Principles of Distributed Computing 03/21, 2018

Lecture 5

Lecturer: Mohsen Ghaffari *Scribe:*

1 Maximal Independent Set

The Maximal Independent Set (MIS) problem is a central problem in the area of local graph algorithms, in fact arguably the most central one. One partial reason for this central role is that many other problems, including graph coloring and maximal matching, reduce to MIS, as we soon see.

1.1 Definition and Reductions

Let us start with recalling the definition of MIS:

Definition 1. *Given a graph $G = (V, E)$, a set of vertices $S \subseteq V$ is called a* Maximal Independent Set (MIS) *if it satisfies the following two properties:*

(1) the set S is an independent *set meaning that no two vertices $v, u \in S$ are adjacent,*

(2) the set S is maximal *— with regard to independence — meaning that for each node $v \notin S$, there exists a neighbor u of v such that $u \in S$.*

Lemma 2. *Given a distributed algorithm $\mathcal{A}$ that computes an MIS on any N-node graph in $T(N)$ rounds, there is a distributed algorithm $\mathcal{B}$ that computes a $\Delta + 1$ coloring of any n-node graph with maximum degree Δ in $T(n(\Delta + 1))$ rounds.*

In particular, we will soon see an $O(\log n)$ round randomized algorithm for computing an MIS on n-node graphs, which by this lemma, implies an $O(\log n)$ round randomized algorithm for $(\Delta+1)$ coloring.

Proof. Let G be an arbitrary n-node graph with maximum degree Δ, for which we would like to compute a $(\Delta + 1)$-coloring. Let $H = G \times K_{\Delta+1}$, that is, the graph H is an $n(\Delta + 1)$-vertex graph generated by taking $\Delta + 1$ copies of G. Hence each node $v \in G$ has $\Delta + 1$ copies in H, which we will refer to as v_1, v_2, $\ldots$, $v_{\Delta+1}$. Then, add additional edges between all copies of each node $v \in G$, that is, each two copy vertices v_i and v_j are connected in H.

Run the algorithm $\mathcal{A}$ on H. The resulting MIS produces a maximal independent set S. For each node $v \in G$, the color of v will the the number i such that the $v_i \in S$. Clearly, each node receives at most one color, as at most one copy v_i of $v \in G$ can be in S. However, one can see that each node $v \in G$ receives actually exactly one color. The reason is that each neighboring node $u \in G$ can have at most one of its copies in S. Node v has at most Δ neighbors u and $\Delta+1$ copies v_i. Hence, there is at least one copy v_i of v for which no adjacent copy u_i of neighboring vertices $u \in G$ is in the set S. By maximality of S, we must have $v_i \in S$. □

Unfortunately, the best deterministic algorithms for computing an MIS remain slow:

- An $O(\Delta+\log^* n)$-round algorithm follows by computing a $\Delta+1$ coloring and then iterating through the colors and greedily adding vertices to the MIS. In the previous lecture we saw a method for computing a $\Delta + 1$ coloring, in $O(\Delta \log \Delta + \log^* n)$ rounds, but that can be sped up slightly to $O(\Delta + \log^* n)$ rounds.
- A $2^{O(\sqrt{\log n})}$-round algorithm follows from network decompositions of Panconesi and Srinivasan [PS92].

Improving these bounds remains a long-standing open problem. In contrast, there is an extremely simple randomized algorithm that computes an MIS in merely $O(\log n)$ rounds, as we see next.

1.2 Luby's MIS Algorithm

We now present the so called Luby's algorithm — presented essentially simultaneously and independently by Luby [Lub85] and Alon, Babai, and Itai [ABI86] — that computes an MIS in $O(\log n)$ rounds, with high probability.

