Pracownia z Analizy Numerycznej

Krzysztof Chrobak, Aleksandra Spyra

10 października 2011

Spis treści

1 Wstęp

Macierze stosuje się do reprezentacji danych i obliczeń w wielu dziedzinach naukowych i technicznych m. in. w matematyce, fizyce czy grafice komputerowej. Często istotne jest również przprowadzanie działań na macierzach, takich jak mnożenie, dodatkowo w jak najkrótszym czasie. Nie dziwi nas więc fakt, że powstało wiele algorytmów obliczjących iloczyn macierzy. W naszej pracy przedstawimy i zbadamy doświadczalnie dwa spośród nich.

Głównym celem naszego zadania jest porównanie pod względem dokładności i szybkości dwuch algorytmów mnożenia macierzy rozmiaru $n \in [4, 500]$: algorytmu naturalnego i algorytmu Strassena. Stawiamy przed sobą dodatkowe zadanie jakim jest rozszerzenie zakresu wielkości danych oraz znalezienie przedziałów wielkości macierzy dla których określony algorytm daje dokładniejsze wyniki czy też jest wydajniejszy czasowo.

W rozdziale §2 znajduje się krótki opis algorytmów, następny rozdział przedstawia realizację zadania. Tam znajduje się raport z przeprowadzonych doświadczeń, jak również przegląd otrzymanych wyników w postaci tabel i wykresów. Rozdział §5 zawiera nie tylko wnioski oraz obserwacje zebrane przez nas podczas rozwiązywania problemu, ale także został wzbogacony o rozważania teoretyczne, które pozwoliły nam uzyskać szersze spojrzenie na obydwa algorytmy.

Do obliczeń wykorzystaliśmy napisany przez nas program w języku C++, implementujący dwie wspomniane powyżej metody liczenia iloczynu macierzy. Pliki z wszstkimi testami oraz wynikami zostały umieszczone w katalogu *testy*.

2 Opis algorytmów

2.1 Agorytm naturalny mnożenia macierzy

Pierwszy badany przez nas algorytm jest bezpośrednią realizacją definicji mnożenia macierzy. Mając dane $A, B \in M_n$, gdzie M_n jest zbiorem macierzy kwadratowych o szerokości i długości n,ich iloczyn określamy jako

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & \dots\dots\dots\dots \\ \dots\dots\dots\dots\dots\dots \\ a_{n1} & \dots\dots\dots & & a_{nn} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & \dots\dots\dots\dots \\ \dots\dots\dots\dots\dots\dots \\ b_{n1} & \dots\dots\dots & & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ a_{21} & \dots\dots\dots\dots \\ \dots\dots\dots\dots\dots\dots \\ c_{n1} & \dots\dots\dots & & c_{nn} \end{bmatrix}$$

, gdzie$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$.

Pozytywną cechą tego algorytmu jest jego prostota, jednakże obliczenie wartości każdego elementu macierzy wynikowej musimy wykonać n mnożeń oraz n dodawań, co przy uwzględnieniu ilości elementów w wyniku daje złożoność $O(n^3)$. Jak się okazuje istnieją algorytmy o lepszej złożoności. Jednym z nich jest algorytm Strassena[1].

2.2 Algorytm Strassena mnożenia macierzy

Rozważamy jedynie macierze kwadratowe, których rozmiar wynosi 2^n, $n \in \mathbf{N}$ (jeżeli wymiar macierzy nie jest tej postaci, uzupełamy badaną macierz zerami tak, aby spełniała to założenie). W pojedynczym wywołaniu algorytmu Strassena, macierze wejściowe X i Y dzielone są na 4 podmacierze tych samych rozmiarów.

$$X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, Y = \begin{bmatrix} E & G \\ F & H \end{bmatrix}, Z = XY = \begin{bmatrix} R & S \\ T & U \end{bmatrix}$$

[1]Tu będzie przypis o hipotezie Strassena, jak dostaniem ksiazke od Michała

3 Przebieg doświadczenia numerycznego

4 Wnioski

5 Bibliografia