Camouflaging Independent Sets in Quasi-Random Graphs

Mark Brockington [*] Joseph C. Culberson [†]

April 18, 1994

Abstract

In this paper, we look at the problem of how one might try to hide a large independent set in a graph in which all other independent sets are significantly smaller.

We observe that the most common methods of generating graphs with known maximal independent sets are subject to attack using quite simple techniques that are successful significantly often for graphs of practical size. We present an improved random graph generation system, DELTA, which increases the range of maximal independent sets that can be hidden from these techniques.

1 Introduction

The maximum independent set problem is, given a graph $G = (V, E)$, find the largest independent set $I \subseteq V$. This problem is known to be NP-hard [3]. Kučera [4] analyzes the independent set problem as the basis of a cryptographic system. He proves it is possible to hide an independent set in a graph such that, *asymptotically*, certain common heuristic approaches have an exponentially small probability of finding a set of the hidden size. However, even for $n = 1000$ these sets can be found with high probability (at least higher than one would want for cryptographic systems). In this paper we will describe improved hiding methods, which attempt to hide larger independent sets and then describe new algorithms that take advantage of the hiding methods.

Kučera's graph generation algorithm selects a set $I \subseteq V$ of s vertices and for each pair of vertices x, y, if $x, y \in I$ then no edge is assigned, otherwise $\{x, y\}$ is an edge with (independent) probability p. (His GGES algorithm requires public and private random bit sequences to meet cryptographic constraints, but this is not relevant to this paper.) The

[*]Supported by Natural Sciences and Engineering Research Council Postgraduate Scholarship
email:brock@cs.ualberta.ca

[†]Supported by Natural Sciences and Engineering Research Council Grant No. OGP8053. Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2H1. email:joe@cs.ualberta.ca

probability p is such that the probability of finding a set of $s-1$ or more vertices in a random graph is small. We call this the *naive set creation* (NSC) algorithm.

A simple class of attack algorithms, designated $\mathrm{SM}^i, i = 0, 1, 2 \ldots$ which use a basic minimum degree heuristic, have a high success rate on graphs of approximately 1000 vertices. We briefly outline the reason this attack succeeds in Section 2.

Based on this reasoning, we describe a succession of graph classes that thwart the SM^i approach by adjusting the degree sequence. But in doing this, we create new properties which can be utilized to identify the independent set. We exploit these properties using a class of algorithms designated SP^i.

We then find a subclass of graphs which are optimized in the sense that they are resistant to both lines of attack, and hide larger sets than can be hidden by NSC. We believe this class should prove interesting for a wide variety of algorithms.

2 The Basic Attack

The basic algorithm, SM^0 is standard [4] and is presented in Figure 1. $N(v)$ indicates the neighbors of the vertex v in G. x is the candidate of minimum degree in the graph G' to be added to the independent set. This algorithm, SM^0, is not effective for 1000 node graphs for finding small sets hidden by the NSC algorithm, although it is effective for sets of size approximately 30 or larger. SM^0 can be implemented to run in $O(n^2)$ time.

INPUT: $G = (V, E)$
OUTPUT: A maximal independent set I
    $Z = V$
    $I = \emptyset$
    while $Z \neq \emptyset$ do
        $G' =$ subgraph induced by Z.
        $r =$ minimum $\{deg_{G'}(v)\, v \in Z\}$
        Select (randomly) $x \in G'$ with $r = deg_{G'}(x)$
        $I = I \cup \{x\}$
        $Z = Z - \{x\} - N(x)$
    end
    return I

Figure 1: Algorithm SM^0

SM^i (see Figure 2) for $i = 1, 2, \ldots$ are algorithms in which every combination of i vertices (which form independent sets) are generated and extended using SM^0. Kučera [4] claims that for fixed i even these will not succeed asymptotically. (Here $N(I')$ means the union of the neighbors of the vertices in I'.) SM^i runs in $O(n^{i+2})$ time for fixed i.