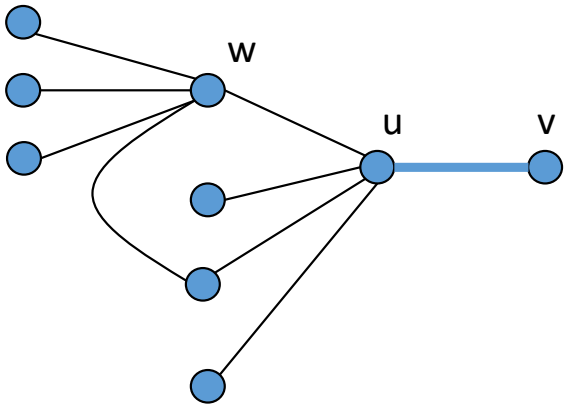

Figure 1: Node w killing edge $e = \{u, v\}$

Luby's Algorithm: The algorithm is made of iterations, each of which has two rounds, as follows:

- In the first round, each node v picks a random real number $r_v \in [0, 1]$ and sends it to its neighbors[1]. Then, node v joins the (eventual) MIS set S if and only if node v has a strict local maxima, that is, if $r_v > r_u$ for all neighbors u of v.
- In the second round, if a node v joined the MIS, then it informs its neighbors and then, node v and all of its neighbors get removed from the problem. That is, they will not participate in the future iterations.

Analysis: It is easy to see that the algorithm always produces an independent set, and eventually, this set is maximal. The main question is, how long does it take for the algorithm to reach a maximal independent set?

Theorem 3. *Luby's Algorithm computes a maximal independent set in $O(\log n)$ rounds, with high probability.*

Proof. Consider an arbitrary iteration i and suppose that the graph induced on the remaining vertices is G_i, which has m_i edges. In the following, we will argue that the graph G_{i+1} which will remain for the next iteration has in expectation at most $m_i/2$ edges. By a repeated application of this, we get that the graph that remains after $4 \log n$ iterations has at most $m_0/2^{4 \log n} \leq 1/n^2$ edges. Hence, by Markov's inequality, the probability that the graph $G_{4 \log n}$ has at least 1 edge is at most $1/n^2$. That is, with high probability, $G_{4 \log n}$ has no edge left, and thus, the algorithm finishes in $4 \log n + 1$ iterations, with high probability.

Given the above outline, what remains is to prove that

$$\mathbb{E}[m_{i+1} \,|\, G_i \text{ is any graph with } m_i \text{ edges}] \leq m_i/2.$$

For that, let us consider an edge $e = \{u, v\}$ and a neighbor w of u, as depicted in Figure 1. If w has the maximum number among $N(w)$, the set of neighbors of w in the remaining graph, then w joins the MIS and hence nodes w and u and thus also the edge $e = \{u, v\}$ get removed. In this case, we say node w killed edge $e = \{u, v\}$. However, unfortunately, there is a possible double counting in that an edge e might be killed by many neighbors w_1, w_2, w_3, etc, and thus we cannot lower bound the number of removed edges easily by counting how many edges are killed in this manner.

To circumvent this, we make a slight adjustment: we say that node w single-handedly kills $e = \{u, v\}$ (from the side of u) if r_w is the maximum random number among those of nodes in $N(w) \cup N(u)$. Notice that this limits the number of double-counting of an edge being killed to 2, meaning that at most one node w might single-handedly kill $e = \{u, v\}$ (from the side of u) and at most one node w' might single-handedly kill $e = \{u, v\}$ (from the side of v). Hence, we can lower bound the number of removed

[1] One can easily see that having real numbers is unnecessary and values with $O(\log n)$-bit precision suffice.

edges by estimating the number of single-handedly killed edges and dividing that by 2. We next make this concrete.

Consider two neighboring nodes w and u. The probability that w has the maximum among $N(w) \cup N(u)$ is at least $\frac{1}{d(w)+d(u)}$. In that case, node w single-handedly kills $d(u)$ edges incident on u (from the side of u). Similarly, the probability that u has the maximum among $N(w) \cup N(u)$ is at least $\frac{1}{d(w)+d(u)}$, and in that case, u single-handedly kills $d(w)$ edges incident on w (from the side of w). Therefore, by linearity of expectation, and given that we are counting each edge killed at most twice (once from each endpoint), we cay say the following: the total expected number of removed edges is at least

$$\mathbb{E}[m_i - m_{i+1}] \geq \sum_{\{w,u\} \in E_i} \big(\frac{d(u)}{d(w)+d(u)} + \frac{d(w)}{d(w)+d(u)}\big)/2 = m_i/2.$$

□

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.

[Lub85] Michael Luby. A simple parallel algorithm for the maximal independent set problem. In *Proc. of the Symp. on Theory of Comp. (STOC)*, pages 1–10, 1985.

[PS92] Alessandro Panconesi and Aravind Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In *Proc. of the Symp. on Theory of Comp. (STOC)*, pages 581–592. ACM, 1992.