Problem grupowania z minimalizacją drzewa rozpinającego

Mikołaj Rozwadowski
127205

18 marca 2019

1 Opis problemu

Rozważany problem polega na podziale danego zbioru X, zawierającego n punktów, na k grup tak, aby suma długości minimalnych drzew rozpinających dla każdej grupy była minimalna. Choć w instancji testowej punkty znajdowały się na płaszczyźnie $\mathbb{R}^2$, a odległości między nimi wyznaczała metryka euklidesowa, to dla zapewnienia ogólności rozwiązania, koszty połączeń przechowywane były w kwadratowej macierzy odległości D.

- **Wejście:** macierz $D = [d_{i,j} \in \mathbb{R}], i, j = 0, 1, \ldots, n-1$.
- **Wyjście:** rodzina $G = \{G_1, G_2, \ldots, G_k\}$, spełniająca warunki podziału zbioru $\{0, 1, \ldots, n-1\}$ i minimalizująca funkcję celu (1), gdzie $MST(G_i)$ to zbiór krawędzi należących do minimalnego drzewa rozpinającego wierzchołków G_i.

$$f(G) = \sum_{G_i \in G} \sum_{\{a,b\} \in MST(G_i)} D_{a,b} \tag{1}$$

2 Pseudokody

```
function GreedyHeuristic(X, D, k)
    G ← InitializeClusters(X)
    for all x ∈ X od k-tego elementu w losowej kolejności do
        InsertToNearest(x)
    return G
function RegretHeuristic(X, D, k)
    G ← InitializeClusters(X)
    for all x ∈ X od k-tego do n-1-go elementu w losowej kolejności do
        if wstawienie punktu x + 1 przed x jest korzystniejsze then
            zamień punkty x i x + 1 miejscami
        InsertToNearest(x)
    InsertToNearest(n)
    return G
function InitializeClusters(X)
    i = 1
    for all x ∈ k pierwszych punktów z X w losowej kolejności do
        G_i ← {x}
        i ← i + 1
    return G
function InsertToNearest(x)
    i* ← arg min_{i=1,2,...,k} min_{j∈G_i} D_{x,j}
    G_{i*} ← G_{i*} ∪ {x}
```

Heurystyka na początku tworzy grupy zawierające po jednym, losowo wybranym punkcie. Następnie kolejne punkty są dodawane do najbliższych im grup. Heurystyka z żalem sprawdza punkty w parach i dołącza je do grup w takiej kolejności, aby było to najmniej kosztowne. Z założenia pozwala to uniknąć sytuacji, w której dołączenie bliskiego punktu (preferowanego przez strategię zachłanną) zwiększa koszt dodania innego punktu i oddala od optimum globalnego.

3 Wyniki eksperymentu

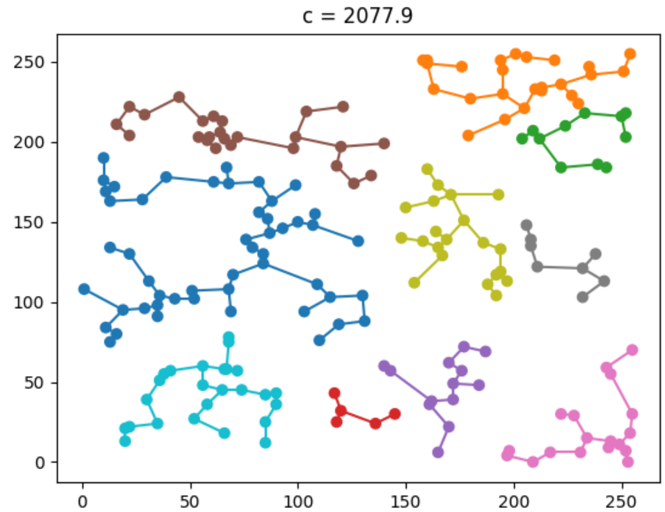
Oba algorytmy uruchomione zostały na instancji testowej 100 razy. Otrzymane wyniki są następujące:

	czas [s]			funkcja celu		
	min	średnio	max	min	średnio	max
greedy heuristic	0,0206	0,0212	0,0275	2077,9	2130,9	2258,2
regret heuristic	0,0977	0,1029	0,1374	2061,2	2130,4	2199,9

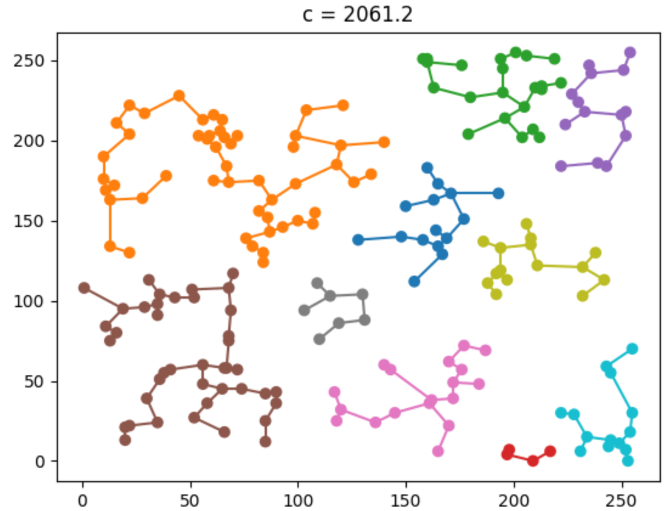
Średnie uzyskane wartości funkcji celu są praktycznie identyczne, choć heurystyka z żalem uzyskała odrobinę lepsze wyniki zarówno w przypadku najlepszym, jak i najgorszym. Heurystyka zachłanna okazała się za to prawie 5 razy szybsza.

4 Wizualizacje rozwiązań

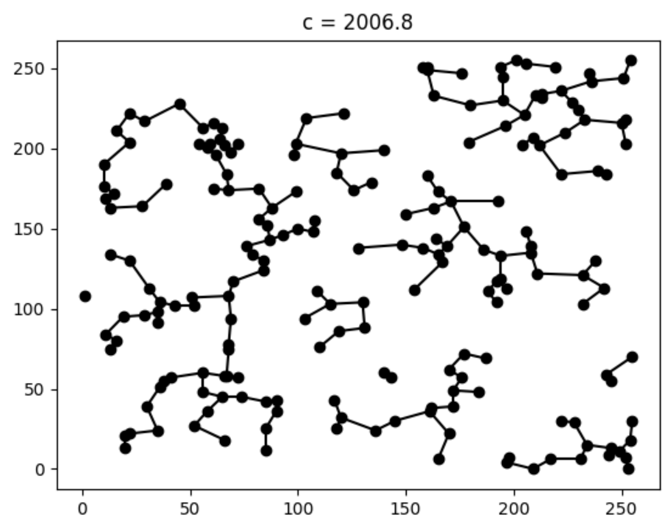
Rysunki 1 i 2 przedstawiają najlepsze rozwiązania znalezione odpowiednio heurystyką zachłanną i z żalem. Ponadto, na rysunku 3 pokazany jest wynik uzyskany poprzez wyznaczenie minimalnego drzewa rozpinającego wszystkich punktów, a następnie usunięcie z niego $k-1$ najdłuższych krawędzi.



Rysunek 1: Najlepsze rozwiązanie uzyskane heurystyką zachłanną.



Rysunek 2: Najlepsze rozwiązanie uzyskane heurystyką z żalem.



Rysunek 3: Najlepsze rozwiązanie uzyskane przez usunięcie krawędzi z MST.