INPUT: $G = (V, E)$
OUTPUT: A maximal independent set I
    $I = \emptyset$
    $\forall I' \subseteq V$, $|I'| = i$
        If I' is an independent set then
            Compute the induced subgraph G' with vertex set
                $V' = V - I' - N(I')$
            $I' = \max\{I, I' \cup \mathrm{SM}^0(G')\}$
    end
    return I

Figure 2: Algorithm SM^i

For $n = 1000$ and using the minimum probability p allowed by the criteria set by [4], SM^2 found the independent set in 14% of fifty trials. As p goes to $\frac{1}{2}$, the success frequency reported by SM^2 increases rapidly to 100%.

To discuss better hiding techniques, we will first discuss why SM^i works. Let us define $s = |I|$, and let S_i of size i be a (random) subset of I. Let $I_i = I - S_i$ and thus $I_0 = I$. The vertices which are independent of S_i are $V_i = V - S_i - N(S_i)$, where $N(S_i)$ is the union of the neighborhoods of vertices in S_i, that is $N(S_i) = \bigcup_{v \in S_i} N(v)$.

We let $\mu(I_i)$ be the average degree of a vertex in I_i in the subgraph induced by V_i, and $\mu(\bar{I}_i)$ the average degree of a vertex in $V_i - I_i$. Similarly, we use σ as the standard deviation. We say that I_i *stands out* if the difference $d_i = \mu(\bar{I}_i) - \mu(I_i) \geq \sigma(\bar{I}_i)$. The idea is that if I_i stands out, selecting a vertex on the basis of degree has a reasonably high chance of choosing a vertex from I_i. This means we have a reasonable chance of finding the independent set I once we have found a set S_i. If I_i stands out, SM^i has a reasonable chance of finding I because it will consider every subset of size i of I.

Note that we have implicitly defined what we mean by *reasonable chance*. For security purposes, we would want to be able to hide so that a search had a far less than reasonable chance, while for a practical algorithm we might wish for something better. However, this gives us a working definition, from which we can hope to learn something of how difficult it is to hide in practice. For this working definition, we will henceforth use an estimator $\hat{\sigma}$ to the degree deviation; namely the deviation on a random graph of the expected size of V_i.

Suppose we have created a graph using NSC with probability p on n vertices, where we may assume that I is larger than the probable size of a maximum independent set in a graph in $\mathcal{G}_{n,p}$. Then $\mu(I) = (n-s)p$, $\mu(\bar{I}) = (n-1)p$, $d_0 = (s-1)p$ and $\hat{\sigma}(\bar{I}) = \sqrt{(n-1)p(1-p)}$. Thus, if $p = \frac{1}{2}$, $n = 1000$, we see that with $s \geq 31$ the independent set will stand out. It is well known [5] that with high probability the maximum independent set of a graph in $\mathcal{G}_{1000,\frac{1}{2}}$ will be of size 15 or less. So for this case a set created by NSC will stand out at $i = 0$ if the

set is twice the background.

Noting that $|I_i| = s - i$ and the expected size of $\bar{I}_i$ is $(n-s)(1-p)^i$, we see that $\mu(I_i) = p(n-s)(1-p)^i$ and $\mu(\bar{I}_i) = p((n-s)(1-p)^i + s - i - 1)$. Thus $d_i = p((s-i)-(1-p)^i) \approx p(s-i)$ and $\hat{\sigma}(\bar{I}_i) = \sqrt{(1-p)p((n-s-1)(1-p)^i + s - i)}$. Let us suppose that NSC is used to create a hidden set of size 16 in $\mathcal{G}_{1000,\frac{1}{2}}$. The results are shown for i up to 3 in Table 1.

i	d_i	$\hat{\sigma}(\bar{I}_i)$
0	7.5	15.68
1	7.25	11.25
2	6.88	8.06
3	6.44	5.83

Table 1: Statistics for $s = 16$ in $\mathcal{G}_{1000,\frac{1}{2}}$

Although I_2 does not quite stand out, nevertheless SM^2 found the set in 36 out of 50 trials. Notice that for $p = \frac{1}{2}$ this is the smallest set that we could hope to hide and still meet the conditions in [4].

