Wydział Matematyki
i Nauk Informacyjnych

POLITECHNIKA WARSZAWSKA

Teoria algorytmów i obliczeń

Projekt zaliczeniowy

Piotr Jacak
Jakub Kindracki
Wiktor Kobielski
Ernest Mołczan

Koordynator: prof. dr hab. inż. Władysław Homenda

Semestr zimowy 2025/2026

Spis treści

1 Wstęp

Niniejsza praca stanowi sprawozdanie z projektu zrealizowanego w ramach przedmiotu **Teoria algorytmów i obliczeń**. Przedmiotem badań są algorytmy operujące na multigrafach, ze szczególnym uwzględnieniem problematyki izomorfizmu podgrafów oraz minimalnych rozszerzeń grafów.

Głównym celem projektu jest opracowanie, analiza teoretyczna oraz implementacja algorytmów rozwiązujących dwa ściśle powiązane problemy. Pierwszym z nich jest weryfikacja, czy dany multigraf H jest izomorficzny z n podgrafami multigrafu G. Drugim, kluczowym zagadnieniem, jest wyznaczenie *minimalnego rozszerzenia* multigrafu G do postaci G', która zawiera co najmniej n podgrafów izomorficznych z H.

Realizacja powyższych celów wymagała formalnego zdefiniowania oraz uzasadnienia kilku fundamentalnych pojęć. W pracy zaproponowano autorskie lub bazujące na literaturze definicje:

- *rozmiaru multigrafu*,
- *metryki* w zbiorze multigrafów,
- *minimalnego rozszerzenia* multigrafu.

Pojęcia te stanowią podstawę do dalszej analizy algorytmicznej oraz oceny kosztu operacji.

W ramach pracy przeprowadzono analizę złożoności obliczeniowej opracowanych algorytmów. Zgodnie z założeniami projektu, w przypadku gdy algorytmy dokładne charakteryzują się złożonością wykładniczą, przedstawiono również propozycje algorytmów aproksymacyjnych o złożoności wielomianowej.

2 Definicje pojęć

Definicja 1 (Graf)**.** Grafem nazywamy parę $G = (V, E)$, gdzie V jest zbiorem wierzchołków, a $E \subseteq V \times V = \{(u, v) : u, v \in V \wedge u \neq v\}$ jest zbiorem krawędzi. Dla każdej pary wierzchołków $u, v \in V$ istnieje co najwyżej jedna krawędź łącząca wierzchołki u i v.

Definicja 2 (Multigraf)**.** Multigrafem nazywamy graf, w którym pomiędzy dowolnymi dwoma różnymi wierzchołkami $u, v \in V$ może istnieć więcej niż jedna krawędź.

Definicja 3 (Graf skierowany)**.** Grafem skierowanym nazywamy parę $G = (V, E)$, gdzie V jest zbiorem wierzchołków, a $E \subseteq V \times V = \{(u, v) : u, v \in V \wedge u \neq v\}$ jest zbiorem krawędzi. Krawędzie w grafie skierowanym mają określony kierunek, co oznacza, że krawędź (u, v) jest różna od krawędzi (v, u). Definicja jest analogiczna dla multigrafów.

Definicja 4 (Izomorfizm grafów)**.** Dwa grafy $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$ są izomorficzne, wtedy i tylko wtedy, gdy istnieje bijekcja $f : V_1 \to V_2$, taka że dla każdej krawędzi $(u, v) \in E_1$ zachodzi $(f(u), f(v)) \in E_2$. Definicja ta jest analogiczna dla multigrafów i grafów skierowanych.

Definicja 5 (Podgraf)**.** Graf $H = (V_H, E_H)$ nazywamy podgrafem grafu $G = (V_G, E_G)$, wtedy i tylko wtedy, gdy $V_H \subseteq V_G$ oraz $E_H \subseteq E_G$. Definicja ta jest analogiczna dla multigrafów i grafów skierowanych.

Definicja 6 (Graf atrybutowy)**.** Graf $G = (V, E, f)$ nazywamy grafem atrybutowym, gdzie V jest zbiorem wierzchołków, a $E \subseteq V \times V = \{(u, v) : u, v \in V \wedge u \neq v\}$ jest zbiorem krawędzi. Dla każdej pary wierzchołków $u, v \in V$ istnieje co najwyżej jedna krawędź łącząca wierzchołki u i v. $f : E \to \Sigma_E$ jest funkcją, przypisującą etykiety wszystkim krawędziom w grafie G.

