Teoria Współbieżności
CW_6
Mateusz Skowron

1. Część Teoretyczna

1.1. Notacja

Mamy macierz kwadratową o rozmiarze *N x N* oraz wektor wyrazów wolnych. Macierz wraz z wektorem zapisujemy jako jedną macierz o rozmiarze *N x (N+1)* oraz wprowadzamy odpowiednie oznaczenia pól tej macierzy. Element leżący w *i*-tym wierszu oraz *j*-tej kolumnie oznaczamy jako $M_{i,j}$.

Macierz wygląda następująco:

$$\begin{pmatrix} M_{1,1} & M_{1,2} & \cdots & M_{1,N} & M_{1,N+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ M_{N,1} & M_{N,2} & \cdots & M_{N,N} & M_{N,N+1} \end{pmatrix}$$

Wektor wyrazów wolnych: $< M_{1,N+1}, M_{2,N+1}, M_{3,N+1}, \dots, M_{N,N+1} >$

1.2. Niepodzielne zadania obliczeniowe

Wyróżniamy następujące niepodzielne zadania obliczeniowe:

- $A_{k,i}$ - znalezienie mnożnika dla wiersza *i*-tego, do odejmowania go od *k*-tego wiersza.
  $m_{k,i} = \frac{M_{k,i}}{M_{i,i}}$
- $B_{k,i,j}$ - pomnożenie *j*-tego elementu wiersza *i*-tego przez mnożnik - do odejmowania od *k*-tego wiersza.
  $n_{k,i,j} = M_{i,j} * m_{k,i}$
- $C_{k,i,j}$- odjęcie *j*-tego elementu wiersza *i*-tego od wiersza *k*-tego.
  $M_{k,j} = M_{k,j} - n_{k,i,j}$

1.3. Alfabet w sensie teorii śladów

Definiujemy alfabet w sensie teorii śladów:

$$\Sigma = \{A_{k,i} \mid 1 \le i < N,\ i < k \le N\} \cup$$
$$\cup \{B_{k,i,j} \mid 1 \le i < N,\ i < k \le N,\ i \le j \le N+1\} \cup$$
$$\cup \{C_{k,i,j} \mid 1 \le i < N,\ i < k \le N,\ i \le j \le N+1\}$$

1.4. Algorytm sekwencyjny eliminacji Gaussa w postaci ciągu symboli alfabetu

Niech *N* - rozmiar macierzy (liczba wierszy).

Używamy pierwszego wiersza do wyprodukowania zer w pierwszej kolumnie.

$Drugi\ wiersz\ =\ drugi\ wiersz\ -\ mnożnik\ *\ pierwszy\ wiersz$

Wykonujemy kolejne zadania obliczeniowe:

$A_{2,1}, B_{2,1,1}, C_{2,1,1}, B_{2,1,2}, C_{2,1,2}, B_{2,1,3}, C_{2,1,3}, \dots, B_{2,1,N+1}, C_{2,1,N+1}$

$Trzeci\ wiersz\ =\ trzeci\ wiersz\ -\ mnożnik\ *\ pierwszy\ wiersz$

Wykonujemy kolejne zadania obliczeniowe:

$A_{3,1}, B_{3,1,1}, C_{3,1,1}, B_{3,1,2}, C_{3,1,2}, B_{3,1,3}, C_{3,1,3}, \dots, B_{3,1,N+1}, C_{3,1,N+1}$

$N\ wiersz\ =\ N\ wiersz\ -\ mnożnik\ *\ pierwszy\ wiersz$

Wykonujemy kolejne zadania obliczeniowe:

$A_{N,1}, B_{N,1,1}, C_{N,1,1}, B_{N,1,2}, C_{N,1,2}, B_{N,1,3}, C_{N,1,3}, \dots, B_{N,1,N+1}, C_{N,1,N+1}$

Analogicznie postępujemy używając kolejnych wierszy do produkowania zer w następnych kolumnach.

Ogólnie możemy zapisać:

$Sub_{k,i}$ - ciąg operacji, który skutkuje odjęciem *i*-tego wiersza od *k*-tego wiersza.

$Sub_{k,i} = (A_{k,i}, B_{k,i,i}, C_{k,i,i}, B_{k,i,i+1}, C_{k,i,i+1}, \dots, B_{k,i,N+1}, C_{k,i,N+1})$

Algorytm eliminacji Gaussa możemy zapisać (bez podstawiania wstecz):

$Sub_{2,1}\, Sub_{3,1}\, Sub_{4,1} \dots Sub_{N,1}\, Sub_{3,2}\, Sub_{4,2} \dots Sub_{N,2} \dots Sub_{N-1,N-2}\, Sub_{N,N-2}\, Sub_{N,N-1}$

1.5. Algorytm sekwencyjny eliminacji Gaussa w postaci pseudokodu

```
function Gauss(M, N):
    for i from 1 to N - 1:
        for k from i + 1 to N:
            perform A_{k,i} on M
            for j from i to N + 1:
                perform B_{k,i,j}
                perform C_{k,i,j}
```

1.6. Relacja zależności

Wyznaczamy relację zależności *D*.

