Algorytmy numeryczne

Zadanie 3
Tomasz Adamczyk | Aleksander Kosma
243217 | 238193
grupa 1 tester-programista

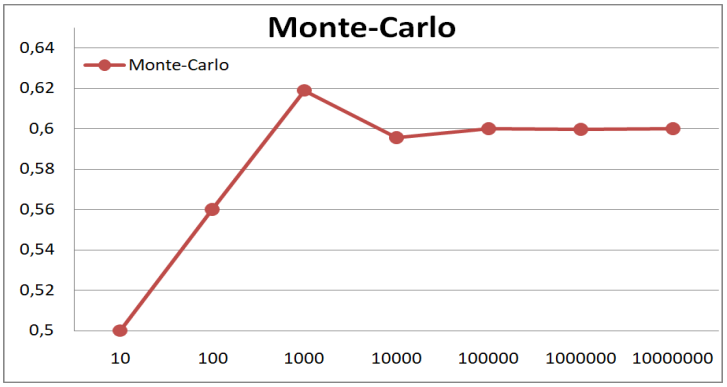
13 Grudzień 2017

Informacje wstępne

Większość testów została przeprowadzona dla danych wejściowych:

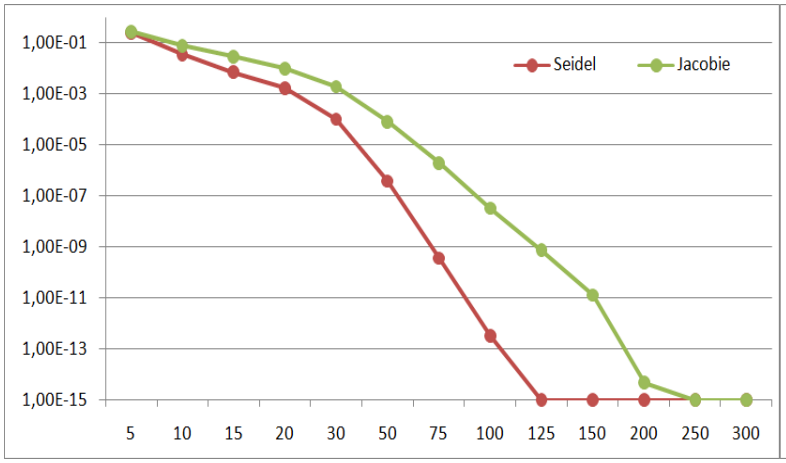
rozmiar planszy	ilość grzybów	wartości kostki
7	4	-4 -3 -2 -1 0 1 2 3 4
pozycje graczy	pozycje grzybów	prawdopodobieństwo kostki
3 4	1 2 5 6	1 1 1 1 1 1 1 1 1

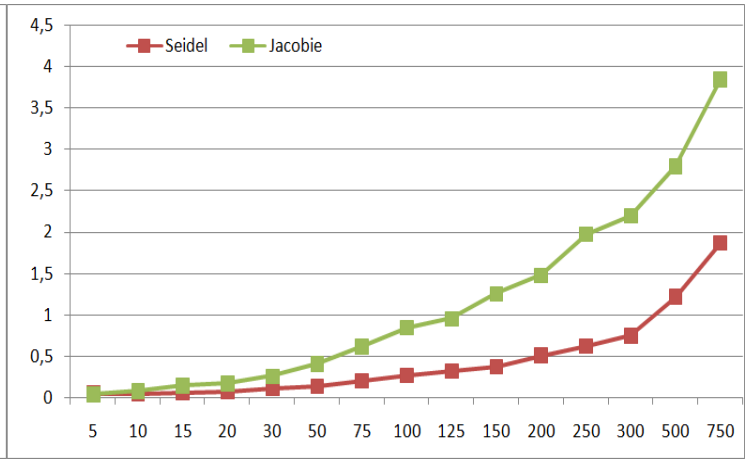
Weryfikacja poprawnego działania metody Monte Carlo. Im więcej iteracji symulacji gry, tym wynik staje się precyzyjniejszy. W tym przypadku, wynik powinien zbiegać do wartości 0,6(wartość obliczona ręcznie).



