Lokální prohledávání

7. března 2022

1. Lokální prohledávání obecně
2. Tabu prohledávání
3. Simulované žíhání
4. Genetické algoritmy

Konstruktivní vs. lokální metody

Konstruktivní metody

- začneme s prázdným rozvrhem
- do rozvrhu přidáváme postupně jednotlivé úlohy tak, aby byl rozvrh stále konzistentní

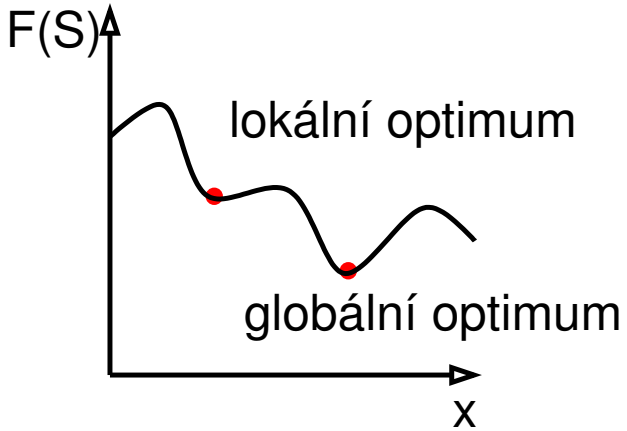
Lokální prohledávání

- začneme s úplným nekonzistentním rozvrhem
  - triviálně: s náhodně vygenerovaným
- snažíme se najít lepší „podobný" rozvrh lokálními změnami
- kvalitu rozvrhu posuzujeme optimalizačními kritérii
  - např. makespan
- optimalizační kritéria vyhodnocují také konzistenci rozvrhu
  - např. počet porušených precedenčních omezení

Hybridní přístupy

- kombinace obou metod

Algoritmus lokálního prohledávání



1. Inicializace
   - $k = 0$
   - výběr iniciálního rozvrhu S_0
   - zaznamenání dosud nejlepšího rozvrhu:
     $S_{best} = S_0$ a $best\_cost = F(S_0)$
2. Výběr a aktualizace
   - výběr rozvrhu z okolí: $S_{k+1} \in N(S_k)$
   - pokud kriterium přijetí rozvrhu nesplňuje
     žádný prvek $N(S_k)$, pak algoritmus končí
   - jestliže $F(S_{k+1}) < best\_cost$ pak
     $S_{best} = S_{k+1}$ a $best\_cost = F(S_{k+1})$
3. Ukončení
   - jestliže platí podmínky ukončení pak algoritmus končí
   - jinak $k = k + 1$ a skok na krok 2.

Jeden stroj + nepreemptivní úlohy

Reprezentace rozvrhu

- permutace n úloh
- příklad se šesti úlohami: 1,4,2,6,3,5

Definice okolí

- párová výměna sousedních úloh
  - příklad: 1,4,2,6,3,5 se změní např. na 1,4,2,6,5,3
  - možných rozvrhů v okolí: $n - 1$
- nebo výběr libovolné úlohy v rozvrhu a umístění na libovolnou pozici
  - příklad: z 1,4,2,6,3,5 náhodně vybereme 4 a dáme ji jinam: 1,2,6,3,4,5
  - možných rozvrhů v okolí: $\leq n(n-1)$

Iniciální řešení & podmínky ukončení

Iniciální řešení generujeme

- náhodně
- heuristicky např. konstruktivním algoritmem jako jsou řídící pravidla

Podmínka ukončení typicky

- zadaný počet iterací
- omezená doba běhu
- počet porovnání účelové funkce
- žádné zlepšení nezískáno po daném počtu iterací

Výběr rozvrhu z okolí

- Lokální změna
  - úprava rozvrhu výběrem rozvrhu z okolí
- Výběr rozvrhu z okolí, tj. prohledání okolí a výběr vhodného kandidáta
  - náhodný výběr
  - výběr nejslibnějšího kandidáta
    - vybíráme rozvrh s nejlepší hodnotou účelové funkce v okolí
    - snaha o zlepšení hodnoty účelové funkce

Kritérium výběru rozvrhu

Kritérium výběru rozvrhu =
kriterium přijetí/odmítnutí rozvrhu

- akceptovat vždy lepší rozvrh?
- někdy akceptovat i horší rozvrh?

