LITERATURE REVIEW: Parallel algorithms for the maximal independent set problem in graphs

Luis Barba

October 16, 2012

1 Introduction

Let $G = (V, E)$ be an undirected graph. An *independent set* of G is a subset I of V such that no two vertex in I are adjacent in G. A *Maximal Independent Set* (MIS) of G is an independent set such that adding any other vertex to it forces the set to contain an edge between two of its vertices. A Maximum independent Set (MaxIS) is a largest maximal independent set contained in V; see Fig. 1. The problem of finding a MaxIS has been proved to be NP-hard. Moreover, it is even hard to find an ε-approximation unless $P = NP$ [EH00]. However, finding a MIS is relatively easy and many (greedy) algorithms are known to solve the problem. The idea to construct an MIS I is usually the following: Given an ordering of the vertices of G, process each vertex by adding it to I as long as it is not adjacent to a vertex already in I. In this way, it is not hard to see that an MIS would be produced. The output of this algorithm is known as the Lexicografical Maximal Independent Set (LFMIS) for the given order. In fact, if the vertices are sorted by degree, from minimum to maximum, the output of the algorithm is a $\frac{1}{\Delta+1}$-approximation of the MaxIS, where Δ is the maximum degree of a vertex in G. That is, the size of the computed approximation MIS has at least $\frac{k}{\Delta+1}$ where k is the size of a MaxIS in G.

In this work we are interested in the study of the MIS and LFMIS problem using parallel algorithms. Finding a LFMIS has been proved hard since it was shown to be a P-complete problem [Coo85]. Therefore, it is very unlikely that there exists an efficient parallel algorithm to solve this problem in its general setting. Furthermore, it is even P-complete to approximate the size of the LFMIS [GHR95]. However, in "average" it is not so hard to parallelize. In other words, for the vast majority of orderings of V, there is an efficient parallel algorithm to find the LFMIS as proved in [BFS12] in 2012. This is certainly surprising and opened the door for new research in a field that has been almost closed for the past 20 years.

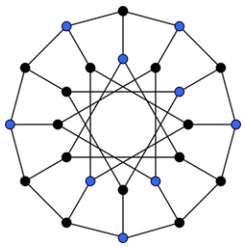

Figure 1: A Maximum Independent Set of the graph is depicted in blue.

The MIS problem is a basic problem in Graph Theory since several problems can be reduced to it. For example, finding small proper colorings or finding maximal matchings in a graph. A *vertex coloring* of a graph $G = (V, E)$ is an assignment of colors to the vertices of G. A vertex coloring is *proper* if no two adjacent vertices are assign with the same color. A proper coloring of G is minimum if G cannot be properly colored with less colors. One important remark is that every chromatic class of a proper coloring is a MIS of G. Therefore, one way to compute a proper coloring with "few" colors is to compute a large MIS, assign the same color to all its vertices and remove it from G afterwards. By repeating this process recursively, we obtain an algorithm to compute a proper coloring of G. However, the quality of the coloring depends heavily on the choice for the MIS algorithm. Therefore, having good and fast algorithms to compute MIS translates into efficient algorithms to compute a proper coloring of G.

Another classical problem in graph theory is the matching problem. A set of edges of a graph G is a *matching* if no two edges in it share an endpoint. The problem of computing a Maximal Matching (MM) can also be solved using an algorithm for MIS. Consider the edge-graph G, that is the graph whose vertex set are the edges of G and where two of these edges are adjacent if they share a common endpoint. Therefore, a MIS of the edge-graph of G corresponds to a MM of G. While this algorithm is straightforward, it requires the computation of the edge-graph of G, which can be expensive. Therefore, direct parallel algorithms are of great interest. In this direction, Blelloch et al. in [BFS12] provided a variation of their techniques to solve the MIS and applied them directly to find a MM.

Other problems such as the vertex covering and maximal clique problems are also closely related to MIS. The consequences and applications of a good algorithm to find a MIS are too many and just a few are mentioned in this work. This remarks, together with the fact that finding a MaxIS is NP-hard and a LFMIS is P-complete, show that the study of this problem is fundamental and that the consequences of any development in this field are of great importance.

