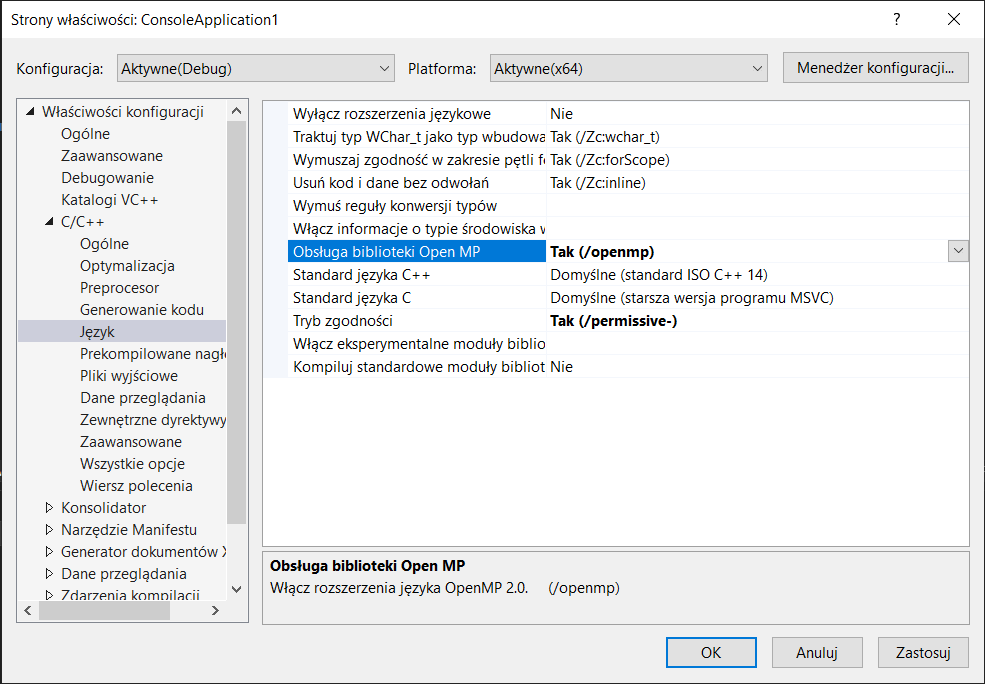
|  |  |
| --- | --- |
| Akademia Nauk Stosowanych w Nowym Sączu  Wydział Nauk Inżynieryjnych | |
| Imię i nazwisko: | Michał Bernardy |
| Grupa: | P1 |
| Ocena: |  |

**Wstęp**

Sprawozdanie prezentuje analizę wydajności programowania wielowątkowego z wykorzystaniem OpenMP w środowisku Visual Studio. Badania skupiały się na kluczowych dyrektywach OpenMP (if, num\_threads, barrier, section) oraz klauzuli nowait. Eksperymenty przeprowadzono na przykładzie programu mnożącego macierz przez wektor oraz na własnej implementacji obliczania długości wektora. Głównym celem było zbadanie wpływu liczby wątków oraz różnych mechanizmów synchronizacji na wydajność obliczeń równoległych.****

**Zadanie1**

Wyniki kompliacji bez nowait:

Iteracja 1: Wypelnienie sekwencyjne w 657 ms

Iteracja 1: Mnozenie sekwencyjne w 43 ms

Iteracja 2: Wypelnienie sekwencyjne w 1318 ms

Iteracja 2: Mnozenie sekwencyjne w 48 ms

Iteracja 3: Wypelnienie sekwencyjne w 887 ms

Iteracja 3: Mnozenie sekwencyjne w 34 ms

Iteracja 4: Wypelnienie sekwencyjne w 495 ms

Iteracja 4: Mnozenie sekwencyjne w 33 ms

Iteracja 5: Wypelnienie sekwencyjne w 485 ms

Iteracja 5: Mnozenie sekwencyjne w 37 ms

Calkowity czas sekwencyjny dla 5 iteracji: 4046 ms

=== Testowanie z 2 watkami ===

Iteracja 1: Wypelnienie rownolegle w 288 ms

Iteracja 1: Mnozenie rownolegle w 30 ms

Iteracja 2: Wypelnienie rownolegle w 293 ms

Iteracja 2: Mnozenie rownolegle w 34 ms

Iteracja 3: Wypelnienie rownolegle w 333 ms

Iteracja 3: Mnozenie rownolegle w 17 ms

Iteracja 4: Wypelnienie rownolegle w 275 ms

Iteracja 4: Mnozenie rownolegle w 23 ms

Iteracja 5: Wypelnienie rownolegle w 300 ms

Iteracja 5: Mnozenie rownolegle w 30 ms

Calkowity czas rownolegly dla 5 iteracji z 2 watkami: 1629 ms

=== Testowanie z 4 watkami ===

Iteracja 1: Wypelnienie rownolegle w 327 ms

Iteracja 1: Mnozenie rownolegle w 43 ms

Iteracja 2: Wypelnienie rownolegle w 284 ms

Iteracja 2: Mnozenie rownolegle w 20 ms

Iteracja 3: Wypelnienie rownolegle w 337 ms

Iteracja 3: Mnozenie rownolegle w 46 ms

Iteracja 4: Wypelnienie rownolegle w 345 ms

Iteracja 4: Mnozenie rownolegle w 23 ms

Iteracja 5: Wypelnienie rownolegle w 189 ms

Iteracja 5: Mnozenie rownolegle w 26 ms

Calkowity czas rownolegly dla 5 iteracji z 4 watkami: 1649 ms

=== Testowanie z 8 watkami ===

Iteracja 1: Wypelnienie rownolegle w 195 ms

Iteracja 1: Mnozenie rownolegle w 23 ms

Iteracja 2: Wypelnienie rownolegle w 192 ms

Iteracja 2: Mnozenie rownolegle w 25 ms

Iteracja 3: Wypelnienie rownolegle w 189 ms

Iteracja 3: Mnozenie rownolegle w 28 ms

Iteracja 4: Wypelnienie rownolegle w 180 ms

Iteracja 4: Mnozenie rownolegle w 25 ms

Iteracja 5: Wypelnienie rownolegle w 194 ms

Iteracja 5: Mnozenie rownolegle w 18 ms

Calkowity czas rownolegly dla 5 iteracji z 8 watkami: 1078 ms

=== Podsumowanie ===

Czas sekwencyjny: 4046 ms

Wyniki z nowait:

=== Testowanie z 4 watkami bez klauzuli nowait ===

Iteracja 1: Wypelnienie bez nowait w 318 ms

Iteracja 1: Mnozenie bez nowait w 26 ms

Iteracja 2: Wypelnienie bez nowait w 233 ms

Iteracja 2: Mnozenie bez nowait w 27 ms

Iteracja 3: Wypelnienie bez nowait w 236 ms

Iteracja 3: Mnozenie bez nowait w 17 ms

Iteracja 4: Wypelnienie bez nowait w 197 ms

Iteracja 4: Mnozenie bez nowait w 28 ms

Iteracja 5: Wypelnienie bez nowait w 239 ms

Iteracja 5: Mnozenie bez nowait w 21 ms

Calkowity czas bez nowait dla 5 iteracji: 1349 ms

=== Testowanie z 4 watkami i klauzula nowait ===

Iteracja 1: Wypelnienie z nowait w 219 ms

Iteracja 1: Mnozenie w 28 ms

Iteracja 2: Wypelnienie z nowait w 263 ms

Iteracja 2: Mnozenie w 28 ms

Iteracja 3: Wypelnienie z nowait w 602 ms

Iteracja 3: Mnozenie w 63 ms

Iteracja 4: Wypelnienie z nowait w 212 ms

Iteracja 4: Mnozenie w 28 ms

Iteracja 5: Wypelnienie z nowait w 245 ms

Iteracja 5: Mnozenie w 17 ms

Calkowity czas z nowait dla 5 iteracji: 1739 ms

=== Testowanie z 8 watkami bez klauzuli nowait ===

Iteracja 1: Wypelnienie bez nowait w 172 ms

Iteracja 1: Mnozenie bez nowait w 21 ms

Iteracja 2: Wypelnienie bez nowait w 179 ms

Iteracja 2: Mnozenie bez nowait w 18 ms

Iteracja 3: Wypelnienie bez nowait w 171 ms

Iteracja 3: Mnozenie bez nowait w 18 ms

Iteracja 4: Wypelnienie bez nowait w 172 ms

Iteracja 4: Mnozenie bez nowait w 18 ms

Iteracja 5: Wypelnienie bez nowait w 181 ms

Iteracja 5: Mnozenie bez nowait w 17 ms

Calkowity czas bez nowait dla 5 iteracji: 974 ms

=== Testowanie z 8 watkami i klauzula nowait ===

Iteracja 1: Wypelnienie z nowait w 187 ms

Iteracja 1: Mnozenie w 17 ms

Iteracja 2: Wypelnienie z nowait w 170 ms

Iteracja 2: Mnozenie w 18 ms

Iteracja 3: Wypelnienie z nowait w 168 ms

Iteracja 3: Mnozenie w 18 ms

Iteracja 4: Wypelnienie z nowait w 172 ms

Iteracja 4: Mnozenie w 17 ms

Iteracja 5: Wypelnienie z nowait w 172 ms

Iteracja 5: Mnozenie w 24 ms

Calkowity czas z nowait dla 5 iteracji: 969 ms

**Zadanie2**

Inicjalizacja wektora o rozmiarze 100000000...

Obliczanie dlugosci wektora - metoda sekwencyjna...

Czas sekwencyjny: 341 ms

Dlugosc wektora (sekwencyjnie): 57269.295655

Obliczanie dlugosci wektora - metoda rownolegle z sekcjami...

Watek 0 rozpoczyna prace na sekcji 1

Watek 1 rozpoczyna prace na sekcji 2

Watek 3 rozpoczyna prace na sekcji 3

Watek 2 rozpoczyna prace na sekcji 4

Czas rownolegly: 399 ms

Dlugosc wektora (rownolegle): 57269.295655

**Wnioski**

Przeprowadzone eksperymenty z OpenMP wyraźnie pokazały zależność między liczbą wątków a wydajnością programu. Wykonanie sekwencyjne zajęło 4046 ms, podczas gdy wykorzystanie 2 wątków przyspieszyło pracę 2,5-krotnie, redukując czas do 1629 ms. Ciekawe jest to, że zwiększenie liczby wątków do 4 nie przyniosło zauważalnej poprawy (1649 ms), co sugeruje, że nie zawsze dodanie większej liczby wątków przekłada się na lepszą wydajność. Najlepsze wyniki uzyskano dla 8 wątków, osiągając czas 1078 ms, czyli przyspieszenie 3,8-krotne względem wersji sekwencyjnej. Analiza działania klauzuli nowait pokazała, że dla 4 wątków wersja bez nowait (1349 ms) była efektywniejsza niż z tą klauzulą (1739 ms), natomiast przy 8 wątkach różnica stała się minimalna, z lekką przewagą wersji wykorzystującej nowait (969 ms wobec 974 ms). Może to wskazywać, że korzyści z eliminacji barier synchronizacyjnych są bardziej widoczne przy większej liczbie równolegle działających wątków. Zaskakujące wyniki zaobserwowano przy programie obliczającym długość wektora - wersja sekwencyjna wykonała się szybciej (341 ms) niż równoległa implementacja wykorzystująca sekcje (399 ms). Ta obserwacja uwidacznia, że narzut związany z tworzeniem i zarządzaniem wątkami oraz koszt synchronizacji mogą przewyższać korzyści płynące z równoległego przetwarzania, szczególnie w przypadku mniej złożonych operacji. Badania jednoznacznie wskazują, że optymalna konfiguracja OpenMP zależy silnie od charakterystyki rozwiązywanego problemu oraz architektury sprzętowej. Projektując rozwiązania równoległe, należy eksperymentalnie dobierać liczbę wątków i mechanizmy synchronizacji, mając na uwadze, że w niektórych przypadkach, zwłaszcza dla mniejszych zbiorów danych, podejście sekwencyjne może okazać się bardziej efektywne.