Akademia Górniczo-Hutnicza

Wydział Informatyki

Projekt nr 1
Z przedmiotu Rachunek Macierzowy

"Mnożenie macierzy"

Autor: *Jacek Tyszkiewicz i Michał Godek*

Kierunek studiów: *Informatyka*

Kraków, 2025

Spis treści

1. Opracowanie algorytmów mnożenia macierzy

1.1. Cel projektu

Celem projektu było zaimplementowanie trzech algorytmów mnożenia macierzy: tradycyjnego, strassena, bineta. Algorytmy zostały zaimplementowane według poniższej konfiguracji:

1. Dla macierzy o rozmiarze mniejszym lub równym $2^l \times 2^l$ algorytm rekurencyjny Strassena. Dla macierzy o rozmiarze większym od $2^l \times 2^l$ rekurencyjny Binéta.

Następnie algorytmy poddano testom dla k=2,3,..,11 oraz l=3,4,5. Wyrysowano wykresy:

1. Pierwszy wykres: oś pozioma rozmiar macierzy $2^k \times 2^k$ dla $k = 2, 3, \ldots, 11$, oś pionowa czas mnożenia metodą rekurencyjną. Wyrysowano różne wykresy dla wybranych l z przedziału $2 < l < 11$.

2. Drugi wykres: oś pozioma rozmiar macierzy $2^k \times 2^k$ dla $k = 2, 3, \ldots, 8$, oś pionowa liczba operacji zmiennoprzecinkowych metodą rekurencyjną. Wyrysowano różne wykresy dla wybranych l z przedziału $2 < l < k$.

1.2. Algorytm Strassena

Algorytm Strassena to metoda rekursywnego mnożenia macierzy, która redukuje liczbę mnożeń skalarnych z $O(n^3)$ do około $O(n^{2.81})$. Jest to szczególnie przydatne w obliczeniach na dużych macierzach.

Rozważmy dwie macierze A i B, podzielone na **cztery podmacierze**:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Zamiast wykonywać 8 blokowych mnożeń, algorytm Strassena definiuje **7 specjalnych iloczynów pośrednich**.

Siedem iloczynów pośrednich Strassena:

$$P_1 = (A_{11} + A_{22})(B_{11} + B_{22})$$

$$P_2 = (A_{21} + A_{22})B_{11}$$

$$P_3 = A_{11}(B_{12} - B_{22})$$

$$P_4 = A_{22}(B_{21} - B_{11})$$

$$P_5 = (A_{11} + A_{12})B_{22}$$

$$P_6 = (A_{21} - A_{11})(B_{11} + B_{12})$$

$$P_7 = (A_{12} - A_{22})(B_{21} + B_{22})$$

Obliczamy macierz wynikową:

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Gdzie:

$$C_{11} = P_1 + P_4 - P_5 + P_7$$

$$C_{12} = P_3 + P_5$$

$$C_{21} = P_2 + P_4$$

$$C_{22} = P_1 - P_2 + P_3 + P_6$$

1.3. Klasyczny algorytm mnożenia macierzy (Binét)

Klasyczna metoda mnożenia macierzy, znana również jako **algorytm Binéta**, polega na standardowej regule mnożenia blokowego macierzy, gdzie każdy element wyniku jest sumą iloczynów odpowiednich wierszy i kolumn.

Dla dwóch macierzy A i B podzielonych na **podmacierze**:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad \text{oraz} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

macierz wynikowa C obliczana jest jako:

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Gdzie bloki C_{ij} są obliczane według klasycznej reguły mnożenia macierzy:

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

1.4. Porównanie metod mnożenia macierzy

Metoda	**Liczba mnożeń**	**Liczba dodawań**	**Złożoność**
Klasyczny (Binét)	8	4	$O(n^3)$
Strassen	7	18	$O(n^{2.81})$

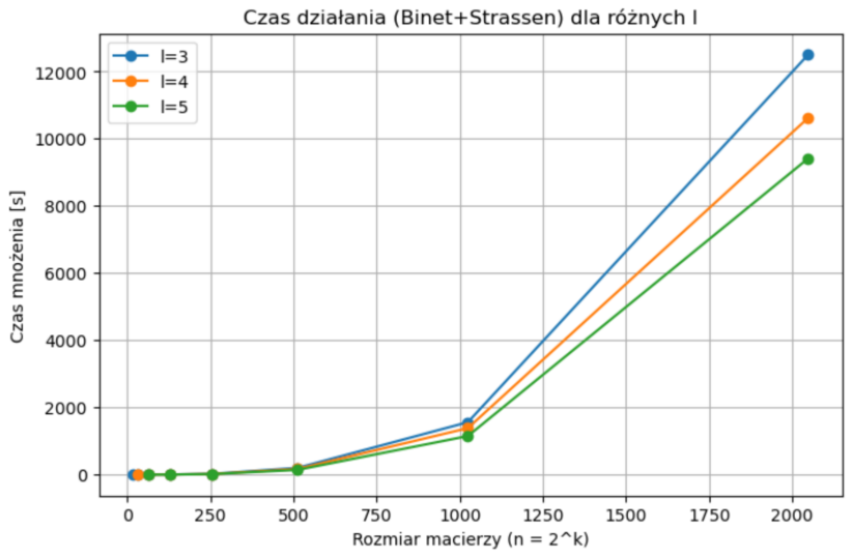
Tabela 1.1. Porównanie klasycznego algorytmu mnożenia macierzy z metodą Strassena

