Experimental Evaluation of the Greedy and Random Algorithms for Finding Independent Sets in Random Graphs*

M. Goldberg, D. Hollinger, M. Magdon-Ismail

Computer Science Department
Rensselaer Polytechnic Institute

Abstract. This work is motivated by the long-standing open problem of designing a polynomial-time algorithm that with high probability constructs an asymptotically maximum independent set in a random graph. We present the results of an experimental investigation of the comparative performance of several efficient heuristics for constructing maximal independent sets. Among the algorithms that we evaluate are the well known randomized heuristic, the greedy heuristic, and a modification of the latter which breaks ties in a novel way. All algorithms deliver online upper bounds on the size of the maximum independent set for the specific input-graph. In our experiments, we consider random graphs parameterized by the number of vertices n and the average vertex degree d. Our results provide strong experimental evidence in support of the following conjectures:

1. for $d = c \cdot n$ (c is a constant), the greedy and random algorithms are asymptotically equivalent;
2. for fixed d, the greedy algorithms are asymptotically superior to the random algorithm;
3. for graphs with $d \leq 3$, the approximation ratio of the modified greedy algorithm is asymptotically < 1.005.

We also consider random 3-regular graphs, for which non-trivial lower and upper bounds on the size of a maximum independent set are known. Our experiments suggest that the lower bound is asymptotically tight.

* This research was partially supported by NSF grants 0324947 and 0346341

1 Introduction.

The problem of constructing a maximum independent set (*MIS*) in a graph is a classical NP-hard computational problem [9], which arises in many applications. It is also NP-hard to approximate the size of the maximum independent set [11]. For general graphs, the best approximation algorithm for the independence number has an approximation ratio of $O(n/(\log n)^2)$ [4]; for bounded-degree graphs, the best ratio known is $O(\Delta/\log\log\Delta)$ ([8]). The problem is 2-approximable on random graphs with fixed edge probability p (the average degre d is $(n-1)p$). Here we consider random graphs in the Erdös-Rényi, or $G(n,p)$-model [2, 5]: given n vertices, each of the $\binom{n}{2}$ edges is generated independently with probability p. Let $\alpha(n,p)$ be a random variable denoting the size of a maximum independent set in a $G(n,p)$ graph. For fixed p, it is known that the standard randomized algorithm *Random* (described below) outputs an independent set of size [1] $\sim \log_{1/(1-p)} n$, with probability $\to 1$ (w.p.1). The 2-approximability of *MIS* on $G(n,p)$ with fixed p follows from the results by Bollobás and Erdös ([3]) and Matula ([12]), where they show that for a $G(n,p)$ graph with fixed p, $\alpha(n,p) \sim 2\log_{1/(1-p)} n$, w.p.1. For $c > 0$, no polynomial algorithm for *MIS* on a $G(n,p)$ is known to find, w.p.1, an independent set of size $\sim (1+c)\log_{1/(1-p)} n$. Frieze and McDiamard [6, page 11] pose the following research problem:

> *Construct a polynomial algorithm that finds an independent set of size* $\sim (\frac{1}{2}+c)\alpha(n,p)$ *with high probability, or show that no such algorithm exists, modulo some reasonable conjecture on the computational complexity hierarchy (e.g.* $P \neq NP$*).*

Two well-known algorithms for *MIS* are *Random* and *Greedy*, which are instantiations of the following general *sequential* algorithm:

1: *Sequential algorithm for MIS*:
2: $I = \emptyset$ (the independent set);
3: **while** $G \neq \emptyset$ **do**
4: select a vertex $v \in G$;
5: $I \leftarrow I \cup \{v\}$;
6: $G \leftarrow G \setminus \{N(v) \cup v\}$

In *Random*, the vertex selected in step 4 is random, whereas in *Greedy* it is a minimum degree vertex (ties are broken randomly). *Random* is generally

[1] We use the notation $f(n) \sim g(n)$ if $\lim_{n\to\infty} f(n)/g(n) = 1$.

