**Programowanie współbieżne i rozproszone – laboratorium**

**Instrukcja do ćwiczenia laboratoryjnego IV**

**Wprowadzenie do programowania w OpenMP**

Autor: S. Samolej

Rzeszów, 2018

Przygotowanie środowiska programistycznego:

***Code::Blocks – linux:***

W nowoutworzonym projekcie należy „kliknąć” lewym przyciskiem myszy na nazwę projektu w oknie „Management”, zakładce „Projects”. Z menu kontekstowego należy wybrać „Build options…”. W zakładce „Linker settings” należy w oknie „Other linker options” dopisać „-**lgomp –pthread**”. W zakładce „Compiler settings” -> „Other options” wpisać „ **–fopenmp**”.

***Kompilowanie przy użyciu linii poleceń linux:***

„**gcc –fopenmp plik\_zrodlowy.c –o plik\_docelowy.o –lgomp –pthread**”

***Kompilator VS:***

W nowoutworzonym projekcie należy „kliknąć” lewym przyciskiem myszy na nazwę projektu w oknie „Soluton Explorer”. Z menu kontekstowego wybrać „Properties”, a następnie gałąź „C/C++” i liść „Language”. W linii „Open MP Support” należy wybrać „**Yes (/openmp)**”.

1. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_01\_first.c”. Program pokazuje podstawową metodę tworzenia programów współbieżnych z zastosowaniem biblioteki OpenMP. Funkcja „omp\_get\_thread\_num()” zwraca numer wykonywanego wątku obliczeniowego.
2. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_02\_for.c”. Program pokazuje metodę zrównoleglania obliczeń w pętli „for”. Proszę zwrócić uwagę na interpretację klauzul „default”, „shared” i „private”. Czy wszystkie pętle „for” zgodne z notacją języka C/C++ można zrównoleglić?
3. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_03\_fork\_fork.c”. W programie pokazano, jak wydzielić fragment kodu wykonywany tylko w jednym wątku. Program zawiera dwie wydzielone współbieżne fazy obliczeniowe, w których następuje zrównoleglenie prac, a następnie synchronizacja na „ukrytej” barierze.
4. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_04\_critical.c”. Program jest rozwiązaniem problemu redukcji z zastosowaniem sekcji krytycznej.
5. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_05\_reduction.c”. Jest on wydajniejszą wersją poprzedniego przykładu, gdzie problem redukcji jest rozwiązywany z zastosowaniem klauzuli „reduction”.
6. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_06\_sequential\_counting.c”. Jest on sekwencyjnym rozwiązaniem wariantu algorytmu sortowania przez zliczanie.
7. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_07\_parallel\_counting.c”. Jest to podstawowy sposób zrównoleglenia sekwencyjnego algorytmu sortowania przez zliczanie.
8. Proszę skompilować, uruchomić i przeanalizować przykładowy program „OpenMP\_08\_pparallel\_counting.c”. Jest to inny sposób zrównoleglenia sekwencyjnego algorytmu sortowania przez zliczanie. Zastosowano zrównoleglenie w stopniu N2, gdzie N jest rozmiarem tablicy do posortowania.
9. Proszę zaproponować współbieżne rozwiązanie znajdowanie największej liczby w zadanej N-elementowej tablicy. Program ma dokonywać obliczeń w M zadanych wątkach, gdzie M <= N.
10. Proszę zaproponować współbieżne rozwiązanie obliczenia przybliżenia liczby PI metodą Monte-Carlo. Stopień zrównoleglenia aplikacji należy pozostawić systemowi obliczeń współbieżnych. Proszę rozważyć zastosowanie „erand48()” na platformie Linux do losowania liczb z zakresu [0.0.1.0).