Teoria współbieżności

Współbieżna eliminacja Gaussa

Łukasz Stępień

1. Temat ćwiczenia.

Zaprojektowanie i zaimplementowanie równoległego algorytmu eliminacji Gaussa bazującego na informacji uzyskanej z grafu zależności. Program ma działać dla zadanych rozmiarów macierzy N oraz macierzy na wejściu/wyjściu w formacie zadanym przez prowadzącego.

2. Implementacja.

Implementacja zadania obejmuje stworzenie równoległego algorytmu eliminacji Gaussa, który korzysta z informacji uzyskanych z grafu zależności. Poniżej przedstawiam opis poszczególnych części implementacji:

- **Generowanie losowej macierzy**: Funkcja generate_random_matrix(n) generuje losową macierz o rozmiarze nxn+1, gdzie elementy macierzy są losowe liczby całkowite z zakresu od 1 do 10.

- **Tworzenie grafu zależności**: Funkcja f(k, n) generuje alfabet oraz zbiór zależności D w zależności od rozmiaru macierzy. Otrzymane zbiory sigma i D reprezentują etykiety wierzchołków grafu i zależności między nimi.

- **Zmodyfikowany BFS**: Funkcja bfs(G, s) implementuje zmodyfikowany algorytm BFS (Breadth-First Search), który etykietuje wierzchołki grafu na podstawie zależności.

- **Główna funkcja testująca**: Funkcja test(n) jest główną funkcją, rozwiązującą zadanie dla zadanych rozmiarów macierzy. Generuje graf zależności i przekształca go w graf zależności fnf. Wykorzystuje modyfikowany BFS do etykietowania wierzchołków grafu. Przeprowadza eliminację Gaussa, a wyniki zapisuje do pliku "outputN.txt". Tworzy graf zależności w postaci graficznej i zapisuje go do pliku "graphN.png".

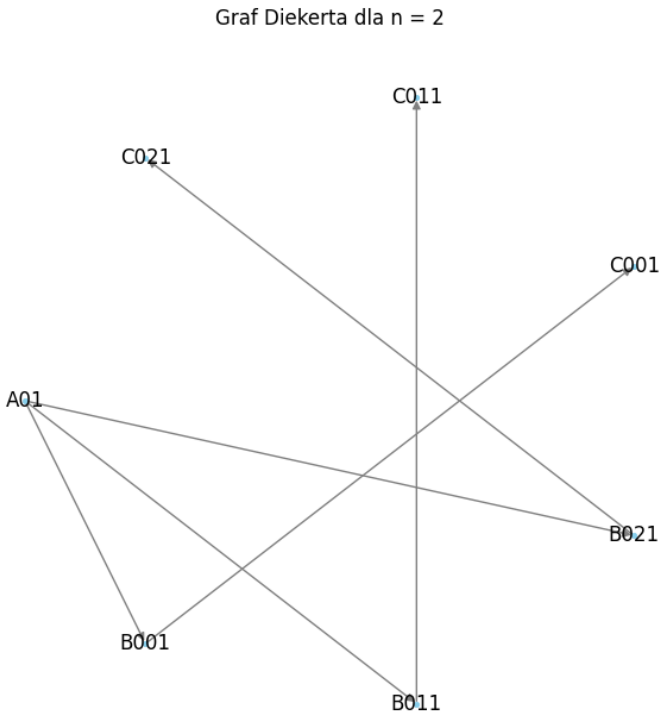
- **Przetwarzanie równoległe klas**: funkcja process_classes(class_set) przetwarza klasy FNF równolegle. Dla każdej grupy klas uruchamiane są osobne wątki, które wykonują funkcję worker dla poszczególnych klas.

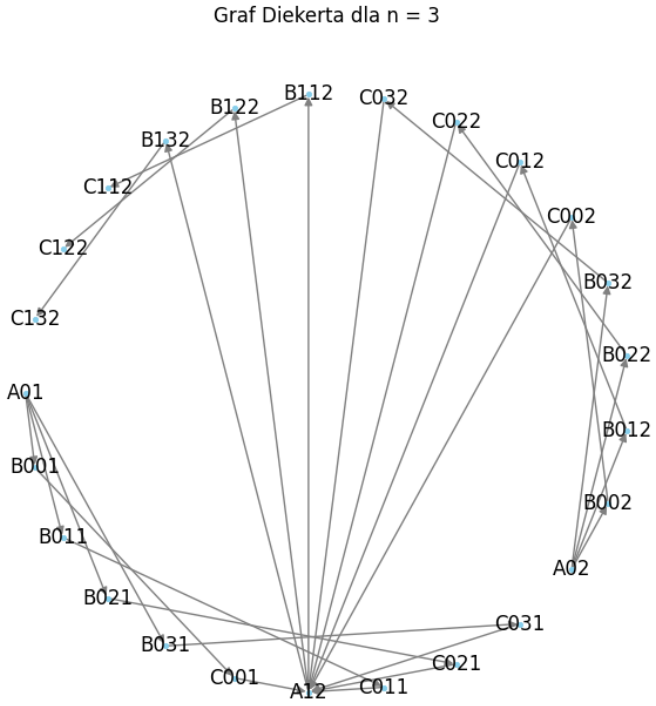
- **Wątki przetwarzające klasy**: funkcja worker(class_name) implementuje operacje dla poszczególnych klas, uwzględniając operacje typu "A", "B", i "C".

