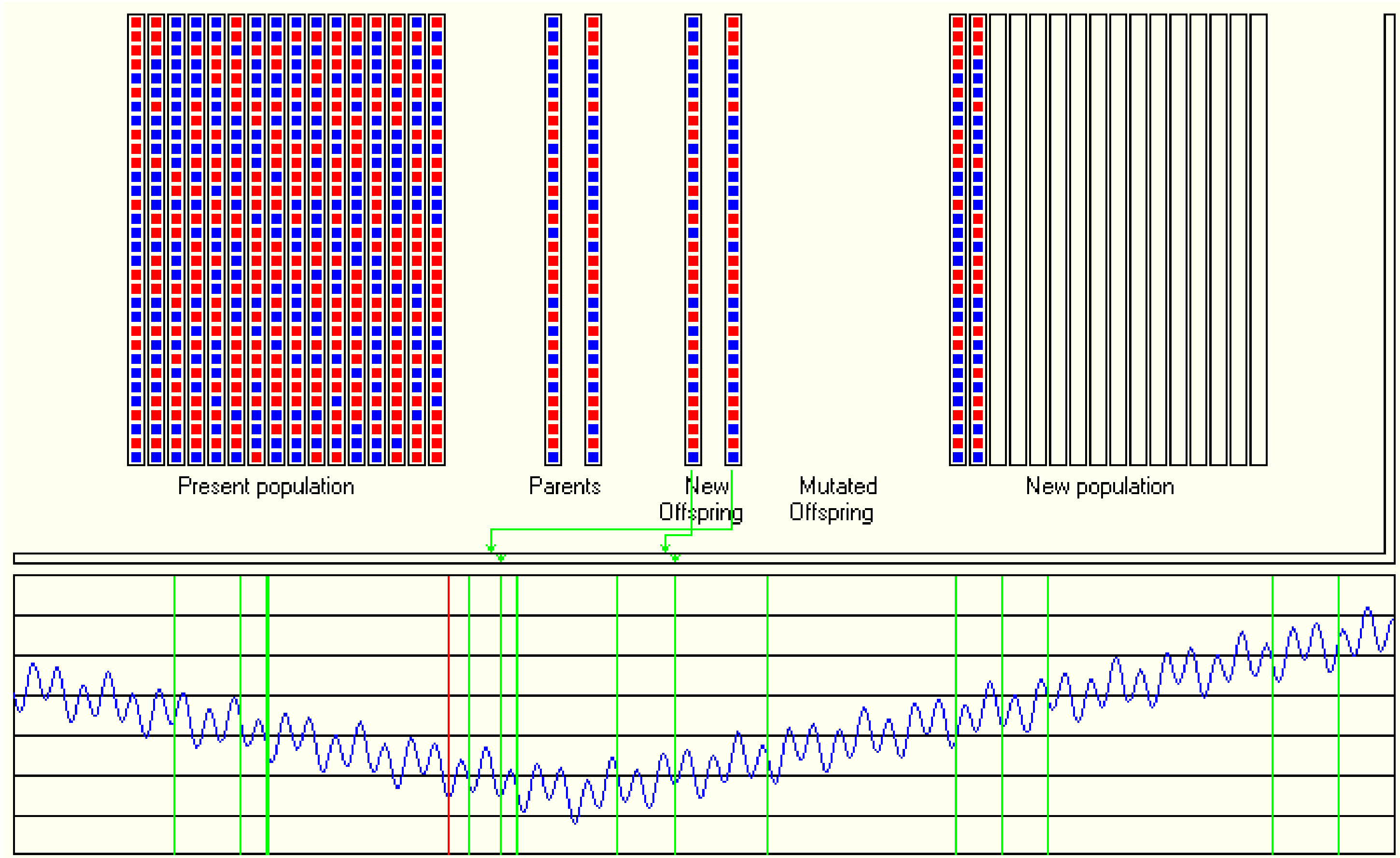
elitářství - nejlepší jedinci přecházejí přímo do nové generace



