Maciej Oziębły 184147
Tomasz Wojciechowski 184132

Sterowanie procesami dyskretnymi
Sprawozdanie 2

Zakres sprawozdania:

- **zastosowanie algorytmu NEH w celu optymalizacji kolejności zadań dyskretnych**
- **akceleracja algorytmu NEH**

Środowisko:

System operacyjny: Windows 7
Platforma programistyczna: .NET 4.5
Język programowania: C#
IDE: Visual Studio 2012 Ultimate (wersja z MSDNAA)
Komputer wypozażony w 2 rdzeniowy procesor korzystający z technologii Hyper-threading (więc mogący wykonywać równolegle 4 operacje)

Wyniki:

Zarówno wyjście programu z użyciem algorytmu NEH bez akceleracji jak i z akceleracją dało identyczne wyniki, co wzorcowy plik z rozwiązaniami.

Wydajność implementacji algorytmu bez akceleracji:

Wzorcowy program wykonał się na moim domowym komputerze w 76.2481s. Nasza implementacja algorytmu wykonała to samo zadanie w 94.4294s (więc o 23.8% dłużej). Trzeba jednak pamiętać, że nasza implementacja powstała w oparciu o platformę .NET, więc wykonanie naszego programu odbywało się za pośrednictwem maszyny wirtualnej. Program został poddany bardzo mocnym optymalizacjom (z performance testów wynika, że 97,1% czasu działania programu to budowanie tablicy Cmax), do tego obliczenia zostały zrównoleglone (na 4 wątki), a i tak program jest nieco wolniejszy. Przed wszystkimi operacjami przyspieszającymi działanie algorytmu program wykonywał się ponad 7 minut (a więc ponad 5 razy dłużej).

Statystyki algorytmu bez akceleracji:

liczba zadań	20	20	20	50	50	50
liczba etapów	5	10	20	5	10	20
średni czas wykoania [ms]	3,4	3,2	3,1	4,8	10,1	14,3

liczba zadań	100	100	100	200	200	500
liczba etapów	5	10	20	10	20	20
średni czas wykoania [ms]	23,1	46,8	91,5	338,3	641	8945,5

Dla małej liczby zadań bardzo ciężko wyciągnąć z zaprezentowanych wyników jakiekolwiek wnioski. Czasy wykonania były tak małe, że ciężko było je zmierzyć.

Przy statystkykach dla liczby zadań 100 i więcej wyraźnie widać, że średni czas wykonania jest w przybliżeniu proporcjonalny do liczby etapów. Można też pokazać sześcienną złożoność algorytmu w zależności od liczby zadań:

liczba zadań	100	200	500
liczba etapów	20	20	20
średni czas wykoania [ms] / liczba zadań^3	0,0000915	0,0000422875	0,000071564

Średnie czasy wykonania podzielone przez liczbę zadań podniesioną do potęgi trzeciej są do siebie podobne, co pokazuje sześcienną złożoność algorytmu w zależności od liczby zadań.

Wydajność implementacji algorytmu z akceleracją:

Czas wykonania programu dla algorytmu z akceleracją wynosi 4.5912s, więc jest ponad 20-krotnie krótszy od czasu wykonania implementacji bez akceleracji. Jest to spowodowane zmniejszeniem złożoności algorytmu do kwadratowej względem liczby zadań, co przy liczbie zadań wynoszącej 500 ma ogromny wpływ na czas wykonania zadania.