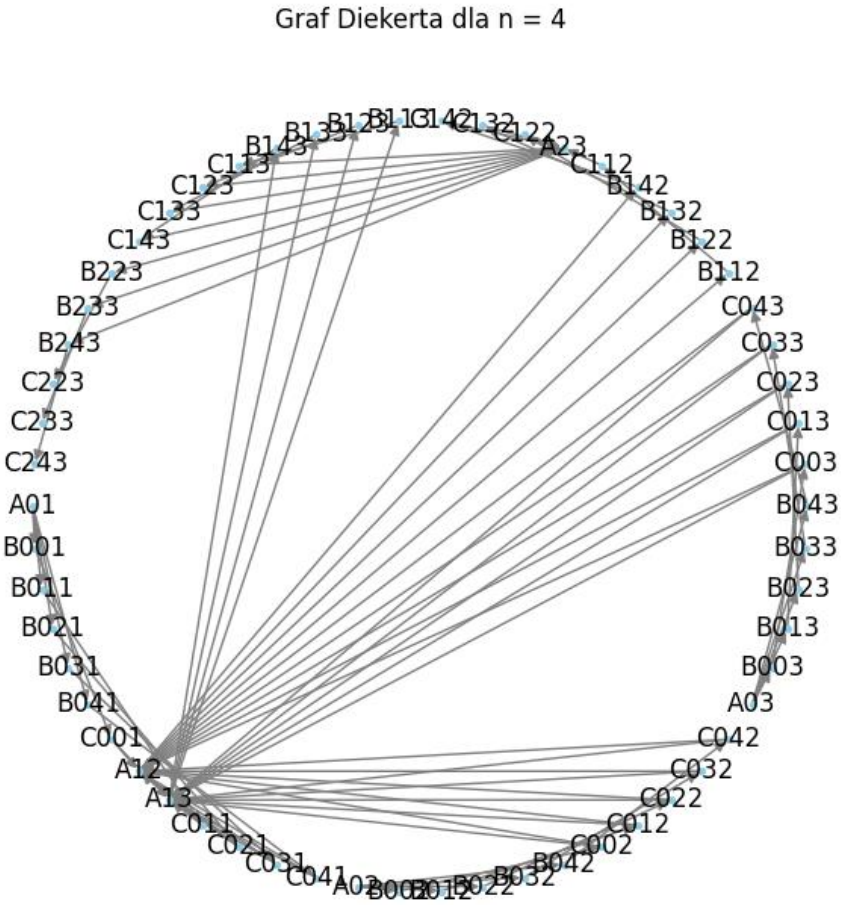
- **Uruchomienie dla różnych rozmiarów maci**erzy: W bloku if __name__ == "__main__": program jest uruchamiany dla macierzy o rozmiarach od 2 do 4. Dla rozmiaru 3 macierzy A jest zdefiniowana ręcznie, dla innych rozmiarów generowane są losowe macierze.

3. Wyniki.

Wykresy przedstawiają poszczególne grafy Diekerta dla różnych rozmiarów macierzy:







W załączonych plikach outputN.txt dla poszczególnych rozmiarów zostały wygenerowane wyniki algorytmu: macierz po współbieżnej eliminacji Gaussa, alfabet w sensie teorii śladów, zaprezentowany w postaci ciągu w postaci algorytmu eliminacji Gaussa, relacja zależności D, postać normalna Foaty oraz wygenerowany graf zależności Diekerta.

4. Podsumowanie

Opracowano program realizujący równoległy algorytm eliminacji Gaussa, oparty na informacjach z grafu zależności. Program generuje losowe macierze, tworzy grafy zależności, a następnie stosuje zmodyfikowany algorytm BFS do etykietowania wierzchołków. Wykorzystując otrzymane informacje, przeprowadza eliminację Gaussa i generuje wyniki w postaci plików tekstowych oraz graficznych, przedstawiających strukturę zależności między elementami macierzy. Zastosowanie wątków przyspiesza przetwarzanie klas równolegle. Implementacja umożliwia testowanie dla różnych rozmiarów macierzy, a wyniki są zapisywane w odpowiednich plikach.