Implementacja metod iteracyjnych

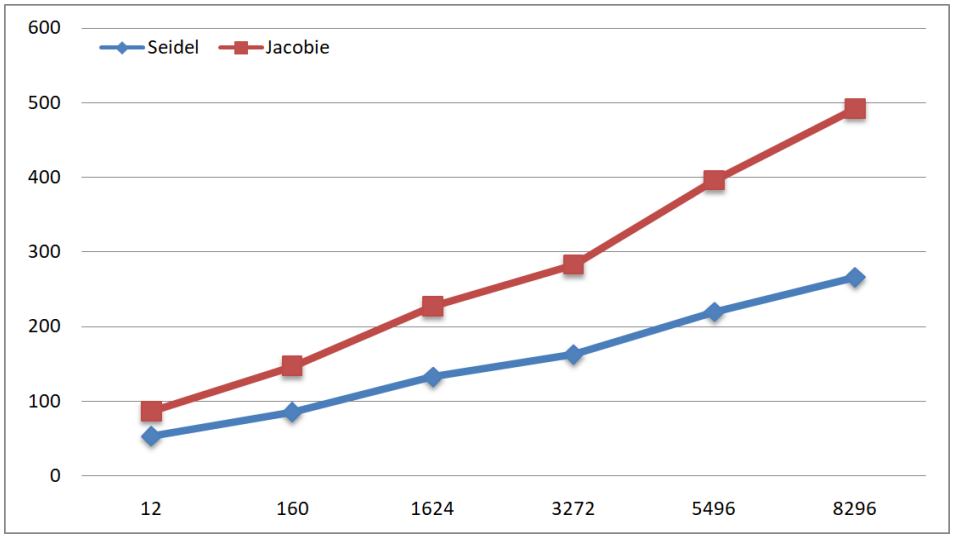
W pierwszej części sprawozdania zbadamy poprawność metod iteracyjnych do rozwiązywania układów liniowych. Wyniki metod porównamy z wynikami z Eigen metodą Gaussa.
Lewy wykres prezentuje różnicę wyniku z metod iteracyjnych i wyniku Eigena. Widać że wyniki te wraz ze wzrostem liczby iteracji zbiegają do zera. Seidel już w około 130 iteracji osiągnął wartość równą Eigenowi. Jacobie potrzebował do tego ponad 250 iteracji. Te wyniki dowodzą, ze metody iteracyjne liczą równie precyzyjnie co metoda Gaussa.
Prawy wykres prezentuje czas potrzebny do obliczenia wyniku dla podanych na osi poziomej iteracji.





Optymalna ilość iteracji w przypadku metod iteracyjnych

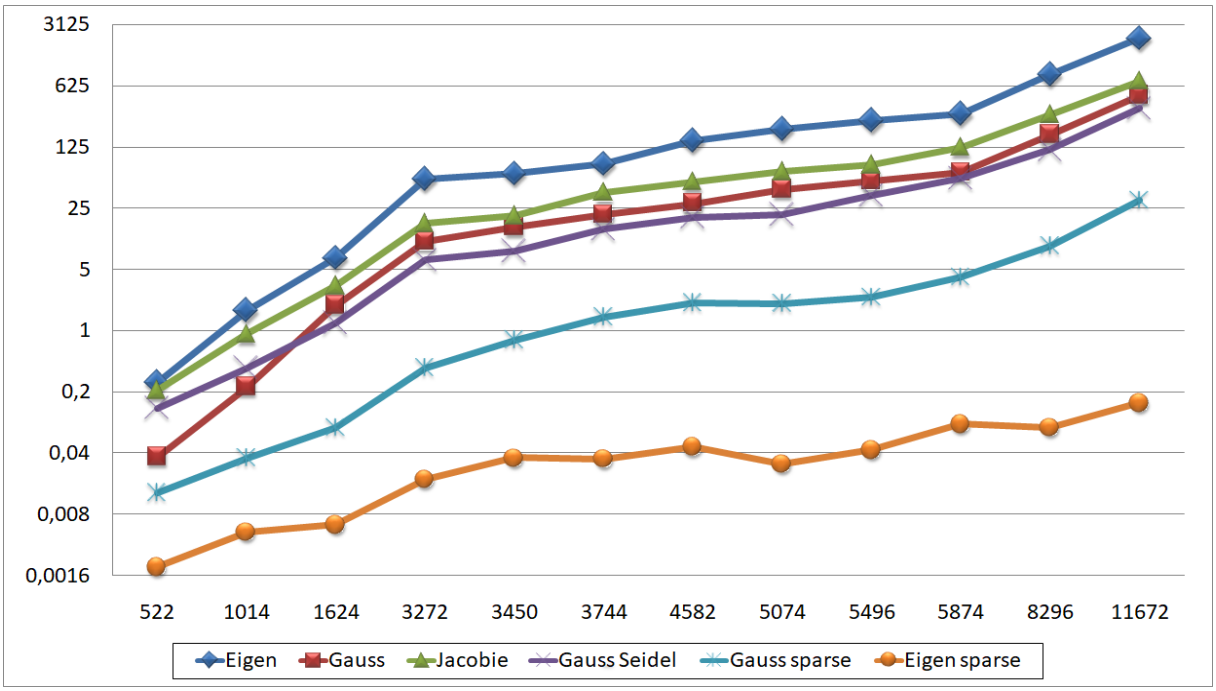
Aby zoptymalizować czas potrzebny od wyliczenia wyniku przez metody iteracyjne, warto wiedzieć ile danych iteracji potrzebujemy. Za duża ilość iteracji może przeważyć na końcowych rezultatach i wnioskach.
Zastosowany został przez nas algorytm, wyliczający normę wektora poprzedniego i aktualnego. Jeśli ułamek tych norm jest mniejszy niż podany epsilon, kończymy kolejne iteracje.
Wykres pokazuje mniejszą potrzebę iteracji dla Gaussa-Seidela. Gdzie oś pionowa to ilość iteracji, a oś pozioma to rozmiar macierzy. Jacobie potrzebuje więcej iteracji, jak i szybciej przyspiesza z kolejnymi wymaganymi iteracjami. Więc wraz ze wzrostem macierzy potrzeba w obu przypadkach więcej iteracji.



