# Comparing Performance of Parallel Maximal Independent Set Algorithms

**Lance Fletcher**

## Abstract

Many parallel algorithms exist for finding a maximal independent set (MIS) of nodes in a graph. Despite the development of many parallel MIS algorithms, the comparison of these algorithms has not been well investigated. The concept of finding a MIS implies finding a large set of independent nodes is ideal. Given this notion of larger MISs being better, it is beneficial to know which parallel algorithm provides the largest MISs. Many other graph algorithms utilize MISs and could benefit from obtaining larger sets of independent nodes. This work investigates MIS sizes returned by parallel algorithms on a variety of graphs. Additionally, the novel parallel algorithm, *Degree One*, is introduced and shows an increase in MIS size ranging from 2.7% to 8% when compared to the next best MIS algorithm. The algorithm contains a hyperparameter which allows for a trade off between MIS size and the amount of parallel rounds needed.

## 1 Introduction

A *maximum* independent set is the largest independent set of nodes within a graph. The problem of finding a maximum independent set is well studied and is known to be NP-complete [13]. A related problem is finding a *maximal* independent set (MIS) of nodes. A MIS of nodes in a graph is a set where no two nodes are adjacent and a node cannot be added to the set without being adjacent to any node already in the set. A trivial linear time sequential algorithm involves iterating over the set of nodes and adding a node to the independent set if none of its neighbors are already in the independent set. To make use of multiple processors simultaneously many parallel MIS algorithms have been developed [2], [4], [5], [6], [11]. These algorithms all involve the general idea of a parallel *round*. A high level explanation is during each parallel round the independent set is grown by each processor iterating over its delegated nodes. These rounds occur until maximality is achieved. Thus, the number of rounds needed to generate an MIS is common in the analysis of parallel MIS algorithms. How the vertices are chosen to be added to the MIS each round is how the algorithms primarily differ.

## 2 Related Work

The first parallel MIS algorithm was introduced by Karp and Wigderson and executed in O($\log^4 n$) time with O($n$/$\log^3 n$) processors [6]. A deterministic parallel MIS algorithm was developed in [5] with O($n$/$\log^3 n$) run-time with a linear amount of EREW-processors. At around the same time, [2] and [11] both improved upon previous algorithms with each developing an algorithm with expected run-time of O($\log^2 n$) w.h.p. when using O($m$) processors. Additionally, an O($\log^2 n$) algorithm which allows for a trade off between parallelism and total work was proposed in [4].

While run-time analysis is vital to justifying the efficiency of these algorithms, the quality of MIS found by each algorithm should also be evaluated. Some of the algorithms, such a Luby's and Alon's utilize vertex degree's when calculating the the probability that a vertex is added to the MIS [2], [11]. A larger MIS would be of use in a variety of applications and algorithms. There are many other graph problems where obtaining a larger MIS would provide better results such as maximal matching,

graph coloring, vertex cover, and correlation clustering [1],[3],[6],[14]. Therefore, knowing which parallel algorithm generates a larger MIS on average, potentially influences the implementation of other algorithms.

# 3 Preliminaries

## 3.1 Notation

The current section intends to define the notation used throughout the paper. A graph is defined as $G = (V, E)$, where $V$ is the set of vertices or nodes within a graph and $E$ is the set of edges which connect two vertices $(i, j)$ in the graph. In this work all graphs are undirected and unweighted. A node's degree is represented as $d(v)$, and a node's set of neighbors is represented as $\mathcal{N}(v)$.

## 3.2 Parallel MIS Algorithms

For the sake of space, the pseudo-code of each parallel MIS algorithm compared in this work will be forgone; however, the following is a brief description of each algorithm. As mentioned previously, each algorithm involves continually iterating over the set of nodes and adding some to the independent set in parallel rounds.

In Luby's algorithm, each parallel round has two main components [11]. First, each $v \in V'$ is added to a set $X$ with a probability $1/(2d(v))$. Where $V'$ is the set of all nodes who can possibly be placed in the MIS, i.e. nodes which do not have neighbors in the MIS and are not in the MIS. Then, for every pair of nodes $u, v \in X$, if $(u, v) \in E$, remove the node with the smaller degree from $X$. Lastly, add the set $X$ to the independent set and remove all nodes in $X$ from $V'$.

The steps of a parallel round in Alon et al.'s algorithm [2] are very similar to Luby's. Once again, each $v \in V'$ is added to a set $X$, but this time with probability $1/d(v)$. Then, for every pair of nodes $u, v \in X$, if $(u, v) \in E$, remove $u$ from $X$ with probability $d(v)/(d(v) + d(u))$, else remove $v$. Meaning the node with the lower degree is more likely to be added to the MIS. Again, the round concludes by adding the set $X$ to the independent set and removing all nodes in $X$ from $V'$.