1.5. Algorytmu Strassena dla macierzy o rozmiarze mniejszym l i Bineta w pozostałym przypadku

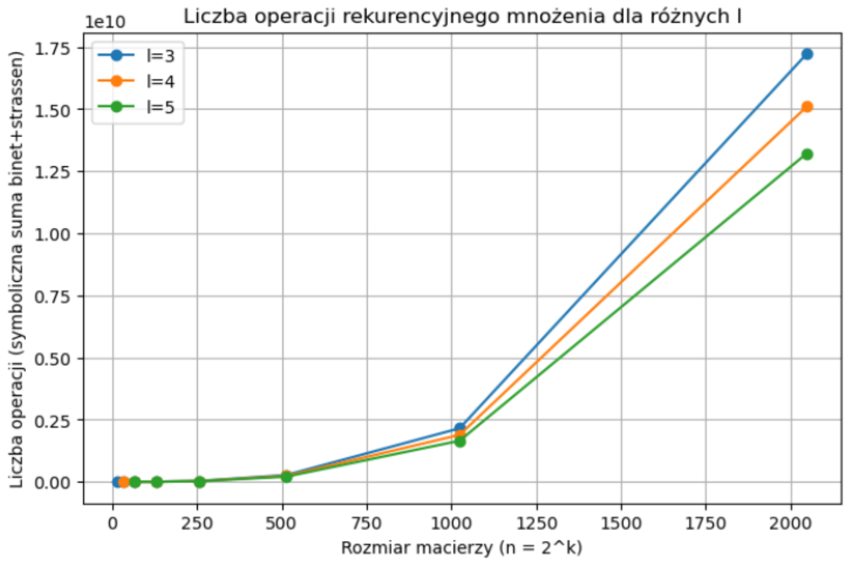
```
Funkcja strassen(A, B):
    n = rozmiar macierzy A

    Jeśli n == 1:
        Zwróć A * B

    half = n / 2

    Podziel A i B na 4 podmacierze:
        A11, A12, A21, A22 = A[:half, :half], A[:half, half:],
                   A[half:, :half], A[half:, half:]
        B11, B12, B21, B22 = B[:half, :half], B[:half, half:],
                   B[half:, :half], B[half:, half:]

    P1 = strassen(A11 + A22, B11 + B22)
    P2 = strassen(A21 + A22, B11)
    P3 = strassen(A11, B12 - B22)
    P4 = strassen(A22, B21 - B11)
    P5 = strassen(A11 + A12, B22)
    P6 = strassen(A21 - A11, B11 + B12)
    P7 = strassen(A12 - A22, B21 + B22)

    C11 = P1 + P4 - P5 + P7
    C12 = P3 + P5
    C21 = P2 + P4
    C22 = P1 - P2 + P3 + P6

    C = Połącz(Połącz(C11, C12, w poziomie), Połącz(C21, C22, w
       poziomie), w pionie)

    d["strassen"] += 18  // Liczba operacji Strassena

    Zwróć C


Funkcja binet(A, B, size, l):
```

```
    Jeśli size <= 2^l:
        Zwróć strassen(A, B)

    half = size / 2

    Podziel A i B na 4 podmacierze:
        A11, A12, A21, A22 = A[:half, :half], A[:half, half:], A[half
            :, :half], A[half:, half:]
        B11, B12, B21, B22 = B[:half, :half], B[:half, half:], B[half
            :, :half], B[half:, half:]

    C1 = Połącz(binet(A11, B11, half, l) + binet(A12, B21, half, l),
        binet(A11, B12, half, l) + binet(A12, B22, half, l), w
        poziomie)
    C2 = Połącz(binet(A21, B11, half, l) + binet(A22, B21, half, l),
        binet(A21, B12, half, l) + binet(A22, B22, half, l), w
        poziomie)

    d["binet"] += 4  // Liczba operacji Bineta

    Zwróć Połącz(C1, C2, w pionie)
```

1.6. Wyniki działania algorytmu



Rys. 1.1. wykres czasu od rozmiaru macierzy



Rys. 1.2. wykres liczby operacji od rozmiaru macierzy

1.7. Wnioski

Z wykresu liczby operacji od rozmiaru macierzy dla różnych wartości parametru l (rys. 1.2) wynika, że algorytm Bineta wykonuje większą liczbę operacji i jest przez to algorytmem wolniejszym od algorytmu Strassena, co znajduje potwierdzenie na wykresie czasu od rozmiaru macierzy (rys. 1.1). Ponadto może to wynikać z tego, że Strassen ma 7 wywołań rekurencyjnych, natomiast Binet 8 wywołań, co predysponuje Strassena dla macierzy o większym rozmiarze. Złożoność algorytmu Binta to n^3 natomiast algorytmu Strassena to $n^{2.81}$ więc wyniki eksperymentu nie są zaskoczneiem.