It should be mentioned that there is another effect that helps SM^i. I_i has one fewer vertex than I_{i-1}, while $\bar{I}_i$ has on average $(1-p) * |\bar{I}_{i-1}|$ vertices. Thus, the ratio of desirable to undesirable vertices is increasing exponentially.

3 Improving The Graph Generator

The SM^i algorithms seem to succeed regularly on graphs created by the NSC algorithm, because the degree difference $d_i \approx \hat{\sigma}(\bar{I}_i)$ with $i \geq 2$. To defeat this form of attack, we attempted to set $d_i = 0$ for an arbitrary level i.

The Split Naive Set Creation algorithm (SNSC) has two distinct probabilities: p_{10} and p_{00}. p_{10} is the probability of connecting two vertices x, y, if exactly one of $\{x \in I, y \in I\}$ holds. Similarly, p_{00} is the probability of connecting two vertices x, y, if both $x, y \notin I$. This degree of freedom allows us to define the probabilities so that the average edge density across the graph equals our original probability p, while enforcing that $d_i = 0$ for some i.

There is a parameter u that is called the *level of hiding* we wish to use on the graph. When $u = 0, p_{00} = p_{10} = p$. If $u \geq 1$, then we set the difference function: $d_{u-1} = 0$. The next two paragraphs outline the procedure for setting p_{00} and p_{10} when $u \geq 1$.

Assume that we have chosen u elements of the independent set I_0 to create the graph on V_u and the set I_u. Let t be the average degree of a vertex in a graph of size $|V_u|$, but with edge probability p. If we assume that $|I_u| \geq 2$, the following formulas hold: $\mu(I_u) = p_{10}(1 - p_{10})^u(n-s)$, $\mu(\bar{I}_u) = p_{00}((1-p_{10})^u(n-s)-1) + p_{10}(s-u)$, $t = p(s-u-1+(1-p_{10})^u(n-s))$.

We wish to enforce that $\mu(I_u) = \mu(\bar{I}_u) = t$. We can solve for p_{10} with $\mu(I_u) = t$ as follows.

If we consider p_{10} as the variable in the function $f(p_{10}) = \mu(I_u) - t$, and let $x = p_{10}$, we get $f(x) = x(1-x)^u(n-s) - p(s-u-1+(1-x)^u(n-s))$ where x is the probability (unknown) of an edge from I to $V-I$. We wish to solve for $f(x) = 0$ given u and p.

Rewriting the formula, we see that $f(x) = (1-x)^u(x-p)(n-s) - p(s-u-1)$ and $f'(x) = (1-x)^{(u-1)}(n-s)((1-x) - u(x-p))$. Note $f(x) < 0$ if $x = 1$ or $x = p$. Solving for $f'(x) = 0$ we find $x_c = \frac{1+up}{1+u}$ is the critical point strictly between p and 1. ($x = 1$ is another critical pt.)

If $f(x_c) >= 0$ then there is one real root in $[p, x_c]$ and this is the value we want to use in depth u hiding at probability p. If $f(x_c) < 0$, we can't do depth u hiding on this graph.

We ran a series of experiments on graphs created with SNSC with $g = 1000$, $p = \frac{1}{2}$, using SM^2 to find the independent set. The results of these tests are given in Table 2. 50 graphs were run for each data point, and a success was registered when SM^2 found a graph of the requisite size. The probability of success increases with the size of the independent set being hidden and decreases as the depth of hiding increases.

