Assignment 2 - Parallel DFS

Ghazal Rafiei

March 14, 2022

1 Introduction

The objective in this assignment is to parallelize iterative depth-first search algorithm for undirected graphs. In the rest, we will mention the platform specifications, implementation approach, experiment method and the results.

2 Platform

The program is written in C++ language using OpenMP library. Furthermore, here is the specification of the system.

```
Linux zenbookux434flcux433flc 5.15.25-1-MANJARO 1 SMP PREEMPT x86_64 GNU/Linux
CPU(s):  8
Vendor ID: GenuineIntel
Model name:  Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
MemTotal:  16179008 kB
```

3 Implementation

Among the well-known recursive and iterative approaches of implementing DFS algorithm, iterative is chosen. The reason is that in recursive approach, the call stack of memory may run out but the user-defined data structure stack can allocate even a large memory. The algorithm is implemented using `std::atomic<bool>` for keeping records of visited nodes, and `std::stack<int>` for keeping the frontier nodes.

3.1 Parallelization

To parallelize the algorithm, it is not optimized to parallelize one execution of DFS and use the mentioned stack as a shared memory because the critical section would be large and it appears to be a serial algorithm. Instead, we start to run DFS from different points of the graph. The number of these points is equal to the number of threads and once a thread finishes its job, another DFS from another node will be started to execute.

4 Experiment

In order to examine how much our proposed algorithm is optimized, we run the sequential and the parallel iterative DFS with a graph including 10 nodes and repeat it 10000 times and store the average time as a time of running. It is worth noting that if we run the two algorithms sequentially in the one program, the second algorithm, no matter which one, will be executed faster. The hypothesis is that when a program starts to run, it will allocate more resources of CPU and memory as the time passes and the final results will not be accurate. Hence, the sequential and the parallel algorithms are run separately.

5 Results

In this section, the duration of running sequential and proposed parallel DFS algorithms, is shown in the table. Each number is indicative of average of 10000 runs.

	Time (milliseconds)
Sequential	0.005
Parallel	0.003