easier to analyze mathematically ([2]), but it under-performs *Greedy* on most graph domains. It is not known whether *Greedy* is asymptotically better than, worse than, or equivalent to *Random* on $G(n,p)$ graphs. It is not known whether there are any polynomial-time algorithms that perform asymptotically better than *Random*. The difficulty encountered by existing analysis techniques in attempting to analyze deterministic algorithms, *e.g. Greedy*, on random objects is that during execution, the algorithm destroys the randomness in an analytically unpredictable way. This difficulty already surfaces when one analyzes *Greedy* on random graphs with average degree 3. The only results that carry through the analysis of *Greedy* are those for finding matchings in random graphs ([10], [1]) and for 3-regular random graphs ([7]). However, it is not known how *Greedy* performs on random d-regular graphs with $d \geq 4$; on random graphs with the average vertex degree d, for any $d > 0$; or on random graphs with a fixed edge-probability $p > 0$ ($d = p(n-1)$).

In this paper, we present the results of a comparative experimental investigation of the performance of several fast polynomial heuristics for constructing maximum independent sets in random graphs. The procedures that we tested include *Random*, *Greedy*, and a modification of the latter, *Greedy*^{m}, which breaks the ties in a novel way. All three heuristics deliver, *on-line*, upper bounds on the size of the maximum independent set in the input-graph. The input domain of all experiments is the set of random graphs with a given number n of vertices and a given average vertex degree d. Based on the experiments, we formulate the following conjectures:

- for $d = c \cdot n$ (c is a constant), *Random* and the greedy algorithms are asymptotically equivalent, i.e., the independent sets they construct are asymptotically of the same size;
- for fixed d, *Greedy* and *Greedy*^{m} are asymptotically superior to *Random*;
- for $d \leq 3$, the approximation ratio of *Greedy*^{m} is asymptotically $<$ 1.005 w.p.1.

We also tested our algorithms on random 3-regular graphs for whichnontrivial lower and upper bounds on the *MIS* exist ([7]). The results suggest that the lower bound is asymptotically tight.

We will use standard graph theory notation as described in [13]. For a graph G, the independence number $\alpha(G)$ is the maximal size of an independent set in G. For $v \in V(G)$, $\alpha_v(G)$ denotes the maximal size of an independent set containing v. Vertex v is **correct** if $\alpha_v(G) = \alpha(G)$.

$N(v)$ is the neighborhood of v, the set of vertices adjacent to v; $deg(v)$, the degree of v, is the size of the neighborhood, $|N(v)|$; $deg(G) = \max_v deg(v)$. For a subset of the vertices, $T \subseteq V(G)$, $N(T)$, the neighborhood of T, is the union of the neighborhoods of the vertices in T.

Let $\mathcal{A}$ be a sequential algorithm for *MIS*, which constructs an independent set by repeatedly selecting vertices, *e.g. Random*, or *Greedy*. When $\mathcal{A}$ is applied to G, suppose that the sequence of vertices selected is $\{v_1, v_2, \ldots, v_k\}$ (k is the size of the independent set). Let G_i be the subgraph obtained from G by removing $N(\{v_1, \ldots, v_{i-1}\})$. The **dynamic degree** $\overline{deg}(v_i)$ is the degree of v_i in G_i (the dynamic degree depends on the algorithm $\mathcal{A}$); $\overline{deg}(G)$, the dynamic degree of G, is $\max_i\{\overline{deg}(v_i)\}$. Let I_ℓ denote the vertices in the output independent set whose dynamic degrees are ℓ ($\ell \geq 0$); thus, $I = \cup_{\ell \geq 0} I_\ell$.

2 On-line upper bounds.

A key feature of our algorithms is that we bound the approximation ratio of the output independent set, for the specific input graph. Our bounds are based on the following observations.

