# JIMP2 Projekt 2025

## Dokumentacja funkcjonalna - Java

Michał Ludwiczak
GR3

29 kwietnia 2025

## Spis treści

**1 Cel projektu** **1**

**2 Problem** **2**

**3 Interfejs graficzny użytkownika** **3**

**4 Pliki wejściowe i wyjściowe** **4**
4.1 Plik wejściowy (`.csrrg` / `.bin`) . . . . . 4
4.2 Plik wyjściowy (`.csrrg2` / `.bin`) . . . . . 4

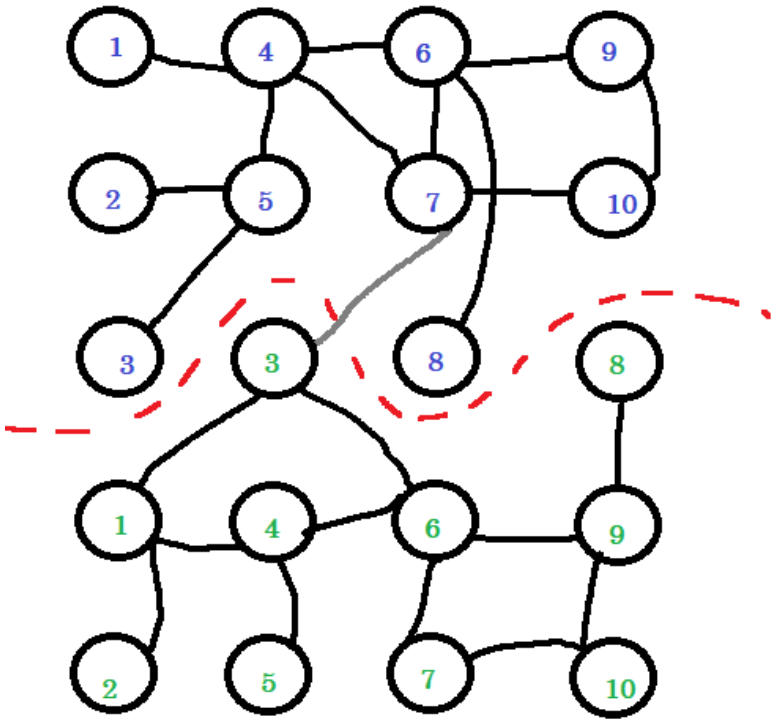
**5 Przykłady użycia** **5**
5.1 Wczytywanie grafu z pliku tekstowego . . . . . 5
5.2 Wczytywanie grafu z pliku binarnego . . . . . 5
5.3 Podział grafu na x części z marginesem y . . . . . 5
5.4 Zapisanie wyniku do pliku wyjściowego . . . . . 5
5.5 Zapisanie wyniku do pliku binarnego . . . . . 5

**6 Wymagania niefunkcjonalne** **6**

# 1 Cel projektu

Celem projektu jest stworzenie aplikacji w języku Java, umożliwiającej podział grafu na określoną przez użytkownika liczbę części, z zachowaniem określonego lub domyślnego 10-procentowego marginesu różnicy liczby wierzchołków pomiędzy częściami. Domyślnie graf dzielony jest na dwie części. Celem podziału jest także minimalizacja liczby przeciętych krawędzi pomiędzy powstałymi częściami grafu. Aplikacja będzie wyposażona w graficzny interfejs użytkownika wykonany w technologii Swing. Użytkownik będzie mógł wczytywać graf z pliku tekstowego lub binarnego, definiować liczbę części oraz margines podziału, a także zapisywać wynikowy graf wraz z danymi wejściowymi do pliku tekstowego lub

binarnego. Każda część grafu będzie prezentowana w interfejsie graficznym w odrębnym kolorze.