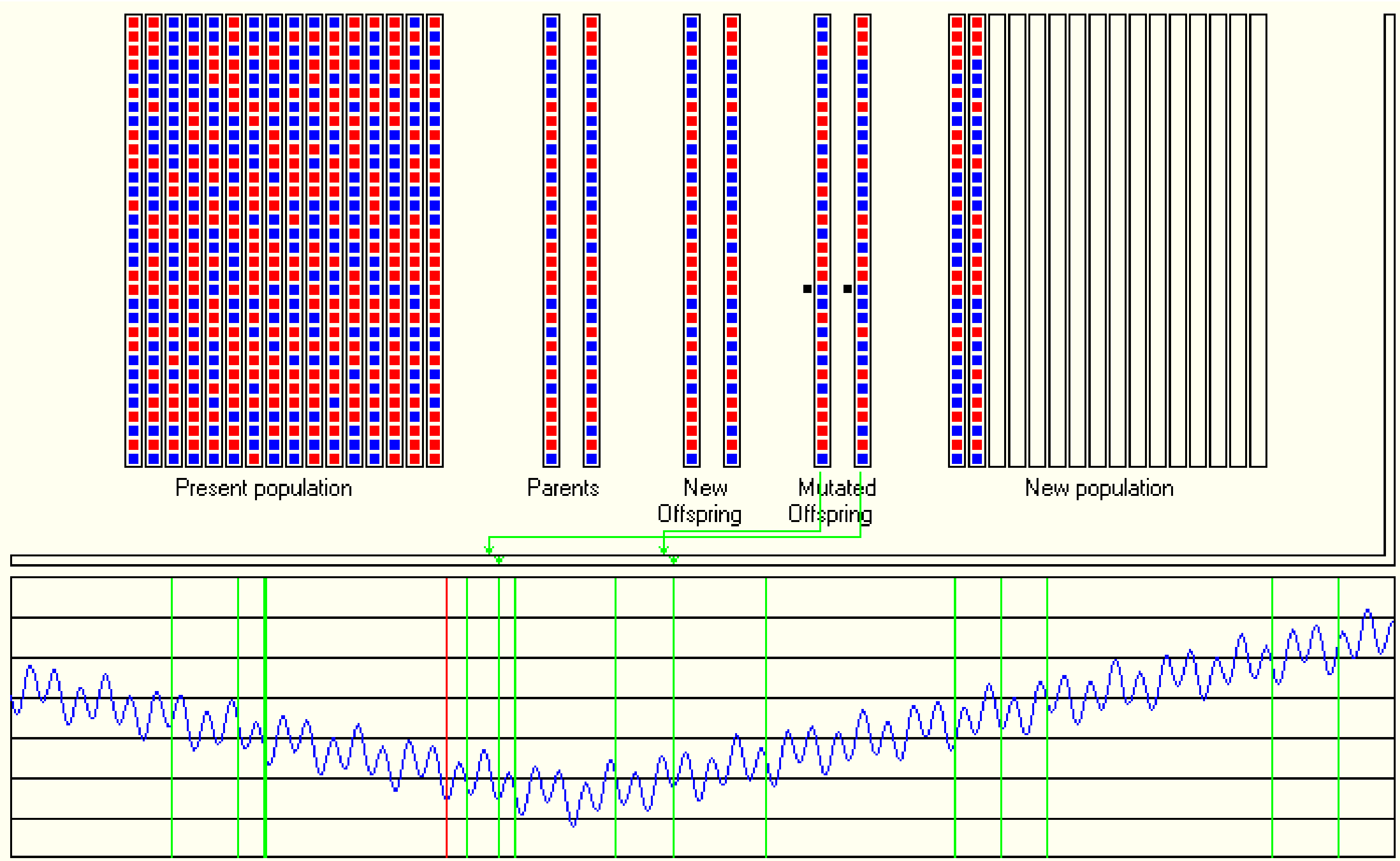
dva jedinci jsou vybráni pro křížení



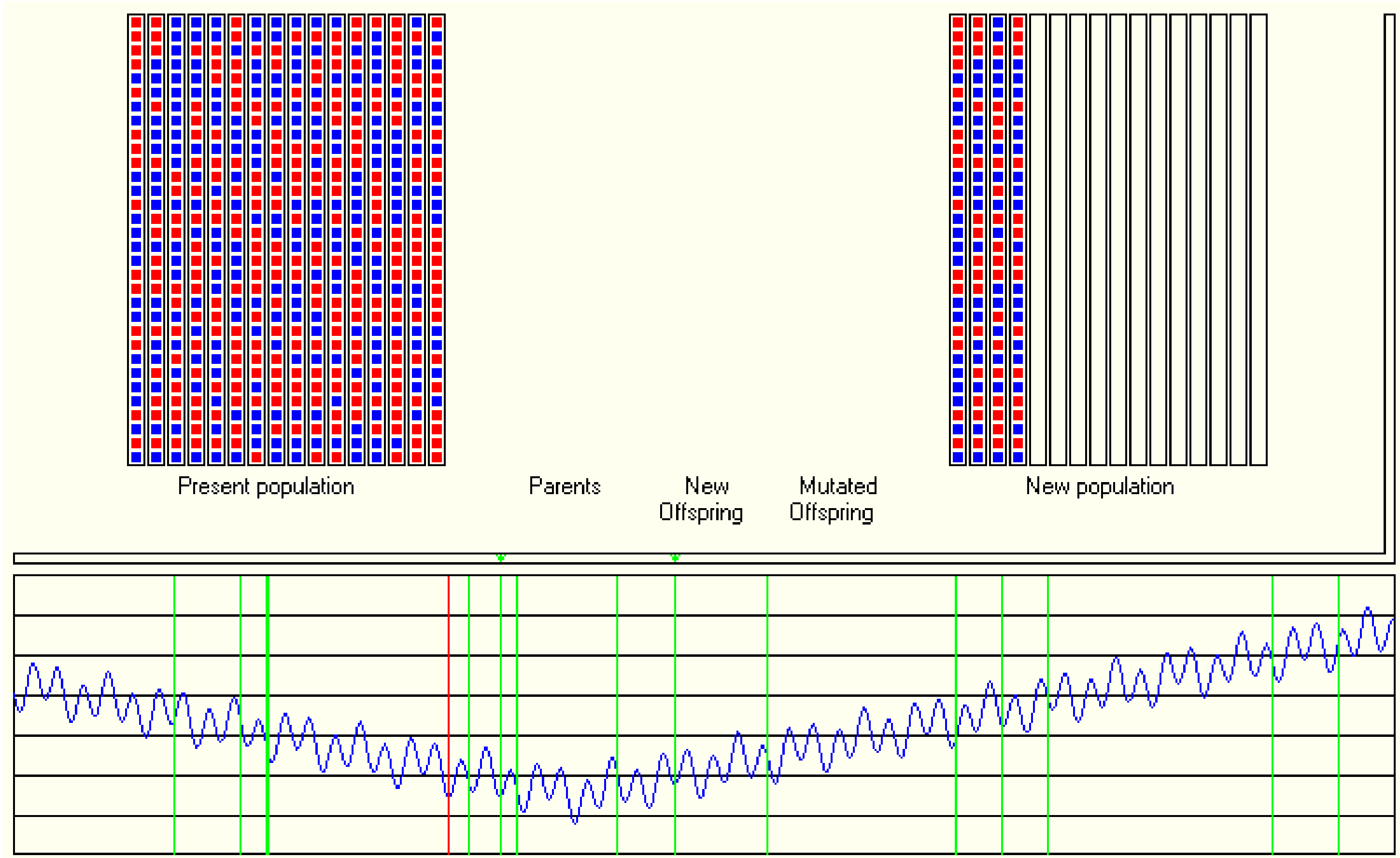
vznikli noví potomci, kteří jsou přesnými klony rodičů



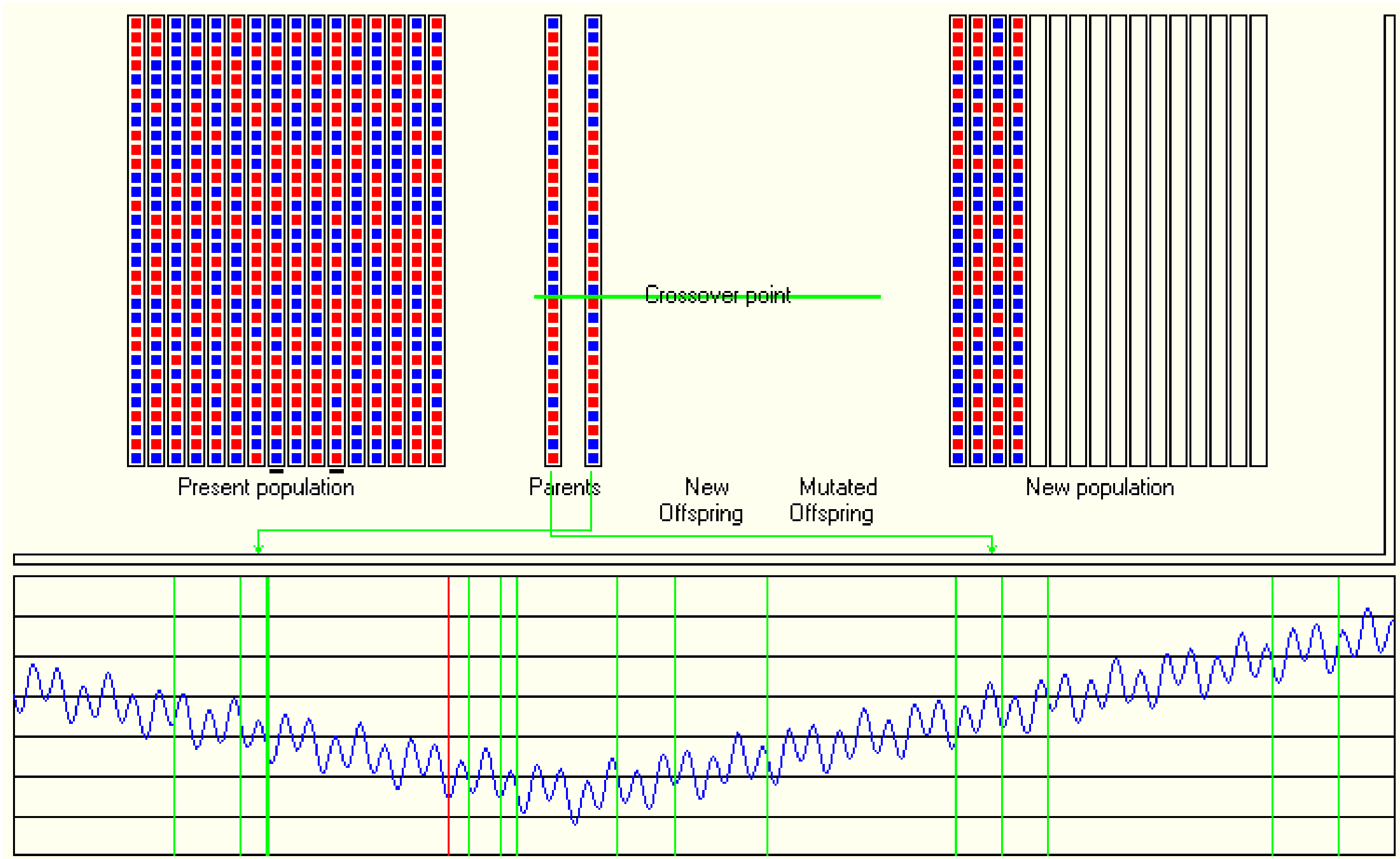
docházi k mutaci genetické informace potomků



