Specifikace pro zápočtový program
Artem Kopyl
14.05.2023

Téma programu

Hlavním cílem zápočtového projektu je naučit se a implementovat evoluční (genetické) algoritmy pro řešení vyhledávacích problémů. Výběr algoritmu je podpořen skutečností, že algoritmus je nejlépe implementován ve stylu OOP, což mi umožní ukázat znalosti získané během semestru. Tento algoritmus budu demonstrovat na řešení klasického Salesmanova problému (TSP). Je dána množina měst a vzdálenost mezi jednotlivými dvojicemi měst, problém spočívá v nalezení nejkratší možné trasy, která navštíví každé město přesně jednou a vrátí se do výchozího bodu.

Teorie

Definice pojmů:
Generace - množina vygenerovaných řešení
Chromozom - instance možného řešení (generační prvky)
Crossover (operator) - genetický mechanismus používaný ke kombinaci genetické informace dvou rodičů za účelem vytvoření nového potomka.
Mutation (operator) - genetický mechanismus k vytvoření rozmanitosti budoucích generací
Selection - proces výběru rodičů, kteří se páří a rekombinují, aby vytvořili potomky pro další generaci

Základ genetických algoritmů
Genetické algoritmy (GA) jsou adaptivní heuristické vyhledávací algoritmy, které patří do větší části evolučních algoritmů. Genetické algoritmy vycházejí z myšlenek přírodního výběru a genetiky. Jedná se o inteligentní využití náhodného hledání opatřeného historickými daty k nasměrování hledání do oblasti lepšího výkonu v prostoru řešení. Běžně se používají ke generování kvalitních řešení optimalizačních problémů a problémů vyhledávání.

Genetické algoritmy simulují proces přírodního výběru, což znamená, že ty druhy, které se dokáží přizpůsobit změnám v prostředí, jsou schopny přežít a rozmnožit se a přejít do další generace. Zjednodušeně řečeno, simulují "přežití nejsilnějších" mezi jedinci po sobě jdoucích generací pro řešení problému. Každá generace se skládá z populace jedinců a každý jedinec představuje bod v prostoru hledání a možné řešení.

Genetické algoritmy jsou založeny na analogii s genetickou strukturou a chováním chromozomů populace.

Na této analogii je založen základ GA -

- Jedinci v populaci soutěží o zdroje a páření
- Ti jedinci, kteří jsou úspěšní (nejschopnější), se pak páří, aby vytvořili více potomků než ostatní
- Geny (některé vlastnosti řešení, které jednotlivec reprezentuje) od "nejschopnějších" rodičů se šíří v celé generaci, to znamená, že někdy rodiče vytvoří potomstvo, které je lepší než kterýkoli z rodičů.
- Každá následující generace je tak vhodnější pro své prostředí.

Podrobnosti o realizací a rozhraní programu

I/O: Program obdrží sadu měst, která jsou zadána svými názvy a souřadnicemi na mapě. Po zpracování dat a spuštění algoritmu se uživateli zobrazí nejkratší cesta a její délka.

- Input - posloupnost měst a jejich souřadnic
- Output - posloupnost měst v nalezeném nejlepším řešení

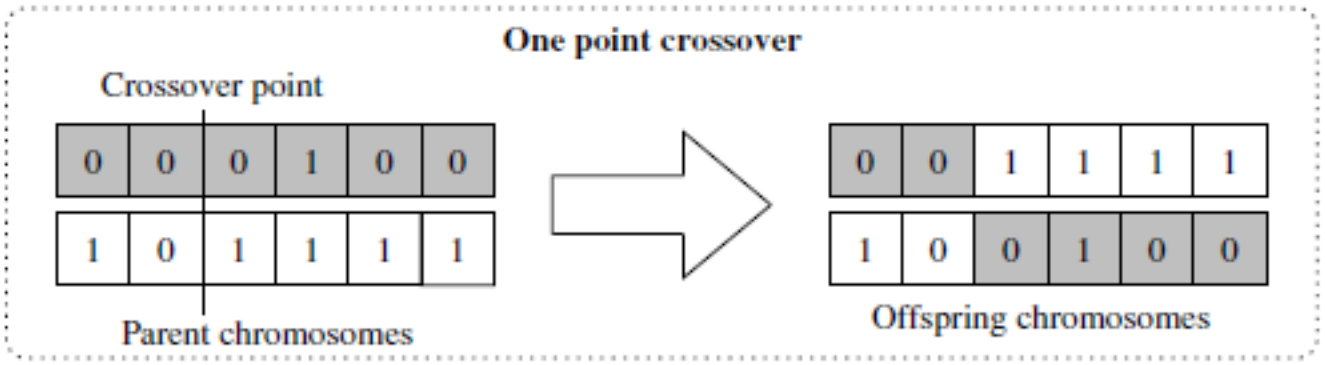
Software také podporuje testovací režim, ve kterém jsou data o městech generována nezávisle (uživatel musí pouze zadat počet měst, která chce otestovat).

