Evoluční algoritmy - cvičení 8

Permutační GA pro řešení TSP

Martin Mézl

Ústav biomedicínského inženýrství
VUT v Brně
Brno

mezl@feec.vutbr.cz

ZS 2013

Osnova

Harmonogram cvičení

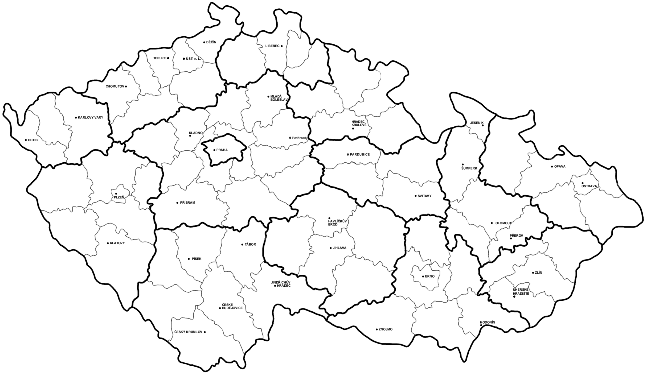
- cv. 1 - úvod do matematické optimalizace, neevoluční techniky
- cv. 2 - Newtonova metoda, stochastické metody
- cv. 3 - simplexová metoda
- cv. 4 - úvod do genetických algoritmů (GA)
- cv. 5 - rozšíření binárních GA
- cv. 6 - GA se spojitým vyjádřením hodnot
- cv. 7 - problém obchodního cestujícího (TSP)
- cv. 8 - permutační GA pro řešení problému TSP
- cv. 9 - řešení TSP s využitím optimalizace pomocí mravenčí kolonie (ACO)
- cv. 10 - samostatná práce
- cv. 11 - optimalizace pomocí roje částic (PSO), světluškový algoritmus
- cv. 12 - zápočtový test

Problém obchodního cestujícího

- neboli *Travelling salesman problem*
- je dáno N měst a úkolem obchodního cestujícího je navštívit všechna města a vrátit se do počátečního města tak, aby celková trasa byla nejkratší
- počet všech řešení problému prudce narůstá s počtem měst - nemůžeme řešit pomocí úplného prohledávání
- heuristická metoda - metoda nejbližšího souseda (NN)
- metoda simulovaného žíhání (SA)
- použitelné výsledky - u SA možnost dosažení globálního extrému

Problém obchodního cestujícího - ukázkový příklad (1)

- je dáno 35 (resp. 13) měst ČR, známe vzájemné vzdálenosti mezi městy (kilometrovník)
- Kolik je možných unikátních řešení tohoto problému?



Problém obchodního cestujícího - ukázkový příklad (2)

- nový úkol - je dáno 263 měst v ČR
- vzdálenosti počítány jako eukleidovské (v pixelech)

Reprezentace dat - připomenutí

- každá trasa může být reprezentována jako permutace vektoru $\{1, 2, 3, \ldots, N\}$
- permutace = každé město navštíveno právě jednou, z posledního města půjdu na začátek
- počet všech permutací je roven tohoto vektoru je $N!$
- v tomto počtu jsou zahrnuty některé trasy vícekrát - stejné trasy s rozdílným poč. bodem a trasy po směru hodinových a proti směru hodinových ručiček

Počet řešení TSP

Pokud odpočítáme duplicitní trasy v permutacích, můžeme počet řešení vyjádřit jako:

$$M = \frac{(N-1)!}{2}$$

Permutační GA

- vývojový diagram stejný jako na předchozích cvičeních - tzn. funkční bloky výběr jedinců, křížení a mutace
- pracujeme s populací jedinců - každý jedinec je reprezentován jako permutace vektrou $\{0, 1, 2, \ldots, N\}$
- populace má velikost MxN, kde M je počet jedinců, N počet měst
- počet jedinců volíme v závislosti na složitosti řešeného problému, pro 35 měst volíme 50-200 jedinců
- bloky křížení a mutace jsou definovány odlišně

Kvalita jedince

- kvalita jedince je úměrná celkové uražené vzdálenosti
- hledáme minimální vzdálenost - tzn. funkci kvality musíme přepočítat
- pro funcki kvality platí:

$$kvalita = (-vzd + max(vzd))^k,$$

kde *vzd* je celková vzdálenost jedince, k je kladná konstanta, typicky z intervalu $(2, 10)$