Blelloch et al. introduced a simpler algorithm which does not utilize the degree of nodes [4]. First produce a random permutation of the set of nodes in the graph. Then, every round for each $v \in V'$ add $v$ to the MIS if none of $v$'s neighbors occur earlier in the random permutation. Additionally, remove $v$ and its neighbors from $V'$.

# 4 Degree One Algorithm

The proposed *Degree One* algorithm was designed with the intention of finding larger maximal independent sets. Simply adding all the nodes which have a degree of one is a simple way of increasing the size of a MIS.

**Theorem 1.** *Given a graph G, the maximum independent set of nodes in G will include every node with a degree of one, unless the degree one node's neighbor also has a degree of one.*

*Proof.* Given a node $v$ with $d(v) = 1$ and its single neighbor $u$ with $d(u) = x$, where $x > 1$. Adding $v$ to an independent set of nodes $I$ will eliminate the possibility of one node being added to $I$. However, adding $u$ to $I$ will eliminate $x$ nodes from being added to $I$. In the case, where the neighboring nodes $v$ and $u$ both have a degree of one, adding either node to $I$ will eliminate one node from being added to $I$. Therefore, the maximum independent set will always contain every node with a degree of one, and contain a single node from a disconnect component consisting of two nodes. □

Theorem 1 references the maximum independent set of nodes in a graph, but with the premise that larger MISs are better, applying the same principle will result in improved MIS sizes. Algorithm 1 shows the pseudo-code for the Degree One algorithm. Every parallel round contains two main parts. The first part identifies nodes within $V'$ which have a degree of one and adds them to the MIS. These degree one nodes and their neighbors are removed from $V'$. If a node which has a degree of one has a neighbor which is also has a degree of one, then only one of the two nodes is added to the MIS. Similar to obtaining the 2-cores of a graph, this process of removing degree one nodes and

their neighbors is done recursively. Unless $G$ is a tree, eventually there will be no degree one nodes remaining in $V'$. When this occurs, the second part of the algorithm executed by performing a single round based on Luby's algorithm. Algorithm 2 show this process. The main difference between this algorithm and Luby's is when two neighboring nodes are selected to be added to the MIS, the node with the lesser degree is added. This follows the same idea behind adding degree one nodes, i.e. adding nodes with lower degree gives more nodes the opportunity to be included in the MIS. Each part of the algorithm respectively can easily be executed in parallel in shared or distributed memory with minor implementation differences. This algorithm also contains a hyperparameter $r_{max}$ which allows for a limit to be set on the amount of recursive degree one removals performed consecutively. This allows for a trade-off between the size of the MIS, and the number of rounds performed by the algorithm. Along with a comparison of other parallel MIS algorithms, the results show how different $r_{max}$ values effect the MIS sizes returned and the number of rounds needed.

**Algorithm 1** DegreeOneMIS

$I \leftarrow \emptyset$
$G' = (V', E') \leftarrow G = (V, E)$
$r \leftarrow 0$
**while** $V \neq \emptyset$ **do**
  **while** $\exists u \in V'$ where $d(u) = 1$ and $r < r_{max}$ **do**
    $D \leftarrow$ All nodes in $V'$ of degree one
    Remove at random nodes from D which are apart of a disconnected two nodes component
    $I \leftarrow I \cup D$
    $V' \leftarrow V' - (D \cup \mathcal{N}(D)$
  **end while**
  $r \leftarrow r + 1$
  $X \leftarrow$ AlteredLubyRound(V')
  $I \leftarrow I \cup X$
  $V' \leftarrow V' - (X \cup \mathcal{N}(X))$
**end while**
**return** $I$

**Algorithm 2** AlteredLubyRound [11]

$X \leftarrow \emptyset$
**for** all $v \in V'$ **do**
  Randomly choose to add $v$ to $X$ with probability $\frac{1}{2 \cdot d(v)}$
  **if** $d(v) = 0$ **then**
    Add $v$ to $X$
  **end if**
**end for**
**for** all $(v, w) \in E'$ **do**
  **if** $v \in X$ **and** $w \in E'$ **then**
    **if** $d(v) \geq d(w)$ **then** ▷ Main difference compared to the original Luby's algorithm
      $X \leftarrow X - w$
    **end if**
  **end if**
**end for**
**return** $X$

## 5 Approach

For simplicity, the parallel algorithms are implemented as sequential algorithms; however, this has have no effect on the MISs produced. Each parallel round involves iterating over $V$ and $E$. Therefore, every "parallel" round, a single processor will iterate over the sets rather than the work being distributed to many processors. Since a true parallel implementation is not utilized, a run-time analysis is not provided; however, the average number of rounds required by each algorithm is analyzed.