Lemma 1 *Let $v \in V(G)$. If $deg(v) = 0$, then $\alpha_v(G) = \alpha(G)$, otherwise $\alpha_v(G) \geq \alpha(G) - deg(v) + 1$. If $\alpha_v(G) < \alpha(G)$, then every MIS of G contains at least $\alpha(G) - \alpha_v(G) + 1$ vertices from $N(v)$.* ■

Lemma 2 *For a sequential algorithm $\mathcal{A}$ which selects vertices $\{v_1, \ldots, v_k\}$, $\alpha(G) \leq |I_0| + \sum_{i=1}^{k} \overline{deg}(v_i)$.* ■

The lemmas imply that selecting any vertex of dynamic degree 0 or 1 is correct, and each vertex of dynamic degree 2 gives an error of size at most 1. Our experiments show that for $G(n,p)$ graphs with average degree ≤ 6, *Greedy* selects a small number of vertices of dynamic degree 2, and the probability that *Greedy* selects a vertex of dynamic degree ≥ 3 is asymptotically 0. The general problem of determining whether a given degree two vertex belongs to an *MIS* is NP-complete. We present polynomial-time sufficient conditions for a degree two vertex to be correct. These conditions are incorporated in *Greedy*^{m}, an enhancement over *Greedy* in which ties between dynamic degree 2 vertices may be broken optimally to yield a correct vertex. Since we can always remove any vertex with dynamic degree 0 or 1, we assume that the dynamic degrees of all vertices are at least 2.

A **straight path** is a path containing only degree two vertices. An **interval** is a straight path which is not a proper subpath of another

straight path. A straight path is **even** (resp. **odd**) if its length is even (resp. odd). (The smallest interval consisting of one vertex is even and has length 0.) The **end-points** of an interval are the first and last vertices of the interval. A **connector** is a vertex of degree > 2 adjacent to the end-points of an interval. A **leaf-interval (or leaf)** is an interval with length > 0 whose end-points are adjacent to the same connector. A **loop-interval (or loop)** is an interval whose end-points are adjacent.

Define a bi-partite graph $Q = (I \cup C; F)$ as follows: the vertex set is $I \cup C$, where I is the set of intervals and C is the set of connectors; the edge (p, c) is in the edge set F *iff* p is an interval in I and c is a connector adjacent to at least one end-point of p. The even (resp. odd) subgraphs Q^{ev} (resp Q^{odd}) are the induced subgraphs after removing the even (resp. odd) intervals from Q. We now give sufficient conditions for a degree 2 vertex to be correct (proofs are postponed to a full version of the paper).

Theorem 1 *If v is an end-point of a loop or leaf, then v is a correct.*

Theorem 2 *Let $\mathcal{C}$ be a connected component of Q^{ev} and v an end-point of an interval in $\mathcal{C}$. Then v is correct if one of the following hold:*

1. *$\mathcal{C}$ contains a cycle;*
2. *two connectors from $\mathcal{C}$ are adjacent in G;*
3. *there is an odd interval whose connectors are both vertices in $\mathcal{C}$.*

2.1 *Greedy*^{m}: Modifying the Greedy Algorithm

Theorem 2 indicates how we can modify *Greedy* to obtain a better algorithm. If a degree 2 vertex v is encountered, we may try to determine if it is correct. If it is correct, then we may safely take v. A polynomial-time scan through all degree 2 vertices can be used to find a correct degree two vertex if one exists. If no degree 2 vertex is found, then the degree 2 vertex that is a member of the longest path is selected, resulting in a mistake of at most 1 for all the vertices selected in the path. Thus, *Greedy*^{m} differs from *Greedy* (described in the introduction) only in how it breaks ties among dynamic degree 2 vertices.

2.2 Efficiently Generating Random Graphs

For dense graphs (fixed edge probability p), the expected size of the input is $\Omega(n^2)$. Thus, one cannot significantly improve upon the algorithm that checks each of the $\binom{n}{2}$ pairs to generate $\binom{n}{2} p$ of edges independently with probability p.