2 Literature Review

In the parallel setting, the MIS problem has been extensively studied as well. Vliant [Val82] noted that the MIS problem, which has an easy sequential algorithm, may be one of the problems for which no fast parallel algorithm exists. Cook [Coo85] strengthened this belief by proving that obtaining the LFMIS for any arbitray ordering of V is P-complete. Where P is the class of all problems solvable in polynomial time. In other words, there is no parallel algorithm to find the LFMIS for any given order of V unless $P = NC$. Therefore, to solve the MIS problem, researches needed to look for a different approach to that present in the sequential algorithm.

The study of the MIS problem in the parallel setting was inspired by the growing number of parallel algorithms using the MIS algorithm as a subroutine. Karp and Wigderron [KW85] gave NC^1 reductions from the Maximum Set Packing and the Maximal Matching problems to the MIS problem, and an NC^2 reduction from the 2- Satisfiability problem to the MIS problem. Luby also showed an NC^1 reduction from the Maximal Coloring problem to the MIS problem.

Karp and Wigderson [KW85] surprised the research community by developing a fast parallel algorithm for the MIS problem. They presented a randomized algorithm with expected running time $O(\log^4 n)$ using $O(n^2)$ processors, and a deterministic algorithm with running

time $O(\log^4 n)$ using $O(\frac{n^3}{\log^3 n})$ processors on a EREW PRAM. That is, they established that the MIS problem is in NC^4. Independently, Goldberg and Spencer showed a deterministic parallel algorithm running in $O(\log^4 n)$ time on the EREW PRAM model [GS89]. Later on, Luby [Lub86] and Alon et al. [ABI86], independently proved the existence of a Monte Carlo algorithm in the EREW PRAM running in expected $O(\log^2 n)$ time using $O(|E|)$ processors. Furthermore, Alon et al. showed that in the CRCW PRAM, the MIS problem can be solved in $O(\log n)$ time using $O(|E|\Delta)$ processors. Both papers show a way to derandomized the algorithms at the expense of using $O(mn^2)$ processors, obtaining in this way efficient deterministic algorithms for the MIS problem.

A few years later, Coppersmith [CRT89] studied the problem in the setting of random graphs. He showed that in a random graph with n vertices, the LFMIS can be found in $O(\log^{O(1)} n)$ expected time using linearly many processors in the CRCW PRAM model for any given order. Recall that LFMIS is P-complete for general graphs. However, this is not contradiction and simply shows that in the "average" case, the LFMIS problem is highly parallelizable. Calkin and Frieze further analyzed this algorithm and proved that the expected running time of Coppersmith's algorithm is in fact $O(\log n)$ for random graphs of arbitrary edge density. Nevertheless, these results were only theoretical and couldn't be implemented to solve real-life applications.

The greedy sequential algorithm to compute an MIS loops over the vertices according to the given order, adding them to the resulting set only if no previous neighboring vertex has been added. In this loop, each iterate depends only on a subset of the previous iterates. That is, to decide the fate of a vertex we need only to know if any one of the vertex's previous (in the order) neighbors is in the partially computed MIS. This leads to a dependence structure among the iterates. If this structure is shallow (has a polylogarithmic depth), then running each of the iterates in parallel, while respecting the dependencies, leads to an efficient parallel implementation that mimics the sequential algorithm. The depth of this dependance structure is called the *dependence length*.

Using these abstraction, Blelloch et Al. [BFS12] revisited the ideas of Coppersmith and Calkin et al., and showed that the dependence length of the sequential algorithm for the LFMIS problem has $O(\log^2 n)$ depth with high probability, where the probability is with respect to the random order taken from the input. That is, for almost every ordering the dependance structure will be shallow. Furthermore, this result was followed by a new parallel implementation providing a continuous trade-off between total work and running time. To achieve this result, they fix an ordering of the vertices and instead of processing all of them at once, they process only a prefix of the vertices in parallel. By reducing the size of the prefix, the parallelism is also reduced but the redundant work is reduced. When the prefix is of size one, the algorithm mimics the sequential algorithm with no redundant work. Using this trade-off, they showed that by choosing and appropriate size for the prefix, they can obtain linear work while computing the solution after a polylogarithmic number of steps. Another interesting property of their algorithm is that even while their algorithm is randomized, it will produce the same output whenever the same ordering of the vertices is given as an input. This can be a desirable property for parallel algorithms as shown in [BAAS09].