	Ind. Set Size Created		
u	$s = 16$	$s = 17$	$s = 18$
0	50	50	50
1	47	47	50
2	38	43	49
3	18	24	33
4	0	0	0

Table 2: SM^2 Successes against $SNSC_{1000,\frac{1}{2}}$

To speed up the results of these tests, the only pairs of vertices considered as candidates in the SM^2 algorithm are those that have two members of the independent set in them. This "cheating" version of SM^2 generates only $\frac{s(s-1)}{2}$ independent sets, while the "non-cheating" version generates $\frac{n(n-1)}{2}$ independent sets, all but $\frac{s(s-1)}{2}$ of which will fail with high probability. This is a large time savings when doing experimentation since $s \leq \sqrt{n} \ll n$ for the majority of the experiments.

Since there are $\frac{s(s-1)}{2}$ attempts to obtain the independent set in SM^2 the chance of finding the set will increase with s. The probability that SM^2 will find the set drops as u increases because the independent set elements must have increased degree in the initial graph to satisfy the constraint. Hence, the minimum degree heuristic will be misled.

As a counterpart to the SM^i algorithms, a different heuristic was tried. The SP^0 algorithm, given in Figure 3, uses the *maximum degree drop* heuristic to generate the first half of the independent set. After half of the expected independent set being searched for is found, the minimum degree heuristic is used to generate the rest of the set. The maximum degree

drop heuristic works well in cases where the degree of the independent set elements has been artificially raised, since the focus is placed on both current and original degree sequences.

```
INPUT: G = (V, E), initial degree sequence deg_G(v)
    expected independent set size = e
OUTPUT: A maximal independent set I
    Z = V
    I = ∅
    while Z ≠ ∅ do
        G' = subgraph induced by Z.
        Compute current degrees deg_G'(v) ∀v ∈ Z
        if (e ≥ 2 * |I|) then
            Comment: Maximum Degree Drop Heuristic
            r = maximum {v ∈ Z | deg_G(v) − deg_G'(v)}
            Select (randomly) x ∈ Z with r = deg_G(x) − deg_G'(x)
        else
            Comment: Minimum Degree Heuristic
            r = minimum {v ∈ Z | deg_G'(v)}
            Select (randomly) x ∈ Z with r = deg_G'(x)
        endif
        I = I ∪ {x}
        Z = Z − {x} − N(x)
    end
    return I
```

Figure 3: Algorithm SP^0

The SP^i algorithm for calling the SP^0 algorithm is similar to the SM^0 algorithm, but must accept the expected independent set size as input to pass to SP^0. As well, it must compute the degrees from the initial graph G of vertices in V'.

The same graphs as in Table 2 were searched by the SP^2 algorithm and the results are given in Table 3. The results show that as the independent set vertices are made artificially higher, making them more difficult for the SM^2 algorithm to find, they are easier for SP^2 to find.

When the results of Table 2 are combined with Table 3, we have an interesting scenario. The background independent sets that we can expect to see on $SNSC_{1000,\frac{1}{2}}$ are of size 15, with a rare appearance of a set of size 16. Using the pair of algorithms in tandem, the results show that there are no independent sets that are difficult to hide from these two algorithms.

Another problem with NSC and SNSC is the possibility of an independent set of the same size appearing that is not the intended one. These *false positive* independent sets are undesirable in a scheme where you are trying to hide the size of the independent set from

	Ind. Set Size Created		
u	$s = 16$	$s = 17$	$s = 18$
0	0	1	0
1	3	6	13
2	22	27	33
3	50	50	50
4	50	50	50

Table 3: SP^2 Successes against $SNSC_{1000,\frac{1}{2}}$

an adversary, or are trying to encode information in the vertex labelings of the maximum independent set. For example, a false positive set will occur when a vertex is attached to only one element of the independent set, and will be selected in about 50% of the trials when this occurs.

In an attempt to fix the false positive problem, we slightly modified our assignment of edges to independent set vertices. The complete graph generation algorithm DELTA, shown in Figure 4, forces all members of $V - I$ to have a nearly equal number of edges leading to independent set vertices.

This improvement not only removes most of the false positive sets, but also reduces the variances on the degree of the vertices. As a result, this generator proved to provide more difficult graphs for the algorithms.

