Distributed Algorithms

Algorithms that were designed to run on many processors "distributed" a large geographical area.

Many different kinds of DAs:

- Interprocess Communication method: accessing shared memory, point-to-point or broadcast messages, or remote procedure calls.
- timing model: synchronous or asynchronous models.
- failure models: reliable or faulty behavior; Byzantine failures (failed processor can behave arbitrarily).
- Various problems: resource allocation, communication, consensus, concurrency control, deadlock detection ..

Local Algorithms

A distributed algorithm is a localized algorithm if it uses only the information of all the local nodes plus the information of a small (preferably a constant) number of other nodes.

Maximal Independent Set Problem

Given an arbitrary topology network we want to find a maximal independent set of nodes.

A set of nodes is called an *independent set* if it contains no pair of neighboring nodes, and it is maximal if it cannot be increased to form a larger independent set by the addition of other nodes.

Applications: resource allocation, dominating sets, clusterhead selection ..

Given a arbitrary network $G = (V, E)$, we want to design a distributed (and local) algorithm which will output a MIS set as follows: every node will know whether it is in the MIS or not.

Let $\Gamma(v)$ be the set of vertices in V that are adjacent to v.

Basic idea: The algorithm proceeds in rounds; in every round find an independent set S, add S to I (initially I is empty) and delete $S \cup \Gamma(S)$ from the graph.

Algorithm

1. $I = \phi$
2. **repeat**
   - 2.1 for all $v \in V$ do
     - if $d(v) = 0$ then
     - add v to I and delete v from V
     - else mark v with probability $1/(2d(v))$
   - 2.2 for all $(u, v) \in E$ do
     - if both u and v are marked then
     - unmark the lower degree vertex
   - 2.3 for all $v \in V$ do
     - if v is marked then add v to S.
   - 2.4 $I = I \cup S$
   - 2.5 delete $S \cup \Gamma(S)$ from V,
   - and all incident edges from E
3. **until** $V = \phi$

Analysis

We show that the expected fraction of edges removed from E during each iteration is bounded from below by a constant.

A vertex $v \in V$ is *good* if it has at least $d(v)/3$ neighbors of degree no more than $d(v)$; otherwise the vertex is bad. An edge is good if at least one of its endpoints is a good vertex, and it is bad if both endpoints are bad vertices.

We show that a good vertex is likely to have one of its lower degree neighbors in S and thereby deleted from V.

Lemma 1. *Let $v \in V$ be a good vertex with degree $d(v) > 0$. Then, the probability that some vertex $w \in \Gamma(v)$ gets marked is at least $1 - e^{-1/6}$.*

Proof. Each vertex $w \in \Gamma(v)$ is marked independently with probability $1/(2d(w))$. The probability that none of the neighbors of v gets marked is at most

$$\left(1 - \frac{1}{2d(v)}\right)^{d(v)/3} \leq e^{-1/6}$$

□

Lemma 2. *During any round, if a vertex w is marked then it is selected to be in S with probability at least $1/2$.*

Proof. The only reason a marked vertex w becomes unmarked is that one of its neighbors of degree at least $d(w)$ is also marked. The probability of this happening is at most

$$\sum_{x \in \Gamma(w)} \frac{1}{2d(w)} \leq 1/2$$

□

Lemma 3. *The probability that a good vertex belongs to $S \cup \Gamma(S)$ is at least $(1 - e^{-1/6})/2$.*

Lemma 4. *In a graph $G = (V, E)$ the number of good edges is at least $|E|/2$.*

Proof. Direct the edges in E from the lower degree end-point to the higher degree end-point breaking ties arbitrarily. Let $d_i(v)$ and $d_o(v)$ be the in-degree and out-degree of v.

For each bad vertex v,

$$d_o(v) - d_i(v) \geq d(v)/3 = \frac{d_o(v) + d_i(v)}{3}$$

Let $E(S, T)$ be the set of edges directed from vertices in S to vertices in T; and $e(S, T) = |E(S, T)|$.

The total degree of the bad vertices is given by

$$2e(V_B, V_B) + e(V_B, V_G) + e(V_G, V_B)$$

$$= \sum_{v \in V_B} (d_o(v) + d_i(v))$$

$$\leq 3 \sum_{v \in V_B} (d_o(v) - d_i(v))$$

$$= 3 \sum_{v \in V_G} (d_i(v) - d_o(v))$$

$$= 3[(e(V_B, V_G) + e(V_G, V_G)) - (e(V_G, V_B) + e(V_G, V_G))]$$

$$= 3[e(V_B, V_G) - e(V_G, V_B)]$$

$$\leq 3[e(V_B, V_G) + e(V_G, V_B)]$$

□