Definicja 7 (Macierz sąsiedztwa)**.** Macierzą sąsiedztwa multigrafu $G = (V, E)$ nazywamy macierz A, której pole $A_{uv} = k$, wtedy i tylko wtedy, gdy istnieje k krawędzi $(u, v) \in E$. W przypadku gdy nie istnieje żadna krawędź pomiędzy wierzchołkami u i v, to $A_{uv} = 0$.

3 Rozmiar multigrafu

Definicja 8 (Rozmiar multigrafu)**.** Rozmiarem $S(G)$ multigrafu $G = (V, E)$ nazywamy parę liczb naturalnych $(|V|, |E|)$, gdzie $|V|$ oznacza liczbę wierzchołków, a $|E|$ liczbę krawędzi w multigrafie G.

Zakładamy, że liczby wierzchołków i krawędzi są zapisanymi wcześniej stałymi, więc obliczenie rozmiaru multigrafów jest operacją o złożoności czasowej $O(1)$.

Definicja 9 (Porządek w zbiorze wszystkich multigrafów)**.** Niech G_1 i G_2 będą dwoma multigrafami. Mówimy, że G_1 jest mniejszy, lub równy G_2 wtedy i tylko wtedy, gdy:

$$|V_1| < |V_2| \vee (|V_1| = |V_2| \wedge |E_1| \leq |E_2|)$$

Żeby udowodnić poprawność powyższej definicji porządku wykazujemy, że spełnia ona trzy wymagane własności:

- **Zwrotność**:

  $$S(G) \leq S(G)$$

  Dla dowolnego multigrafu $G = (V, E)$, zachodzi $|V| = |V| \wedge |E| = |E|$. Więc w szczególności spełnia on warunek $|V| = |V| \wedge |E| \leq |E|$ z definicji porządku. Stąd $S(G) \leq S(G)$.

- **Przechodniość**:

  $$S(G_1) \leq S(G_2) \wedge S(G_2) \leq S(G_3) \Rightarrow S(G_1) \leq S(G_3)$$

  Weźmy dowolne trzy multigrafy $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ oraz $G_3 = (V_3, E_3)$ takie, że $S(G_1) \leq S(G_2)$ oraz $S(G_2) \leq S(G_3)$.

  Załóżmy, że $S(G_1) \geq S(G_3)$. Z definicji to implikuje, że $|V_1| > |V_3| \vee (|V_1| = |V_3| \wedge |E_1| > |E_3|)$.

  Z założeń wiemy też, że $|V_2| > |V_1|$, lub $|V_2| = |V_1| \wedge |E_2| \geq |E_1|$.

  W pierwszym przypadku z założeń wynika, ze $|V_2| > |V_3|$, co stoi w sprzeczności z $S(G_2) \leq S(G_3)$.

  W drugim przypadku, z założeń wynika, że $|V_2| = |V_3|$ oraz $|E_2| > |E_3|$, co również stoi w sprzeczności z $S(G_2) \leq S(G_3)$.

  W obu przypadkach dochodzimy do sprzeczności, więc nasze początkowe założenie było fałszywe. Stąd $S(G_1) \leq S(G_3)$.

- **Antysymetryczność**:

$$S(G_1) \leq S(G_2) \wedge S(G_2) \leq S(G_1) \Rightarrow S(G_1) = S(G_2)$$

Weźmy dowolne dwa multigrafy $G_1 = (V_1, E_1)$ oraz $G_2 = (V_2, E_2)$ takie, że $S(G_1) \leq S(G_2)$ oraz $S(G_2) \leq S(G_1)$. Z definicji porządku, z pierwszego założenia wynika, że $|V_1| < |V_2| \vee (|V_1| = |V_2| \wedge |E_1| \leq |E_2|)$. Z drugiego założenia wynika, że $|V_2| < |V_1| \vee (|V_2| = |V_1| \wedge |E_2| \leq |E_1|)$.

Jeśli $|V_1| < |V_2|$, to z drugiego założenia wynika, że $|V_2| < |V_1|$, co jest sprzeczne. Analogicznie, jeśli $|V_2| < |V_1|$, to z pierwszego założenia wynika, że $|V_1| < |V_2|$, co również jest sprzeczne. Zatem musi zachodzić $|V_1| = |V_2|$.

Wtedy z pierwszego założenia wynika, że $|E_1| \leq |E_2|$, a z drugiego, że $|E_2| \leq |E_1|$. Stąd $|E_1| = |E_2|$.

W rezultacie mamy $S(G_1) = S(G_2)$.