Rysunek 1: Przykładowy graf podzielony na 2 równe części

## 2 Problem

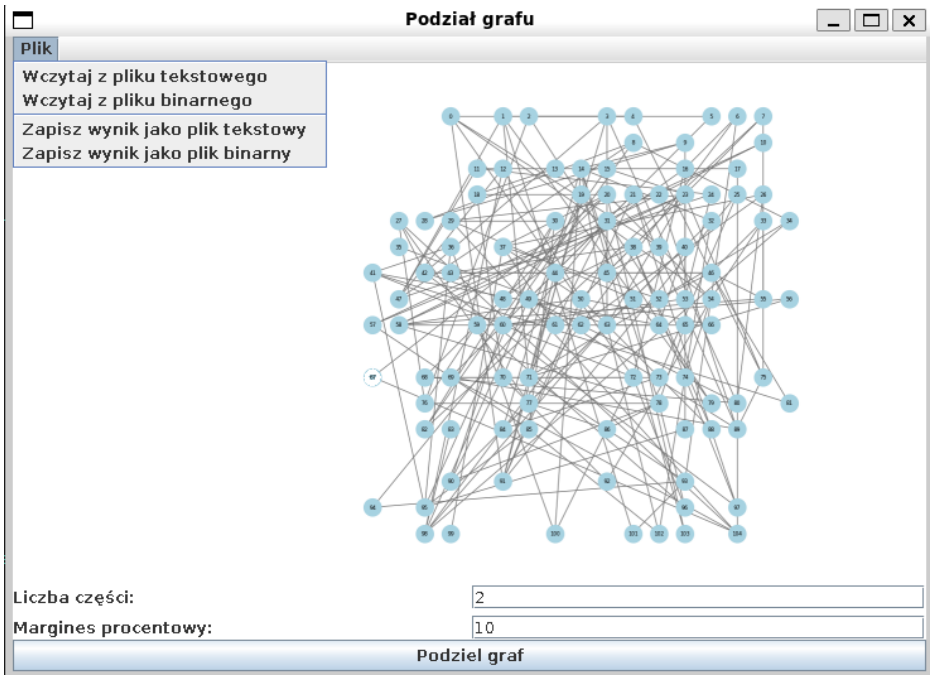
Podział grafu na części w taki sposób, aby liczba przeciętych krawędzi była jak najmniejsza, jest problemem należącym do klasy NP-trudnych. Oznacza to, że znalezienie optymalnego rozwiązania dla dużych grafów jest praktycznie niewykonalne w rozsądnym czasie, ponieważ liczba możliwych podziałów rośnie wykładniczo wraz z liczbą wierzchołków. W przypadku podziału grafu na dwie części istnieje $2^{n-1} - 1$ możliwych sposobów podziału, gdzie $n$ oznacza liczbę wierzchołków. Z tego powodu zamiast sprawdzania wszystkich możliwych podziałów stosuje się algorytmy przybliżone, heurystyki lub algorytmy zachłanne, które pozwalają na szybkie znalezienie dobrego, choć niekoniecznie optymalnego rozwiązania.

# 3 Interfejs graficzny użytkownika

Aplikacja została zaprojektowana z wykorzystaniem biblioteki Swing i oferuje graficzny interfejs użytkownika, który umożliwia interaktywną obsługę procesu podziału grafu. Interfejs składa się z następujących komponentów:

- **Pasek menu** zawierający opcje:
  - `Wczytaj z pliku tekstowego`
  - `Wczytaj z pliku binarnego`
  - `Zapisz wynik jako plik tekstowy`
  - `Zapisz wynik jako plik binarny`
- **Komponent prezentujący wczytany lub podzielony graf**
  Wierzchołki i krawędzie będą rysowane z wykorzystaniem metod graficznych Swing, a poszczególne części grafu po podziale są oznaczane różnymi kolorami.
- **Panel narzędziowy** zawierający:
  - Pole do wprowadzenia liczby części, na które ma zostać podzielony graf (domyślnie 2).
  - Pole do określenia dopuszczalnego marginesu procentowego różnicy w liczbie wierzchołków między częściami (domyślnie 10%).
  - Przycisk `Podziel graf`, który inicjuje proces podziału grafu zgodnie z podanymi parametrami.

Interfejs zostanie zaprojektowany z myślą o intuicyjnej obsłudze, umożliwiając użytkownikowi łatwe wczytywanie grafów, definiowanie parametrów podziału oraz wizualizację wyników.