4 Results

The DELTA graph generator was tested over a range of probabilities with a fixed graph size of 1000. It was also tested over a fixed probability of $p = \frac{1}{2}$, with a varying graph size. For each pair of graph size and probability, the tests were run for each level of hiding u from 0 to 4. For each of these trials in the tables, the background was computed using the formulas found in [1]. The largest probable background set is listed under Background in the tables. To compute each entry in the tables, we started by creating fifty graphs using different random number seeds, setting s one larger than the computed background. "Cheating" versions of SM^2 and SP^2 were used to examine the fifty graphs. If neither algorithm found at least twenty five of the fifty independent sets, the test was re-run with s incremented by 1. In Table 4 and Table 5, we report where one of the two algorithms first finds the independent set size with a probability of 50% or greater. Dashes in Table 4 indicate that the level of hiding selected did not give a proper probability for p_{10}.

It is extremely interesting to note that in every experiment summarized with Table 4 and Table 5, the SP^2 algorithm achieves better results when $u \geq 2$, and SM^2 achieves better results when $u \leq 1$. This tends to indicate that the level of highest difficulty for the pair of

INPUT: Depth of hiding: u
    Graph size: n, Ind. Set Size: s
    Probability: p
OUTPUT: $G = (V, E)$ with maximum ind. set size is
    and an average edge density p.
$V = \{v_1, \ldots, v_n\}$
Compute p_{10} and p_{00} based on u.
Choose independent set elements $I = i_1, \ldots, i_s$.
$d = 0.0$
for $v \in V - I$ do
    Comment: Add edges from v to $\bar{I}$
    for $w \in V - I$ s.t. $w > v$ do
        if rrandom $< p_{00}$
            $E = \{v, w\} \cup E$
        fi
    od
    $d = d + is * p_{10}$
    Comment: Add edges from v to I
    while $d > 0$ do
        repeat
            $x =$ Uniform$[1 \ldots is]$
        until $(v, x) \notin E$
        $E = \{v, x\} \cup E$
        $d = d - 1$
    od
od
return$((V, E))$

Figure 4: The DELTA Graph Generation Algorithm

Graph Size	Prob.	Back-ground	Independent Set Size of First Success				
			Depth 0	Depth 1	Depth 2	Depth 3	Depth 4
1000	0.25	30	44	48	51	43	37
1000	0.30	25	37	41	40	34	28
1000	0.35	22	32	35	34	28	23
1000	0.40	19	28	30	28	22	20
1000	0.45	17	25	26	23	18	18
1000	0.50	15	21	22	19	16	16
1000	0.55	13	19	19	16	14	–
1000	0.60	12	16	17	14	13	–
1000	0.65	11	14	14	12	12	–
1000	0.70	9	12	12	10	–	–
1000	0.75	8	10	10	9	–	–

Table 4: Summary of Runs: $\mathrm{DELTA}_{1000,p}$

Graph Size	Prob.	Back-ground	Independent Set Size of First Success				
			Depth 0	Depth 1	Depth 2	Depth 3	Depth 4
125	0.50	9	10	10	10	10	10
250	0.50	11	12	12	12	12	12
500	0.50	13	15	16	14	14	14
750	0.50	14	19	19	16	15	15
1000	0.50	15	21	22	19	16	16
1250	0.50	15	24	24	22	16	16
1500	0.50	16	26	27	23	18	17
1750	0.50	16	28	29	26	19	17
2000	0.50	17	30	32	29	20	18

Table 5: Summary of Runs: $\mathrm{DELTA}_{G,\frac{1}{2}}$

algorithms lies in between $u = 1$ and $u = 2$.

At the top of Table 4, we notice that for $p = 0.25$, $u = 2$ hides larger sets than $u = 1$. In all other cases, $u = 1$ hides larger sets than $u = 2$. As the probability p decreases, the effectiveness of SP^2 tends to be decreasing faster than for SM^2.

