

Zadanie 3 - Trigramy - Analiza dokumentów

Programem spełniającym polecenie zadania 3. jest taki, który przyjmując listę plików wejściowych, przeanalizuje ich zawartość tworząc mapę dystrybucji n-gramów o rozmiarze 3 (trigramów).

Część kodu odpowiedzialna za uzupełnienie informacji o częstości występowania trigramów w tekście źródłowym prezentuje się następująco:

```
1  #pragma omp parallel
2  {
3      unsigned int threadNumber = omp_get_thread_num();
4      unsigned int startPos = chunkSize * threadNumber;
5      unsigned int endPos = chunkSize * (threadNumber + 1);
6      endPos = std::min(endPos, (unsigned int)contents.size());
7      for (unsigned int i = startPos; i < endPos; i += 3)
8      {
9          string trigram = contents.substr(i, 3);
10         ++trigramDistribution[trigram];
11     }
12 }
```

Wcześniejsza implementacja:

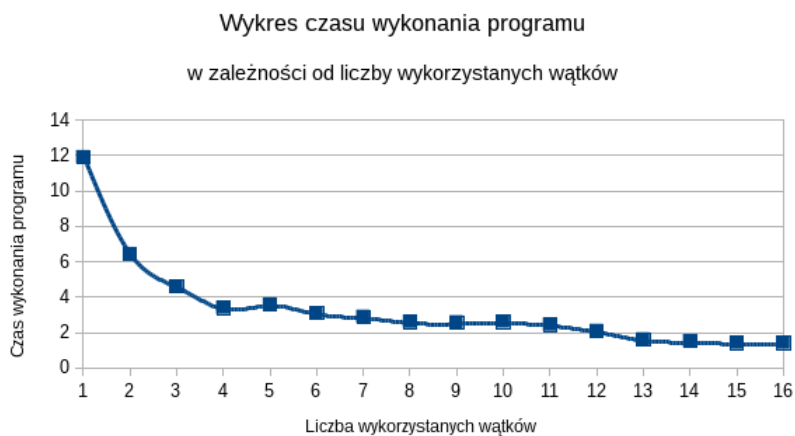
```
1  #pragma omp parallel num_threads(threadCount)
2  {
3      std::string threeLetters;
4      unsigned int endPosition = portion * (omp_get_thread_num() + 1);
5
6      if (endPosition > contents.size())
7      {
8          endPosition = contents.size();
9      }
10
11     #pragma for default(none) shared(contents, trigram) firstprivate(
12         portion) private(threeLetters)
13     for (int i = portion * omp_get_thread_num();
14         i != portion * (omp_get_thread_num() + 1); i += 3)
15     {
16         threeLetters = std::string(contents.substr(i, 3));
17         trigram[threeLetters]++;
18     }
19 }
```

Była podatna na błędy współbieżnego dostępu do mapy wystąpień trigramów, a także jej wydajność była dużo niższa. Wynikało to zapewne z tego w jaki sposób dane są porcjowane – porcjowanie ręczne pozwala na większą kontrolę nad rozmiarem porcji, co więcej, pozwala ustawić nieregularny rozmiar porcji, co jest szczególnie przydatne, gdy rozmiar danych wejściowych nie jest podzielny przez $3 * \text{[liczba wątków]}$.

Przebieg

Obliczenia zostały wykonane na serwerze CUDA.

Poniżej wykresy przedstawiający rezultat przeprowadzonych operacji na wątkach.



Wnioski

Zauważalny spadek wydajności można tłumaczyć liczbą rdzeni maszyny, na której wykonywano pomiary (serwer CUDA) oraz sposobem przydziału zadań – algorytmem karuzeli – klauzula `schedule(static)`.