K tomu musí být program spuštěn s argumentem v příkazovém řádku (ve složce projektu):

dotnet run (# měst)

Kódování chromozomu: *permutace měst na cestě*

Crossover: upravený *One Point Crossover*
Vyberou se dva chromozomy z aktuální generace. V jejich sadě genů je vybrán bod křížení, podle kterého se části těchto chromozomů přeskupí.
Protože chromozomy v našem případě představují permutace posloupnosti indexů měst, může se stát, že se některý index města bude opakovat (což není povoleno).



Proto algoritmus opravíme:
1) řekněme, že máme dva chromozomy Táta a Máma.
2) Vybereme bod křížení (část chromozomu před tímto bodem je hlava chromozomu a zbytek je ocas).
První syn - hlava otce + geny od matky, které nebyly v hlavě otce (jejich pořadí je zachováno).

PŘÍKLAD

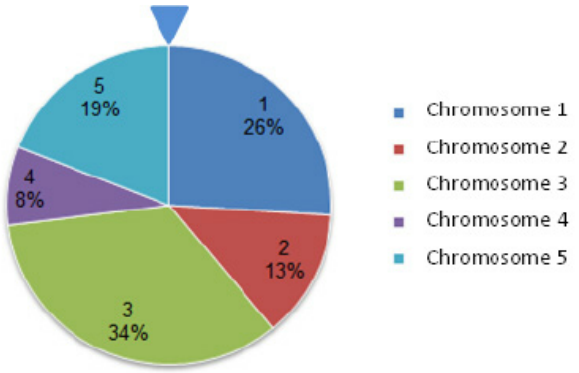
Father = [1, 3, 5, 4, 6, 7, 0, 2, 9, 8]
Mother = [1, 5, 6, 7, 2, 4, 8, 9, 3, 0] (geny v otce, zbytek)

První syn = [1, 3, 5, 4, 6, 7, 2, 8, 9, 0]
Druhý syn - symetricky

Selection: *Roulette Wheel Selection (schematický popis principu)*
Vezměme si kruhové kolo. Kolo je rozděleno na n částí, kde n je počet jedinců v populaci. Každý jedinec dostane část kruhu, která je úměrná jeho hodnotě fitness. Na obvodu kola je zvolen pevný bod, jak je znázorněno na obrázku, a kolo se otáčí. Oblast kola, která se nachází před pevným bodem, je vybrána jako rodičovská. Pro

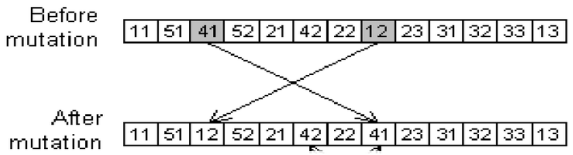
druhého rodiče se opakuje stejný postup. Je zřejmé, že zdatnější jedinec má na kole větší část, a tudíž větší šanci, že se při otáčení kola ocitne před pevným bodem. Pravděpodobnost výběru jedince tedy přímo závisí na jeho zdatnosti.

Při implementaci používáme následující kroky:
- Vypočítejte S - součet fitnessů
- Vypočítejte pravděpodobnost výběru jedince (fitness jedince / S).
- Normalizujte pravděpodobnosti
- Poté můžeme vybírat jedince k chovu podle jejich pravděpodobností

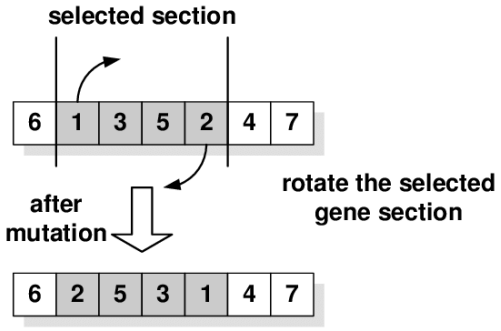


Mutation: *Swap Mutation & Inversion Mutation*
Při psaní programu jsem zjistil, že kombinace několika mutačních operátorů může vyhledávání urychlit, na rozdíl od situace, kdy jsem použil pouze jeden z nich. Jedná se o kombinaci Swap Mutation a Inversion Mutation.

Swap Mutation - Při výměnné mutaci vybereme náhodně dvě pozice na chromozomu a vyměníme jejich hodnoty.



Inversion Mutation - Z celého chromozomu je vybrána podmnožina genů, která je invertována.



(Pravděpodobnost mutace v programu byla 5 %.)

Podmínka ukončení

Program byl doplněn o kontrolu maximálního počtu vypočtených generací a maximálního počtu generací bez změny nejlepšího chromozomu.