Metody akceptování horšího rozvrhu

- pravděpodobnostní
  - náhodná procházka: s malou pravděpodobností (např. 0.01) akceptujeme i horší rozvrh
  - simulované žíhání
- deterministická
  - tabu prohledávání: udržujeme tabu seznam několika posledních stavů/změn, které jsou pro další výběr nepřípustné

Tabu prohledávání

Deterministické kritérium přijetí/odmítnutí rozvrhu

Udržován tabu seznam několika posledních změn v rozvrhu

- tabu seznam = seznam zakázaných změn
- každá nová změna je umístěna na vrchol tabu seznamu
  - př. uchovávané změny: výměna úloh j a k
- okolí omezeno na rozvrhy, které nepožadují změnu z tabu seznamu
  - zabraňuje cyklení
  - příklad triviálního cyklení:
    první krok: prohození úloh 3 a 4, druhý krok: prohození úloh 4 a 3
- pevná délka seznamu (typicky: 5-9)
  - nejstarší změny z tabu seznamu odstraněny
  - příliš malá délka: nebezpečí cyklení
  - příliš velká délka: může omezit prohledávání příliš

Aspirační kritérium

- určuje, kdy je možné akceptovat i změny v tabu seznamu
- př. změna z tabu seznamu povolena, pokud zlepšeno $F(S_{best})$

Algoritmus tabu prohledávání

1.
   - $k = 1$
   - výběr iniciálního rozvrhu S_1 použitím heuristiky, $S_{best} = S_1$
2.
   - výběr $S_c \in N(S_k)$
   - jestliže je změna $S_k \to S_c$ zakázána
     protože je v tabu seznamu
     a není splněno aspirační kritérium
     pak běž na krok 2
3.
   - jestliže změna $S_k \to S_c$ není zakázána tabu seznamem
     nebo je splněno aspirační kritérium
     pak $S_{k+1} = S_c$
     ulož reversní změnu na vrchol tabu seznamu
     posuň další pozice v tabu seznamu o pozici níže
     smaž poslední položku z tabu seznamu
   - jestliže $F(S_c) < F(S_{best})$ pak $S_{best} = S_c$
4.
   - $k = k + 1$
   - jestliže platí podmínka ukončení pak konec
     jinak běž na krok 2.

Příklad: tabu seznam

- Uvažujte rozvrhovací problém s $1||\sum w_j T_j$
  - opakování: $T_j = \max(C_j - d_j, 0)$

úlohy	1	2	3	4
p_j	10	10	13	4
d_j	4	2	1	12
w_j	14	12	1	12

- Okolí: všechny rozvrhy získané párovou výměnou sousedních úloh
- Výběr rozvrhu z okolí: vybereme nejlepší rozvrh
- Tabu seznam: páry úloh (j, k), které byly přehozeny při posledních dvou změnách

$S_1 = (2, 1, 4, 3)$
$F(S_1) = \sum w_j T_j = 12 \cdot 8 + 14 \cdot 16 + 12 \cdot 12 + 1 \cdot 36 = 500 = F(S_{best})$
$F(1, 2, 4, 3) = 480$
$F(2, \underline{4}, \underline{1}, 3) = 436 = F(S_{best})$
$F(2, 1, 3, 4) = 652$
Tabu seznam: $\langle(1, 4)\rangle$

Příklad: tabu seznam (pokračování)

$S_2 = (2, 4, 1, 3), F(S_2) = 436$
$F(\underline{4}, \underline{2}, 1, 3) = 460$
$F(2, 1, 4, 3)(= 500)$ tabu!
$F(2, 4, 3, 1) = 608$
Tabu seznam: $\langle (2, 4), (1, 4) \rangle$

