Parallel Graph Development

Soham Tamba
B.Tech. Computer Science, Optimization Enthusiast

Introduction

Augment the LightGraphs.jl package with respect to:

• **Parallelization** of sequentially implemented graph algorithms.

• **Optimization** of existing sequential implementations .

• **Implementation** of greedy heuristics for NP-hard graph problems.

Why Graphs?

Many real world scenarios can be modeled as graphs for which various Graph algorithms can be used to obtain the solution to problems like Shortest Path or insightful inferences such as Centrality measures.

Pre-Requisites

•Basic knowledge of **Graphs**.
•Proficiency in **Data Structures and Algorithms**.
•Basic knowledge of **Parallel Computing**.

Benchmarks

Algorithm	1	2	3
Breadth-First Search	1.80	2.08	2.12
PageRank	1.77	1.54	1.65
Bellman Ford SSSP	-	-	1.88
Floyd Warshall APSP	-	-	1.27
Johnson APSP	-	-	2.10
Randomized Heuristic	1.88	1.70	1.66
Closeness Centrality	-	-	2.17

Speed-up: Parallel vs Sequential

Algorithm	1	2	3
PageRank	3.05	3.37	3.17
Dijkstra SSSP	2.80	2.10	1.68
Prim MST	7.65	4.25	4.05
Kruskal MST	7.70	3.37	2.80

Speed-up: Sequential Optimizaton

Results

Benchmarks were conducted on a 64-bit linux machine with **4 cores**. The input was set to a scale-free graph in SNAPDatasets.jl with random edge weights.

No.	Graph	Vertices	Edges
1	**Twitter Social Circles**	81,306	1,342,310
2	**Astro Physics Collaboration**	17,903	197,031
3	**FaceBook Social Circles**	4,039	88,234

Greedy Heuristics Implemented :-
•Minimum Vertex Cover
•Minimum Dominating Set
•Maximum Independent Set
•Maximum Edge Matching
•Vertex Coloring
•Minimum Steiner Tree
•Travelling Salesman

Utility Subroutines Implemented:-
•Load-balanced partitioning
•Parallel Heuristic Repetition

Unsuccessful Attempts:-
•Parallel Kruskal using parallel Union-Find datastructure.
•Parallel Dijkstra using parallel Priority Queue

Future Work

Shared memory parallelism is a relatively recent feature in Julia. When this functionality of Julia matures and has a performant interface for thread calls, relaxed atomics, etc. it should be easier to implement other parallel graph algorithms such as Kruskal, Dijkstra etc.

Acknowledgements

I would like to thank my mentor, **Divyansh Srivastav** and LightGraphs co-owner **Seth Bromberger** for reviewing my code and providing valuable suggestions to improve it.