4 Metryka w zbiorze wszystkich multigrafów

Definicja 10 (Metryka w zbiorze multigrafów)**.** Niech $\mathcal{G}$ będzie zbiorem wszystkich multigrafów. **Metryką** w zbiorze $\mathcal{G}$ nazywamy funkcję:

$$d : \mathcal{G} \times \mathcal{G} \to \mathbb{N}_0$$

Wartość $d(G_1, G_2)$ nazywamy **odległością** między multigrafami G_1 i G_2, a definiujemy ją, jako **minimalną** liczbę operacji dodawania lub usuwania pojedynczej krawędzi lub wierzchołka, za pomocą których można przekształcić graf G_1 w graf izomorficzny z G_2.

Powyższa definicja spełnia następujące własności metryki:

- **Identyczność nierozróżnialnych**:
  Dla dowolnych multigrafów G_1 oraz G_2, $d(G_1, G_2) = 0$ wtedy i tylko wtedy, gdy G_1 jest izomorficzny z G_2. Wynika to bezpośrednio z definicji naszej metryki.

- **Symetria**:
  Dla dowolnych multigrafów G_1 oraz G_2, $d(G_1, G_2) = d(G_2, G_1)$. Dodawanie i usuwanie krawędzi lub wierzchołków jest operacją odwracalną, więc liczba operacji potrzebnych do przekształcenia G_1 w G_2 jest równa liczbie odwrotnych operacji potrzebnych do przekształcenia G_2 w G_1.

- **Nierówność trójkąta**:
  Dla dowolnych multigrafów G_1, G_2 oraz G_3, $d(G_1, G_3) \leq d(G_1, G_2) + d(G_2, G_3)$. Oznacza to, że najkrótsza droga między dwoma multigrafami nie może być dłuższa niż droga przechodząca przez trzeci multigraf. Jest to prawda, ponieważ każda sekwencja operacji przekształcających G_1 w G_2 oraz G_2 w G_3 może być złożona w jedną sekwencję przekształcającą G_1 w G_3.

5 Minimalne rozszerzenie multigrafu

5.1 Algorytm dokładny dla problemu izomorfizmu podgrafu

Mając dane dwa grafy G i H, chcemy znaleźć podgrafy G izomorficzne do H. Do rozwiązania tego problemu posłuży nam algorytm, który wykorzystuje procedurę Backtrackingu do sprawdzania struktury grafów.
Przed przejściem do algorytmu, zdefiniujmy sobie struktury przydatne nam do implementacji. Niech $n_G = |V(G)|$ - ilość wierzchołków w grafie G oraz $n_H = |V(H)|$ - ilość wierzchołków w Grafie H.

Opis algorytmu:

1. Inicjalizacja macierzy sąsiedztwa grafów G i H odpowiednio $S_G \in \mathbb{N}^{n_G \times n_G}$ i $S_H \in \mathbb{N}^{n_H \times n_H}$. Wartość $S[i,j]$, to ilość krawędzi pomiędzy i-tym, a j-tym wierzchołkiem dla danego grafu.

2. Inicjalizacja kandydatów - Zdefiniujmy sobie listę $mozliwe\_dopasowania$, $len(mozliwe\_dopasowania) = n_H$, gdzie pod i-tym indeksem, będziemy mieli listę możliwych dopasowań dla wierzchołka $i \in V(H)$.
   Algorytm Ullmana dla grafów prostych zakłada inicjalizację:
   $u \in V(G), u \in mozliwe\_dopasowania[i] \iff deg_G(u) \geq deg_H(i)$
   Jest ona działającą inicjalizacją dla multigrafów, jednak w celach optymalizacji algorytmu, możemy zmienić tę inicjalizację tak, aby zmniejszyć liczbę potencjalnych dopasowań, a co za tym idzie zmniejszyć liczbę gałęzi, które będzie musiał przejść algorytm. Możemy zauważyć, że w macierzach sąsiedztwa na głównej przekątnej pod indeksami $[i,i]$ znajduje się liczba pętli danego wierzchołka, zatem naszym warunkiem będzie także $S_G[i,i] \geq S_H[i,i]$. Biorąc to wszystko razem, otrzymujemy
   $u \in mozliwe\_dopasowania[i] \iff (deg_G(u) \geq deg_H(i)) \wedge (S_G[i,i] \geq S_H[i,i]))$

3. Dla każdej krawędzi, która istnieje między już dopasowanymi wierzchołkami z H, sprawdź czy istnieje krawędź między ich dopasowaniami z G i czy ilość krawędzi między dopasowaniami jest większa lub równa niż ilość krawędzi między wierzchołkami. Można to osiągnąć przez przejrzenie wszystkich par już dopasowanych wierzchołków i krawędzi między nimi. Jeśli nie, zwróć False

4. Sprawdź, czy wszystkie wierzchołki nie zostały już dopasowane. Jeśli tak, zwróć True.

5. Dla każdego wierzchołka $v \in mozliwe\_dopasowania[i]$, jeśli v *not in* $dopasowania$, przypisz $dopasowania[i] = v$ oraz wywołaj funkcję ponownie dla następnego wierzchołka $\in V(H)$ z przekazaną kopią. W przypadku wyniku True z tej funkcji, zwróć True, w przypadku False, $dopasowania[i] = null$ i przejdź do następnego kroku tej pętli.

