Specyfikacja funkcjonalna aplikacji do podziału grafu

Gniewko Wasilewski
Jan Szulc

11 marca 2025

1 Cel projektu

Celem aplikacji jest podział grafu na zadaną liczbę części w sposób minimalizujący liczbę przeciętych krawędzi. Dodatkowym założeniem jest to, że liczba wierzchołków w każdej części nie może różnić się o więcej niż określony margines procentowy. Program działa w trybie terminalowym i akceptuje pliki wejściowe w formacie `.csrrg`. Aplikacja ma być szybka i efektywna, a jej wyniki powinny być możliwe do ponownego wykorzystania w kolejnych analizach.

2 Środowisko pracy

Do implementacji aplikacji wykorzystujemy **Visual Studio Code** jako główne środowisko programistyczne. Kod jest napisany w języku **C** i kompilowany przy użyciu `GCC` w systemie **Windows**. Projekt jest zarządzany przy pomocy **GitHub**, umożliwiając śledzenie zmian i współpracę zespołową.

3 Funkcjonalność programu

Program umożliwia:

- Wczytywanie grafu z pliku `.csrrg`.
- Podział grafu na zadaną liczbę części z minimalizacją przeciętych krawędzi.
- Kontrolowanie maksymalnej różnicy liczby wierzchołków w podgrupach.
- Zapis wyników w formacie tekstowym lub binarnym.
- Obsługę argumentów wiersza poleceń do konfiguracji działania programu.
- Wyświetlanie komunikatów błędów w przypadku problemów z danymi wejściowymi.
- Możliwość ponownego wykorzystania wyników do kolejnych wywołań programu.

4 Argumenty wiersza poleceń

Program akceptuje następujące argumenty:

```
./graph_partition input.csrrg output.txt -p 3 -m 10 -b
```

Gdzie:

- `input.csrrg` – plik wejściowy zawierający graf w formacie `.csrrg`.
- `output.txt` – plik wyjściowy z wynikami.
- `-p <liczba>` – liczba części, na które zostanie podzielony graf (domyślnie 2).
- `-m <liczba>` – maksymalny procentowy margines różnicy liczby wierzchołków między częściami (domyślnie 10%).
- `-b` – zapis wyników w formacie binarnym (domyślnie zapis tekstowy).

5 Format pliku wejściowego

Plik `.csrrg` opisuje graf w postaci skompresowanej reprezentacji macierzy sąsiedztwa. Składa się z kilku sekcji:

1. Pierwsza linia: maksymalna liczba węzłów w wierszu.
2. Druga linia: lista indeksów węzłów.
3. Trzecia linia: wskaźniki na pierwsze indeksy węzłów w liście wierszy.
4. Czwarta i kolejne linie: listy sąsiedztwa (połączenia między węzłami).

Przykładowy plik wejściowy:

```
18
3;5;6;9;10;13;14;15;...
0;0;8;11;18;27;...
0;72;39;91;4;54;...
```

6 Obsługiwane błędy i komunikaty

Program obsługuje następujące sytuacje wyjątkowe:

Błąd	**Komunikat**	**Kod powrotu**
Brak pliku wejściowego	`Błąd: Nie podano pliku wejściowego`	1
Błędny format pliku	`Błąd: Niepoprawny format pliku .csrrg`	2
Błąd odczytu pliku	`Błąd: Nie można otworzyć pliku wejściowego`	3
Nieprawidłowa liczba części	`Błąd: Liczba części musi być większa od 1`	4

Tabela 1: Obsługiwane błędy i ich kody powrotu

7 Przykłady użycia

7.1 Podział grafu na 3 części, margines 5%, zapis tekstowy

```
./graph_partition graf.csrrg wynik.txt -p 3 -m 5
```

7.2 Podział grafu na 4 części, domyślny margines, zapis binarny

```
./graph_partition graf.csrrg wynik.bin -p 4 -b
```

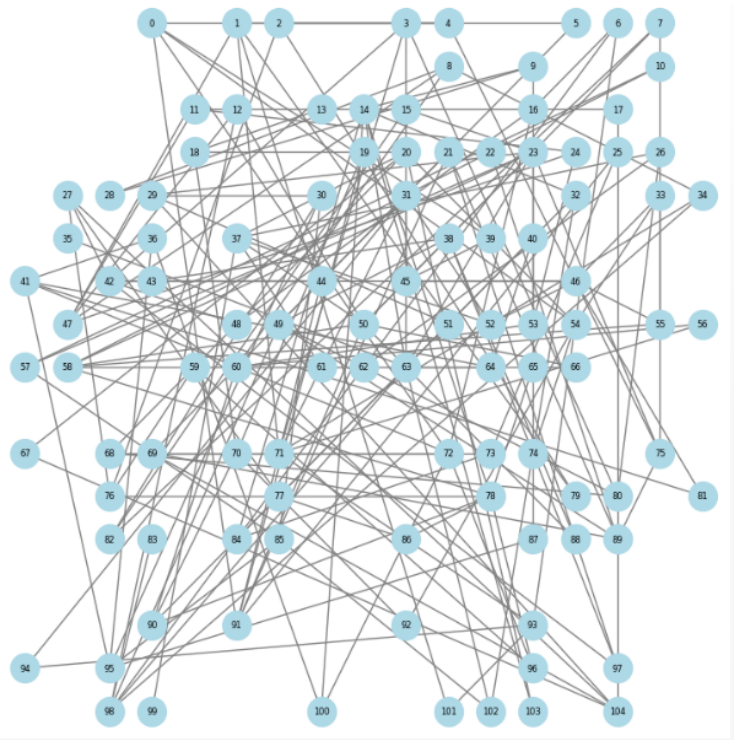
8 Przykładowy graf

Poniżej znajduje się wizualizacja przykładowego grafu używanego w programie:

9 Repozytorium aplikacji

Kod źródłowy projektu znajduje się w repozytorium GitHub:

https://github.com/JanSzulc/jimp2



Rysunek 1: Przykładowy graf podlegający podziałowi

10 Podsumowanie

Aplikacja do podziału grafu została zaprojektowana jako narzędzie do analizy dużych struktur sieciowych. Dzięki elastycznym parametrom podziału użytkownik może dostosować liczbę części oraz poziom równomierności podziału. Obsługa formatu `.csrrg` zapewnia kompatybilność z popularnymi metodami przechowywania danych grafowych. System komunikatów błędów pozwala na szybkie diagnozowanie problemów związanych z wczytywaniem plików i konfiguracją parametrów.

Aplikacja jest gotowa do wykorzystania w analizach grafowych, a jej rozwój może obejmować dalszą optymalizację algorytmów podziału oraz integrację z systemami wizualizacji wyników.