When $p = o(1)$, generating the edges by examining all pairs of vertices is excessive since only $O(n^2 p)$ edges need be generated. It is therefore more efficient to first generate the number of edges M, and then select the specific edges. For sparse graphs, with $p = d/(n-1)$, this approach yields considerable computational savings.

Specifically, the number of edges in the graph M is a binomial random variable $B(p, \binom{n}{2})$. The following code can be used to generate M efficiently,

```
1: x = 0; y = 0; N = (n 2); c = ln(1 − p);
2: if c = 0 then
3:    return 0;
4: while  TRUE do
5:    Generate a uniform random variate u ∈ [0, 1];
6:    y ← y + ⌊ ln u/c ⌋ + 1;
7:    if y ≤ N then
8:       x ← x + 1;
9:    else
10:      BREAK;
11: return x;
```

The runtime of the algorithm is $O(x)$, where x is the output value for M. Since the expected value of M is $\binom{n}{2} p = O(n)$ (sparse graphs), we see that generating M is quite efficient.

Given M, it is now a simple matter to uniformly pick M edges from the available $\binom{n}{2}$ edges. The following algorithm accomplishes this task,

```
1: Let S = ∅ be the set of selected edges;
2: while |S| < M do
3:    Generate a uniform integer random variate u ∈ [1, (n 2)];
4:    if edge u ∉ S then
5:       S ← S ∪ u;
```

The probability that a sampled edge is not placed in S is $O(1/n)$ for sparse graphs. Therefore in $M = O(n)$ samples, $O(1)$ edges are rejected. Thus, $O(2M)$ samples should suffice. If the set S is stored in a data structure that allows for efficient searching in time $O(\log |S|)$, for example a balanced binary search tree, then the entire algorithm has expected run time $O(M \log M)$.

3 Experimental Results

The experiments described here involve running *Random*, *Greedy*, and *Greedy*^{m} on randomly generated graphs from different domains: $G(n,p)$ for a fixed p; $G(n,p)$ for $p = d/(n-1)$, where d is an integer in [1,10]; and 3-regular graphs. All our experiments are reproducible, since we use a seeded pseudo-random number generator. The machines used to run the experiments range in size and power from Sun Ultra 10 workstations with 256MB RAM running Solaris, to Intel IA64 based workstations with 16GB RAM running Linux.

3.1 *Greedy* vs. *Random* For Fixed Edge Probability

It is not known whether *Greedy* outperforms *Random* on $G(n,p)$ graphs for fixed p – no theoretical or experimental evidence to the contrary is available. We present experimental data comparing *Random* to *Greedy* on $G(n,p)$ graphs for fixed p. We show the results for $p = 0.1$ and n up to 100,000 (other values of p gave similar results).

For each n, we generated 1000 graphs, and compared the average independent set size found by *Greedy* to that found by *Random*. Let $g(n,p)$ (resp. $r(n,p)$) be the average independent set size found by *Greedy* (resp. *Random*). We are interested in the ratio $R(n,p) = g(n,p)/r(n,p)$, which is plotted in Figure 1(a). Figure 1(a) shows that $R(n,p)$ is not a monotone function (as was expected initially) but a unimodular function with the maximum near $n = 1000$.

Although *Greedy* consistently finds a larger independent set than *Random*, and $R(100,000, 0.1) > 1.15$, the analysis of $R(n, 0.1)$ suggests that indeed $R(n, 0.1) \to 1$, as $n \to \infty$. We analyze the experimental data for $R(n, 0.1)$ by fitting a curve of the form $R(n,p) = c_0 + c_1 f(n)$, where $f(n) \to 0$. We have tried many functional forms for $f(n)$ (yielding similar results), but the functional forms $f(n) = 1/\log n$ and $f(n) = \log\log n/\log n$ are suggested by the theoretical analysis of *Random*, and so we only show the results for these two functional forms. For $n \geq n_L$, we can obtain the optimal least squares fit for c_0, denoted $c_0(n_L)$. As n_L gets larger, $c_0(n_L)$ should converge to the true value of c_0. We show the convergence of $R(n, 0.1)$ as a function of n and the convergence of $c_0(n_L)$ in Figure 1. From Figure 1(b) it appears that c_0 converges to 1 for $f(n) = 1/\log n$. $f(n) = \log\log n/\log n$ does not give a reasonable value for c_0 as it is below 1. Thus the data indicates that $c_0 \to 1$ and that $f(n) = 1/\log n$. In particular, this indicates that *Greedy* only outperforms *Random* by a constant number of vertices.