6. W przypadku niedopasowania po wszystkich iteracjach pętli, zwróć False.

5.1.1 Dowód poprawności

Najpierw zbadajmy, czy algorytm dobrze inicjalizuje *mozliwe_dopasowania*. W tym celu rozbijmy wszystkie 3 warunki. Pierwszy warunek mówi o tym, że potencjalne dopasowanie v dla wierzchołka u, musi mieć stopień co najmniej równy stopniowi wierzchołka u. Gdyby tak nie było, w grafie G nie istniałaby co najmniej jedna krawędź wychodząca z v, która istniałaby w H i wychodziłaby z u, zatem v nie mogłoby być dopasowaniem dla u. Drugi warunek mówi o tym, że liczba pętli dla v musi być co najmniej równa liczbie pętli dla u. Idea jest taka sama jak warunku pierwszego, gdyby warunek nie był spełniony, nie istniałaby co najmniej jedna pętla da danego wierzchołka, a co za tym idzie, nie mógłby on być dopasowaniem dla u.

Dalej w algorytmie, przechodzimy po kolei po wierzchołkach z H. Najpierw sprawdzamy, czy struktura się zgadza dla tych wierzchołków, do których znaleźliśmy już dopasowania. Jeśli choć 1 krawędź istniejąca w H pomiędzy dwoma wierzchołkami nie będzie istnieć między ich dopasowaniami w G, algorytm wychodzi z tej ścieżki dopasowań i szuka innych, zatem działa poprawnie.

Następnie sprawdzamy wszystkie z możliwych dopasowań dla danego wierzchołka, zatem sprawdzając tak wszystkie wierzchołki, mamy pewność, że przejdziemy po wszystkich możliwych permutacjach.

5.1.2 Złożoność obliczeniowa

Zauważmy, że inicjalizacja *mozliwe_dopasowania* w taki sposób, że dla każdego wierzchołka $u \in V(H)$ możliwym dopasowaniem są wszystkie $v \in V(G)$, to algorytm przejdzie po wszystkich poddrzewach, zatem w przypadku pesymistycznym do 1 wierzchołka wykona n_G potencjalnych dopasowań, do drugiego $n_G - 1$, ..., a do n_H-tego, $(n_G - n_H + 1)$ dopasowań. Zatem mamy

$$\underbrace{(n_G)(n_G-1)\dots(n_G-n_H+1)}_{n_H \text{ razy}} \leq n_G^{n_H}$$

W każdej takiej pętli wykonujemy sprawdzenie, czy struktura grafu się zgadza, (krok 4). Zauważmy, że wykonamy tam i^2 porównań, gdzie i to indeks aktualnie obliczanego wierzchołka. Wiemy że $i < n_H$, zatem możemy ograniczyć tę operację: $i^2 < n_H^2$. W sumie możemy stwierdzić, że złożoność tego algorytmu wyniesie $O(n_G^{n_H} n_H^2)$

5.2 Aproksymacyjne minimalne rozszerzenie multigrafu

Do problemu można zastosować pewną modyfikację algorytmu LeRP (Length-R Paths), opracowanego przez Freda W DePiero oraz Davida Krouta [?]. Algorytm opiera się na założeniu, że o podobieństwie strukturalnym dwóch wierzchołków można wnioskować na podstawie porównania liczby ścieżek (*sygnatur*) o długości r w ich sąsiedztwie.

Modyfikacja algorytmu jako argumenty przyjmuje dwa multigrafy $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$, maksymalną długość ścieżki R oraz liczbę szukanych kopii grafu k. Im większa długość ścieżki R, tym algorytm jest dokładniejszy. Zwraca natomiast przekształcenie $f(g_{1i}) = g_{2k}$, gdzie $g_{1i} \in G_1$ i $g_{2k} \in G_2$, które opisuje najlepszy (w rozumieniu aproksymacji) wspólny podgraf G_1 oraz G_2.
Oznaczmy przez $H = (V_H, E_H)$ najlepszy wspólny podgraf G_1 oraz G_2. Do multigrafu H należy dołożyć zbiór krawędzi X, tak aby grafy G_1 oraz $H' = (V_H, E_H \cup X)$ były izomorficzne. Wówczas multigraf $G_3 = (V_2, E_2 \cup X)$ będzie minimalnym rozszerzeniem G_2, aby ten zawierał G_1 jako podgraf. Przez minimalne rozszerzenie, rozumie się minimalną liczbę dodanych krawędzi do grafu.
Następnie usuwamy z początkowego grafu G_2 wierzchołki podgrafu H i powtarzamy proces dla grafów G_1 i $G_2' = (V_2 - V_H, E_2')$, aby znaleźć k rozdzielnych kopii G_1 w grafie G_2.

