Approximating Independent Set in Semi-Random Graphs

Bodo Manthey [a] Kai Plociennik [b]

[a] *University of Twente, Department of Applied Mathematics*
P. O. Box 217, 7500 AE Enschede, The Netherlands

[b] *TU Chemnitz, Fakultät für Informatik*
Straße der Nationen 62, 09107 Chemnitz, Germany

Abstract

We present an algorithm for the independent set problem on semi-random graphs, which are generated as follows: An adversary chooses an n-vertex graph, and then each edge is flipped independently with a probability of $\varepsilon > 0$. Our algorithm runs in expected polynomial time and guarantees an approximation ratio of roughly $O(\sqrt{n\varepsilon})$, which beats the inapproximability bounds.

1 Introduction and Our Results

Given an undirected graph $G = (V, E)$, the goal of the maximum independent set problem (IS) is to find an *independent set* $I \subseteq V$ (i.e., no edge of E connects two vertices of I) of maximum cardinality. The size of the largest such set is G's *independence number* $\alpha(G)$. IS is NP-hard and even hard to approximate with a ratio of $O(n^{1-\delta})$ for every $\delta > 0$, where n is the number of vertices [5]. The best worst-case polynomial-time approximation algorithm for IS achieves an approximation ratio of $O(n\frac{(\log\log n)^2}{(\log n)^3})$ [2]. Often, however, approximation algorithms show a better performance than their worst-case guarantees promise. To explain the gap between worst-case and observed approximability, the average-case approximability of IS with respect to random graphs in the $G(n, p)$ model has been analyzed by Krivelevich and Vu [4]. They achieve an approximation ratio of $O(\sqrt{np}/\log n)$ in expected polynomial time. A drawback of such an average-case analysis is that it often says little about

Email addresses: `b.manthey@utwente.nl` (Bodo Manthey), `kai.plociennik@informatik.tu-chemnitz.de` (Kai Plociennik).

```
GreedyIS(G)
1: Set C_1 := {1} and χ := 1.
2: For v = 2, ..., n: If there is an i such that C_i ∪ {v} is an independent set
   of G, then C_i := C_i ∪ {v} for the smallest such i. Otherwise, create a new
   class by setting χ := χ + 1 and C_χ := {v}.
3: Choose i that maximizes |C_i|. Output I = C_i.
```

Algorithm 1: A simple greedy algorithm for independent set.

typical performance: random instances have very special properties with high probability and often do not reflect real instances. To circumvent this, graph problems have been analyzed with semi-random inputs [1,3].

In this paper, we analyze the approximability of IS for a semi-random input model: An adversary specifies a graph $G = (V, E)$. Then, a random graph $\mathcal{G} = (V, \mathcal{E})$ is produced by flipping each potential edge in G independently with a probability of $\varepsilon > 0$. More precisely, for every $e \in E$, we have $e \in \mathcal{E}$ with a probability of $p_e = 1 - \varepsilon$, while for every $e \notin E$, we have $e \in \mathcal{E}$ with a probability of $p_e = \varepsilon$. We call this distribution $\mathcal{G}(G, \varepsilon)$. In the extreme case $\varepsilon = 0$, the adversary has full power and we have $\mathcal{G} = G$. For larger values of ε, the adversary loses power. We adapt the algorithm of Krivelevich and Vu [4] to our semi-random model and show that it guarantees an approximation ratio of roughly $O(\sqrt{n\varepsilon})$ in expected polynomial time.

2 Approximating Independent Set

For simplicity, we always assume $V = \{1, \ldots, n\}$ from now on. `GreedyIS` (Algorithm 1) is a simple greedy algorithm, which outputs an independent set. `GreedyIS` is later used as a subroutine of `ApproxIS`.

Lemma 1 *Let $G = (V, E)$ be a graph, and let ε be arbitrary with $n^{-1/2} \leq \varepsilon \leq 1/2$. Let $\mathrm{gis}(G, \varepsilon) = \frac{1}{32} \cdot \min\{\frac{\ln n}{\varepsilon}, \frac{n^2 \ln n}{|E| \ln(1/\varepsilon)}\}$. Let I be the independent set of $\mathcal{G}$ computed by* ***GreedyIS****, where $\mathcal{G}$ is drawn from $\mathcal{G}(G, \varepsilon)$. Then* $\Pr[|I| < \mathrm{gis}(G, \varepsilon)] \leq e^{-n \ln n}$.

