An efficient distributed algorithm for constructing small dominating sets

Introduction

We would like to find small *dominating set* for given graph, typically graph of connections in distributed system. A *dominating set* in general graph is set of marked vertices S such that every other vertex has neighbor in S (equivalently $N(S) = V$). Trivially whole set V is dominating. Generally finding minimal dominating set is NP-complete problem. The algorithm presented in [1] approximates optimal solution and works in expected slightly-over-linear time.

We work under assumption that nodes are connected with two-way channels and whole system is synchronized.

Algorithm

Algorithm proceeds in iteration, every one tries to augment the existing dominating set until every node is *covered*, i.e. is chosen or have neighbor in chosen set. Every iteration consists of few phases:

1. **Span calculation** Every node v computes $d(v)$ that is, number of uncovered nodes in its neighborhood (including itself). We are interested in $\overline{d(v)} = \lfloor \log_b d(v) \rfloor$, where $b > 1$ is hiperparameter of algorithm.

2. **Candidate selection** Node declares itself as candidate if $\overline{d(v)} \geq \overline{d(w)}$ for every node $w \in N^2(v)$.

3. **Support calculation** Every uncovered node v computes its support $s(v)$, that is, by how many candidates it is covered.

4. **Selection** Every candidate adds itself to the dominating set with probability $1/m$, where m is median of $s(w)$ for w in neighborhood.

Authors named it *Local Randomized Greedy* algorithm, `LRG` for short. Every phase can be calculated in one to two rounds. The hiperparameter b determines the trade-off between time complexity and quality of approximation - larger b means more phases but better approximation.

Correctness and time complexity

It can be easily seen that when uncovered node exits, then some node will be chosen as candidate and every candidate at the end have strictly positive propability to be added to dominating set, thus covering at least one previously uncovered node. It can be shown that algorithm finds a solution in $O(\log n \log \Delta)$ rounds and achieves approximation ratio of $O(\log \Delta)$ (where Δ is maximum degree in the graph) in comparison with optimal solution (both expectations are met with high probability). Calculations however are quite involved, so instead of them I am going to explain some intuition behind them. First of all, how b affect the output of the algorithm. It can be seen that large b implies more candidates (as $\lfloor \log_b d(v) \rfloor$ will assume less values, and there will be many more draws), which later causes big support for uncovered nodes and thus small probability

of selecting candidate. Nevertheless, on average, every uncovered node has average (or median) support, so there is more or less constant probability that it will be covered in every iteration, thus logarithmic number of rounds.

References

[1] Jia, L., Rajaraman, R. and Suel, T. *An efficient distributed algorithm for constructing small dominating sets*. Distrib Comput **15**, 193–205 (2002). https://doi.org/10.1007/s00446-002-0078-0.