Rysunek 2: Przykładowy interfejs

# 4 Pliki wejściowe i wyjściowe

## 4.1 Plik wejściowy (`.csrrg` / `.bin`)

Wejście programu może przyjąć dwie formy:

- Plik `.csrrg` z dodatkowymi sekcjami po piątej linii, opisującymi kolejne podgrafy powstałe z podziału grafu. Format pliku składa się z pięciu sekcji zapisanych w kolejnych liniach:
  1. Maksymalna liczba wierzchołków w dowolnym wierszu macierzy sąsiedztwa.
  2. Lista sąsiadów wszystkich wierzchołków zapisana sekwencyjnie.
  3. Wskaźniki (indeksy) na początki list sąsiedztwa dla poszczególnych wierzchołków.
  4. Lista grup wierzchołków połączonych krawędziami (reprezentacja krawędzi).
  5. Wskaźniki na początki grup węzłów z poprzedniej listy.
- Plik binarny `.bin` — binarna forma pliku `.csrrg`, mniej czytelna dla człowieka, ale szybsza w odczycie przez program.

## 4.2 Plik wyjściowy (`.csrrg2` / `.bin`)

Po przetworzeniu danych program generuje plik wyjściowy w jednym z dwóch formatów:

- `.csrrg2` — tekstowa forma pliku wyjściowego, wzorowana na formacie wejściowym .csrrg, ale zawierająca dodatkową linię nagłówkową z wynikiem działania programu.
- `.bin` — binarna wersja pliku tekstowego `.csrrg2`.

W pierwszej linii pliku wyjściowego zapisywany jest rezultat działania programu w formacie:

```
<wynik (S - sukces, F - porażka)> <liczba_części> <liczba_przecięć>
                        <zachowany_margines>
```

Przykład:

```
S 3 2 5
```

# 5 Przykłady użycia

## 5.1 Wczytywanie grafu z pliku tekstowego

Aby wczytać graf z pliku tekstowego `graf.csrrg`, wybierz opcję `Wczytaj z pliku tekstowego` z paska menu. Program odczyta dane grafu, a następnie wyświetli je w głównym obszarze aplikacji.

## 5.2 Wczytywanie grafu z pliku binarnego

Aby wczytać graf z pliku binarnego `graf.bin`, wybierz opcję `Wczytaj z pliku binarnego` z paska menu. Program odczyta dane grafu w formacie binarnym i wyświetli je w głównym obszarze aplikacji.

## 5.3 Podział grafu na x części z marginesem y

Po załadowaniu grafu, w panelu narzędziowym ustaw liczbę części na `x` oraz margines na `y%`. Następnie kliknij przycisk `Podziel graf`, aby rozpocząć proces podziału. Program spróbuje podzielić graf na x części, minimalizując liczbę przeciętych krawędzi, z zachowaniem określonego marginesu różnicy liczby wierzchołków.

## 5.4 Zapisanie wyniku do pliku wyjściowego

Po zakończeniu podziału, aby zapisać wynik do pliku tekstowego, wybierz opcję `Zapisz wynik jako plik tekstowy` z paska menu. Program zapisze dane grafu w formacie `.csrrg2`, zawierającym dodatkową linię nagłówkową z wynikiem działania programu.

## 5.5 Zapisanie wyniku do pliku binarnego

Po zakończeniu podziału, aby zapisać wynik do pliku binarnego, wybierz opcję `Zapisz wynik jako plik binarny` z paska menu. Program zapisze dane grafu w formacie `.bin`, co pozwoli na szybszy odczyt danych w przyszłości.

# 6 Wymagania niefunkcjonalne

Poniżej przedstawiono kluczowe wymagania niefunkcjonalne dla aplikacji:

- **Wydajność:** Czas podziału grafu powinien być jak najkrótszy dla danej wielkości grafu. Dla grafów zawierających do 1000 wierzchołków, czas przetwarzania nie powinien przekraczać minuty na standardowym sprzęcie klasy PC.
- **Użyteczność:** Interfejs użytkownika powinien być intuicyjny i zgodny z ogólnie przyjętymi standardami projektowania GUI.
- **Skalowalność:** Aplikacja powinna umożliwiać obsługę grafów o różnej wielkości, od małych do dużych.
- **Przenośność:** Aplikacja powinna działać poprawnie na różnych systemach operacyjnych wspierających środowisko Java.
- **Utrzymywalność:** Kod źródłowy aplikacji powinien być czytelny i dobrze udokumentowany, aby ułatwić przyszłe modyfikacje i rozwój.

# Literatura

[1] Leonid Zhukov, *Lecture 7. Graph partitioning algorithms.*, YouTube, 24 luty 2021, Dostępny na 28 kwietnia 2025 w: `https://youtu.be/zZae_C2BU_4`