Algorytmy ewolucyjne

Piotr Lipiński

Lista zadań nr 2 – AE dla kombinatorycznych problemów optymalizacji

Zadanie 1. *(6 punktów)*
Przypomnij sobie algorytm Simple Genetic Algorithm (SGA) i operator krzyżowania PMX, a następnie zapoznaj się ze skryptem Pythona umieszczonym w materiałach do wykładu. Przedstawia on zastosowanie algorytmu SGA-PMX do rozwiązywania problemu komiwojażera. Następnie:
a) uzupełnij skrypt o usuniętą implementację operatora PMX,
b) uruchom skrypt dla instancji Berlin52 i porównaj wyniki umieszczonych w nim trzech metod rozwiązywania problemu (obliczenia powtórz kilkukrotnie),
c) zrób wykres zmian wartości funkcji celu w kolejnych iteracjach algorytmu ewolucyjnego, podobny do wykresu dla symulowanego wyżarzania,
d) zamiast mutacji Reverse Sequence Mutation spróbuj użyć mutacji polegającej na transpozycji losowo wybranych dwóch elementów permutacji,
d) sprawdź działanie algorytmu na innych instancjach problemu komiwojażera (co najmniej bayg29, bays29, kroA100, kroA150, kroA200).

Zadanie 2. *(4 punkty)*
Rozszerz algorytm ewolucyjny dodając lub zmieniając operatory krzyżowania i mutacji (według własnych pomysłów, można też sprawdzić popularne w literaturze tematu operatory OX, CX, PBX, OBX, PPX, LCSX, LOX). Przeanalizuj działanie nowego algorytmu i otrzymane wyniki.

Zadanie 3. *(4 punkty)*
Spróbuj ulepszyć algorytm ewolucyjny z Zadania 1 wprowadzając dodatkową mutację polegającą na przeszukiwaniu lokalnym (dla danej permutacji początkowej, sprawdzamy wszystkie permutacje różniące się od niej na nie więcej niż K pozycjach i permutację początkową zamieniamy na najlepszą znalezioną) i iterowanym przeszukiwaniu lokalnym (dla danej permutacji początkowej, sprawdzamy wszystkie permutacje różniące się od niej na nie więcej niż K pozycjach, permutację początkową zamieniamy na najlepszą znalezioną i powtarzamy procedurę dopóki możliwa jest poprawa rozwiązania).

Zadanie 4. *(4 punkty)*
Wykorzystaj zaimplementowany algorytm do rozwiązywania problemu QAP. Użyj danych testowych z biblioteki QAPLIB. Przeprowadź obliczenia co najmniej dla następujących zestawów danych: Nug12, Nug14, … Nug30, Tai50a, Tai60a, Tai80a. Przeanalizuj działanie algorytmu i otrzymane wyniki.

Zadanie 5. *(nieobowiązkowe - 4 punkty bonusowe)*
Wykorzystaj zaimplementowane algorytmy do rozwiązywania problemu Q3AP. Użyj (odpowiednio przekształconych – patrz notatka o problemach QAP i Q3AP) danych testowych z biblioteki QAPLIB. Przeprowadź obliczenia co najmniej dla następujących zestawów danych: Nug12, … Nug30. Przeanalizuj działanie algorytmu i otrzymane wyniki.

UWAGA 1: Nie oczekuję, że opracowane algorytmy będą znajdować DOKŁADNE rozwiązania wyżej wymienionych testowych problemów optymalizacji. Dla prostszych problemów oczekuję jednak znalezienia

dość dobrych przybliżeń rozwiązań dokładnych. Dla trudniejszych problemów znalezione rozwiązania mogą być dość odległe od rozwiązań dokładnych, ale muszą być znacznie lepsze niż rozwiązania generowane "losowo" (tzn. z jednostajnym rozkładem prawdopodobieństwa na przestrzeni poszukiwań).

UWAGA 2: Obliczenia mogą być BARDZO czasochłonne (nawet kilkanaście godzin na starszym sprzęcie). Proponuje używać komputerów z pracowni 110, które mają odpowiednio szybkie procesory i odpowiednio dużo pamięci.

UWAGA 3: Najlepsze znane mi rozwiązania problemu Q3AP dla wyżej wymienionych testowych zestawów danych to:

- Nug8: 134 (znajdowane po średnio 1s obliczeń)
- Nug10: 430 (znajdowane po średnio 5s obliczeń)
- Nug12: 580 (znajdowane po średnio 90s obliczeń)
- Nug13: 1912 (znajdowane po średnio 732s obliczeń)
- Nug14: 2320 (znajdowane po średnio 764s obliczeń)
- Nug15: 2230 (znajdowane po średnio 2032s obliczeń)
- Nug20: 7750 (znalezione po ok. 12h obliczeń)
- Nug30: 28706 (znalezione po ok. 12h obliczeń)

Podane czasy obliczeń dotyczą mojego algorytmu ewolucyjnego do rozwiązywania problemu Q3AP uruchamianego na komputerze z procesorem Intel Core 2 Due 3.0 GHz z kartą graficzną nVidia GeForce GTX 280 (240 rdzeni CUDA).