Praktikum 01

IF3230 Sistem Paralel dan Terdistribusi

OpenMP - Radix Sort

Dipersiapkan oleh:
Asisten Laboratorium Sistem Terdistribusi

Didukung oleh:

Start Date: 21 Februari 2019

End Date: 1 Maret 2019

A. Persiapan Praktikum

Pada praktikum kali ini anda ditugaskan untuk mengeksplorasi OpenMP, suatu API yang dapat digunakan untuk pemrograman berbasis *shared memory multiprocessing*.

Anda akan menggunakan sebuah mesin dengan IP address **167.205.32.40**. Mesin tersebut dapat diakses dari jaringan dalam ITB atau menggunakan fasilitas VPN. Gunakan username berupa NIM dengan password "guest" untuk login. Praktikum menggunakan OpenMP dapat dilakukan pada mesin tersebut. Jika menggunakan mesin tersebut untuk melakukan praktikum, **diwajibkan** mengganti password terlebih dahulu. Selain menggunakan mesin yang telah disediakan, Anda juga diperbolehkan melakukan praktikum ini di mesin yang Anda miliki jika *compiler* yang Anda gunakan telah mengimplementasikan OpenMP API. Daftar *compiler* yang telah mensupport OpenMP dapat dilihat pada halaman ini.

Berikut akan diberikan contoh file `omp_hello.c` serta cara melakukan kompilasi dan menjalankan program.

***File* `omp_hello.c`**

```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>

void Hello(void); /* Thread function */

int main(int argc, char *argv[]) {
   int thread_count = strtol(argv[1], NULL, 10);

   #pragma omp parallel num_threads(thread_count)
   Hello();

   return 0;
}

void Hello(void) {
   int my_rank = omp_get_thread_num();
   int thread_count = omp_get_num_threads();

   printf("Hello from thread %d of %d\n", my_rank, thread_count);
}
```

Petunjuk Kompilasi

Lakukan kompilasi dengan menambahkan argumen `-fopenmp`. Berikut merupakan contoh perintah kompilasi untuk file `omp_hello.c`

```
gcc -g -Wall -o omp_hello omp_hello.c -fopenmp
```

Petunjuk Menjalankan Program

Untuk menjalankan program yang telah dibuat, perintah yang dapat digunakan adalah sebagai berikut. Angka 4 menunjukkan banyaknya *threads* yang dapat digunakan pada program.

```
./omp_hello 4
```

B. Spesifikasi Tugas

Pada tugas ini, Anda diminta untuk mengimplementasikan *radix sort* secara paralel pada OpenMP. Radix Sort adalah algoritma *sorting* yang mengurutkan data menggunakan kunci bilangan (*integer*) dengan melakukan pengelompokan kunci berdasarkan angka yang memiliki kesamaan posisi dan nilai (ratusan dengan ratusan, puluhan dengan puluhan). Ide utama dari algoritma ini adalah melakukan pengurutan untuk setiap angka (digit) dari *least significant digit* sampai dengan *most significant digit*.

Sebagai contoh, berikut merupakan *source code* radix sort dalam bahasa C yang dijalankan secara sekuensial.

Untuk mengukur apakah paralelisasi radix sort berhasil dilakukan, anda diminta untuk melakukan uji kinerja (*performance*) pada program yang anda buat. Pengujian kinerja dilakukan dengan metode sebagai berikut:

1. Inisialisasi array of integer berisi **N** elemen. Setiap elemen berisi nilai random yang di-*generate* oleh rand(). Gunakan seed yang sama untuk setiap pembangkitan array. Gunakan fungsi di bawah untuk membangkitkan array.

```
void rng(int* arr, int n) {
    int seed = 13515000; // Ganti dengan NIM anda sebagai seed.
    srand(seed);
    for(long i = 0; i < n; i++) {
        arr[i] = (int)rand();
    }
}
```

2. Terapkan radix sort pada array sehingga array terurut dari kecil ke besar.
3. Pengukuran waktu dilakukan pada saat radix sort dimulai hingga selesai. Jangan masukkan pembangkitan array pada pengukuran waktu.
4. Gunakan *microsecond* sebagai satuan dasar pada perhitungan waktu yang Anda gunakan.

Pada pengujian ini, **N** yang digunakan adalah 5.000, 50.000, 100.000, 200.000, dan 400.000. Agar pengujian lebih akurat, jalankan setiap kasus uji setidaknya 3 (tiga) kali. Tuliskan hasil pengujian Anda pada laporan pengerjaan yang berisi:

- Deskripsi solusi paralel. Berikan ilustrasi jika perlu.
- Analisis solusi yang anda berikan. Apakah mungkin terdapat solusi yang memberikan kinerja lebih baik?
- Jumlah *thread* yang digunakan. Kenapa anda memilih angka tersebut?
- Pengukuran kinerja untuk tiap kasus uji (jumlah N pada array) dibandingkan dengan radix sort serial.
- Analisis perbandingan kinerja serial dan paralel. Analisis yang diharapkan adalah analisis yang minimal dapat menjelaskan setiap hasil pengukuran kinerja sebelumnya.

C. Pengumpulan dan Deliverables

Tugas dikerjakan dalam kelompok sebanyak maksimal 2 orang (anggota kelompok tidak boleh dari kelas yang berbeda). Fork spesifikasi tugas ini serta contoh source code OpenMP dari repository http://gitlab.informatika.org/IF3230-2019/OpenMP . Repository yang digunakan wajib **private,** jika ada yang membuat repository **public** dan mengerjakan di repository tersebut akan diperingatkan oleh asisten, dan jika dalam waktu tertentu masih tidak merubah repository yang digunakan menjadi **private** maka asisten berhak untuk tidak menilai pekerjaan tersebut. Deadline pekerjaan persoalan yang diberikan pada deskripsi di atas maksimal **1 Maret 2019** pukul **23:59 WIB**.

Setiap kelompok wajib mengisi data pada link google sheet berikut. Nantinya setiap kelompok wajib untuk mengundang salah satu asisten pada repository **private**-nya, hal ini digunakan untuk penilaian sehingga diwajibkan untuk setiap kelompok mengundang salah satu asisten. Nama asisten dan id gitlab asisten akan diberitahukan nantinya pada **google sheet** yang sudah diberikan.

Lakukan merge request pada repository awal paling lambat pada waktu dan tanggal yang sama. Merge request dilakukan dengan judul **Praktikum1_K0[1|2|3]_<NIM1>_<NIM2>.** Perhatikan bahwa **keterlambatan pengumpulan dapat mengakibatkan nilai 0 (nol)**.

Beberapa file yang harus ada dalam repositori tersebut diantaranya:

- Direktori **src** yang berisi source code yang anda buat.
- File **output** yang berisi hasil uji radix sort pada data uji. Contoh output serta data uji akan diberikan pada repository gitlab.
- **Makefile**. Buatlah sehingga kompilasi program dapat dilakukan hanya dengan pemanggilan command `'make'` saja.
- Buatlah sehingga program yang anda buat dapat dijalankan dengan pemanggilan command **`./radix_sort <N>`.** Dengan N adalah jumlah elemen pada array.
- File **README.md** yang berisi:
    - Petunjuk penggunaan program.
    - Pembagian tugas. Sampaikan dalam list pengerjaan untuk setiap mahasiswa. Sebagai contoh: XXXX mengerjakan fungsi YYYY, ZZZZ, dan YYZZ.

- Laporan pengerjaan, dengan struktur laporan sesuai dengan deskripsi pada bagian sebelumnya.

Segala bentuk kecurangan yang terjadi akan ditindaklanjuti oleh asisten dan dikenakan konsekuensi.