Analysis of a Greedy Heuristic for Finding Small Dominating Sets in Graphs

Abhay K. Parekh

Laboratory For Information and Decision Systems, M.I.T., Cambridge, MA 02139, USA

Abstract

We analyze a simple greedy algorithm for finding small dominating sets in undirected graphs of N nodes and M edges. We show that $d_g \leq N + 1 - \sqrt{2M+1}$, where d_g is the cardinality of the dominating set returned by the algorithm.

Keywords: approximate algorithms, analysis of algorithms.

Introduction

Let $G(V, E)$ be an undirected graph with N nodes and M edges. A dominating set of G is a set of nodes such that every node not in the set is adjacent to at least one node in the set. A dominating set of smallest cardinality is known as a minimum dominating set. The problem of finding a minimum dominating set is combinatorially hard (its decision version is NP-complete [2]), so as a practical matter, one has to settle for approximate, but fast algorithms. In this note, we analyze the performance of a simple, greedy algorithm of this type. Let d_g be the size of the dominating set returned by the greedy algorithm, and let d_o be the cardinality of a minimum dominating set. We show that an upper bound on d_o due to Vizing [5], applies to d_g as well:

$$d_g \leq N + 1 - \sqrt{2M+1}.$$

Previous Work

The greedy algorithm considered here is an analog of one that has been analyzed by Chvatal [1] and others [3], [4] for finding small set covers. The focus there has been on comparing the cardinality of the set cover returned by the algorithm to that of the smallest set cover, in the worst case. Since any dominating set problem can be formulated as a set covering problem, the results for the set covering algorithm can be

This work was supported by a Vinton Hayes Fellowship.

specialized to our problem. A result almost directly obtained from the work of Chvatal [1] is that

$$\frac{d_g}{d_o} \leq \sum_{i=1}^{(} \delta + 1)\frac{1}{i},$$

where δ is the maximum degree of a node in the graph. The result of this paper describes the performance of the algorithm in terms of the dimensions of the graph, and should be viewed as complementary to the above result.

Greedy: The Approximation Algorithm

Let $V = \{1, ..., N\}$, and define $D = \phi$. *Greedy* add a new node to D in each iteration, until D forms a dominating set. A node, j, is said to be "covered" if $j \in D$ or if any neighbor of j is in D. A node that is not covered is said to be uncovered. (Since $D = \phi$ at the beginning of the algorithm, it follows that all the nodes are initially uncovered.) In each iteration, put into D the least indexed node that covers the maximum number of uncovered nodes. Stop when all the nodes are covered. (Selecting the least numbered element is just a way of breaking ties.) An example of *Greedy* at work is provided in Figure 1.

Worst Case Performance in terms of N and M

Here we show that an upper bound on d_o due to Vizing [5] is met by d_g as well:

Theorem 1. *For an undirected graph, G, with N nodes and M edges:*

$$d_g \leq N + 1 - \sqrt{2M + 1}. \tag{1}$$

Proof: First convert G into a directed graph by replacing every edge (i,j) by 2 directed edges (i,j) and (j,i), and by adding self-loops (i,i) at every node i. Interpret a directed edge (i,j) to mean that the uncovered node j would be covered if i were included in the dominating set. Define the outdegree of a node as the number of edges emanating from it. We can interpret *Greedy* on the constructed directed graph as follows: Include the least indexed node with the greatest outdegree in the dominating set; delete all edges coming into the neighborhood of that node; and if there are any edges left, include another node in the dominating set, else stop. To see that this is identical to *Greedy*, interpret a directed edge (i,j) to mean that if i were included in the dominating set, the previously uncovered node, j, would be covered by i (see figure 2).

Suppose *Greedy* picks node v_i in the i^{th} iteration. Let $S(i)$ be the set of previously uncovered nodes which were covered by v_i, and let $|S(i)| = m_i$. Finally, define E_i to be the number of edges left at the end of the i^{th} iteration *coming from uncovered nodes* i.e. E_i does not include edges from covered nodes. Set $E_0 = 2M + N$. Our strategy in the proof is to lower bound $E_i - E_{i-1}$ in order to estimate the the maximum number of iterations d_g until there are no edges left, i.e. $E_{d_g} = 0$.