$D_1 = \{(A_{k,i}, B_{k,i,j}) \mid A_{k,i}, B_{k,i,j} \in \sum\}$

- aby pomnożyć *j*-ty element wiersza *i*-tego przez mnożnik $m_{k,i}$ najpierw należy wyznaczyć ten mnożnik.

Zależne zmienne:

$(A_{k,i})\ m_{k,i} = \frac{M_{k,i}}{M_{i,i}}$

$(B_{k,i,j})\ n_{k,i,j} = M_{i,j} * m_{k,i}$

$D_2 = \{(B_{k,i,j}, C_{k,i,j}) \mid B_{k,i,j}, C_{k,i,j} \in \sum\}$

- aby odjąć *j*-ty element wiersza *i*-tego od wiersza *k*-tego należy najpierw przemnożyć *j*-ty element wiersza *i*-tego przez $m_{k,i}$.

Zależne zmienne:

$(B_{k,i,j})\ n_{k,i,j} = M_{i,j} * m_{k,i}$

$(C_{k,i,j})\ M_{k,j} = M_{k,j} - n_{k,i,j}$

$D_3 = \{(C_{kc,ic,jc}, B_{kb,ib,jb}) \mid C_{kc,ic,j}, B_{kb,ib,j} \in \sum\ jc = jb \wedge kc = ib \wedge ic = ib - 1 \wedge j \neq ib\}$

- operacje związane z odejmowaniem elementów danego wiersza od innego są zależne od operacji wymnażania elementów tego wiersza przez odpowiedni mnożnik

Zależne zmienne:

$(C_{kc,ic,j})\ M_{kc,j} = M_{kc,j} - n_{kc,ic,j}$

$(B_{kb,ib,j})\ n_{kb,ib,j} = M_{ib,j} * m_{kb,ib}$

$D_4 = \{(C_{kc,ic,j}, A_{ka,ia,}) \mid C_{kc,ic,jc}, A_{ka,ia,} \in \sum\ \wedge j = ia \wedge ic = ia - 1 \wedge (kc = ia \vee kc = ka)\}$

- aby wyznaczyć $m_{ka,ia}$ należy najpierw odjąć *j*-ty element w wierszu *ic* od wiersza *ka*, gdzie indeksy operacji *C* i *A* spełniają powyższe warunki (wcześniej muszą zajść wszystkie operacje modyfikujące wiersze *kc* oraz *ka*).

Zależne zmienne:

$(C_{kc,ic,j})\ M_{kc,j} = M_{kc,j} - n_{kc,ic,j}$

$(A_{ka,ia,})\ m_{ka,ia} = \frac{M_{ka,ia}}{M_{ia,ia}}$

$D_5 = \{(C_{k,i1,j}, C_{k,i2,j}) \mid C_{k,i1,j}, C_{k,i2,j} \in \sum\ \wedge i1 = i2 - 1 \wedge i2 \neq j\}$

- odejmowanie wierszy *i1* oraz *i2* od wiersza *k* jest zależne.

Zależne zmienne:

$(C_{k,i1,j})\ M_{k,j} = M_{k,j} - n_{k,i1,j}$

$(C_{k,i1,j})\ M_{k,j} = M_{k,j} - n_{k,i2,j}$

Relacja zależności z uwzględnieniem przechodniości i symetrii może być zapisana jako:

$$D = sym\{\{D_1 \cup D_2 \cup D_3 \cup D_4 \cup D_5\}^+\} \cup I_\Sigma$$

1.7. Relacja niezależności

Wyznaczamy relację niezależności *I*.