Optymalny dobór metody w grze Grzybobranie

Druga część opisuje wnioski, oparte o porównania wyników i czasu działania podanych metod:
-metoda Gaussa z częściowym wyborem elementu podstawowego
-metoda Gaussa z częściowym wyborem elementu podstawowego z optymalizacją dla macierzy rzadkich
-metoda iteracyjna Gaussa-Seidela
-metoda iteracyjna Jacobiego
-metoda z biblioteki Eigen partialPivLu, z częściowym wyborem elementu podstawowego
-metoda z biblioteki Eigen SparseLU, z częściowym wyborem przy użyciu macierzy rzadkich

Poniższy wykres prezentuje czas potrzebny do rozwiązania równania, gdzie oś pionowa to czas liczony w sekundach, oś pozioma rozmiar macierzy. Eigen sprase ma miażdżącą przewagę nad wymienioną resztą. Z pośród podanych najlepiej korzystać z tej metody. Jedynym jej mankamentem jest potrzeba dostarczenia wektora z informacją o ilości wartości miejsc niezerowych w kolumnach. Trzeba jednak mieć na uwadze metody Seidela i Jacobiego w wersjach sparse, które mogą okazać się lepsze od wspominanego Eigena sparse. Dlatego nie można stwierdzić czy Eigen sparse jest najlepszą metodą z nam znanych.



Precyzja wyników obliczona przez każdą z powyższych metod była praktycznie ta sama. W lwiej części sytuacji wyniki pokrywały się idealnie. Ma to związek z bardzo małą ilością wartości niezerowych wiec i operacji matematycznych. W konsekwencji strata precyzji nie jest zauważalna. Prawdopodobnie wraz ze wzrostem macierzy, mogłyby się pojawić pewne różnice w wynikach.

Treść zadania zakładała również kostkę, która ma dowolny rozkład wartości i ich prawdopodobieństw. Dane te mają jedynie wpływ na końcowy rezultat. Przez nieparzystą ilość pól na mapie, i rozmiar kostki, każde pole jest osiągalne dla obu graczy. Koniec końców możliwości rozegrania partii jest taka sama. Dlatego ciężko odnotować tu coś ciekawego i zostało to pominięte w sprawozdaniu.

Obliczenia wraz ze wzrostem rozmiaru macierzy były liczone od 10 do 2 próbek. Zazwyczaj tyle wystarczało by zauważyć tendencje. Wynik metody Monte Carlo do weryfikacji wyników, był obliczany każdorazowo dla miliona próbek.

Podział obowiązków	
Aleksander Kosma	**Tomasz Adamczyk**
-Implementacja Monte Carlo	-Implementacja metod iteracyjnych
-Iteracyjne generowanie układu równań	-Ustalenie i implementacja warunków wygranej
-Szukanie gry bliskiej 50%	-Wprowadzanie układu równań do macierzy
-Optymalizacja metod iteracyjnych i Gaussa	-Generowanie wyniku poprzez bibliotekę Eigen
-Testy i generowanie wyników	-Napisanie skryptów do generowania wyników
-Obróbka wyników, wykresy i opracowanie	-

Dodatkowo

W momencie kiedy chcemy osiągnąć wynik jak najbardziej zbliżony do 50% szans na wygraną, naszą intencją jest wydłużanie i odwlekanie wygranej przez któregoś z graczy. W tym celu należy:

- pozbyć się grzybów. Grzyby są ścieżką na skróty do wygrania. Określona ilość grzybów w rozgrywce, zdobyta wcześniej, od razu premiuje nas wygraną.
- zmniejszyć kostkę do najmniejszych przeskoków. Najbardziej optymalna wersja to kostka ze ściankami o wartościach {-1, 0 ,1}. Zero na kostce jest kolejną szansą dla przeciwnika by odwrócić losy rozgrywki.
- prawdopodobieństwo kostki powinno być jak największe dla zera. W momencie kiedy zero wypada najczęściej, dynamika rozgrywki spada. Dzięki temu, któremuś z graczy trudniej stworzyć sobie szybką przewagę.
- Ostatnią zmienną na którą mamy wpływ to pozycje startowe graczy. Znów chcemy by opóźnić wygraną, więc stawiamy obu graczy możliwie daleko od mety.

Najlepszy uzyskany wynik: **0.5000000034688313**

dane wejściowe:

rozmiar planszy	pozycje startowe	wartości kostki	prawdopodobieństwo
101	50, 51	-1, 0, 1	1, 100000, 1