First we show that at most m_i^2 edges *from previously uncovered nodes* are deleted in the i^{th} iteration. Consider some $j \in S(i)$. The outdegree of j can be at most m_i just before the i^{th} iteration. Now notice that no edges into an uncovered node can be deleted before the node is covered. Since j is uncovered before the i^{th} iteration, for each edge coming into j from a uncovered node, there is also an edge going out of j to that uncovered node. Thus there can be at most m_i edges coming into j from uncovered nodes, and the total number of edges running from previously uncovered nodes to members in $S(i)$ is at most m_i^2. There may also be edges from previously covered nodes to members of $S(i)$, but we need not consider them, since we are counting (in E_i) only edges from uncovered nodes.

Next, we estimate the number of edges from $S(i)$ to uncovered nodes which are not in $S(i)$. These edges are not deleted by *Greedy*, but they are not counted in the definition of E_i either. It is clear that the outdegree of every node in $S(i)$ must be $\leq m_i - 1$, since the self loops of all such nodes will have been deleted in the i^{th} step. Thus the number of outgoing edges from $S(i)$, after iteration i is $\leq m_i(m_i - 1)$. However, in the first iteration, we can tighten this bound slightly. We know that $v_1 \in S(1)$, and so *all* edges entering and leaving v_1 will be deleted after the first iteration. Further, the other nodes in $S(1)$ can have outdegree of at most m_2 at the end of the first iteration. Thus, the number of edges (at the end of the first iteration) from $S(1)$ to uncovered nodes not in $S(1)$ is at most $(m_1 - 1)m_2$. Finally, note that no edges remain at the end of the last, i.e. d_g^{th} iteration, and so for this iteration the bound can be set to 0.

By definition of E_i, we conclude that:

$$E_i \geq E_{i-1} - m_i^2 - m_i(m_i - 1)$$

$$E_1 \geq E_o - m_1^2 - (m_1 - 1)m_2.$$

$E_{d_g} = 0$ by definition of d_g. Thus,

$$0 = E_{d_g} \geq -\left(\sum_{i=1}^{d_g} m_i^2 + (m_1 - 1)m_2 + \sum_{i=2}^{d_g - 1} m_i(m_i - 1)\right) + E_0.$$

Solving for E_0:

$$E_0 \leq \sum_{i=1}^{d_g} m_i^2 + (m_1 - 1)m_2 + \sum_{i=2}^{d_g - 1} m_i(m_i - 1). \tag{2}$$

Now notice that $\sum_{i=1}^{d_g} m_i = N$, and that $m_i \geq 1$. E_0 can be upper bounded by maximizing the RHS of (2) with respect to the m_i's subject to the constraints just mentioned. We claim that this maximum occurs when

$$m_1 = N - d_g + 1, m_2 = m_3 = ...m_{d_g} = 1.$$

This is easily seen to be true by contradiction. Suppose the maximum is achieved so that the highest order m_i which is greater than 1 is not m_1, but say $m_j, j > 1$. Now reduce m_j to 1 and add $m_j - 1$ to m_1 (we can do this because none of the constraints are violated), and the difference in the RHS is seen to be positive. This contradicts the assumption.

Substituting the maximum values in the RHS of (2) we have:

$$E_0 = 2M + N \leq (N - d_g + 1)^2 + d_g - 1 + (N - d_g),$$

$$\Rightarrow d_g \leq N + 1 - \sqrt{2M + 1}.$$

Done

The bound of Theorem 1 is met exactly for graphs of the type shown in Figure 3.

Acknowledgment

I am grateful to Professor Robert Gallager for his help in the performing and writing of this work.

References

[1] V. Chvatal, A Greedy Heuristic for the Set Covering problem, *Mathematics of Operations Research*, **4**, (1979) 233-235.

[2] M. R. Garey, D. S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness*, (W. H. Freeman, San Francisco CA, 1979.)

[3] D. S. Hochbaum, Approximation Algorithms for the Set Covering and Vertex Covering Problems, *SIAM Journal on Computing*, **11**, (1982) 555-556.

[4] D. S. Johnson, Approximate Algorithms for Combinatorial Problems, *Journal of Computer System Science*, **9**, (1974) 256-278.

[5] V. G. Vizing, A Bound on the External Stability Number of a Graph, *Doklady A. N.*, **164**, pp. 729-731.