$S_3 = (4, 2, 1, 3), F(S_3) = 460$
$F(2, 4, 1, 3)(= 436)$ tabu!
$F(4, \underline{1}, \underline{2}, 3) = 440$
$F(4, 2, 3, 1) = 632$
Tabu seznam: $\langle (2, 1), (2, 4) \rangle$

$F_4 = (4, 1, 2, 3), F(S_4) = 440$
$F(\underline{1}, \underline{4}, 2, 3) = 408 = F(S_{best})$
$F(4, 2, 1, 3)(= 460)$ tabu!
$F(4, 1, 3, 2) = 586$
Tabu seznam: $\langle (4, 1), (2, 1) \rangle$
$F(S_{best}) = 408$

Cvičení: tabu seznam

- Uvažujte rozvrhovací problém s $1||\sum w_j T_j$

úlohy	1	2	3	4
p_j	10	10	13	4
d_j	4	2	1	12
w_j	14	12	1	12

- Aplikujte tabu prohledávání pro iniciální řešení $(2, 1, 4, 3)$
- Okolí: všechny rozvrhy získané párovou výměnou sousedních úloh
- Výběr rozvrhu z okolí: vybereme nejlepší rozvrh
- Proveďte čtyři iterace
- Tabu seznam: páry úloh (j, k), které byly přehozeny při
  1. jedné poslední změně — výsledek: $F(S_{best}) = 408$
  2. třech posledních změnách — výsledek: $F(S_{best}) = 436$

Simulované žíhání: principy

- Myšlenka: simulace procesu ochlazování kovů
  - na začátku při vyšší teplotě atomy více kmitají a pravděpodobnost změny krystalické mřížky je vyšší
  - postupným ochlazováním se atomy usazují do „nejlepší polohy" s nejmenší energií a pravděpodobnost změny je menší

  $\Rightarrow$ na začátku je tedy pravděpodobnost toho, že akceptujeme zhoršování řešení, vyšší
- Aplikace u lokálního prohledávání
  - lepší nebo stejné řešení akceptováno
  - algoritmus umožňuje akceptovat horší řešení, abychom unikli z lokálního minima
  - pravděpodobnost přijetí horšího řešení se postupně snižuje

Simulované žíhání: realizace

- Parametr chlazení t
  - t je iniciálně vysoké: hodně změn je akceptováno
  - t postupně klesá: horší změny jsou skoro vždy odmítnuty
- Rozdíl mezi kvalitou nového a existujícího řešení
  - $\Delta c = F(S_{new}) - F(S_{old})$
- Metropolisovo kritérium
  - předpoklad: minimalizace F
  - lepší nebo stejné řešení akceptováno: $\Delta c \leq 0$, tj. $F(S_{new}) \leq F(S_{old})$
  - pravděpodobnostní přijetí/odmítnutí rozvrhu:
    horší řešení ($\Delta c > 0$) akceptováno pokud