$$I = \Sigma^2 - D$$

1.8. Graf Diekerta

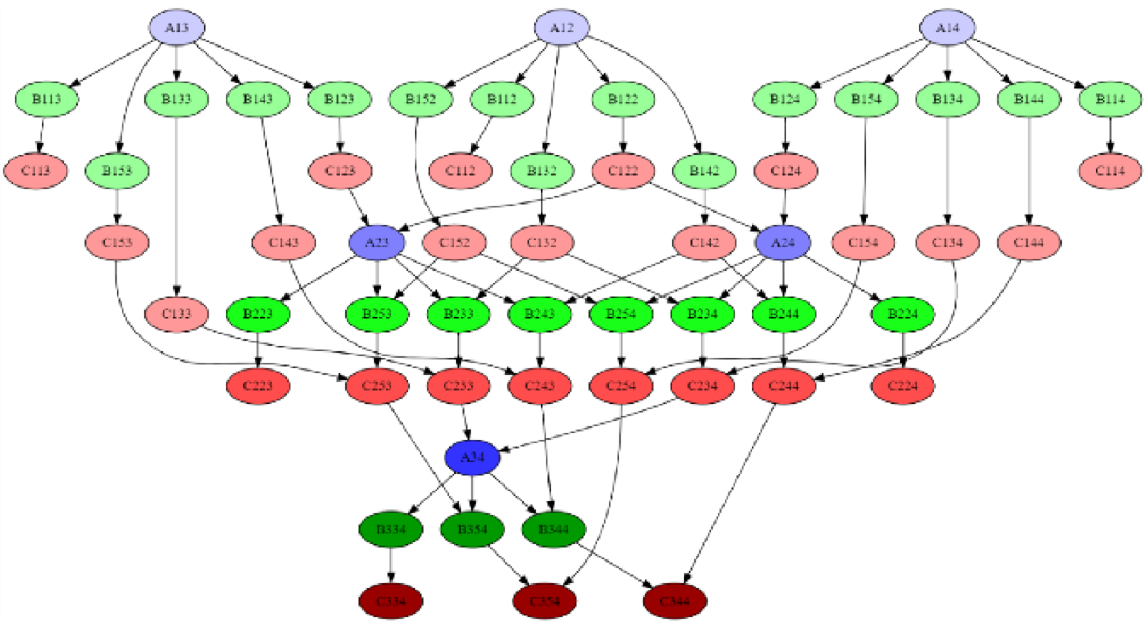
Mając zbiór niepodzielnych zadań obliczeniowych oraz relacje między nimi w prosty sposób otrzymujemy graf Diekerta.
Zbiór wierzchołków *V,* to zbiór niepodzielnych zadań obliczeniowych.

$$V = \Sigma$$

Zbiór krawędzi *E* odpowiada sumie zbiorów D_1, D_2, D_3, D_4, D_5 . Zostały one wcześniej wyznaczone tak, by były ze sobą rozłączne. Zawierają one tylko zależności bezpośrednie (bez przechodniości). Skierowane są zgodnie ze schematem działania algorytmu, a więc wyznaczają one krawędzie grafu w sposób następujący:

$$E = D_1 \cup D_2 \cup D_3 \cup D_4 \cup D_5$$

Przykładowy graf Diekerta dla *N=4*



1.9. Klasy Foaty

Klasy Foaty mają prostą postać. Są one łatwe do zaobserwowania na powyższym grafie. Możemy je podzielić na 3 rodzaje:

- operacje znalezienia wszystkich mnożników, które są potrzebne, aby wyzerować kolumnę o numerze *x* (F_{A_x}).
- operacje przemnażania przez wcześniej znalezione mnożniki kolejnych elementów, aby wyzerować kolumnę o numerze *x* (F_{B_x}).
- operacje odejmowania od kolejnych elementów wyliczonych wcześniej współczynników, aby wyzerować kolumnę o numerze *x* (F_{C_x}).

Definiujemy:

$F_{A_x} = \{A_{k,x} \mid x < k \leq N\}$

$F_{B_x} = \{B_{k,x,j} \mid x < k \leq N \ \land x < j \leq N + 1\}$

$F_{C_x} = \{C_{k,x,j} \mid x < k \leq N \ \land x < j \leq N + 1\}$

dla $x \in \{1, 2, 3, \dots, N - 1\}$

Wtedy klasy Foaty we właściwym porządku:

$FNF = [F_{A_1}][F_{B_1}][F_{C_1}][F_{A_2}][F_{B_2}][F_{C_2}] \dots [F_{A_{N-1}}][F_{B_{N-1}}][F_{C_{N-1}}]$

$[F_{A_x}][F_{B_x}][F_{C_x}]$ możemy interpretować jako zerowanie kolumny o numerze x.

2. Część Implementacyjna

Algorytm został zaimplementowany przy użyciu języka C++ z zachowaniem automatycznej kompilacji przy użyciu Makefile'a. W Makefile'u w regule *run* podajemy jako pierwszy argument ścieżkę do pliku wejściowego, a jako drugi ścieżkę do pliku wyjściowego.

Domyślnie program uruchamiany jest z plikiem wejściowym *input1.txt*, który zawiera przykładowe wejście programu.

Program uruchamiamy wykonując następujące polecenie w katalogu głównym projektu:

make run