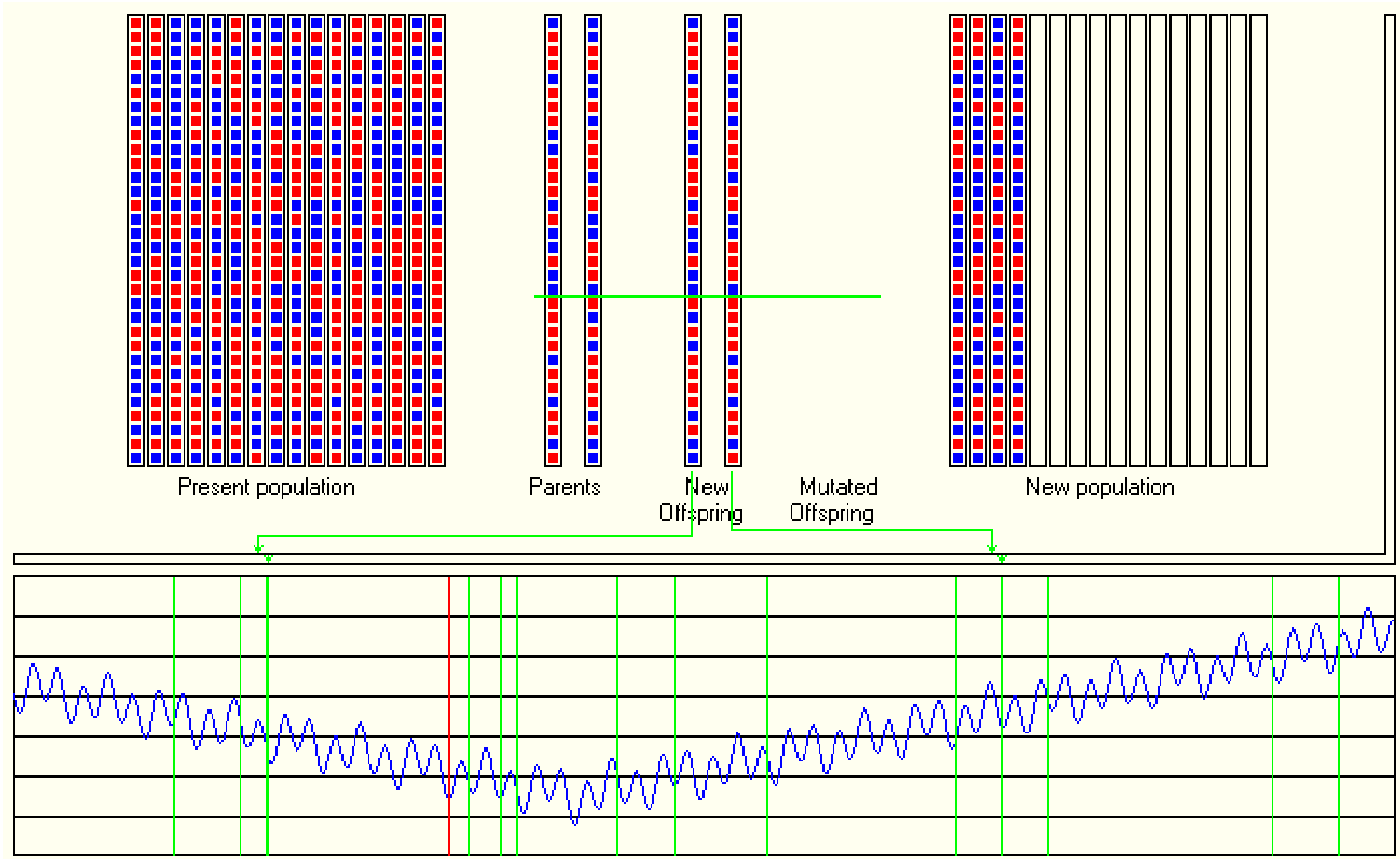
potomci jsou přidáni do nové populace



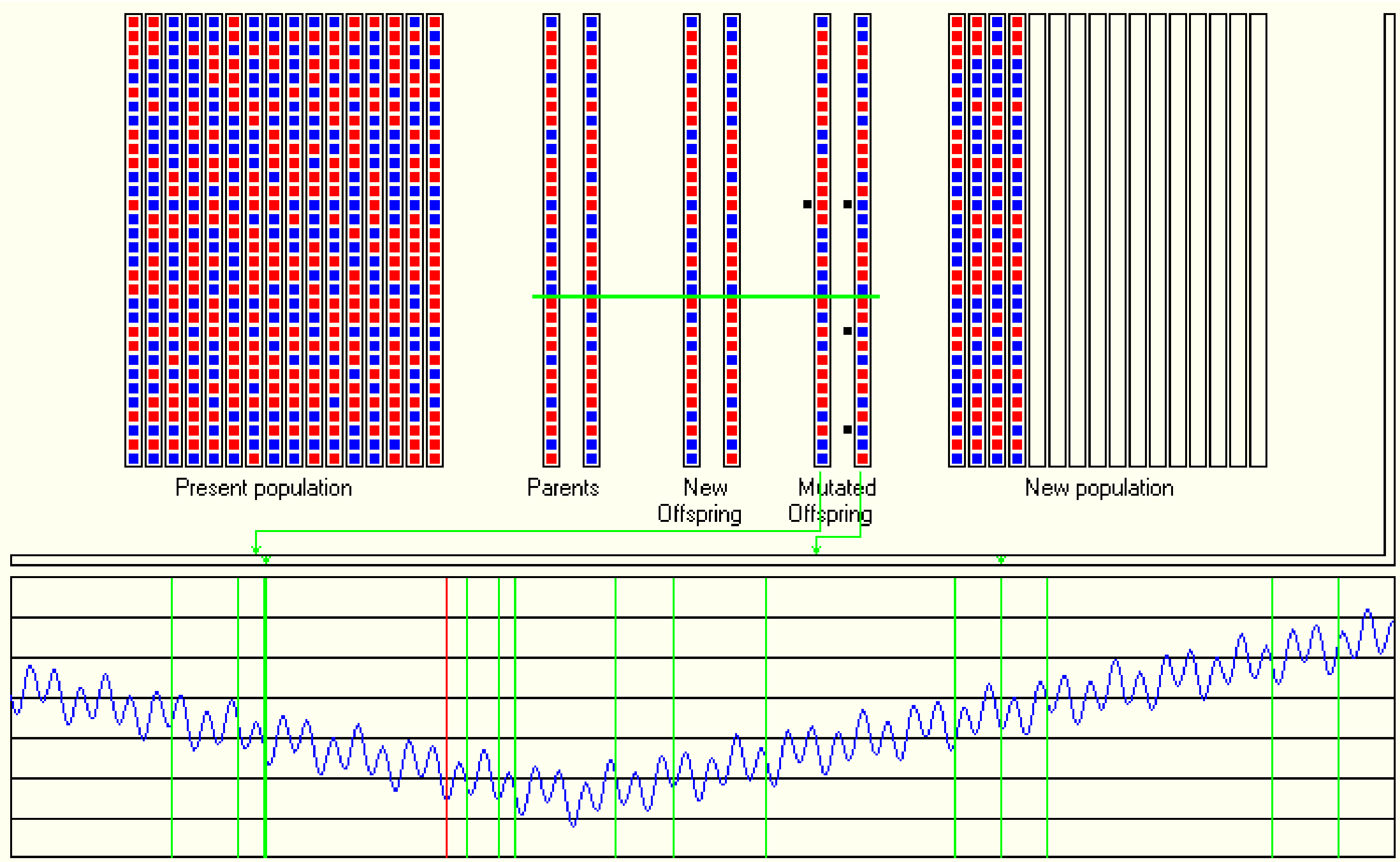
opět dva jedinci jsou vybráni pro křížení