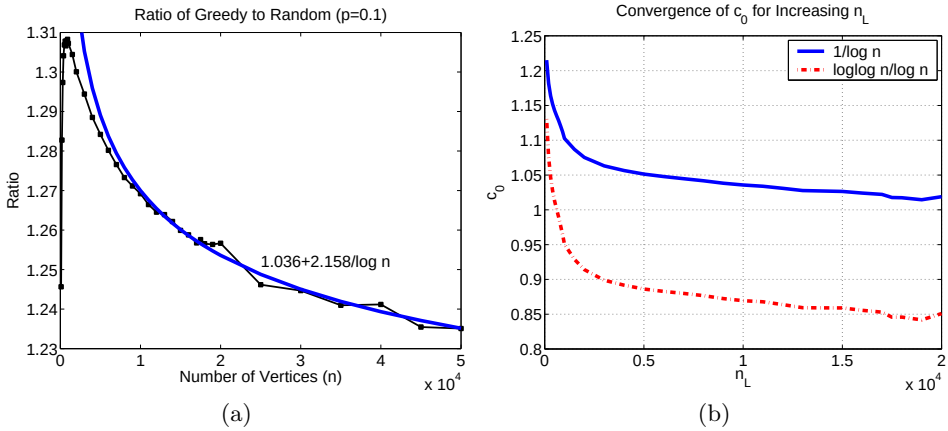

Fig. 1. (a) $R(n,p)$ for $p = 0.1$, $n \leq 50000$. Also shown is the fit to $1/\log n$ for $n \geq 10,000$. (b) Convergence of $c_0(n_L)$ for $f(n) \in \{1/\log n, \log\log n/\log n\}$.

3.2 *Greedy* vs. *Random* For Fixed Average Degree

Frieze and Suen [7] show that for random 3-regular graphs, *Greedy* constructs an independent set of the size at least $(6\log\frac{3}{2} - 2)n \approx 0.432791n$ w.p.1., which is asymptotically larger than the independent set constructed by *Random* (the asymptotics for *Random* can be found in [2]). The data in Table 1 show that the lower bound on the performance of *Greedy* is tight. Each entry is an average over 1000 randomly generated graphs.

n	*Random*	*Greedy*
1,000	0.3749 n	0.4320 n
5,000	0.3751 n	0.4324 n
10,000	0.3750 n	0.4326 n
50,000	0.3750 n	0.4327 n

Table 1. Size of independent sets found in 3-Regular graphs

Our experiments on random graphs with fixed average degree show a similar performance gain for *Greedy* over *Random*. Figure 2 shows the ratio $R(n,p)$ for $d = 1,2,3,4,5,6$; and $n \in [1000, 50, 000]$; each data point is an average over 1000 graphs. The main conclusion: on average, *Greedy* outperforms *Random* by a multiplicative constant which is increasing in d for small d.