Figure 5 illustrates the difference in capability between the SNSC graph generation algorithm and the DELTA graph generation algorithm on graphs of 1000 vertices.

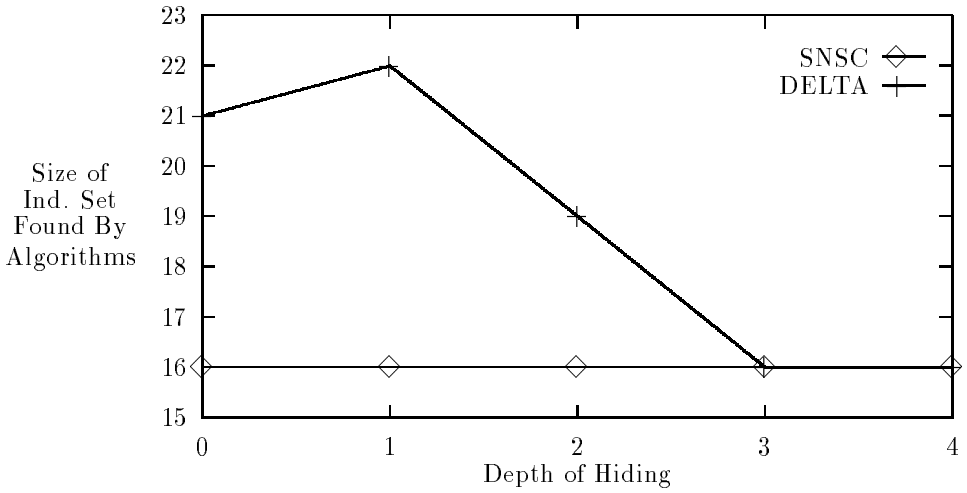

Figure 5: Independent Set Size First Found with Varying Graph Generators, $n = 1000$, $p = 1/2$

Figure 6 and Figure 7 illustrate the ratio of the size of the independent set found with greater than 50% probability to the background over the ranges in Table 4 and Table 5, respectively. As a ratio of the size of the hidden independent set to the background, both increasing the size of the graph and reducing the edge density allow larger independent sets to be hidden from the two algorithms. This makes intuitive sense when we have fixed the size of the full search to two levels. Any change in the graph which increases the number of vertices left in the graph after two vertices have been removed causes more noise and gives SM^2 and SP^2 a harder time to find the independent set.

5 Future Research

It was realized after the experiments that there is nothing to limit u to a whole number. Values in between 1 and 2 seem promising for generating larger independent sets that neither SM^2 nor SP^2 can find.

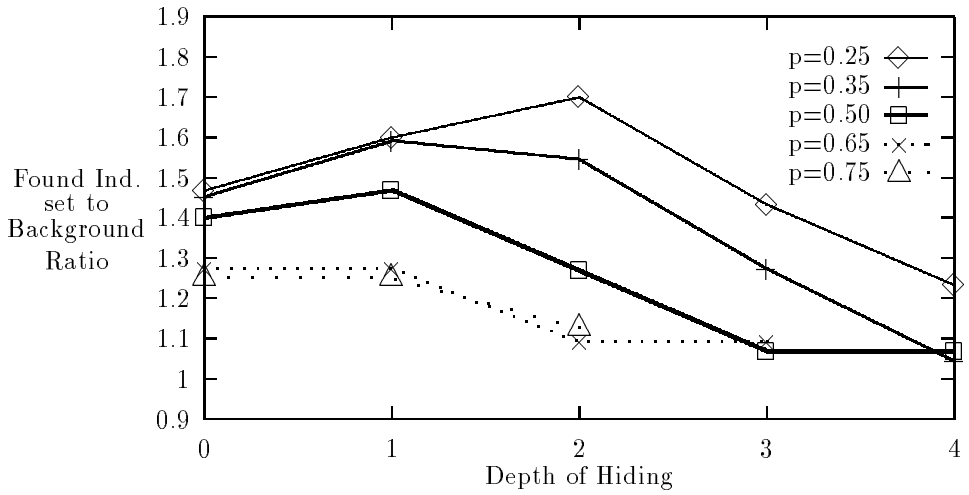

Figure 6: Ratio of Hidden Independent Sets over Varying Probabilities, $\text{DELTA}_{1000,p}$