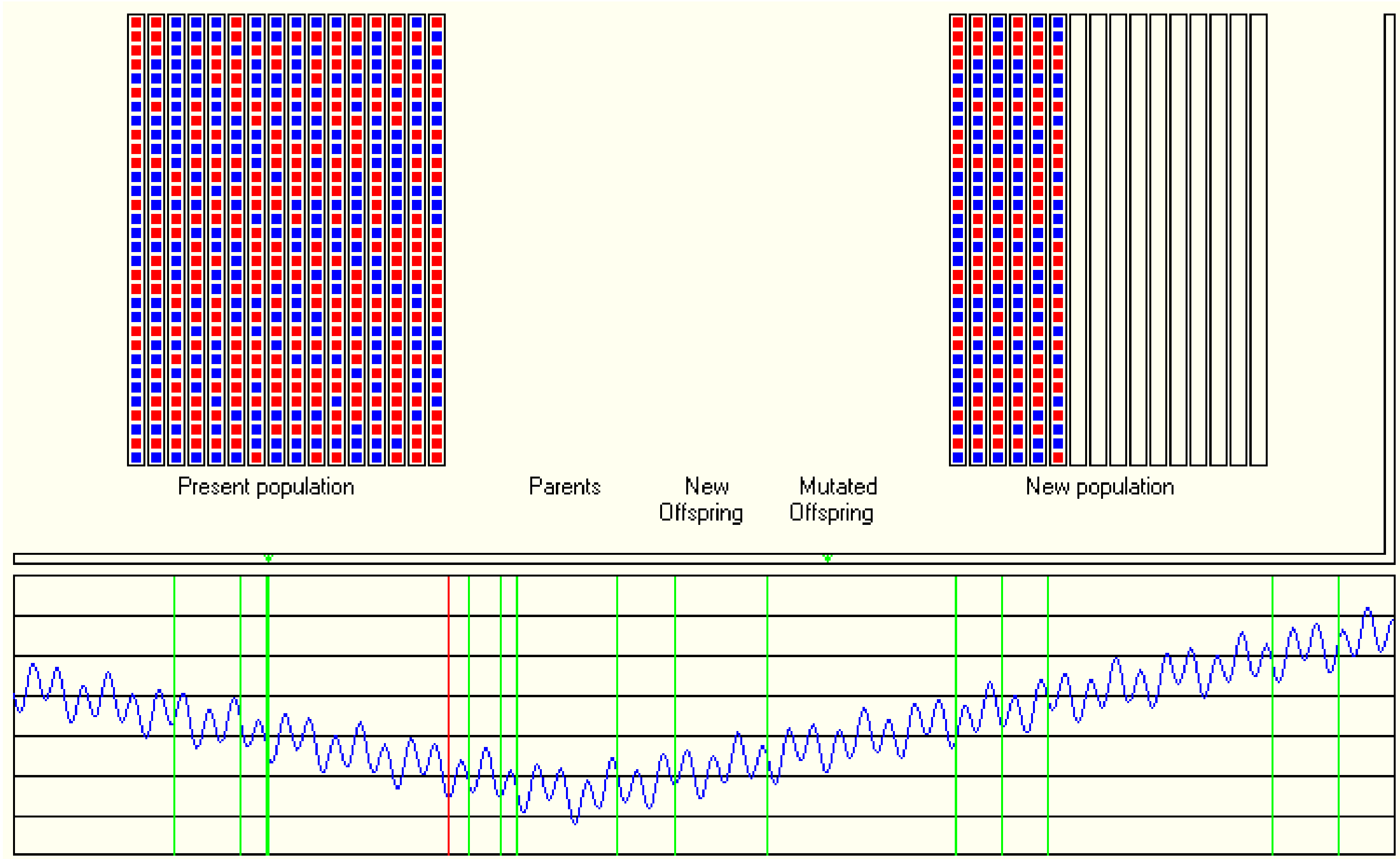
docházi ke křížení



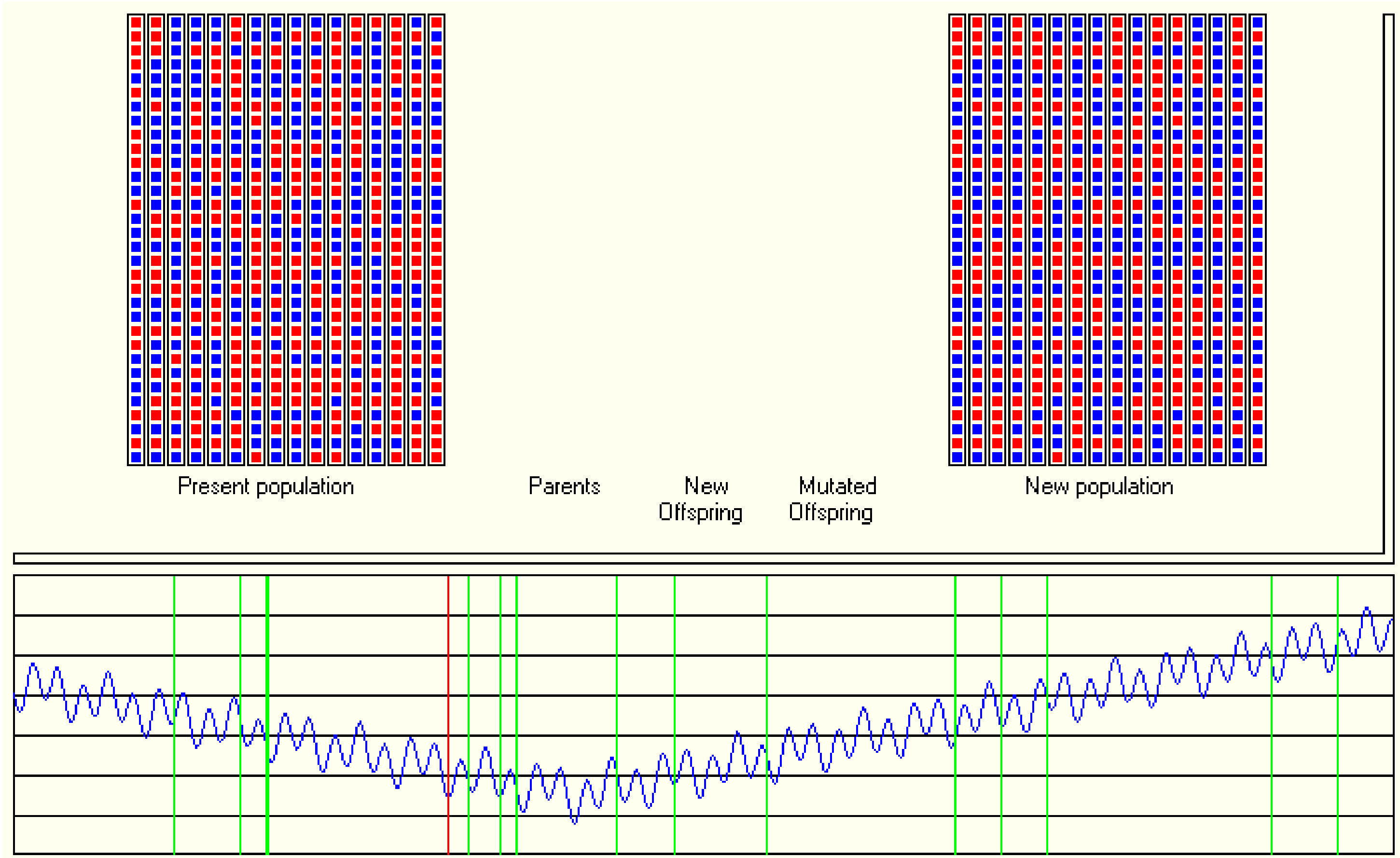
docházízí k mutaci



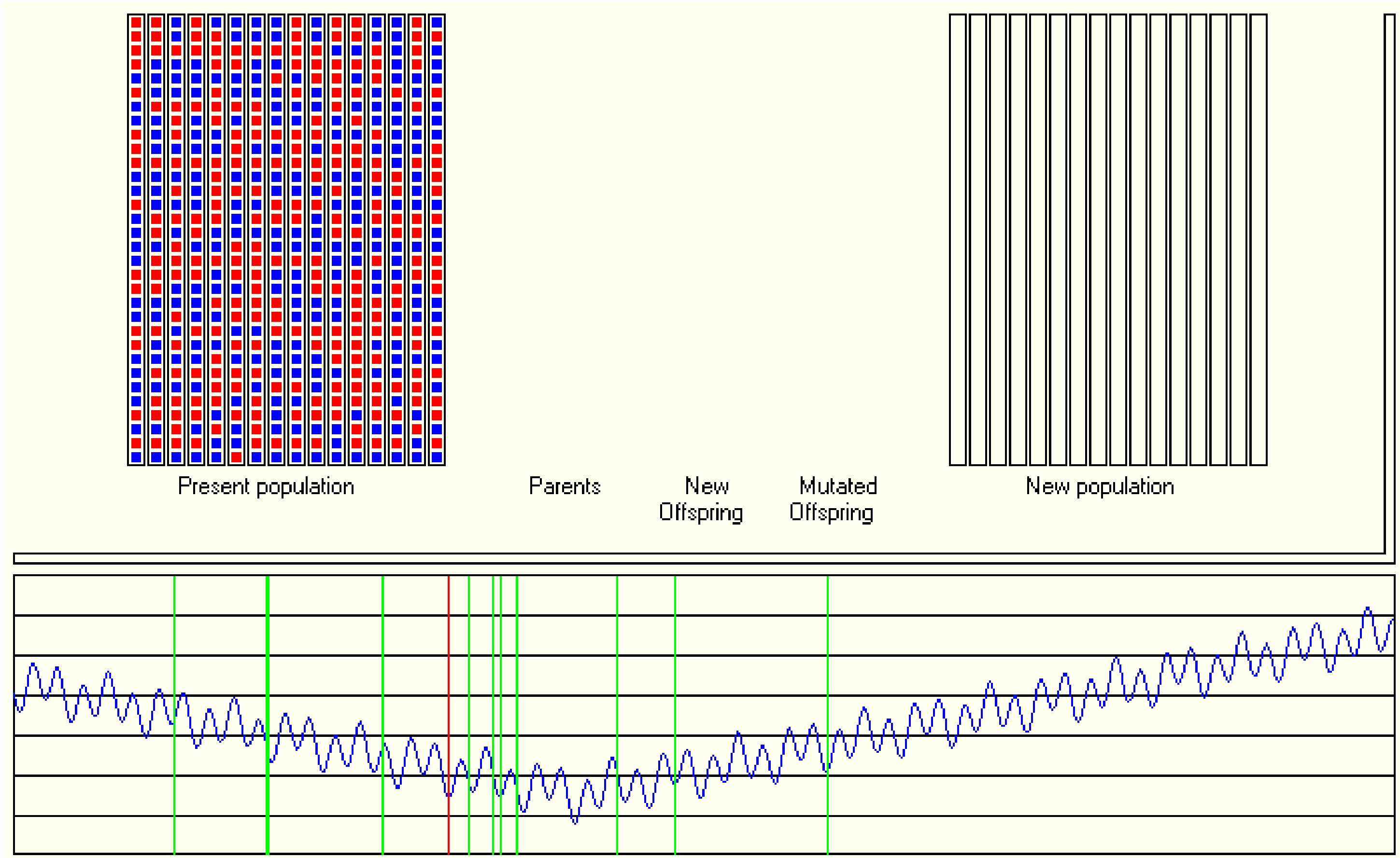