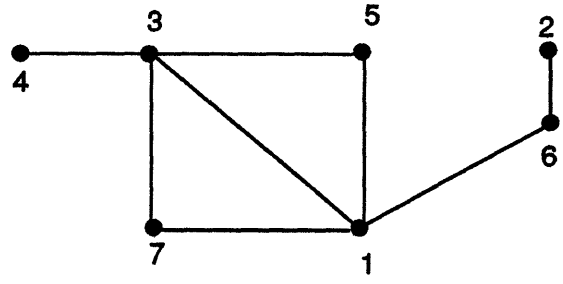

Figure 1. Greedy will return the set $\{1, 2, 3\}$ (note that $\{2, 3\}$ is a minimum cardinality dominating set).

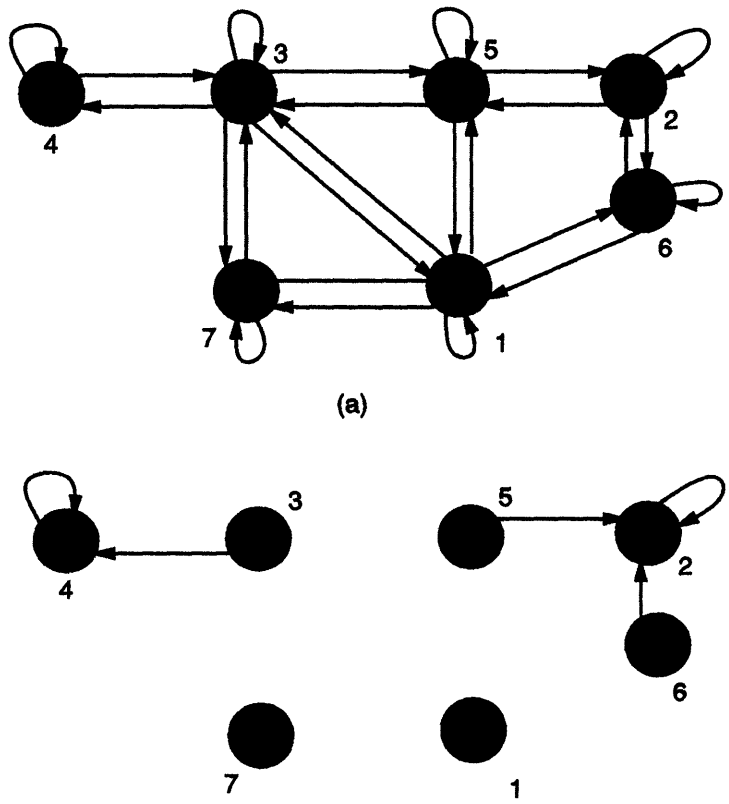

Figure 2. (a) The graph of Figure 1 converted to a directed graph. (b) The graph at the end of the first iteration of *Greedy*. Note that $E_1 = 2$.

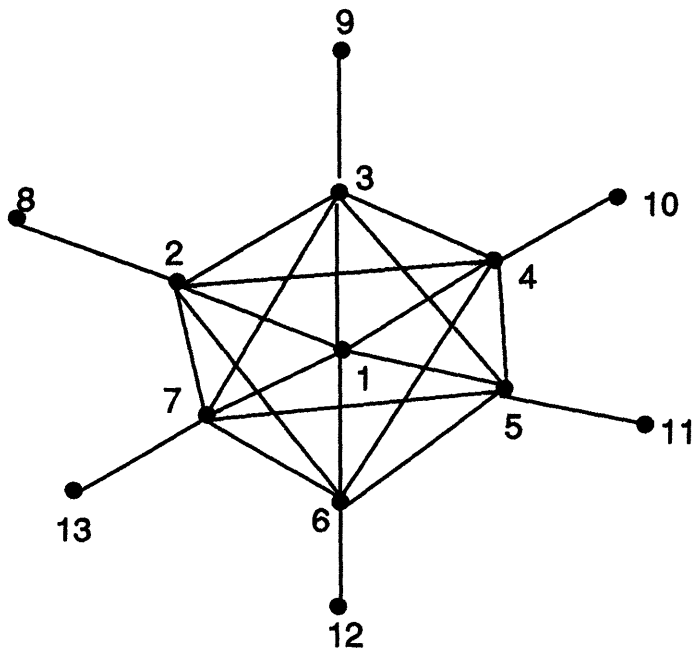

Figure 3. Greedy will return the set $\{1, 2, ..., 7\}$. Since $N = 13$, $M = 24$, we have $7 = d_g \leq 14 - \sqrt{49} = 7$. Notice that $d_o = 6$, for example, $\{2, 3, ..., 7\}$