Principles of Distributed Computing
Roger Wattenhofer

9 Graph Algorithms

9.1 Independent Set

Definition 9.1 (Independent Set) *We are given a graph $G = (V, E)$. An independent set is a subset of nodes $U \subseteq V$, where no two nodes in U are adjacent. An independent set is maximal if no node can be added without violating independence. An independent set of maximum cardinality is called maximum.*

Remarks:

- In this section we concentrate on the maximal independent set (MIS) problem.
- Computing a MIS sequentially is easy. Scan the nodes in arbitrary order. If a node u does not violate independence, add u to the MIS. If the node u does violate independence, discard u.
- Computing a *maximum* independent set is hard. It is the same problem as maximum clique on the complementary graph. Both problems are NP-hard, and in fact not approximable within $n^{1/2-\epsilon}$.

Algorithm 9.2 (Slow MIS) *Nodes have unique identifiers. A node with identifier u must join the MIS if all neighbors with larger identifiers have decided not to join.*

Remark:

- Not surprisingly the slow algorithm is not better than the sequential algorithm in the worst case, because there might be one single point of activity at any time. Specifically:

Theorem 9.3 (Analysis) *Algorithm 9.2 features time complexity $O(n)$ and message complexity $O(|E|)$.*

Remarks:

- There is a relation between independent sets and node coloring (Chapter 1), since each color class is an independent set, however not a MIS. Starting with a good coloring, one can easily derive a MIS algorithm by first choosing all nodes of color 1, and then (for each color in parallel since there is no conflict) adding as many nodes as possible. Thus the following corollary holds:

Corollary 9.4 *Given a coloring algorithm that needs C colors and runs in time T, we can construct a MIS in time $C + T$.*

Remarks:

- Using Theorem 1.23 and Corollary 1.24 we get a distributed deterministic MIS algorithm for trees and for bounded degree graphs with time complexity $O(\log^* n)$.

- With a lower bound argument one can show that the deterministic MIS algorithm for rings is asymptotically optimal.
- Using the Remark of Corollary 1.24 we get a distributed deterministic MIS algorithm for arbitrary graphs with time complexity $O(\Delta \cdot \log n)$. Is there a faster (if necessary: randomized) algorithm? Yes! See below. (Note that this algorithm is not identical with the algorithm in Peleg's book.)

Algorithm 9.5 (Fast MIS) *The algorithm operates in synchronous rounds, grouped into phases. A single phase is as follows:*

1) *Node v marks itself with probability $\frac{1}{2d(v)}$, where $d(v)$ is the current degree of v.*
2) *If no higher degree neighbor of v is also marked, node v joins the MIS. If a higher degree neighbor of v is marked, node v unmarks itself again. (If the neighbors have the same degree, ties are broken arbitrarily, e.g. by identifier).*
3) *Delete all nodes that joined the MIS and their neighbors (that cannot anymore join the MIS).*

Lemma 9.6 (Joining MIS) *A node v joins the MIS in round 2 with probability $\frac{1}{4d(v)}$ or more.*

Proof. Let M be the set of marked nodes in round 1. Let $H(v)$ be the set of neighbors of v with higher degree, or same degree and higher identifier. We have

$$\begin{aligned} Pr[v \notin MIS | v \in M] &= Pr[\exists w \in H(v), w \in M | v \in M] \\ &\leq \sum_{w \in H(v)} Pr[w \in M] = \sum_{w \in H(v)} \frac{1}{2d(w)} \\ &\leq \sum_{w \in H(v)} \frac{1}{2d(v)} \text{ (because } d(v) \leq d(w)\text{)} \\ &\leq \frac{d(v)}{2d(v)} = \frac{1}{2} \end{aligned}$$

Then,

$$Pr[v \in MIS] = Pr[v \in MIS | v \in M] \cdot Pr[v \in M] \geq \frac{1}{2}\frac{1}{2d(v)}.$$

□

Lemma 9.7 (Good Nodes) *We call a node v good if $\sum_{w \in N(v)} \frac{1}{2d(w)} \geq 1/6$. A good node will be removed in round 3 with probability at least $1/36$.*

Proof. Let node v be good. Intuitively, good nodes have lots of low-degree neighbors, thus chances are high that one of them goes into the independent set, in which case node v will be removed in round 3.

Case 1: There is a neighbor $w \in N(v)$ with degree at most 2. With Lemma 9.6 the probability that node w joins the MIS is at least 1/8.