potomci jsou opět přidáni do nové populace



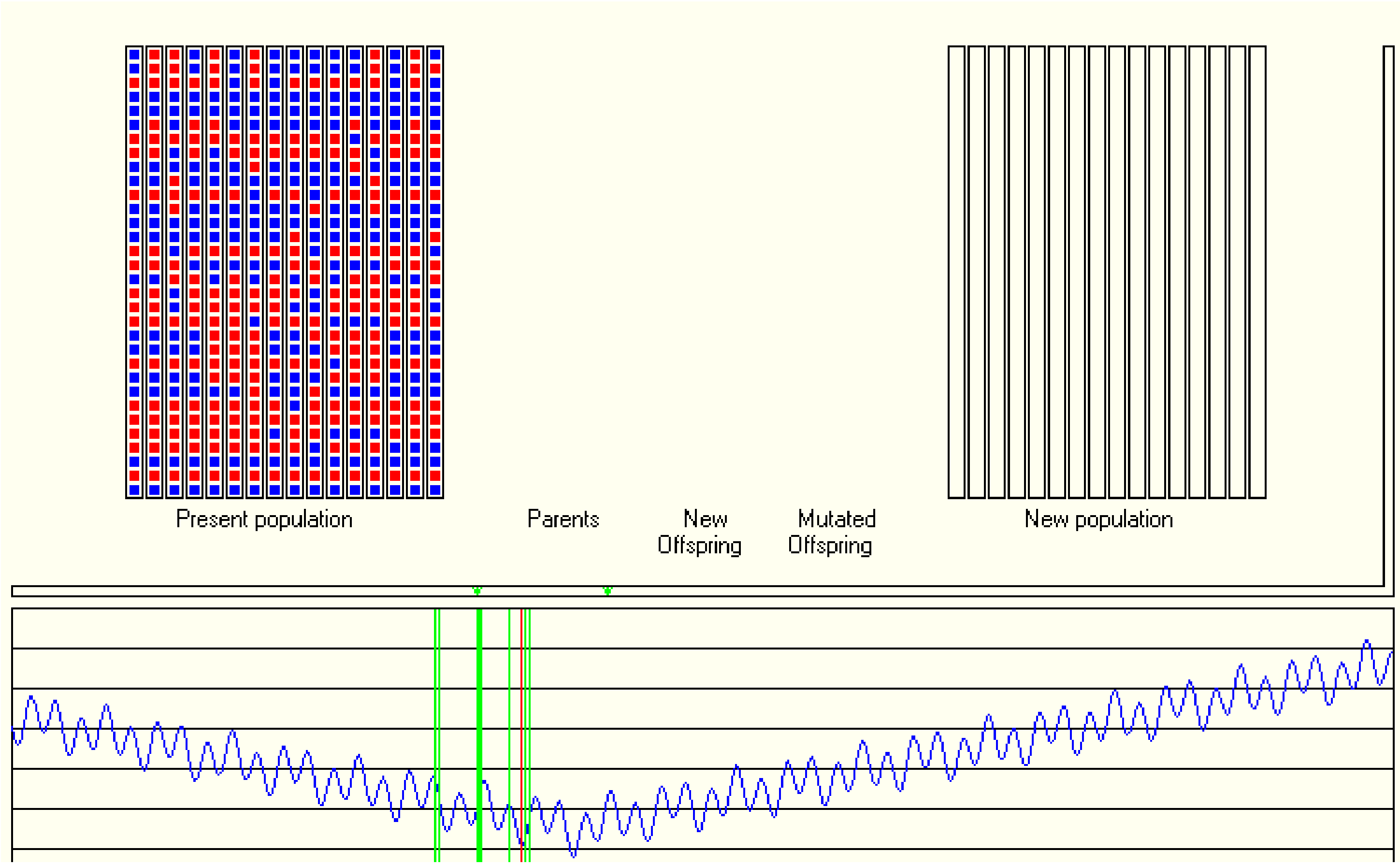
a po čase je uvedeným postupem vytvořena nová generace