- umocnění na mocninu k lépe rozlišuje mezi dobrými a špatnými jedinci

Permutační křížení a mutace

- jednobodové nebo dvoubodové křížení nelze použít, protože výsledkem křížení by nebyly permutace
- mutace ve formě inverze bitu také nelze použít
- v nejjednodušší podobě definujeme křížení jako náhodné promíchání části řetězce (ve skutečnosti to není křížení, protože upravuje pouze jednoho jedince)
- mutace je definována jako záměna dvou pozic v řetězci
- pravděpodobnost křížení a mutace má stejný význam

Křížení:

$$(165|\boxed{423}|87)$$

$$(165|\boxed{342}|87)$$

Mutace:

$$(16\boxed{5}342\boxed{8}7)$$

$$(16\boxed{8}342\boxed{5}7)$$

PMX křížení

- partially matched crossover (PMX)
- skutečné křížení - ze dvou rodičovských jedinců tvořím dva potomky
- spočívá v dvoubodovém překřížení a následné detekci a korekci konfliktů - tedy čísel (měst), které se opakují
- do pozic, kde se čísla opakují umístíme náhodně zvolené číslo, které nebylo použito tak, aby výstupem byla opět permutace vektoru $\{1, 2, \ldots, N\}$
- lepší výsledky než výše uvedená metoda křížení

PMX křížení - příklad

(1) Dvoubodové křížení:
(156|423|87) - rodič 1
(271|684|35) - rodič 2
(156|684|87) - potomek 1
(271|423|35) - potomek 2

(2) Detekce konfliktů:
(156|684|87) - 6 a 8 v konfliktu, čísla 2 a 3 nepoužita
(271|423|34) - 2 a 3 v konfliktu, čísla 6 a 8 nepoužita

(3) Náhrada konfliktů:
(153|684|27) - náhodně zvolím z dostupných čísel
(671|423|84)

(4) Takto křížíme s určitou pravděpodobností celou populaci

"Hladové" křížení

- greedy crossover
- opět skutečné křížení - dva rodiče a dva potomci
- využívám i informaci o vzdálenostech dvou sousedních měst v řetězci - heuristika
- křížím opět s pravděpodobností křížení
- Postup:
  1. náhodně vyberu počáteční město
  2. vypíšu město na pozici napravo od tohoto města v obou rodičích a zjistím jejich vzdálenosti od poč. města
  3. město, které má menší vzdálenost, zapíši do výstupního řetězce (potomek 1)
  4. opakuji body 2 a 3, tedy zjistím vzdálenosti měst napravo od posledně přidaného města v obou řetězcích a bližší města zapíšu do výstupního řetězce, pokud není v konfliktu s již navštívenými městy, pokud je v konfliktu, vyberu náhodné město ze zbývajících (bez ohledu na vzdálenost)
  5. potomek 2 vznikne stejně, pouze se dívám na pozice nalevo od aktuálního města

"Hladové křížení"- příklad

Řetězce ke křížení:

(634 1 7285) - rodič 1

(23 1 54678) - rodič 2

náhodně vyberu počáteční město: 1

města napravo jsou 7 a 5, vzdálenosti (1,7) a (1,5) jsou 142 a 107
vybírám proto město 5, výstupní řetězec je tedy zatím **(15)**

opakuji předchozí body - aktuální město je 5

města napravo jsou 6 a 4, vzdálenosti (5,6) a (5,4) jsou 157 a 154
vybírám proto město 4, výstupní řetězec je tedy zatím **(154)**

opakuji předchozí body - aktuální město je 4

města napravo jsou 1 a 6, vzdálenosti (4,1) a (4,6) jsou 318 a 371
vybírám město 1, ale to už je ve výstupním řetězci obsazeno, volím proto náhodně ze zbývajících, např. 3
výstupní řetězec je tedy zatím **(1543)**
opakuji dokud nejsou použita všechna města....
... potomek 2 vznikne z pozic nalevo - tedy od počátečního města leží nalevo města 4 a 3, určím vzdálenosti, atd.

Úkoly

1. naprogramujte funkci pro výpočet kvality populace jedinců
2. realizujte křížení formou záměny části řetězce a mutaci prohozením dvou měst
3. seznamte se s dalšími metodami křížení a metody otestujte
4. realizujte hybridní algoritmus, kdy počáteční populací pro permutační GA bude výstup z metody nejbližšího souseda - matice "reseni" z předchozího cvičení