We also need an upper bound on the independence number of a graph drawn from $\mathcal{G}(G, \varepsilon)$. The proof is an adaption of Krivelevich and Vu's technique [4] to our semi-random model. For a graph G and a random graph $\mathcal{G} = (V, \mathcal{E})$ drawn from $\mathcal{G}(G, \varepsilon)$, let $A = A(\mathcal{G}, G, \varepsilon) = (a_{ij})_{1 \leq i,j \leq n}$ be the $n \times n$-matrix given by $a_{ij} = 1$ if $e = \{i, j\} \notin \mathcal{E}$ and $a_{ij} = -(1 - p_e)/p_e$ if $e = \{i, j\} \in \mathcal{E}$, where $p_e = \varepsilon$ if $e \notin E$ and $p_e = 1 - \varepsilon$ if $e \in E$. Note that A depends on G since G defines the values p_e. ¿From the results of Krivelevich and Vu [4, Lemma 2.4], we easily get $\alpha(\mathcal{G}) \leq \lambda_1(A(\mathcal{G}, G, \varepsilon))$ for any G, ε, and $\mathcal{G}$, where λ_1 denotes the largest eigenvalue of a matrix. We have to show that $\lambda_1(A(\mathcal{G}, G, \varepsilon)) =$

$\texttt{ApproxIS}(\mathcal{G} = (V, \mathcal{E}), G, \varepsilon)$

1: $I := \texttt{GreedyIS}(\mathcal{G})$. If $|I| < \mathrm{gis}(G, \varepsilon)$, go to Step 5.
2: Compute $\lambda_1(A(\mathcal{G}, G, \varepsilon))$. If $\lambda_1 < 2^8 \cdot (\log n) \cdot \sqrt{n/\varepsilon}$, then output I.
3: Compute $|\overline{N}(S')|$ for all sets $S' \subseteq V$ with $|S'| = (8 \log n)/\varepsilon$. If $|\overline{N}(S')| \leq (2 \log n) \cdot \sqrt{n/\varepsilon}$ for all such subsets S', output I.
4: Check all subsets $S'' \subseteq V$ with $|S''| = (8 \log n) \cdot \sqrt{n/\varepsilon}$. If none of them is independent, output I.
5: Try all subsets of V and output a largest independent set found.

Algorithm 2: Approximation algorithm for independent set with guaranteed approximation ratio and expected polynomial running-time.

$O((\log n) \cdot \sqrt{n/\varepsilon})$ with high probability. Krivelevich and Vu's proof [4] of the corresponding result for $G(n, p)$ graphs is based on a concentration result for the largest eigenvalue of a matrix. To apply their techniques, we first have to determine the expected value $\mathrm{E}[\lambda_1(A)]$ of the largest eigenvalue (Lemma 2). Using this, we can derive a tail bound for $\lambda_1(A)$ (Lemma 3). Together with $\alpha(\mathcal{G}) \leq \lambda_1(A)$, we then get a concentration result for the size of the largest independent set in $\mathcal{G}(G, \varepsilon)$ graphs (Theorem 4).

Lemma 2 *Fix a graph $G = (V, E)$, and let $\varepsilon = \Omega((\log n)^2/n)$, $\varepsilon \leq 1/2$. Let $A = A(\mathcal{G}, G, \varepsilon)$ for $\mathcal{G}$ drawn from $\mathcal{G}(G, \varepsilon)$. Then* $\mathrm{E}[\lambda_1(A)] \leq 2^7 (\log n)\sqrt{n/\varepsilon}$.

Lemma 3 *Under the same assumptions as in Lemma 2, we have* $\Pr[\lambda_1(A) \geq 2^8 (\log n)\sqrt{n/\varepsilon}] \leq 4 \exp(-2^9 n\varepsilon (\log n)^2)$.

Theorem 4 *Fix a graph $G = (V, E)$, and let $\varepsilon = \Omega((\log n)^2/n)$, $\varepsilon \leq 1/2$. Let $\mathcal{G}$ be a random graph drawn from $\mathcal{G}(G, \varepsilon)$. Then* $\mathrm{E}[\alpha(\mathcal{G})] \leq 2^7 \cdot (\log n) \cdot \sqrt{n/\varepsilon}$ *and* $\Pr[\alpha(\mathcal{G}) \geq 2^8 \cdot (\log n) \cdot \sqrt{n/\varepsilon}] \leq 4 \exp(-2^9 \cdot n\varepsilon \cdot (\log n)^2)$.

