15418 Project Proposal — Parallel Maximal Independent Set

Clarissa Xu, Arvind Mahankali

March 2022

1 Summary

We will implement parallel algorithms for finding maximal independent sets in the shared memory, message passing, and data-parallel models.

2 Background

We will be considering the problem of finding a maximal independent set of a graph $G = (V, E)$, that is, a subset $S \subset V$ of vertices such that S is an independent set, and moreover, for any $v \in V \setminus S$, $S \cup \{v\}$ is not an independent set. As described in Algorithm 1 of this survey by Shi, Wang, Shang, there is a simple sequential algorithm for this problem: we can iterate through the vertices of G and maintain an independent set S, adding new vertices v to S if v is not adjacent to any of the vertices in S.

In this project, we will implement existing parallel algorithms for maximal independent set. One such algorithm is Luby's algorithm (shown in Algorithm 2 of Shi, Wang and Shang's survey) which maintains an independent set S over the course of several rounds, as well as a set of vertices $U \subset V \setminus S$ of G which are guaranteed to not be adjacent to any vertices in S. In each round, the vertices in U are assigned (in parallel) a random priority, and the vertices in U which have a higher priority than all of their neighbors are added to S — these vertices and their neighbors are then removed from U. It can be shown that $O(\log n)$ rounds suffice with high probability to process all the vertices (here $n = |V|$).

We also plan to implement the algorithms of Blelloch, Fineman and Shun. Blelloch, Fineman and Shun observe that if the vertices are processed according to a random permutation (fixed at the beginning of the algorithm) then the span of a parallel version of the greedy sequential algorithm is $O(\log^2 n)$ with high probability over the initial random permutation. This algorithm also effectively maintains an independent set S over the course of several rounds, and a set $U \subset V \setminus S$ consisting of vertices which are not adjacent to S. In each round, for all the vertices in U whose neighbors in G do not come before them in the initial random permutation, the algorithm will take those vertices and add them to S,

and then remove those vertices and their neighbors from U (and then recurse on U). This algorithm can be shown to require at most $O(\log^2 n)$ rounds with high probability over the initial permutation. The work required by this algorithm is $O(m \log^2 n)$, but Blelloch, Fineman and Shun also propose algorithms with $O(m)$ work.

3 Challenge

Due to the iterative nature of these algorithms, one key challenge in achieving good speedup is reducing the overhead of synchronization. Minimizing communication in the message-passing and data-parallel models is also likely to be nontrivial.

4 Resources

Conceptual resources we are using include the survey of Shi, Wang, Shang mentioned above, as well as the work of Blelloch, Fineman and Shun. We are likely to implement these algorithms in the message-passing and data-parallel models from scratch. For implementations of these algorithms in the shared memory model, we will potentially use this implementation of Luby's Algorithm or the Ligra framework as starter code. We will be making use of the PSC machines and NVIDIA GPUs.

5 Goals and Deliverables

Overall, we hope to determine how to minimize communication and synchronization overhead — we will be showing speedup graphs at the poster session. Our goals are as follows:

5.1 75% Goals

In case the work goes slowly, we would aim to implement Luby's algorithm and Algorithms 2 and 3 in the paper by Blelloch, Fineman and Shun in the shared memory and message passing models (i.e. OpenMP and MPI).

5.2 100% Goals

In addition to those, we hope to implement the algorithms mentioned above in CUDA.

5.3 125% Goals

If we make rapid progress on the above goals, we will empirically evaluate the following heuristic in the message-passing model: in each round, we will simply

sort the vertices by degree and then remove vertices with low degree. This heuristic could potentially have the benefit of low communication, and may result in larger independent sets compared to the algorithms mentioned above which make progress by including vertices with large degrees.

6 Schedule

- During the week of 3/27, we will aim to get a working shared memory implementation of Luby's algorithm, potentially based on existing repositories.
- During the week of 4/3, we will aim to get a working shared memory implementation of the algorithms of Blelloch, Fineman and Shun, and a message passing implementation of Luby's algorithm.
- During the week of 4/10, we will aim to get a working message passing implementation of the remaining algorithms.
- During the week of 4/17, we will implement all of the above algorithms in CUDA.
- During the week of 4/24, we will attempt to implement our proposed heuristic in the shared-memory model.

7 References

- Shi, Wang and Shang's survey `http://web.mit.edu/jeshi/www/public/papers/parallel_MIS_survey.pdf`
- Guy E. Blelloch, Jeremy T. Fineman, Julian Shun. Greedy Sequential Maximal Independent Set and Matching are Parallel on Average. Published in SPAA 2012. `https://arxiv.org/pdf/1202.3205.pdf`