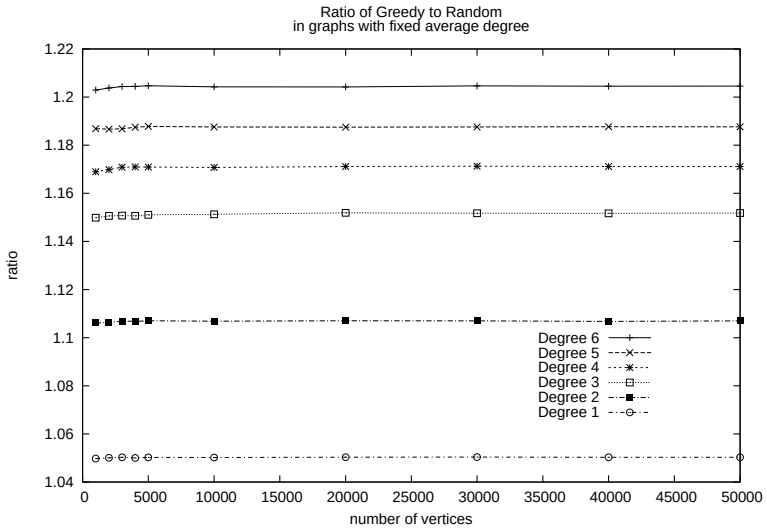

Fig. 2. Ratio of *Greedy* vs. *Random* for graphs with fixed average degree

3.3 Dynamic Degrees

We now consider the dynamic degrees of the vertices in the independent sets found by *Greedy* and *Random*. The dynamic degrees allow us to establish non-trivial upper bounds on the independence number of the input-graph. The dynamic degree distributions with respect to *Greedy* for random average degree 3 graphs are shown in Table 2. Tables Tables 3, 4 and 5 give the corresponding distributions for average degrees 4, 5 and 6 respectively. These statistics (averaged over 1000 graphs) were

n	I_0/n	I_1/n	I_2/n
1000	0.076	0.442	0.013
5000	0.073	0.449	0.009
10000	0.073	0.451	0.009
50000	0.072	0.452	0.008
100000	0.072	0.452	0.008

Table 2. Distribution of dynamic degrees for graphs with average degree 3

n	I_0/n	I_1/n	I_2/n
1000	0.025	0.363	0.083
5000	0.023	0.368	0.081
10000	0.023	0.368	0.080
50000	0.022	0.369	0.080

Table 3. Distribution of dynamic degrees for graphs with average degree 4

collected from the same data used in the previous section. Although there

are a few vertices of dynamic degree 3 for average degree and 6 graphs, it is clear that the number of such vertices is approaching zero as the graph size increases.

n	I_0/n	I_1/n	I_2/n
1000	0.010	0.255	0.160
5000	0.008	0.260	0.157
10000	0.008	0.260	0.157
50000	0.008	0.262	0.156

Table 4. Distribution of dynamic degrees for graphs with average degree 5

n	I_0/n	I_1/n	I_2/n	I_3/n
1000	0.004	0.161	0.224	0.0002
5000	0.003	0.165	0.222	0.0000
10000	0.003	0.166	0.222	0.0000
50000	0.003	0.166	0.221	0.0000

Table 5. Distribution of dynamic degrees for graphs with average degree 6

Using the data collected above we are able to evaluate the accuracy of *Greedy* using the bound in Lemma 2. Table 6 shows the approximation ratio for *Greedy* on average degree graphs with up to 50,000 vertices. The ratio shown is the average upper bound on the size of independent set derived from the dynamic degrees over the average size found by the *Greedy* algorithm.

Average Degree	Approximation Ratio			
	$n = 1000$	$n = 5000$	$n = 10000$	$n = 50000$
1	1.0000784	1.0000195	1.0000077	1.0000014
2	1.0006551	1.0001155	1.0000603	1.0000111
3	1.0249426	1.0174993	1.0161715	1.0150454
4	1.1767924	1.1709950	1.1708517	1.1694409
5	1.3761916	1.3700855	1.3695010	1.3674423
6	1.5764024	1.5693010	1.5685679	1.5670232

Table 6. Accuracy of *Greedy* on random graphs with a fixed average degree

Our interpretation of the data presented in Table 6 is that *Greedy* is very near optimal for graphs with average degree 1 and 2, and probably 3 as well (asymptotically).