Table 1 shows the various graphs used in the analysis. Most of the graphs are based on social networks; however, the topologies and sizes vary. The Powergrid graph is the most unique of the group. All data collected is based on generating 100 MISs for each graph for every algorithm.

Table 1:

| Name | Description | n | m | Source |
|---|---|---|---|---|
| Enron | Email communication network from Enron | 36692 | 183831 | [9] |
| Powergrid | Western Power grid of the United States | 4941 | 6594 | [7] |
| Citation | Arxiv Physics paper citation network | 34546 | 421578 | [8] |
| Facebook1 | Social circles from Facebook | 4039 | 88234 | [10] |
| Facebook2 | Facebook page-page network | 22470 | 171002 | [12] |

## 6 Results

As seen in figure 1, the Degree One algorithm out performs all other MIS algorithms with respect to MIS size. Among the established algorithms, Alon's generated, on average, larger MISs; however, the increase is small compared to the other two algorithms. In comparison, Blelloch's algorithm was the worst. This is likely due to Blelloch's algorithm not considering a node's degree when adding it to the MIS. When comparing the Alon and Degree One algorithms, Degree One significantly increases the average MIS size on many of the graphs evaluated. The Enron, Powergrid, Citation, Facebook1, and Facebook2 graphs saw an increase in average MIS size of 2.7%, 8%, 7.5%, 4.6%, and 7.3% respectively. These results show the low density and tree like topology of the Powergrid graph, are ideal for the Degree One algorithm.

Despite Degree One utilizing a slightly altered version of Luby's algorithm, it still greatly outperforms Luby's. During a parallel round of Luby's algorithm, if two nodes which share an edge are to be added to the MIS, only the node with the greater degree is added. Logically, this means more neighboring nodes are eliminated from possibly being included in the MIS. Whereas in Degree One, the smaller degree node is added to the MIS; thus, less neighbors are eliminated and the resulting MIS is larger.

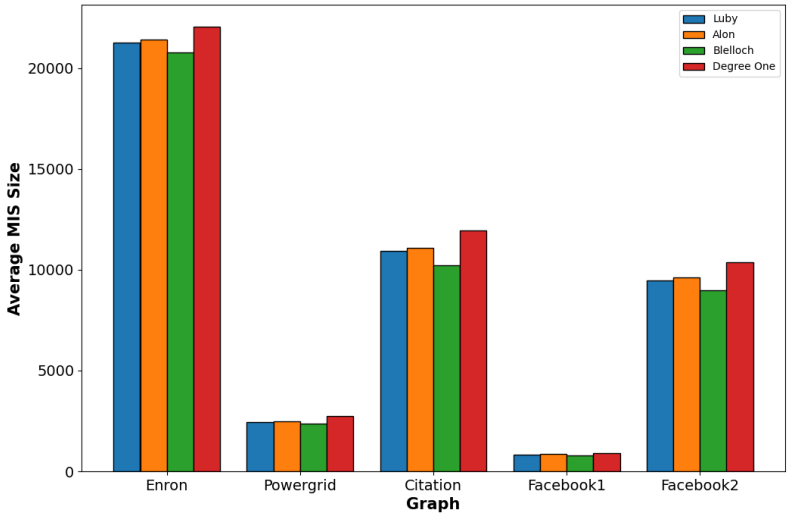


Figure 1: Average MIS size produced by each parallel algorithm on a variety of graphs. For each graph, the average is calculated from 100 MISs generated by each algorithm.

The Degree One algorithm returns larger MISs; however, table 2 shows the average number of parallel rounds needed is significantly greater than the other algorithms. In this analysis, a round is considered

to be each time $V'$ is iterated over in parallel. Therefore, every iteration which searches for degree one nodes to add to the MIS is considered a round. In table 2, the average rounds needed by Degree One is based on the unbounded $r_{max}$ value. Table 3 shows that lower $r_{max}$ values decrease the average number of parallel rounds. An $r_{max}$ value of zero means that degree one nodes were never searched for and only algorithm 2 was utilized to add nodes to the MIS.

Table 4 reveals that smaller $r_{max}$ values produce MIS sizes that are comparable to when $r_{max}$ is None (unbounded). When $r_{max}$ equals zero, the average MIS size is still larger than Luby's, Alon's, and Blelloch's, algorithms. This result indicates that much of the increase in MIS size produced by Degree One can be attributed to the altered Luby's algorithm.

