Tugas Pemrograman Parallel

Oleh:

D121171519 - Glenn Claudio Ivan Petrus

Departemen Teknik Informatika

Fakultas Teknik

Universitas Hasanuddin

2020

Execution Time Fibonacci Serial – Rekursif

n = 40

```
102334155
exec_time 0.859000
```

T*(n) = 0,859000

Execution Time Fibonacci Parallel – Rekursif

n = 40

```
102334155
exec_time 568.040000
```

Tp(n) = 565,040000

Cost of The Parallel Program

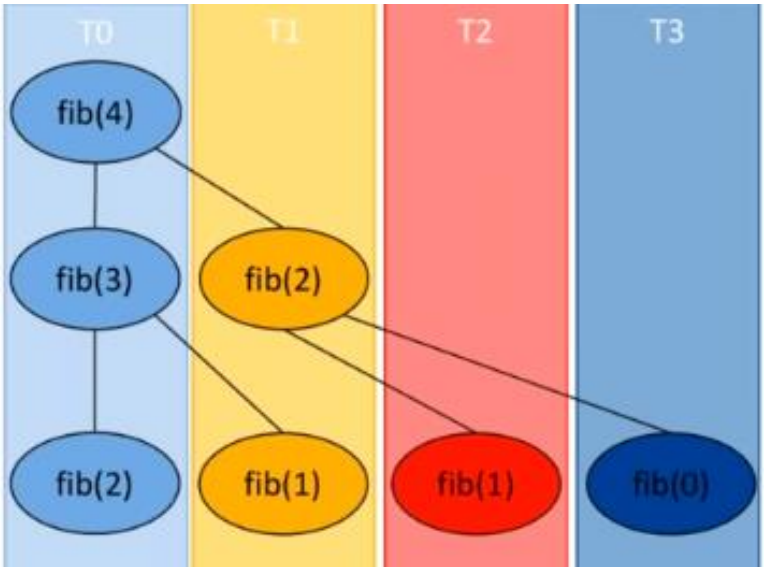
p = 4

Cp(n) = p x Tp(n)

Cp(n) = 4 x 565,040000

Cp(n) = 2.260,16

Program parallel tersebut tidak *cost optimal*, program parallel dikatakan *cost optimal* jika Cp(n) = T*(n) atau T*(n) / Cp(n) ∈ Θ(1), penyebabnya kita dapat mengamati bahwa implementasi ini melakukan banyak pekerjaan yang berulang, seperti contoh pohon rekursif berikut.



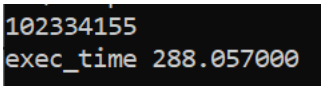
Gambar 1 https://www.youtube.com/watch?v=Wx4eQQihP6I&t=935, n = 4

Penyebab utamanya juga karena adanya *overhead* saat penciptaan dan manajemen *tasks*.

Optimasi Tanpa Mengubah Algoritma – Parallel - Fibonacci Rekursif

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

int fib(int n){
      int x, y;
      if(n<2) return n;
      else{
            #pragma omp task shared(x) if(n>20)
            x = fib(n-1);

            y = fib(n-2);

            #pragma omp taskwait
            return x+y;
      }
}

int main(){
      int n = 40;
      int R = 0;
      long start, finish;
      double exec_time;

      start = clock();
      #pragma omp parallel
      {
            #pragma omp single
            R = fib(n);
      }

      printf("%d\n", R);
      finish = clock();

      exec_time = (double) (finish-start)/CLOCKS_PER_SEC;
      printf("exec_time %1f\n", exec_time);

      return 0;
}
```

Output Optimasi Tanpa Mengubah Algoritma – Parallel - Fibonacci Rekursif



Jika n < 20, kode akan dieksekusi secara serial. Hal tersebut akan memangkas *task tree*. Optimasi lain yang dilakukan dengan tidak membuat *task* kedua. Jika hal tersebut tidak dilakukan maka *parent thread* tidak akan melakukan apa-apa ketika *task-task* dibuat. Namun masih terdapat *overhead* pada progam parallel ini dibandingkan program serialnya.

Optimasi Dengan Mengubah Algoritma – Serial - Fibonacci Dengan Rumus

```c
#include<stdio.h>
#include<math.h>
#include<time.h>

int fib(int n) {
  double phi = (1 + sqrt(5)) / 2;
  return round(pow(phi, n) / sqrt(5));
}

int main ()
{
  int n = 40;
  int R = 0;
  long start, finish;
  double exec_time;

  start = clock();
  R = fib(n);
  printf("%d\n", R);
  finish = clock();

  exec_time = (double) (finish-start)/CLOCKS_PER_SEC;
  printf("exec_time %1f\n", exec_time);

  return 0;
}
```

Optimasi Dengan Mengubah Algoritma – Serial - Fibonacci Dengan Rumus

```
102334155
exec_time 0.000000
```