Case 2: All neighbors have at least degree 3. For any neighbor w of v we have $1/2d(w) \leq 1/6$. Since $\sum_{w \in N(v)} \frac{1}{2d(w)} \geq 1/6$ there is a subset of neighbors $X \subseteq N(v)$ such that

$$\frac{1}{6} \leq \sum_{w \in X} \frac{1}{2d(w)} \leq \frac{1}{3}.$$

We can now bound the probability that node v will be removed. Again, if a neighbor of v joins the MIS in round 2, node v will be removed in round 3, we have:

$$\begin{aligned} Pr[v \text{ will be removed}] &\geq Pr[\exists u \in X, u \in MIS] \\ &\geq \sum_{u \in X} Pr[u \in MIS] - \sum_{u,w \in X, u \neq w} Pr[u \in MIS \text{ and } w \in MIS] \end{aligned}$$

For the last inequality we used the inclusion-exclusion principle truncated after the second order terms. Let M again be the set of marked nodes after round 1. Then,

$$\begin{aligned} Pr[v \text{ will be removed}] &\geq \sum_{u \in X} Pr[u \in MIS] - \sum_{u,w \in X, u \neq w} Pr[u \in M \text{ and } w \in M] \\ &\geq \sum_{u \in X} Pr[u \in MIS] - \sum_{u \in X} \sum_{w \in X} Pr[u \in M] Pr[w \in M] \\ &\geq \sum_{u \in X} \frac{1}{4d(u)} - \sum_{u \in X} \sum_{w \in X} \frac{1}{2d(u)} \frac{1}{2d(w)} \\ &\geq \sum_{u \in X} \frac{1}{2d(u)} \left(\frac{1}{2} - \sum_{w \in X} \frac{1}{2d(w)} \right) \geq \frac{1}{6} \left(\frac{1}{2} - \frac{1}{3} \right) = \frac{1}{36}. \end{aligned}$$

□

Remark:

- It would be nice if many nodes are good in each phase. Unfortunately this is not the case: In a star-graph, for example, only the center node is good.

Lemma 9.8 (Good Edges) *An edge is good if one of its endpoints is good; else the edge is bad. At any time at least half of the edges are good.*

Proof. Direct each edge towards the higher degree node (if both nodes have the same degree direct it towards the higher identifier). We need a little helper Lemma before we can continue with the proof.

Lemma 9.9 *A node is bad if it is not good. A bad node has outdegree at least twice its indegree.*

Proof. Assume for the sake of contradiction, that a bad node v does not have outdegree at least twice its indegree. In other words, at least one third of the neighbor nodes (set S) have degree at most $d(v)$. But then

$$\sum_{w \in N(v)} \frac{1}{2d(w)} \geq \sum_{w \in S} \frac{1}{2d(w)} \geq \sum_{w \in S} \frac{1}{2d(v)} \geq \frac{d(v)}{3} \frac{1}{2d(v)} = 1/6$$

which is the definition for v being good – a contradiction. ⊓

With Lemma 9.9 the number of edges directed into bad nodes is at most half the number of edges directed out of bad nodes. Thus, the number of edges directed into bad nodes is at most half the number of edges. Thus, at least half of the edges are directed into good nodes. Since these edges are not bad, they must be good. ⊓

Theorem 9.10 (Analysis) *Algorithm 9.5 terminates in expected $O(\log n)$ time.*

Proof. With Lemma 9.7 a good node (and therefore a good edge!) will be deleted with constant probability. Since half of the edges are good (Lemma 9.8) a constant number of edges will be deleted in each phase. After $O(\log |E|)$ phases $= O(\log n)$ rounds, all edges are deleted. ⊓

Remark:

- This algorithm/proof is a mixture of an algorithm by Luby, an analysis method stolen from Israeli and Itai (developed for the related matching problem), and everything put together by my wife. Surprisingly there have been half a dozen papers published after the original work (1986), with worse running times. In 2002, for example, there was a paper at PODC with linear running time! In the PODC paper the authors write that they improve on a DISTCOMP 1994 journal paper that does matching in $O(n^3)$ time, for trees only...

9.2 Matching

Definition 9.11 (Matching) *We are given a graph $G = (V, E)$. A matching is a subset of edges $M \subseteq E$, such that no two edges in M are adjacent (or, alternatively: where no node is adjacent to two edges in the matching). A matching is maximal if no edge can be added without violating the constraints. A matching of maximum cardinality is called maximum (or minimum maximal). A matching is perfect if each node is adjacent to an edge in the matching.*

Remarks:

- A maximum matching can be found in polynomial time (Blossom algorithm by Jack Edmonds), and is also easy to approximate (in fact, already any maximal matching is a 2-approximation).
- An independent set algorithm is also a matching algorithm: Let graph $G = (V, E)$ be the graph for which we want to construct the matching. Graph G' is as follows: for every edge in G there is a node in G'; two nodes in G' are connected if the respective edges in G are adjacent. A (maximal) independent set in G' is a (maximal) matching in G, and vice versa.
- The Algorithm 9.5 directly gives a $O(\log n)$ algorithm for maximal matching.