Algorytmy Metaheurystyczne lista nr 2

Sebastian Woźniak 268491

December 2023

Porównanie z poprzednimi wynikami:

Graph	Avg. solution cost	Best solution
xit1083	3844.42	3794
icw1483	4819.41	4788
djc1785	6547.51	6490
dcb2086	7156.85	7128
pds2566	8285.1	8195

Tabela 1: MST + Local search

1 Symulowane wyżarzanie

Parametry:

- T - początkowa temperatura
- δ - stała przez, którą jest mnożona temperatura
- $EPOCH\_IT$ - liczba iteracji w jednej epoce
- MAX_EPOCH - maksymalna liczba epok
- $MAX\_EPOCH\_WITHOUT\_NEW\_BEST$ - maksymalna liczba epok bez nowego najlepszego rozwiązania

1.1 Dobór parametrów

Rozpoczynając z losowej permutacji potrzbujemy wielu kroków aby osiągnąć lokalne ekstremum. Szczególnie gdy przeszukiwane są losowe ruchy, a nie całe sąsiedztwo. Nie chcemy aby na początku często dobierane były nieoptymalne ruchy więc początkowa temperatura T jest ustawiona tak wysoka jak koszt rozwiązania wejściowego. $\delta = 0.95$. Aby zwiększyć prawdopodobieństwo wylosowania optymalnego ruchu $EPOCH\_IT$ zostało ustawione na wartość średnią liczby kroków poprawy z poprzedniej listy która wynosiła mniej więcej $0.3V$. MAX_EPOCH

zostało ustawione na $10V$ po nieudanych próbach powiązania tej wartości z temperaturą i jej tempem spadania. Najlepsze wyniki zostały osiągnięte dla dużej liczby epok. Ograniczenie $MAX\_EPOCH\_WITHOUT\_NEW\_BEST$ zostało ustawione na $\frac{1}{10}MAX\_EPOCH$ po zobserwowaniu dużej ilości nieudanych epok pod koniec wyżarzania, które wydłużały czas działania z bardzo niską szansą na znalezienie nowego najlepszego działania przez zbyt duże prawdopodobieństwo przyjmowania gorszych rozwiązań.

1.2 Wyniki

Graph	Avg. solution cost	Best solution
xit1083	3989.83	3860
icw1483	4873.09	4771
djc1785	6864.59	6721
dcb2086	7419.95	7214
pds2566	8651.12	8485

Tabela 2: Symulowane wyżarzanie

Uzyskaliśmy niedużo gorsze wyniki do tych z algorytmu local search z MST w dużo krótszym czasie. Eksperymentowanie z parametrami mogłoby doprowadzić do uzyskania jeszcze lepszych wyników.

2 Tabu search

Parametry:

- $TABU\_CAPACITY$ - pojemności listy tabu
- $MAX\_IT\_WITHOUT\_NEW\_BEST$ - maksymalna liczba iteracji bez nowego najlepszego rozwiązania
- Ustalono badanie całego sąsiedztwa w każdej iteracji.

2.1 MST jako rozwiązanie wejściowe

Wyciągając wnioski z poprzedniej listy zaczynając od minmalnego drzewa rozpinającego szybko dotrzemy do lokalnego ekstremum, więc wolniej będzie trwało zepełnianie listy tabu. Pojemnosć listy może być mniejsza niż w przypadku przyjmowania jako rozwiązanie początkowe losowej permutacji.

Dla $TABU\_CAPACITY$ większego niż V nie ma różnicy w najlepszym znalezionym rozwiązaniu w 100 wywołaniach algorytmu. Istnieje bardzo nieznaczna róznica w średnim koszcie rozwiązania. Dalsze zwiększanie pojemności listy może wiązać się ze znaczym wydłużeniem czasu działania bez poprawy optymalnego wyniku, więc przyjęte $TABU\_CAPACITY = \frac{1}{5}V$.

$MAX\_IT\_WITHOUT\_NEW\_BEST = \frac{1}{7}TABU\_CAPACITY$. Zwiększanie wartości parametrów nie wpływa na najlepszy uzyskany wynik w 100 powtórzeniach.

2.2 Wyniki

Graph	**Avg. solution cost**	**Best solution**
xit1083	3843.35	3787
icw1483	4819.85	4778
djc1785	6583.23	6499
dcb2086	7158.93	7106
pds2566	8275.21	8189

Tabela 3: MST + Tabu search

Uzyskany średni koszt rozwiązanie jest porównywalny do algorytmu local search jednak poprawił się najlepszy uzyskany wynik.