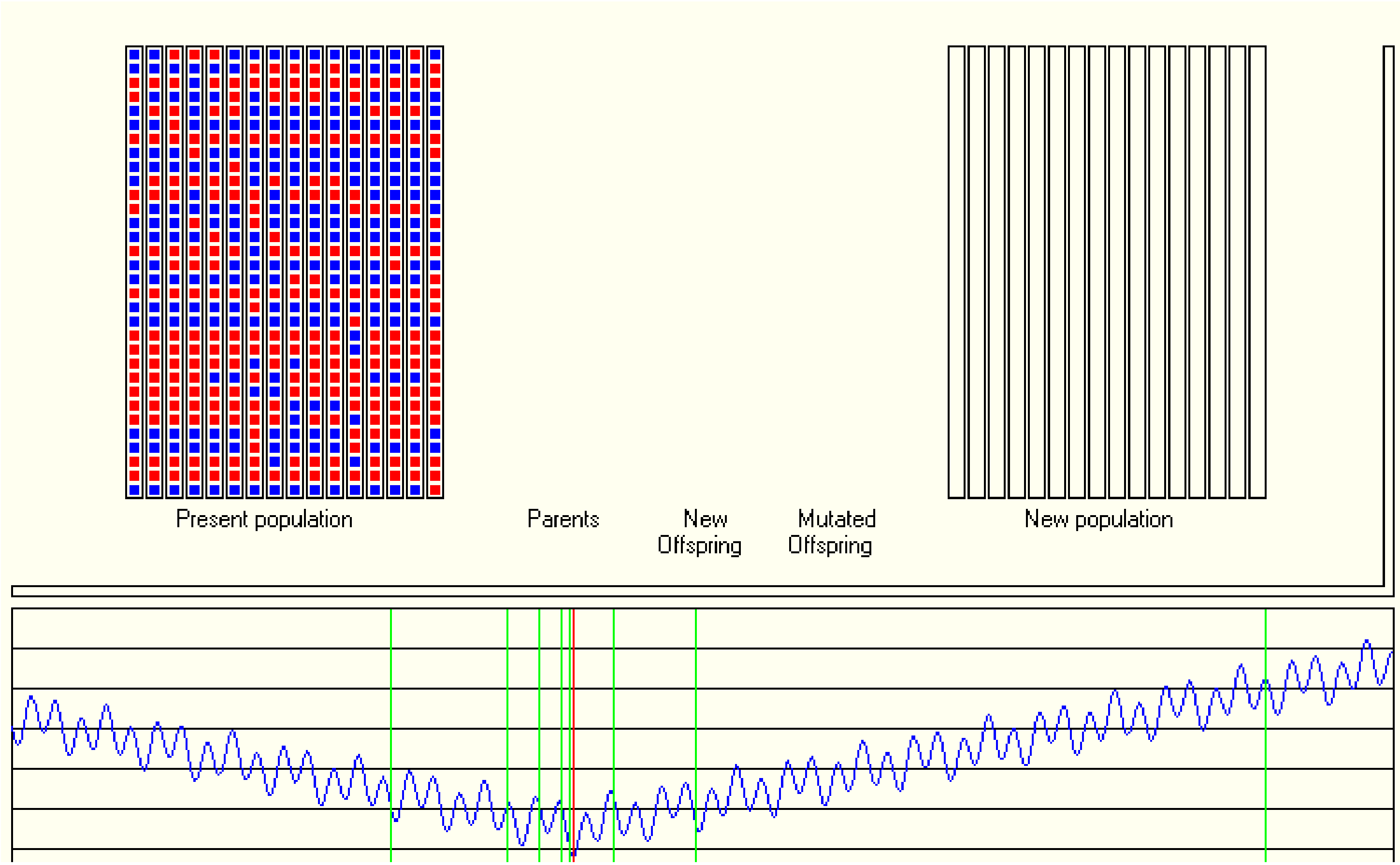
nová generace nahrazuje starou



po několika dalších generacích



závěrečná generace



Genetický algoritmus - pojmy

h jedinec (hypotéza)

f hodnotící funkce (angl. fitness) jedinců

- $f(h)$ obvykle udává kvalitu jedince h , t.j. chceme f maximalizovat.

t prahová maximální hodnota pro hodnotící funkci

p velikost populace

P_c pravděpodobnost křížení

P_m pravděpodobnost mutace

Genetický algoritmus

- **Inicializace**: H je náhodná počáteční populace
- **Ohodnocení**: pro každé $h \in H$, spočti $f(h)$
- Pokud $\max_h f(h) < t$ opakuj
  - **Pravděpodobnostní výběr** $(1 - P_c) \cdot p$ jedinců z H do H'.
  - **Křížení**:

    Pravděpodobnostní výběr $P_c \cdot \frac{p}{2}$ dvojic jedinců z H.

    Pro každou dvojici vytvoř dva potomky křížením.

    Přidej potomky do H'.
  - **Mutace**: Invertuj náhodně vybrané bity u jedinců z H' s pravděpodobností P_m.
  - **Aktualizace**: $H \leftarrow H'$
  - **Ohodnocení**: pro každé $h \in H$, spočti $f(h)$
- Navrať jedince $h \in H$ nabývajícího nejvyšší hodnoty $f(h)$

Metody náhodného výběru

- Proporcionálně k hodnotící funkci, t.j. $P(h_i) = \frac{f(h_i)}{\sum_{j=1}^{p} f(h_j)}$.

  Existuje nebezpečí sekupení všech jedinců blízko sebe.

- Pomocí turnaje.
  - Vyber náhodně dva jedince h_1, h_2. Každý jedinec má stejnou pravděpodobnost výběru.
  - S pravděpodobností P_t vyber jedince s vyšší hodnotou f, jinak vyber jedince s nižší hodnotou f.

- Podle pořadí.
  - Seřaď jedince podle jejich hodnotící funkce (sestupně).
  - Pravděpodobnost výběru jedince je inverzně proporcionální vzhledem k jeho pořadí.

Reprezentace jedinců

Volba vhodné reprezentace:

- Řetězec by měl nějakým způsobem odrážet vlastnosti objektu, který reprezentuje.
- Je žádoucí, aby všichni jedinci reprezentovali přípustná řešení problému, neboť vyřazování nepřípustných řešení může výrazně zpomalit algoritmus.

Jedinec může být například reprezentován

- řetězcem nul a jedniček - binární reprezentace nebo
- řetězcem čísel - číselná reprezentace nebo
- řetězcem písmen abecedy - znaková reprezentace
- stromem nějakých objektů (např. funkcí nebo příkazů programovacího jazyka) - genetické programování

Binární reprezentace

$(+)$ snadná implementace genetických operátorů

$(-)$ pro mnoho problémů není však přirozená

Příklady

Hledání minima funkce:

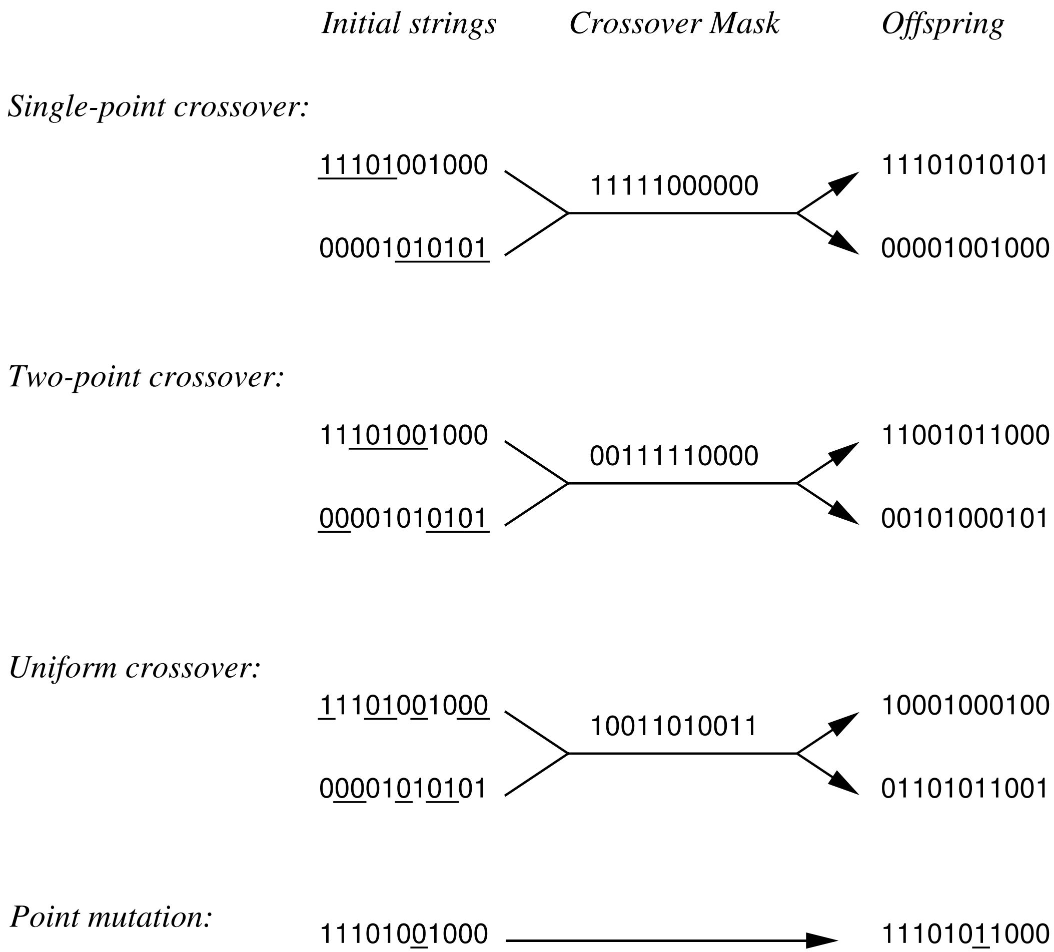
$f : N \mapsto \mathbb{R}$, kde N je množina celých čísel od 0 do 255.
Pro celá čísla použijeme binární řetězce délky 8.
Např. $23 = (00010111)$

Problém batohu (Knapsack problem):

- V místnosti jsou věci různé velikosti a různé ceny.
- Zloděj chce do batohu určité omezené kapacity zabalit věci tak, aby maximalizoval celkovou hodnotu věcí v batohu.

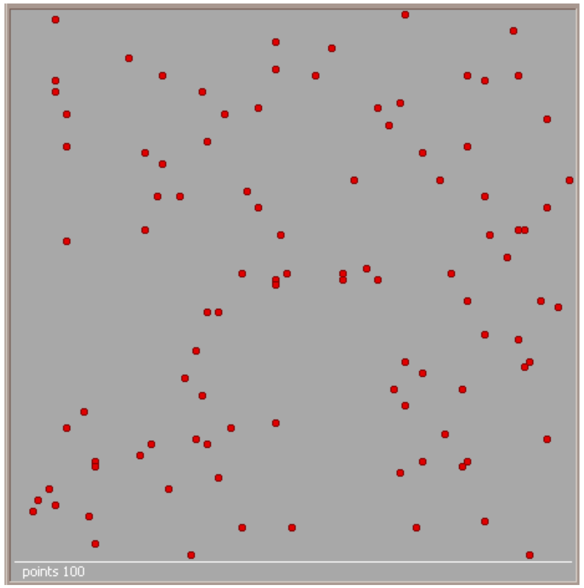
Každý bit v reprezentaci říká, zda-li odpovídající věc je nebo není v batohu.

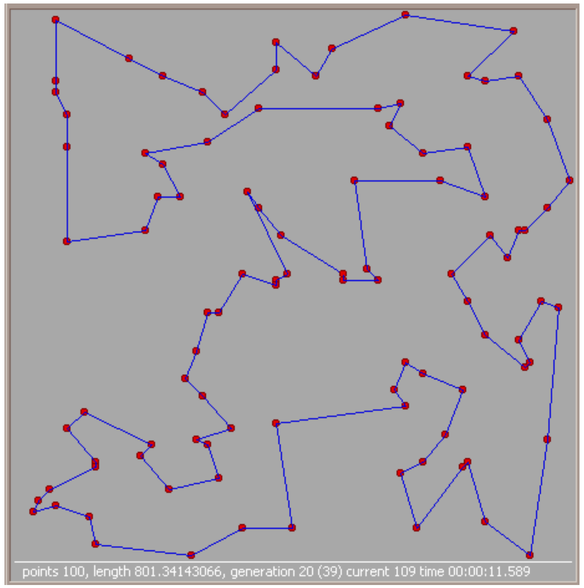
Způsoby křížení



Problém obchodního cestujícího (TSP)

- Obchodní cestující dostane seznam měst, která má navštívit.
- Jsou mu známy vzdálenosti mezi jednotlivými městy. Obchodní cestující má navštívit všechna města (právě jednou) a vrátit se do výchozího bodu.
- Cílem je minimalizovat celkovu vzdálenost, kterou urazí.





TSP - kódování a křížení

Kódování

- Každému městu je přiřazeno celé číslo.
- Města se v řetězci vyskytují v pořadí jakém jsou navštívena.
- Např. (9340125768)

Hladové křížení (angl. Greedy crossover)

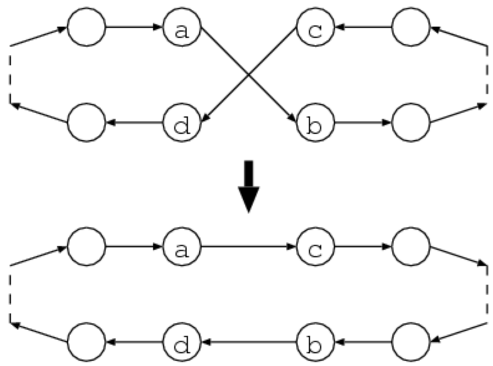
- Vyber první město jednoho rodiče.
- Porovnej druhá města u obou rodičů a vyber to, které je blíže k prvnímu vybranému.
- Jestliže město je již v řetězci, vyber město od druhého rodiče.
- Jestliže i toto město je již v řetězci vyber náhodně nějaké město, které v řetězci ještě není.
- Obdobně pokračuj pro třetí, čtvrté, ... město.

TSP - mutace

Hladové přehození (angl. Greedy swap)

- Vyber náhodně dvě města a prohoď je v řetězci.
- Pokud je nově vytvořená cesta kratší, přijmi mutaci, jinak zachovej původní cestu.
- Např. $(0\textcolor{red}{1}2\textcolor{blue}{3}456) \longrightarrow (0\textcolor{blue}{3}2\textcolor{red}{1}456)$.

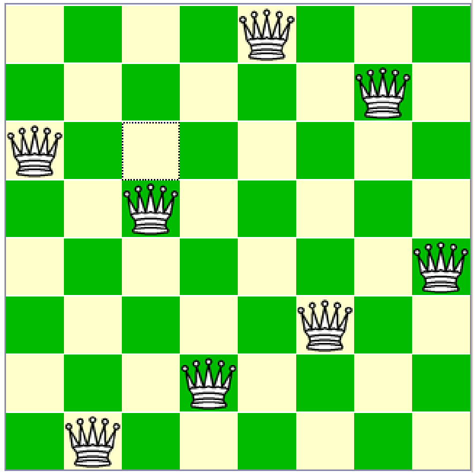
Mutace 2opt



Demo: `TSPApp.exe` 200 měst v kruhu - porovnání $p = 10$ a $p = 100$, heuristics (2opt mutace) 0 a 5.

Cvičení - problém osmi dam na šachovnici

Příklad řešení:



Úkolem je umístit osm dam na šachovnici 8×8, tak aby žádná dáma neohrožovala žádnou jinou.

Cvičení: Navrhněte genetický algoritmus pro řešení problému osmi dam. To znamená navrhnout:

- vhodnou reprezentaci řešení problému pomocí řetězce,
- hodnotící funkci,
- operátor křížení, a
- operátor mutace.