$$U < e^{-\Delta c/t}$$

  - U náhodné číslo z intervalu $(0, 1)$

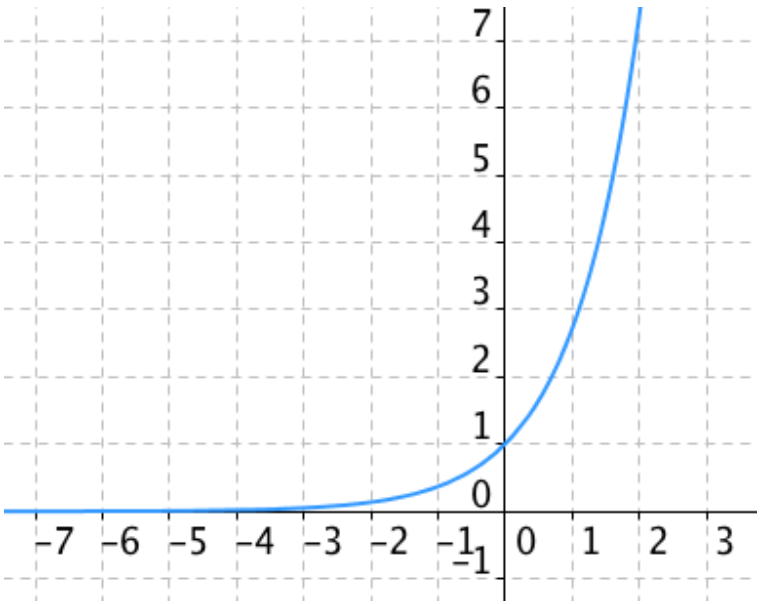
Metropolisovo kritérium

Horší řešení ($F(S_{new}) - F(S_{old}) = \Delta c > 0$) akceptováno pokud

$$U < e^{-\Delta c/t}$$

- U náhodné číslo z intervalu $(0, 1)$
- $\Delta c > 0$, tj. $-\Delta c < 0$
- pokud klesá t nebo klesá $-\Delta c$, tak klesá i $e^{-\Delta c/t}$
- pomůcka: porovnej $e^{-10/100}$ vs. $e^{-100/100}$ a $e^{-10/100}$ vs. $e^{-10/1}$

Graf e^x



Algoritmus simulovaného žíhání

1.
   - $k = 1$
   - výběr iniciálního rozvrhu S_1 použitím heuristiky, $S_{best} = S_1$
   - nastavení iniciální teploty $t_0 > 0$
   - výběr funkce redukce teploty $\alpha(t)$
2.
   - výběr $S_{new} \in N(S_k)$
   - jestliže $F(S_{new}) < F(S_{best})$ pak $S_{best} = S_{new}$, $S_{k+1} = S_{new}$
     jinak
     jestliže $F(S_{new}) \leq F(S_k)$ pak $S_{k+1} = S_{new}$
     jinak
     generuj náhodné číslo U_k
     jestliže $U_k < e^{\frac{F(S_k) - F(S_{new})}{t}}$ pak $S_{k+1} = S_{new}$
     jinak $S_{k+1} = S_k$
3.
   - $t_k = \alpha(t_k)$
   - $k = k + 1$
   - jestliže platí podmínka ukončení pak konec
     jinak běž na krok 2.

Příklad: simulované žíhání

- Opakování: $T_j = \max(C_j - d_j, 0)$
- Uvažujte rozvrhovací problém s $1||\sum w_j T_j$

úlohy	1	2	3	4
p_j	9	9	12	3
d_j	10	8	5	28
w_j	14	12	1	12

- Použijte simulované žíhání s iniciálním rozvrhem $(3, 1, 4, 2)$
- Okolí: všechny rozvrhy, které dostaneme sousedními párovými výměnami
- Výběr rozvrhu z okolí náhodně
- Zvolte $\alpha(t) = 0.9 \times t$
- $t_0 = 0.9$
- Použijte následující náhodná čísla: $0.17, 0.91, \ldots$

Příklad: simulované žíhání (řešení)

$S_{best} = S_1 = (3, 1, 4, 2)$
$F(S_1) = \sum w_j T_j = 1 \cdot 7 + 14 \cdot 11 + 12 \cdot 0 + 12 \cdot 25 = 461 = F(S_{best})$
$t_0 = 0.9$

$S_{new} = (1, 3, 4, 2)$
$F(S_{new}) = 316 < F(S_{best})$
$S_{best} = (1, 3, 4, 2)$
$F(S_{best}) = 316$
$S_2 = (1, 3, 4, 2)$
$t = 0.9 \times 0.9 = 0.81$

$S_{new} = (1, 3, 2, 4)$
$F(S_{new}) = 340 > F(S_{best}) = 316$
$F(S_{new}) > F(S_2) = 316$
$U_1 = 0.17 > e^{-(340-316)/0.81} = 1.35 \times 10^{-13}$
$S_3 = S_2 = (1, 3, 4, 2)$
$t = 0.729$

$S_{new} = (1, 4, 3, 2)$
$F(S_{new}) = 319 > F(S_{best}) = 316$
$F(S_{new}) > F(S_3) = 316$
$U_3 = 0.91 > e^{-(319-316)/0.729} = 0.016$
$S_4 = S_3 = (1, 3, 4, 2)$
$t = 0.6561$
...

