Maksymalizacja liczby celów podróży

Łukasz Szwed 264147

Wprowadzenie:

Problem, który opisuję to szczególna modyfikacja Problemu Komiwojażera. W zwyczajnym przypadku tego zagadnienia mamy wyznaczyć cykl Hamiltona o możliwie najkrótszej długości. W mojej modyfikacji nie interesuje nas pełen cykl Hamiltona, ale cykl wychodzący z zadanego wierzchołka i zawierający jak największą liczbę pozostałych. Maksymalizujemy więc liczbę odwiedzonych punktów, jednakże musimy wprowadzić dodatkowe ograniczenie – limit trasy, którą możemy pokonać. W mojej interpretacji problemu wierzchołkami grafu są miasta, które chcemy odwiedzić, krawędziami – ilość benzyny, którą spalimy przejeżdżając z wierzchołka początkowego krawędzi do wierzchołka końcowego, a limitem trasy pojemość baku w samochodzie, którym się poruszamy. Sam graf przedstawiłem w postaci macierzy sąsiedztwa.

Opis matematyczny:

Oznaczenia:

n – liczba miast w grafie

c_{ij} – macierz sąsiedztwa między miastami przedstawiona jako liczba litrów benzyny, którą trzeba spalić aby przejechać z miasta i do miasta j

t – pojemność baku w litrach

Zmienne decyzyjne:

$$x_{ij} = \begin{cases} 1, & \text{jeżeli wybrana trasa zawiera krawędź z wierzchołka i do wierzchołka j} \\ 0, & \text{w.p.p.} \end{cases}$$

$u_i \in R+$ - zmienna pomocnicza służąca do wykluczenia rozwiązań zawierających podcykle

Funkcja celu:

$$F(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}$$

Funkcja celu wyraża liczbę wierzchołków (miast), które decydujemy się odwiedzić.

Ograniczenia:

1. Liczba spalonych na trasie litrów nie może być większa od pojemności zbiornika paliwa (zakładamy, że na trasie nie ma stacji benzynowych):

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}c_{ij} \leq t$$

2. Do każdego miasta możemy przyjechać co najwyżej raz:

$$\sum_{i=1}^{n} x_{ij} \leq 1, \quad j = 1,\ldots,n$$

3. Z każdego miasta możemy wyjechać co najwyżej raz:

$$\sum_{j=1}^{n} x_{ij} \leq 1, \quad i = 1,\ldots,n$$

4. Jeśli przyjeżdżamy do jakiegoś miasta, to musimy też z niego wyjechać, jeśli zaś nie decydujemy się go odwiedzić, to nie możemy ani do niego przyjechać ani z niego wyjechać:

$$\sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n} x_{ji}\ , \quad i = 1,\ldots,n$$

5. Ograniczenie eliminujące podcykle poprzez zapisywanie kolejności odwiedzanych wierzchołków:

$$u_i + x_{ij} \leq u_j + (n - 1) * (1 - x_{ij}), \quad i = 1,\ldots,n; \quad j = 2,\ldots,n$$

6. Ograniczenia pomocniczej zmiennej decyzyjnej u:

$u_i \in R+$ – kolejność nie może być ujemna

$u_1 = 1$ – zaczynamy budowę cykli od wierzchołka 1

Szukane:

x* = argmax(F(x))

Podejście z wykorzystaniem metaheurystyki Symulowanego Wyżarzania:

Założenia:

Temperatura początkowa: $T_0 = 10$

Sposób obniżania temperatury: $T_n = \alpha * T_{n-1}$

Współczynnik obniżania temperatury: $\alpha = 0.98$

Liczba prób zmiany x w ramach jednej epoki: 1

Warunek stopu: $T < 1$

Działania podczas każdej iteracji:

1. Wylosuj, czy chcesz dodać czy odjąć losowy wierzchołek (prawdopodobieństwo: dodaj – 75%, odejmij – 25%)
2. Jeżeli dodajesz:
    1) Wylosuj wierzchołek
    2) Wylosuj krawędź spośród aktualnych krawędzi x
    3) Usuń wylosowaną krawędź i wstaw alternatywną ścieżkę prowadzącą przez wylosowany wierzchołek

   Jeżeli odejmujesz:

    1) Wylosuj krawędź spośród aktualnych krawędzi x
    2) Usuń wierzchołek, do którego prowadziła krawędź ze ścieżki (połącz w jedną poprzednią i następną krawędź)
3. Sprawdź, czy ograniczenia są spełnione. Jeżeli tak, idź dalej. Jeżeli nie, wróć do punktu 1.
4. Sprawdź, czy wartość funkcji celu dla nowego x jest większa, niż dla starego x.

