COMP90025 Parallel and Multicore Computing

Project 1A - Diameter of Graph

Aaron Harwood and Lachlan Andrew

School of Computing and Information Systems
The University of Melbourne

2019 Semester II

Summary

- The first project is group work (size at most 2) and is divided into two parts (so that you can get feedback earlier). This is Part A.
- A sequential algorithm for computing the diameter of a graph has been provided on LMS. The program reads a graph from standard input (`stdin`) and outputs the diameter.
- The diameter of a graph is defined as the maximum of the shortest path lengths between all pairs of nodes.
- The graph is assumed to be directed and connected, and edge weights are always greater than 0.

Input format

- The first line of input is a single integer giving the number of vertices in the graph, N.
- The second line is a single integer giving the number of edges, M. Each subsequent line of the input file provides a tuple of positive integers:

  fromVertex toVertex weight

- Vertices are numbered 0 to $N - 1$.

Example Input

```
5
10
0 1 5
0 2 2
0 4 15
1 2 6
1 3 1
3 1 4
3 0 9
3 4 2
4 1 6
2 4 1
```

Example Graph

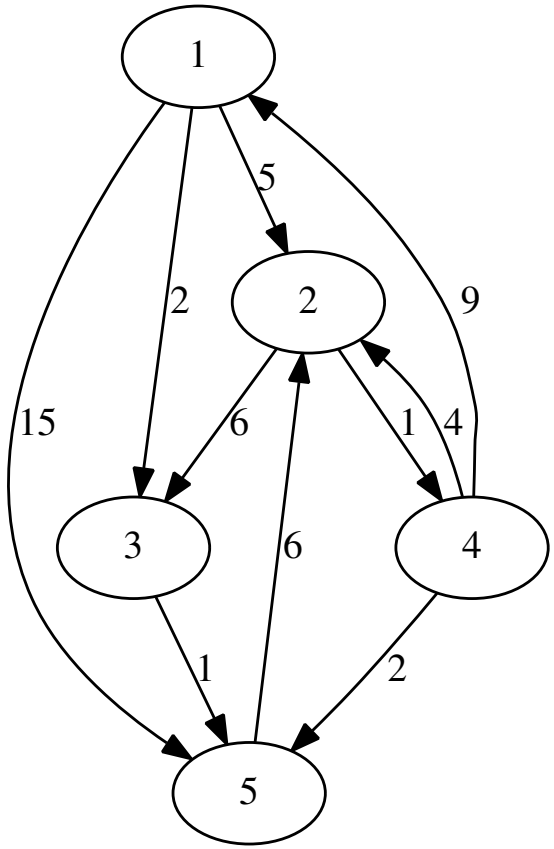

Figure: Example Graph

Diameter 17, given by the shortest path from vertex 3 to vertex 1.
(In this figure nodes are numbered from 1, unlike in the data file.)

Tasks

- Write an OpenMP program that computes and outputs, in the same way, the diameter of a given graph as done by the sequential program.
- Aim to have your program run as fast as possible. In doing so, you may alter the calculations of the program, so long as the final output is correct.
- Use the sequential code as a base. Do not change the file before the end of `main()`. If you need to include additional headers, they can be included after `main()`. In particular, do not alter the timing statements or add any of your own timing statements.
- Write at most 550 words that outlines how you achieved parallelism/high performance. Include tables and/or charts of your own measurements that support your discussion.
- Write at most 200 additional words that discuss Illiac. Describe the important features of the architecture in the terms introduced in this course. Discuss whether the algorithm you used in this project would be suitable for running on any of the Illiac computers.

Assessment

- Project 1 A is worth **7%** of your total assessment. It is group work in size of 2 at most.
- Assessment of the report (3/7) is based on the level of details and presentation.
- Assessment of the program (4/7) is based on correctness and performance. Incorrect programs (i.e. that give incorrect outputs or that fail to compile/run) will attract few if any marks. The top 5 fastest running programs, when given a mystery work load (you will not be told the work load in advance), will be given a bonus mark; i.e. the maximum mark for this project part is 8/7.
- At most five people will get the bonus mark. If a sixth person is tied with anyone in the top five, then anyone tied with that sixth person will not receive a bonus mark.

Submission

- Submit a PDF of your report (use PDF only, no other format will be assessed) via LMS on or before **Saturday 31st August**. As well you will need to submit your program (either via LMS or directly on Spartan). Instructions for doing this will be given closer to the deadline.
- Optimal code often requires specific compiler options. As a comment in the last line of your C code, put the exact command line used to compile your code. Do not put anything else on that line
- Use 10pt font, single line spacing, 25 mm margins all around and double column formatting. Use appropriate headings and clearly label and refer to tables, figures and references.
- Clearly put your name and login name at the top of the report.