Genetické programování - příklad

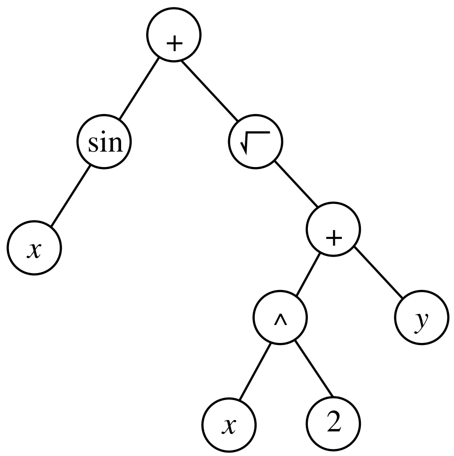
Aproximující funkce

Cílem je nalezení funkce, která by nejlépe aproximovala dané trojice hodnot

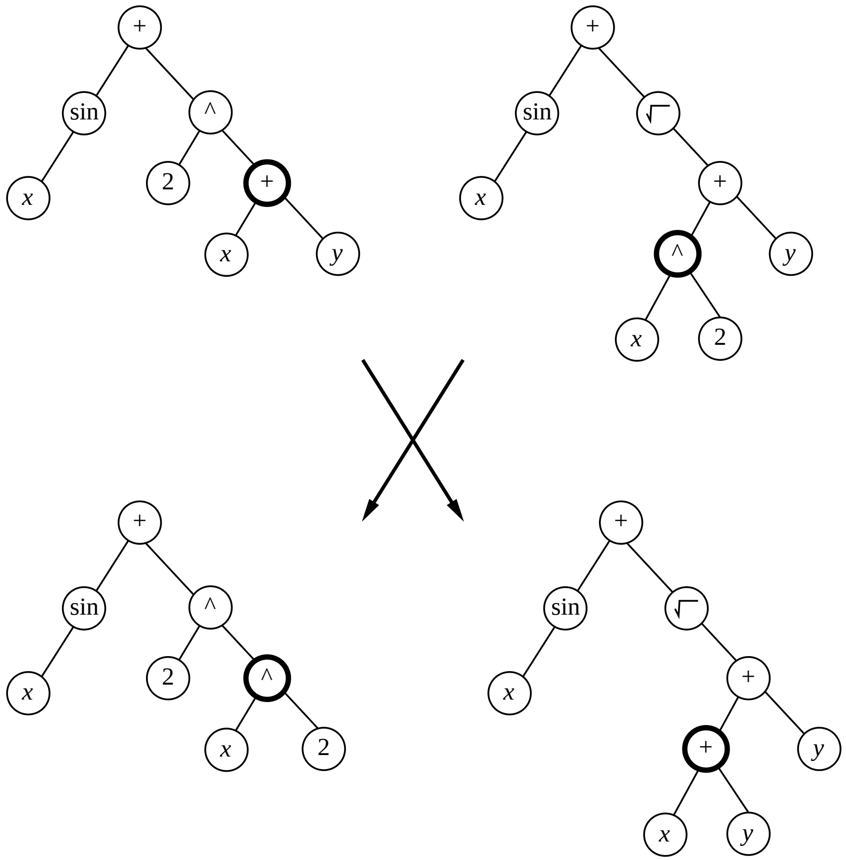
$$(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n).$$

T.j. pro daná první dvě čísla z trojice x_i, y_i by vrátila výstup z_i' co nejblíže třetímu číslu trojice z_i.

Jedinci jsou funkce ve stromu. Např. funkce $sin(x) + \sqrt{x^2 + y}$



Genetické programování - příklad křížení



Schémata

Schéma = řetězec obsahující 0, 1, * ("cokoliv")

- Příklad schématu:

$$10{*}{*}0{*}$$

- Jedinci odpovídající výše uvedenému schématu:

$$100000, 100001, 100100, 100101, 101000, 101001, 101100, 101101.$$

Charakteristika populace:

p	...	počet jedinců v populaci
$m_t(s)$	...	počet jedinců odpovídajících schématu s v čase t
$f(h)$	...	hodnota (fitness) jedince (hypotézy) h
$\bar{f}_t$	...	průměrná hodnota populace v čase t
$\mathcal{H}_t(s)$	...	množina jedinců odpovídajících schématu s v čase t

Věta o schématech (pro výběr)

$\bar{f}_t(s)$	$\ldots$	průměrná hodnota jedinců z množiny $\mathcal{H}_t(s)$
	=	$\frac{1}{m_t(s)} \cdot \sum_{h \in \mathcal{H}_t(s)} f(h)$
$E[m_{t+1}(s)]$	$\ldots$	očekávaný počet jedinců odpovídajících schématu s v čase $t+1$
$P(h)$	$\ldots$	pravděpodobnost výběru jedince h

$$P(h) \stackrel{df}{=} \frac{f(h)}{\sum_{i=1}^{p} f(h_i)} = \frac{f(h)}{p \cdot \bar{f}_t}$$

$$P(h \in \mathcal{H}_{t+1}(s)) = \sum_{h' \in \mathcal{H}_t(s)} \frac{f(h')}{p \cdot \bar{f}_t} = \frac{\bar{f}_t(s) \cdot m_t(s)}{p \cdot \bar{f}_t}$$

$$E[m_{t+1}(s)] = p \cdot P(h \in \mathcal{H}_{t+1}(s)) = \frac{\bar{f}_t(s)}{\bar{f}_t} \cdot m_t(s)$$

Věta o schématech
(jednobodové křížení a mutace)

P_c	...	pravděpodobnost aplikace operátoru křížení
ℓ	...	délka řetězce, který reprezentuje jedince
$d(s)$	...	vzdálenost mezi definovanými prvky schématu s nejvíce vlevo a nejvíce vpravo. Např. u $s = (***0*1*110**)$ je vzdálenost $\lvert 9-3 \rvert = 6$.
$o(s)$	...	počet definovaných prvků ve schématu s
$\left(1 - P_c \cdot \frac{d(s)}{\ell}\right)$	...	dolní odhad pravděpodobnosti, že křížení nenaruší schéma
$(1-P_m)^{o(s)}$	...	pravděpodobnost, že mutace nenaruší schéma

$$E[m_{t+1}(s)] \;\; \geq \;\; \frac{\bar{f}_t(s)}{\bar{f}_t} \cdot m_t(s) \cdot \left(1 - P_c \cdot \frac{d(s)}{\ell}\right) \cdot (1-P_m)^{o(s)}$$

Vlastnosti genetických algoritmů

(+) Dají se použít pro řešení problémů jinak těžko řešitelných (například, když interakce mezi jednotlivými částmi jsou těžko popsatelné).

(+) Většinou neuváznou v lokálním maximu.

(+) Vždy poskytnou nějaké řešení.

(+) Jsou snadno implementovatelné a paralerizovatelné.

(−) Nemáme žádnou záruku, že nalezené řešení je optimální.

(−) Někdy mohou být velmi pomalé (obzvášť pokud nejsou dobře navrženy reprezentace jedinců a operátory křížení a mutace).

(−) Vyžadují vhodné nastavení většího množství parametrů algoritmu (např. P_C, P_M, p).

Stručná historie genetických algoritmů

- **1960**: Ingo Rechenberg představuje myšlenku evolučních výpočtů ve své práci ”Evolution strategies”
- **1975**: John Holland poprvé popisuje genetický algoritmus a vydává svoji knihu ”Adaptation in Natural and Artificial Systems”
- **1992**: John Koza použil genetický algoritmus pro vývoj programů, které mají plnit určité zadané úlohy. Svoji metodu nazval genetické programování.

Příklady aplikací genetických algoritmů

dle encyklopedie `wordIQ.com`

- Optimalizace nákládání kontejnerů.
- Učení chování robotů.
- Optimalizace infrastruktury pro mobilní komunikaci.
- Optimalizace struktury molekul.
- Návrh uspořádání výrobních hal.
- Různé plánovací problémy (např. když jednotlivé úlohy jsou navzájem závislé).
- Predikce akciových trhů.