   Jeżeli $F(x_{nowy}) > F(x_{stary})$:

      Przyjmij $x = x_{nowy}$

   W. p. p.:

      1) $P(x_{nowy}) = exp\left(\frac{F(x_{nowy}) - F(x_{stary})}{T}\right)$

      2) Wylosuj wartość p z rozkładu jednostajnego na przedziale [0;1]

      3) Jeżeli $p < P(x_{nowy})$:

         Przyjmij $x = x_{nowy}$

       W. p. p.:

         Przyjmij $x = x_{stary}$
5. Obniż T.

Wyniki:

Warunki początkowe:

t = 15

n = 17

c_{ij} = [[9999, 3, 5, 48, 48, 8, 8, 5, 5, 3, 3, 0, 3, 5, 8, 8, 5],
[3, 9999, 3, 48, 48, 8, 8, 5, 5, 0, 0, 3, 0, 3, 8, 8, 5],
[5, 3, 9999, 72, 72, 48, 48, 24, 24, 3, 3, 5, 3, 0, 48, 48, 24],
[48, 48, 74, 9999, 0, 6, 6, 12, 12, 48, 48, 48, 48, 74, 6, 6, 12],
[48, 48, 74, 0, 9999, 6, 6, 12, 12, 48, 48, 48, 48, 74, 6, 6, 12],
[8, 8, 50, 6, 6, 9999, 0, 8, 8, 8, 8, 8, 8, 50, 0, 0, 8],
[8, 8, 50, 6, 6, 0, 9999, 8, 8, 8, 8, 8, 8, 50, 0, 0, 8],
[5, 5, 26, 12, 12, 8, 8, 9999, 0, 5, 5, 5, 5, 26, 8, 8, 0],
[5, 5, 26, 12, 12, 8, 8, 0, 9999, 5, 5, 5, 5, 26, 8, 8, 0],
[3, 0, 3, 48, 48, 8, 8, 5, 5, 9999, 0, 3, 0, 3, 8, 8, 5],
[3, 0, 3, 48, 48, 8, 8, 5, 5, 0, 9999, 3, 0, 3, 8, 8, 5],
[0, 3, 5, 48, 48, 8, 8, 5, 5, 3, 3, 9999, 3, 5, 8, 8, 5],
[3, 0, 3, 48, 48, 8, 8, 5, 5, 0, 0, 3, 9999, 3, 8, 8, 5],
[5, 3, 0, 72, 72, 48, 48, 24, 24, 3, 3, 5, 3, 9999, 48, 48, 24],
[8, 8, 50, 6, 6, 0, 0, 8, 8, 8, 8, 8, 8, 50, 9999, 0, 8],
[8, 8, 50, 6, 6, 0, 0, 8, 8, 8, 8, 8, 8, 50, 0, 9999, 8],
[5, 5, 26, 12, 12, 8, 8, 0, 0, 5, 5, 5, 5, 26, 8, 8, 9999]]

Użycie środowiska IBM CPLEX za każdym razem owocowało uzyskaniem tej samej odpowiedzi:
Wartość funkcji celu 9 (9 odwiedzonych miast) przy zużyciu benzyny 13 l.

Z kolei wyniki otrzymywane przy użyciu implementacji symulowanego wyżarzania w języku Python różniły się od siebie przy każdym uruchomieniu, jednak najlepszy, jaki udało się uzyskać to:

Wartość funkcji celu 7 (7 odwiedzonych miast) przy zużyciu benzyny 12l.

Wnioski:

Środowisko IBM CPLEX zwraca najlepsze możliwe wyniki przy akceptowalnym czasie obliczeń dla niedużych problemów. Z kolei rozwiązania otrzymywane dzięki zastosowaniu metaheurystyki symulowanego wyżarzania są akceptowalne i „mieszczą się” w ograniczeniach, jednak tylko zbliżają się do rozwiązania optymalnego.