Praktické úvahy

- Iniciální teplota
  - musí být „vysoká”
  - kritérium přijetí rozvrhu: 40%–60% vykazuje dobré výsledky v mnoha situacích
- Parametr chlazení
  - několik změn při dané teplotě
  - jedna změna při každé teplotě

  $t = \alpha \times t$

  $t = \frac{t}{1+\beta t}$

  α typicky v intervalu [0.9,0.99]

  β typicky blízké k 0

Genetické algoritmy

Srovnání

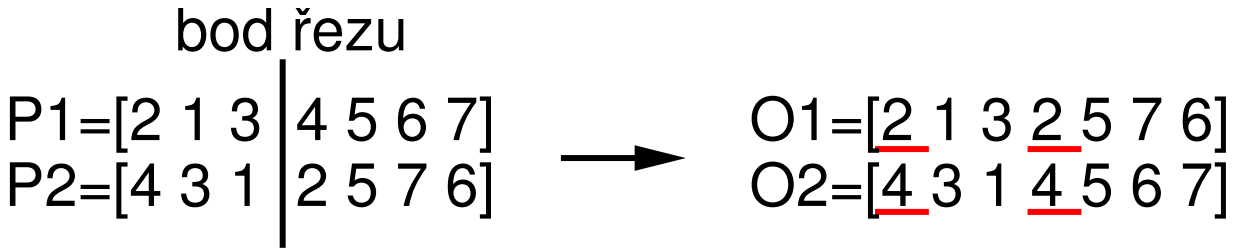
- simulované žíhání, tabu prohledávání
  - jedno řešení je přenášeno z jedné iterace do druhé
- genetické algoritmy
  - udržována populace (několika přenášených) řešení

Genetické algoritmy

- rozvrhy jsou jednotlivci (chromozomy), kteří tvoří populaci
- rozhodovací proměnná (např. čas jedné úlohy) je gen
- hodnota rozhodovací proměnné (např. konkrétní čas) je alela
- každý jednotlivec je vyhodnocen kritériem vhodnosti *(fitness)*
- někteří jednotlivci mutují
- jednotlivci jsou vybráni k reprodukci křížením a mají děti tvořící následující populaci
- nejvhodnější jedinci přežijí do další populace

Reprodukce

- Křížení *(crossover)*: kombinování posloupnosti operací na jednom stroji v jednom rodičovském rozvrhu s posloupností operací na jiném stroji u jiného rodiče
- Operátor jednoduchého křížení není užitečný!