Given a graph G, the *Set Cover* problem asks for the smallest set of points (called a vertex cover) such that every edge of the graph is incident to an element of the vertex cover. The relation between Set Cover and MIS is tight since a set of vertices is a vertex cover, if and only if its complement is an independent set. An immediate consequence is that a big

MIS produces a small vertex cover.

While a vertex cover can be found in polylogarithmic time using parallel algorithms for MIS, its output may be arbitrarily big (bad) compared with the size of a minimum vertex cover. Therefore, another branch of study is dedicated to compute good approximations for graph problems using parallel algorithms for MIS and related problems.

While the Set Cover problem is NP-hard, good approximation algorithm exist in the sequential setting as shown by Chvatal in [Chv79]. Blelloch et Al. [BPT11] studied the Set Cover problem in the parallel setting and showed the existence of a $(1+\varepsilon)H_n$-approximation parallel algorithm using linear work, where $H_n = \sum_{k=1}^{n} \frac{1}{k}$. To obtain this result they use a set of tools related to MIS's. They introduced the concept of Maximal Nearly Independent Set (MaNIS) and presented an $O(m)$ work and $O(\log^2 m)$-time parallel algorithm for the MaNIS problem in the EREW PRAM model. The MaNIS problem is to find a subset of the power set of the vertices of G such that they are nearly independent (their elements do not overlap too much), and maximal (no set can be added without introducing too much overlap). The MaNIS abstraction generalize de MIS problem and allows them to use the duality to solve several set-cover like problems.

References

[ABI86] Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of Algorithms*, 7(4):567 – 583, 1986.

[BAAS09] Robert L. Bocchino, Jr., Vikram S. Adve, Sarita V. Adve, and Marc Snir. Parallel programming must be deterministic by default. In *Proceedings of the First USENIX conference on Hot topics in parallelism*, HotPar'09, pages 4–4, Berkeley, CA, USA, 2009. USENIX Association.

[BFS12] Guy E. Blelloch, Jeremy T. Fineman, and Julian Shun. Greedy sequential maximal independent set and matching are parallel on average. In *Proceedinbgs of the 24th ACM symposium on Parallelism in algorithms and architectures*, SPAA '12, pages 308–317, New York, NY, USA, 2012. ACM.

[BPT11] Guy E. Blelloch, Richard Peng, and Kanat Tangwongsan. Linear-work greedy parallel approximate set cover and variants. In *Proceedings of the 23rd ACM symposium on Parallelism in algorithms and architectures*, SPAA '11, pages 23–32, New York, NY, USA, 2011. ACM.

[Chv79] V. Chvatal. A greedy heuristic for the set-covering problem. *Math. of Oper. Res.*, 4(3):233–235, 1979.

[Coo85] Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64:2 – 22, 1985. International Conference on Foundations of Computation Theory.

[CRT89] Don Coppersmith, Prabhakar Raghavan, and Martin Tompa. Parallei graph algorithms that are efficient on average. *Information and Computation*, 81(3):318 – 333, 1989.

[EH00] Lars Engebretsen and Jonas Holmerin. Clique is hard to approximate within $n^{1-o(1)}$. In Ugo Montanari, José Rolim, and Emo Welzl, editors, *Automata, Languages and Programming*, volume 1853 of *Lecture Notes in Computer Science*, pages 2–12. Springer Berlin / Heidelberg, 2000.

[GHR95] Raymond Greenlaw, H. James Hoover, and Walter L. Ruzzo. Limits to parallel computation: P-completeness theory, 1995.

[GS89] Mark Goldberg and Thomas Spencer. A new parallel algorithm for the maximal independent set problem, 1989.

[KW85] Richard M. Karp and Avi Wigderson. A fast parallel algorithm for the maximal independent set problem. *J. ACM*, 32(4):762–773, October 1985.

[Lub86] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15(4):1036–1053, 1986.

[Val82] L. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11(2):350–361, 1982.