POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY

Programowanie równoległe i rozproszone
Projekt część II – użycie OpenMP
Zadanie BK27 - ,,Słownik równoległy”

Prowadzący: dr inż. Bartłomiej Kubica
Wykonał: Szymon Nogieć
Nr indeksu: 234446

1. Implementacja

Dla potrzeb zadania zmieniono implementację części sekwencyjnej, dostosowując ją do używania nowoczesnego C++. Dodatkowo usunięto zbędny kod oraz zwiększono rozmiary generowanych słowników. Dla celów zadania dopisano w klasie dwie metody – find_linear i find_mp. Obie te funkcje przyjmują wektor stringów do znalezienia w słowniku.

Funkcja find_linear przeszukuje strukturę danych w sposób liniowy.

Funkcja find_mp przeszukuje strukturę danych w sposób równoległy, używając do tego ustawionej wcześniej liczby wątków. Równoległość uzyskano poprzez zastosowanie makra #pragma omp for. Niemniej jednak problematycznym jest używanie struktur z STL razem z open_mp.

W związku z faktem, iż dostęp do std::vector metodami .at, dodawanie elementów poprzez push_back() etc. nie jest bezpieczne w przypadku obliczeń równoległych, należałoby te miejsca w kodzie otoczyć dyrektywą #pragma critical, co bardzo mocno spowolniłoby obliczenia (w szczególności biorąc pod uwagę, iż dostęp do tablicy jest bardzo szybki). W związku z tym zastosowano deklarację rozmiaru wektora ze wskaźnikami elementów, podając jako parametr konstruktura std::vector rozmiar poszukiwanej tablicy stringów. Dzięki temu w pętli for można bezpiecznie użyć dostępu do elementów szablonu poprzez odwoływanie się poprzez konkretny indeks ([i] w pętli for). [2]

Przed wywołaniem pętli for, należy zadeklarować makro

```
#pragma omp for
```

2. Wyniki obliczeń

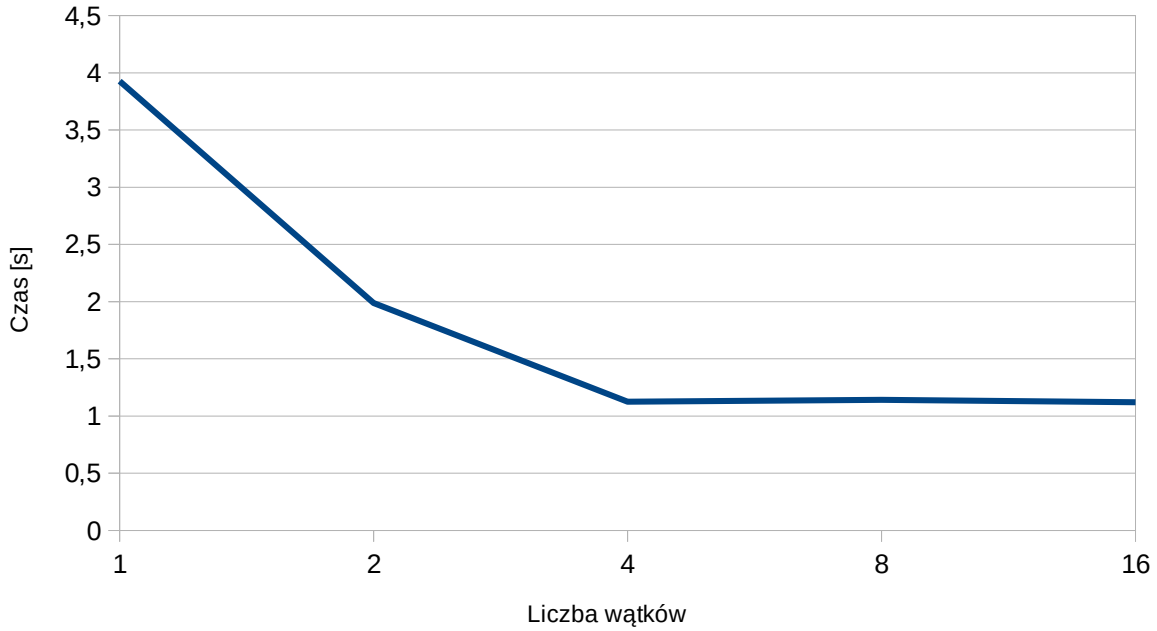
Dla porównania obliczeń wykorzystano tekst o rozmiarze około 2MB. Obliczenia przeprowadzono na procesorze 4-rdzeniowym intel i5-4590 w systemie operacyjnym Linux dystrybucja Ubuntu 18.04. Wykorzystano kompilator g++ w wersji 7.3.0.

Liczba wątków	Czas [s]
1	3.92676
2	1.9877
4	1.12562
8	1.14172
16	1.12037

Całość wyników programu:

```
/home/szymi/src/studies/PRiR/v2/seq/cmake-build-debug/openmp_prir
----LINEAR SEARCH----
----OPENMP PARALLEL SEARCH----
----OPENMP PARALLEL SEARCH----
----OPENMP PARALLEL SEARCH----
----OPENMP PARALLEL SEARCH----
----OPENMP PARALLEL SEARCH----
Linear --- 4.66877
omp 1 --- 3.92676
omp 2 --- 1.9877
omp 4 --- 1.12562
omp 8 --- 1.14172
```

omp 16 --- 1.12037



3. Wnioski

Z wartości czasu w testach wynika, że największe przyspieszenie uzyskano przy zwiększeniu liczby wątków z 1 na dwa. Można było się spodziewać, że czas obliczeń skracany powinien być o około 50 procent (mniej dla krótkich obliczeń – większy wpływ czasu potrzebnego na inicjalizacje, a większy dla dłuższych). Niemniej jednak nie mogłem wykonać obliczeń dla większych plików wejściowych, ponieważ rozmiary alokowanych wektorów/stringów był zbyt dyże. Kolejne dodanie wątków nie wprowadza takiego przyspieszenia, Przy dużej liczbie niezależnych obliczeń zyski z przyspieszenia są coraz mniejsze.

Teoretycznie, przyspieszenie spowodowane zrównolegleniem mogłoby być co najwyżej liniowe – podwojenie liczby jednostek obliczeniowych nie może zmniejszyć czasu obliczeń o więcej niż połowę, jednak w praktyce osiągnięcie optymalnego przyspieszenia nie jest możliwe. Większość realizacji osiąga przyspieszenie bliskie optymalnemu tylko dla małej liczby jednostek obliczeniowych. [1]

4. Bibliografia

[1] https://pl.wikipedia.org/wiki/Obliczenia_r%C3%B3wnoleg%C5%82e
[2] https://www.openmp.org/wp-content/uploads/openmp-examples-4.5.0.pdf