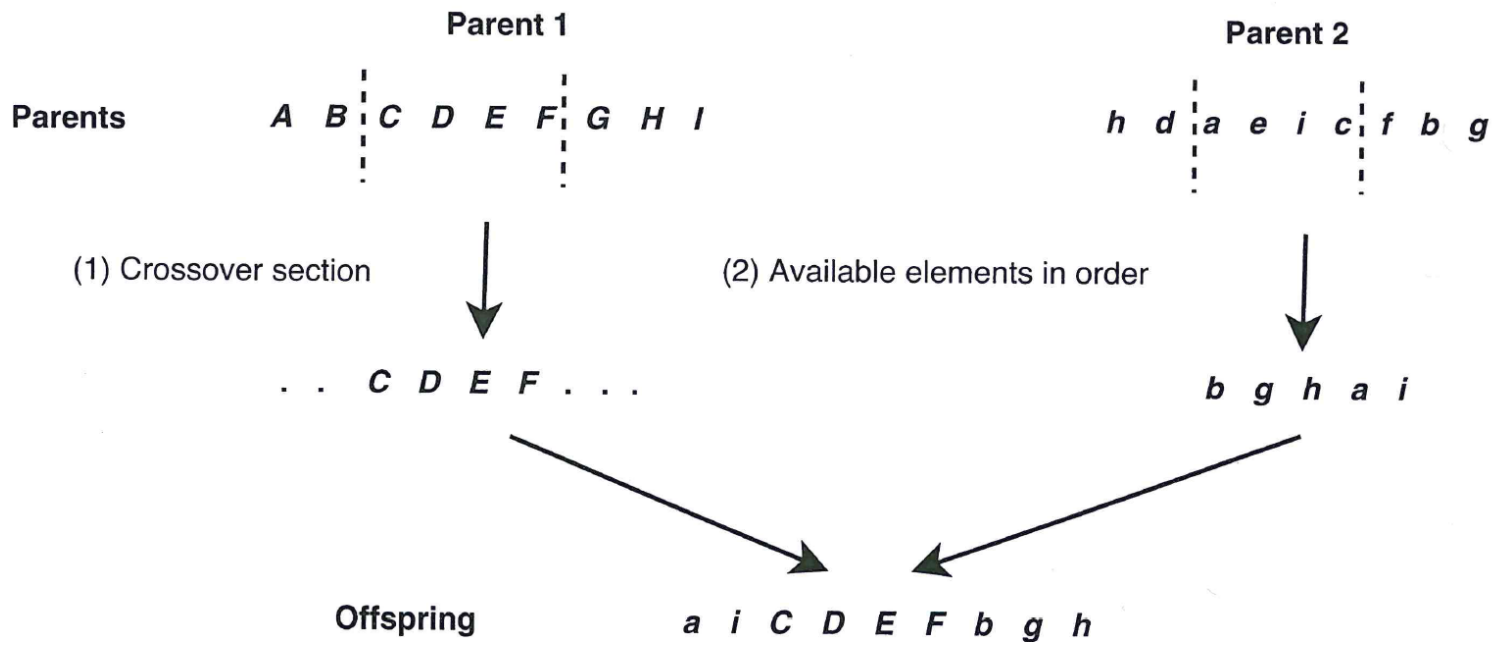


- Každá alela (hodnota) se musí výskytnout v jedinci právě jednou
  - používány různé formy mapování

Křížení dané pořadím

Křížení dané pořadím (Order crossover, OX)

1. Vybrány náhodně dva body křížení
2. Z rodiče 1 hodnoty mezi nimi zkopírovány na stejné pozice v potomkovi
3. Z rodiče 2 začneme od druhého bodu křížení vybírat prvky, které již nebyly vybrány z rodiče 1, a dávame je do potomka od 2. bodu křížení



Mutace

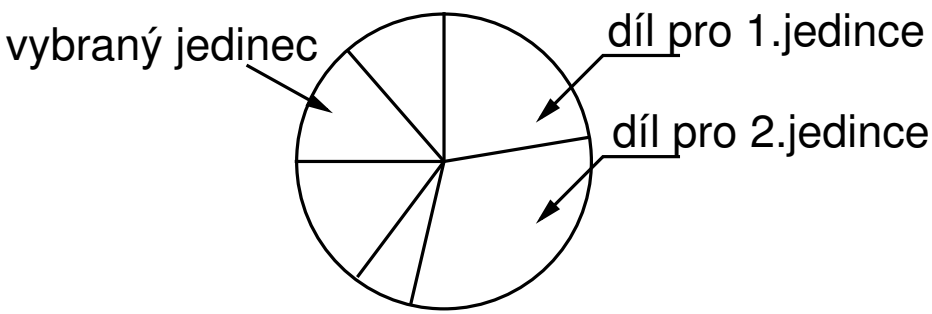
- Mutace umožňuje genetickému algoritmu prohledávat prostor nedosažitelný operátorem křížení
- Párová výměna sousedů v posloupnosti

$$[1, 2, \ldots, n] \rightarrow [2, 1, \ldots, n]$$

- Mutace výměnou: změna dvou náhodně vybraných prvků permutace
- Mutace posunem: přesun náhodně vybraného prvku o náhodný počet míst doleva nebo doprava
- Mutace promícháním podseznamu: výběr dvou bodů v řetězci náhodně a náhodná permutace prvků mezi těmito dvěma pozicemi