Our algorithm `ApproxIS` (Algorithm 2) gets the adversarial graph G, the flip probability ε, and the random graph $\mathcal{G}$ from $\mathcal{G}(G, \varepsilon)$ as input. It runs `GreedyIS` and tries to certify that this yields a good approximation. If this fails, `ApproxIS` turns to brute-force search. Let us briefly discuss the influence of G. With decreasing ε and increasing $|E|$, the graph G gains influence. This is reflected in the approximation ratio below. In Step 3, the following definition is used: For a graph $G = (V, E)$ and $S \subseteq V$, the *non-neighborhood* $\overline{N}(S)$ of S is the set of vertices not adjacent to S.

Theorem 5 *Fix a graph $G = (V, E)$ and a flip probability $n^{-1/2} \leq \varepsilon \leq 1/2$. Let $\mathcal{G}$ be drawn from $\mathcal{G}(G, \varepsilon)$. Then $\texttt{ApproxIS}(\mathcal{G}, G, \varepsilon)$ has polynomial expected running time. If ε is sufficiently large, i.e., $\frac{\ln(1/\varepsilon)}{\varepsilon} \leq \frac{n^2}{|E|}$, it guarantees an approximation ratio of $O(\sqrt{n\varepsilon})$. Otherwise, it guarantees an approximation ratio of $O\Big(\frac{|E| \log(1/\varepsilon)}{n^{3/2}\sqrt{\varepsilon}}\Big)$.*

`GreedyIS` alone is an algorithm with worst-case polynomial running-time, but the approximation ratio then holds only in expectation and with high probability. This complements the performance of `ApproxIS`. The analysis follows from Lemma 1 and Theorem 4.

Corollary 6 *Let $G = (V, E)$ be a graph, and let $n^{-1/2} \leq \varepsilon \leq 1/2$. Then* `GreedyIS` *achieves an expected approximation ratio of* $O\Big(\frac{\sqrt{n/\varepsilon}}{\min\{1/\varepsilon, n^2/(|E| \ln(1/\varepsilon))\}}\Big)$ *on graphs drawn from* $\mathcal{G}(G, \varepsilon)$*. If* $\frac{\ln(1/\varepsilon)}{\varepsilon} \leq \frac{n^2}{|E|}$*, i.e.,* ε *is large enough, this simplifies to* $O(\sqrt{n\varepsilon})$*. The approximation ratio is not only achieved in expectation, but also with a probability of at least* $1 - \exp(-\Omega(n\varepsilon(\log n)^2))$*.*

3 Discussion

We have analyzed the approximability of IS in a semi-random input model. We have presented an approximation algorithm that guarantees an approximation ratio of roughly $O(\sqrt{n\varepsilon})$ in expected polynomial time. A simple greedy algorithm (Algorithm 1) alone achieves an expected approximation ratio of $O(\sqrt{n\varepsilon})$ in worst-case polynomial time. A subtlety of our Algorithm 2 is that it needs the original graph as an additional input. We believe that avoiding this requires new techniques if it can be avoided at all. The reason is that our goal was a *guaranteed approximation ratio*. To achieve this, the algorithm has to know when to switch to brute-force search in order to maintain also an expected polynomial running-time. `GreedyIS` alone, which needs neither the original graph nor ε, has complementary properties: worst-case polynomial running-time but the approximation ratio is only achieved in expectation.

References

[1] Amin Coja-Oghlan. Finding large independent sets in polynomial expected time. *Combin. Probab. Comput.*, 15(5):731–751, 2006.

[2] Uriel Feige. Approximating maximum clique by removing subgraphs. *SIAM J. Discrete Math.*, 18(2):219–225, 2004.

[3] Uriel Feige and Joe Kilian. Heuristics for semirandom graph problems. *J. Comput. System Sci.*, 63(4):639–671, 2001.

[4] Michael Krivelevich and Van H. Vu. Approximating the independence number and the chromatic number in expected polynomial time. *J. Comb. Optim.*, 6(2):143–155, 2002.

[5] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007.