3.4 *Greedy*^{m} for Average Degree 3.

Greedy^m is similar to *Greedy* except that the ties between dynamic degree 2 vertices are broken by incorporating the sufficient conditions in Theorem

2. Thus, not only will the independent set found be larger, but the bound obtained in Lemma 2 can be improved by subtracting the number of dynamic degree two vertices that are correct. The approximation ratios in Table 7 uses this improved upper bound from $Greedy^m$. As before, each entry is based on an average over 1000 random graphs.

n	*Random*	*Greedy*	$Greedy^m$
1000	1.183317	1.021652	1.013076
5000	1.170949	1.012581	1.006536
10000	1.171350	1.012373	1.006349
50000	1.166931	1.009917	1.004922
100000	1.166914	1.009740	1.004830

Table 7. Approximation ratios for *Random*, *Greedy*, and $Greedy^m$; average degree = 3.

4 Conclusions.

The main objective of this research is the development of a database of experimental results to aid the theoretical investigation of the problem of constructing large independent sets in random graphs. Our experiments support a conclusion that traditional randomized algorithms are not optimal on a variety of random graphs domains, and may give clues as to what results may hold and how to prove them.

Specifically, for sparse random graphs, the greedy algorithm asymptotically outperforms the randomized algorithm by a constant factor. Up to average degree 3, the modified greedy algorithm which breaks the ties among dynamic degree 2 vertices appears to be asymptotically optimal or near optimal (approximation ratio < 1.005).

For dense graphs, with fixed edge probability p, our results indicate that the greedy and randomized heuristics are asymptotically equivalent. In particular, the ratio of *Greedy* to *Random* appears to have the dependence $Ratio = 1 + c_1/\log n$. Since the asymptotics of *Random* are well known on this random graph domain, $Random = \log_{1/(1-p)} n + o(\log n)$, our results indicate that *Greedy* is asymptotically only a constant better than *Random*.

Our results indicate that for sparse graphs, the modified greedy algorithm is asymptotically superior to the randomized algorithm. However, for fixed edge probability p the greedy and randomized algorithms are

asymptotically equivalent. An interesting open question raised by our results is to determine the threshold $p(n)$ for the edge probability below which the greedy algorithm is superior to the randomized algorithm, and above which the two are equivalent.

References

1. J. Aronson, A. M. Frieze, and B. G. Pitel. Maximum matchings in sparse random graphs: Karp-Sipser re-visited. *Random Structures and Algorithms*, pages 111–178, 1998.
2. B. Bollobás. *Random Graphs*. Cambridge University Press, New York; second edition, 2001.
3. B. Bollobás and P. Erdös. Cliques in random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 80:419–427, 1976.
4. R. Boppana and M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT*, 20:180–196, 1992.
5. P. Erdös and A. Rényi. On the evolution of random graphs. *Magyar Tud. Akad. Mat. Kut. Int. Közl*, 4:17–61, 1960.
6. A. M. Frieze and B. Read. Probabilistic analysis of algorithms. *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 36–92, 1998.
7. A. M. Frieze and S. Suen. On the independence number of random cubic graphs. *Random Structures and Algorithms*, 5:649–664, 1994.
8. M. Halldórsson and J. Radhakrishnan. Greed is good: Approximating independent sets in sparce and bounded-degree graphs. *Algorithmica*, 18:145–163, 1997.
9. R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computatiolns*, pages 85–103, 1972.
10. R. M. Karp and M. Sipser. Maximum matchings in sparse graphs. *Proceedings of the 22nd Annual IEEE Symposium on Foundations of Computing*, pages 364–375, 1982.
11. C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Proceedings of 25th Annual ACM Symp. on Theory of Computing*, pages 286–293, 1993.
12. D. Matula. The largest clique size in a random graph. *Southern Methodist University, Tech. Report, CS 7608*, 1976.
13. D. B. West. *Introduction to Graph Theory –Second edition*. Prentice Hall, New York, 2001.