Výběr

- Ruletové kolo
  - šance na reprodukci jsou proporcionálně závislé na vhodnosti jedince
  - velikost každého dílu ruletového kola záleží na vhodnosti konkrétního jedince
  - př. kritérium vhodnosti pro 6 jedinců: 5,7,1,3,3,3
    první díl má velikost 5, druhý 7, třetí 1, čtvrtý 3, páty 3, šestý 3



- Kroky pro ruletové kolo
  1. sečti vhodnost všech jednotlivců populace: TF
     - př. $TF = 5 + 7 + 1 + 3 + 3 + 3 = 22$
  2. generuj náhodné číslo m mezi 0 a 22
  3. vrať prvního jednotlivce populace do jehož dílu spadá m
     - čím větší vhodnost jedince, tím větší pravděpodobnost, že se na něj strefíme

Výběr (pokračování)

Turnajový výběr

1. výběr jednoho jedince
   - náhodně vyber skupinu t jedinců z populace
   - vyber nejlepšího jedince
2. pokud potřebujeme vybrat k jedinců, pak postup opakujeme k-krát

Jak garantovat, že nejlepší člen/členové populace přežijí?

Elitářský model:

- nejlepší člen populace je vybrán jako člen následující populace

  nebo
- nejlepší potomci a rodiče jsou vybíráni do následující populace

Základní implementace genetického algoritmu

1.
   - $k = 1$
   - vyber N iniciálních rozvrhů $S_{1,1}, \ldots, S_{1,N}$ použitím heuristiky
   - vyhodnoť jednotlivce populace
2.
   - vytvoř nové jednotlivce spojením jednotlivců současné populace pomocí křížení a mutace
   - smaž členy existující populace, aby udělali místo novým jednotlivcům
   - vyhodnoť nové jednotlivce a přidej je do populace $S_{k+1,1}, \ldots, S_{k+1,N}$
3.
   - $k = k + 1$
   - jestliže platí podmínka ukončení pak vrať nejlepšího jedince jako řešení a skonči jinak běž na krok 2.

Příklad: genetické algoritmy

- Uvažujte rozvrhovací problém s $1||\sum T_j$

úlohy	1	2	3	4	5
p_j	4	3	7	2	2
d_j	5	6	8	8	17

- Velikost populace: 3
- Výběr
  - v každé populaci se reprodukuje nejvhodnější jedinec
    - triviální verze pro snadný výpočet!
  - pro reprodukci je použita párová výměna sousedů
  - počet možných potomků: 4
    pro reprodukci náhodně vybrán jeden z nich
  - potomek nahradí nejhoršího rodiče
- Iniciální populace: náhodné posloupnosti permutací

Příklad: genetické algoritmy (řešení)

Populace 1	Jedinec	25314	14352	12345
	Cena	25	17	16

Vybraný jedinec: 12345 s potomkem 13245, cena 20

Populace 2	Jedinec	13245	14352	12345
	Cena	20	17	16

Vybraný jedinec: 12345 s potomkem 12354, cena 17

Populace 3	Jedinec	12354	14352	12345
	Cena	17	17	16

Vybraný jedinec: 12345 s potomkem 12435, cena 11

Populace 4	Jedinec	14352	12345	12435
	Cena	17	16	11

Vybraný jedinec: 12435 – optimální řešení

Nevýhoda takto jednoduchého nastavení algoritmu:
výběr nejlepšího jedince způsobuje opakovanou reprodukci nejlepšího jedince, dokud není nahrazen lepším potomkem

Praktické úvahy

- Velikost populace
  - malá populace riskuje příliš malé pokrytí prohledávacího prostoru
  - velká populace má velké výpočetní nároky
  - dle empirických výsledků
    - velikost populace kolem 30 často adekvátní
    - velikost populace 20-100 je běžná
- Mutace: obvykle použita s velmi malou pravděpodobností
  - např. mutace jednoho genu jedince