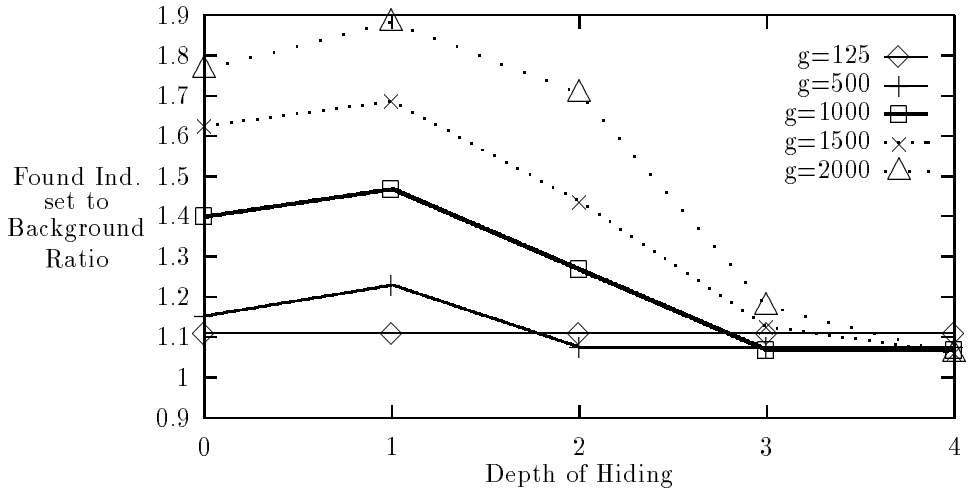

Figure 7: Ratio of Hidden Independent Sets over Varying Graph Sizes, $\text{DELTA}_{n,1/2}$

We have received a number of independent set finders, but we haven't been able to test the results of the graphs generated by DELTA on algorithms such as GRASP$^{\text{tm}}$ [2] due to time constraints. Although it seems clear that both the number of tuples (`ntup`) considered and the number of vertices initially considered (`ntop`) must increase from values used on $\mathcal{G}_{1000,\frac{1}{2}}$, it is not clear whether or not the independent sets are "easy" for the randomized search to discover once a tuple from the independent set is under examination.

Initially it seemed that we could increase the hiding capability of a graph by increasing the "noise"; that is, by creating more variance in the degree sequence of the graph. (DELTA originally allowed for this, and hence the name). However, the problem is much more tightly constrained than we initially believed. It seems that the "noise" must be carefully tuned to hide the set.

We hope that the DIMACS Challenge will be helpful in determining the success of the DELTA graph generation algorithm against a wide variety of independent set finders.

References

[1] Béla Bollobás. Random Graphs. Chapter XI, "Cliques, Independent Sets and Colouring", pages 251–279. Academic Press, London, 1985.

[2] Thomas A. Feo, Mauricio G.C. Resence and Stuart H. Smith A greedy randomized adaptive search procedure for Maximum Independent Set. In *Operations Research*, to appear, 1993.

[3] Michael R. Garey and David S. Johnson. Computers and Intractability: a Guide to the Theory of NP-completeness. W. H. Freeman, 1985

[4] Luděk Kučera. A generalized encryption scheme based on random graphs. In *17th Annual Workshop on Graph-Theoretic Concepts in Computer Science (WG91)*, volume 570 of *Lecture Notes in Computer Science*, pages 180–186. Springer-Verlag, Berlin, 1991.

[5] Bennet Manvel. Extremely greedy coloring algorithms. In *Graphs and applications (Boulder, Colo., 1982)*, Wiley-Intersci. Pub., pages 257–270, New York, New York, 1985. John Wiley & Sons, Inc.