5.2.1 Opis algorytmu

Algorytm można podzielić na kilka etapów.

1. W pierwszym kroku, należy wykonać transformację multigrafów wejściowych $G_1 = (V_1, E_1)$ oraz $G_2 = (V_2, E_2)$ na grafy atrybutowe G_1' oraz G_2'. Transformacja przebiega w następujący sposób:

   - Zbiory wierzchołków pozostają bez zmian ($V_1' = V_1, V_2' = V_2$).
   - Dla każdej pary wierzchołków (u, v) w G_1 (i analogicznie w G_2): Jeśli między u a v w G_1 istnieje k równoległych krawędzi, to w G_1' tworzona jest pojedyncza krawędź (u, v) z atrybutem $k \in \mathbb{N}^+$.

2. W etapie drugim, dla obu grafów G_1' i G_2' obliczane są potęgi ich macierzy sąsiedztwa, odpowiednio A^r i B^r aż do maksymalnej długości R. Wartość A^r_{ij} w macierzy A^r reprezentuje liczbę ścieżek o długości dokładnie r z wierzchołka i do wierzchołka j (na ścieżkach mogą się powtarzać wierzchołki i krawędzie). W

kontekście omówionej transformacji, macierz A jest macierzą, gdzie A_{ij} przechowuje atrybut k - liczbę równoległych krawędzi między wierzchołkami i i j w multigrafie G_1.

3. Każdy wierzchołek $g_{1i} \in G'_1$ można porównać z każdym wierzchołkiem $g_{2k} \in G'_2$, stosując przykładowo podobieństwo cosinusowe między histogramem wartości w wierszu macierzy A^r_i a histogramem wartości w wierszu macierzy B^r_k. Następnie tworzymy macierz z wartościami podobieństw między wierzchołkami. Wierzchołki najbardziej podobne zostają ziarnem mapowania.

4. Następnie iteracyjnie (przykładowo DFS), algorytm porównuje sąsiadów wierzchołków zmapowanych. Do następnego mapowania, wybiera tych sąsiadów, do których liczba ścieżek o długości co najwyżej R jest największa i jest równa dla obu wierzchołków z dwóch grafów. Algorytm sprawdza również, czy wybrane wierzchołki nie zostały już zmapowane.

5. Zmapowane wierzchołki w grafie G'_2 są usuwane, tworząc nowy graf G''_2 i proces jest powtarzany dla grafów G'_1 oraz G''_2. W ten sposób znajdowane jest k rozdzielnych kopii G_1 w minimalnym rozszerzeniu G_2.

5.2.2 Złożoność obliczeniowa

Złożoność pesymistyczna omówionej modyfikacji algorytmu LeRP jest wielomianowa i wynosi $O(N^2 \cdot (N + E) \cdot D^2 \cdot R \cdot k)$, gdzie:

- N to liczba wierzchołków w grafach (zakładając, że oba grafy mają rozmiar rzędu N).
- E to liczba krawędzi w grafach (zakładając, że oba grafy mają liczbę krawędzi rzędu N).
- D to średni stopień wierzchołków w grafach.
- R to maksymalna długość ścieżki brana pod uwagę.
- k to liczba szukanych kopii w minimalnym rozszerzeniu

Uzasadnienie złożoności: porównanie par wierzchołków wymaga N^2 porównań. Każdy wierzchołek ma średnio D sąsiadów, więc porównywanie sąsiadów daje D^2 dodatkowych operacji. Analiza różnych długości ścieżek to czynnik R. Podczas budowania dopasowania, algorytm iteracyjny ma złożoność $(N + E)$. Czynnik k odpowiada za znalezienie k kopii mniejszego multigrafu G_1.

W kontekście tego algorytmu aproksymacyjnego nie rozważano formalnego dowodu poprawności. Dostarczono empiryczną gwarancję jakości - algorytm testowano na dużych zbiorach danych, wykazując, że algorytm konsekwentnie zwraca wyniki bliskie optimum w praktyce.

6 Bibliografia