Table 2: Average parallel rounds needed by each algorithm. The $r_{max}$ value for the Degree One algorithm is unbounded.

| | Algorithm | | | |
|---|---|---|---|---|
| Graph | Luby | Alon | Blelloch | Degree One |
| Enron | 11.56 | 6.15 | 5.02 | **56.53** |
| Powergrid | 7.03 | 3.61 | 3.72 | **30.01** |
| Citation | 11.05 | 6.26 | 5.88 | **72.26** |
| Facebook1 | 10.77 | 6.55 | 5.14 | **49.48** |
| Facebook2 | 10.24 | 6.08 | 5.25 | **69.04** |

Table 3: Average parallel rounds needed to find MIS using Degree One algorithm with various $r_{max}$ values. The $r_{max}$ value none is when no limit was set on the number of consecutive degree one rounds.

| | $r_{max}$ value | | | | | | |
|---|---|---|---|---|---|---|---|
| Graph | 0 | 1 | 2 | 3 | 4 | 5 | None |
| Enron | 23.34 | 30.99 | 36.05 | 38.77 | 41.79 | 44.37 | **56.53** |
| Powergrid | 10.76 | 15.21 | 17.28 | 19.07 | 21.46 | 22.14 | **30.01** |
| Citation | 22.0 | 30.85 | 37.51 | 43.97 | 49.7 | 54.36 | **72.26** |
| Facebook1 | 21.76 | 30.92 | 37.41 | 43.39 | 48.14 | 47.5 | **49.48** |
| Facebook2 | 20.46 | 28.69 | 34.47 | 39.71 | 44.03 | 47.47 | **69.04** |

Table 4: Average MIS size returned using Degree One algorithm with varying $r_{max}$ values. The $r_{max}$ value none is when no limit was set on the number of consecutive degree one rounds.

| | $r_{max}$ value | | | | | | |
|---|---|---|---|---|---|---|---|
| Graph | 0 | 1 | 2 | 3 | 4 | 5 | None |
| Enron | 21951 | 21993 | 22013 | 22023 | 22026 | 22035 | **22039** |
| Powergrid | 2671 | 2697 | 2712 | 2717 | 2722 | 2724 | **2727** |
| Citation | 11864 | 11902 | 11930 | 11937 | 11938 | 11939 | **11940** |
| Facebook1 | 908 | **914** | 912 | **914** | 913 | 912 | **914** |
| Facebook2 | 10237 | 10293 | 10320 | 10338 | 10340 | 10343 | **10353** |

## 7 Conclusion

In this work, we presented the novel parallel algorithm *Degree One* which improves the size of MISs returned from a variety of graphs. The $r_{max}$ hyperparameter proved to provide a trade-off between the number of parallel rounds and the size of the MIS calculated. Additionally, it was found that smaller $r_{max}$ values provide slightly smaller MIS sizes, but with less computational cost, when compared to larger $r_{max}$ values. Among the established parallel algorithms, Alon's showed to provide the largest MISs while also requiring few parallel rounds.

Many questions remain to be explored regarding this work. Developing theoretical bounds on the number of parallel rounds needed by Degree One should be further explored. Similar to the polylogarithmic bounds in other MIS algorithms, Degree One could posses a similar property. Furthermore, work should be conducted on developing other parallel algorithms which are capable of generating larger MISs while also being computationally cheap. Lastly, applying these larger MISs to suitable applications such as graph coloring or correlation cluster could provide motivation to further this work.

## References


[1] JR Allwright, R Bordawekar, PD Coddington, K Dincer, and CL Martin. A comparison of parallel graph coloring algorithms. *SCCS-666*, pages 1–19, 1995.

[2] 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.

[3] Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Almost 3-approximate correlation clustering in constant rounds, 2022.

[4] Guy E. Blelloch, Jeremy T. Fineman, and Julian Shun. Greedy sequential maximal independent set and matching are parallel on average. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '12, page 308–317, New York, NY, USA, 2012. Association for Computing Machinery.

[5] Mark Goldberg and Thomas Spencer. A new parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 18(2):419–427, 1989.

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

[7] Jérôme Kunegis. Konect: The koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, page 1343–1350, New York, NY, USA, 2013. Association for Computing Machinery.

[8] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2–es, 2007.

[9] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *CoRR*, abs/0810.1355, 2008.

[10] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[11] M Luby. A simple parallel algorithm for the maximal independent set problem. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, page 1–10, New York, NY, USA, 1985. Association for Computing Machinery.

[12] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding, 2019.

[13] Robert Endre Tarjan and Anthony E. Trojanowski. Finding a maximum independent set. *SIAM Journal on Computing*, 6(3):537–546, 1977.

[14] Nate Veldt. Growing a random maximal independent set produces a 2-approximate vertex cover. *arXiv preprint arXiv:2209.04673*, 2022.