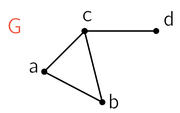
Approximation Algorithms in Semi-Random Models

Theo McKenzie {mentor}, Hermish Mehta {mentee}, Luca Trevisan
Department of Mathematics, University of California, Berkeley: Berkeley, CA

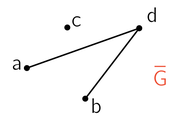
graph theory

A *graph* represents a way to model relationships or connections within a network. Abstractly, a graph consists *vertices* or *nodes* connected by *edges*. Formally, we express a graph $G = (V, E)$ as a set of vertices V and a set of edges $E \subseteq V \times V$, containing pairs of vertices.



An *clique* in a graph is a subset of vertices $S \subseteq V$ where every pair of vertices is connected by an edge. $\{a, b, c\}$ is clearly the largest clique for the graph above, but finding maximum cliques in arbitrary graphs is NP-hard, meaning no efficient algorithms are known.

This problem is identical to finding *independent sets* in graphs, subsets of vertices with no edges between them. Taking the *complement* of the graph, swapping which vertices are connected, reveals the connection between the two problems.



Any clique in a graph is a corresponding independent set in the complement, so now $\{a, b, c\}$ forms the largest independent set. Analogously, finding the maximum independent set in a graph's complement immediately yields the largest clique.

semi-definite programming

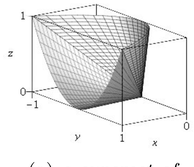
A common technique in approximation algorithms involves constructing *semi-definite programs*, which are optimization problems of the form

$$\min_{\mathbf{x}_i \in \mathbb{R}^n} \sum_{1 \le i,j \le n} C_{ij} \langle \mathbf{x}_i, \mathbf{x}_j \rangle.$$

These vectors are further subject to some additional linear constraints, and the *objective function* above is minimized only with respect to the *feasible* vector assignments which satisfy the constraints.

$$\sum_{1 \le i,j \le n} A_{ij}^{(k)} \langle \mathbf{x}_i, \mathbf{x}_j \rangle \le b^{(k)}$$

The *ellipsoid method* gives an efficient algorithm to find an optimum for any specific objective function C and constraints $A^{(1)}, \ldots, A^{(m)}$ with bounds $\mathbf{b}$. Over the feasible region, the algorithm iteratively shrinks an ellipsoid containing the optimum, converging on the solution.



(a) *a superset of the feasible region*

problem

Problem (independent set). given a graph $G = (V, E)$ find the size of the largest independent set in G.

The maximum independent set problem is a classic graph problem believed to have no polynomial-time algorithm (1). More recent hardness of approximation results show that no efficient algorithm exists to even approximate this within a polynomial factor in the worst-case, unless P = NP (2).

However, real-world instances often differ from pathological worst-case examples, so these results motivate average-case analysis over *random graphs*. However, purely probabilistic models rarely reflect in real instances, so we consider *semi-random graphs*, generated by a mix of random and arbitrary or adversarial choices, accounting for this difference.

Our algorithm to find cliques semi-random instances uses *crude semi-definite programming*, a technique introduced by Kolla, Makarychev, Makarychev (4) and later Makarychev, Makarychev, Vijayaraghavan (5). Here, we carefully design a semi-definite program which reveals information about the planted solution.

main result

Theorem 1. When $k \ge cn^{2/3}$ for some constant c, we can recover an independent set of size at least k from a semi-random graph in polynomial time; furthermore, we can return at most n candidate solutions such that one is the original independent set, both with high probability.

Proof. We provide a deterministic polynomial time algorithm for the recovery problem.

1. Solve the following crude semi-definite programming to obtain a vector assignment of the vertices, $\{u^*\}$.

$$\begin{aligned} \min \sum_{u,v \in V} ||x_u - x_v||^2 \\ \forall u \in V: \ ||x_u||^2 = 1 \\ \forall u, v \in V: \ \langle x_u, x_v \rangle \ge 0 \\ \forall (u,v) \in E: \ \langle x_u, x_v \rangle = 0 \end{aligned}$$

2. For each vertex, take the ball of radius one around its vector and list all vertices with vectors inside this ball.

3. To each list, add all vertices independent to those already on the list. Return all such lists created.

Applying Markov's inequality to Lemma 3, there must be some vertex $u \in S$ such that the probability a vertex in S is greater than least distance 1 is at most $c_1 k^{-3/2} n$. Hence, if $k \ge c_2 n^{2/3}$ for sufficiently large c_2, $\frac{1}{3}$ of vertices in S will lie in this ball.

The probability some outside vertex $v \in \overline{S}$ shares no edges with these vertices is $2^{-k/3}$, so no such vertex will exist with high probability, using a union bound. Therefore, adding additional independent vertices will add only the remaining vertices in S, recovering the original independent set.

semi-random model

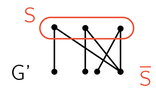
We consider a *planted semi-random model* for independent set first described by Feige and Killian (3).

1. An adversary chooses size-k set $S \subseteq V$, $|S| = k$.

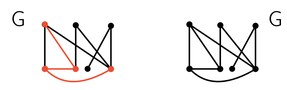
2. Create a random graph $G' = (V, E')$ where each pair of vertices u, v has edge with probability $P(u, v)$, where

$$P(u,v) = \begin{cases} 0 & : (u,v) \in S \times S \\ 0.5 & : (u,v) \in S \times \overline{S} \\ 0 & : (u,v) \in \overline{S} \times \overline{S} \end{cases}$$

3. The adversary can add any edge (u, v) not within S, creating a graph $G = (V, E)$ containing independent set S with at least the random edges E' on its boundary.



(a) *random edges between S and $\overline{S}$*



(b) *adversarial edges added, giving G*

analysis

Lemma 1. For the optimal solution to the crude SDP,

$$\sum_{u,v \in S \times S} ||u^* - v^*||^2 + 2 \sum_{u,v \in S \times \overline{S}} ||u^* - v^*||^2 \le 4k(n-k)$$

Proof Sketch. Consider a feasible solution constructed by taking the optimal solution and assigning all vertices in S to the same vector orthogonal the remaining solution. Setting the objective value of this solution greater than the optimal and rearranging immediately yields this inequality.

Lemma 2. With high probability over E',

$$\max_{\substack{x_1,\ldots,x_n \\ ||x_u||=1, \\ \forall u \in V}} \left| \sum_{\substack{(u,v) \in E' \\ u \in S, v \in \overline{S}}} ||x_u - x_v||^2 - \frac{1}{2} \sum_{u \in S, v \in \overline{S}} ||x_u - x_v||^2 \right| = O(k\sqrt{n})$$

Proof Sketch. Expanding the above expression, denoted D,

$$D \le \left| k(n-k) - 2|E'| \right| + \max_{\substack{x_1,\ldots,x_n \\ ||x_u||=1, \\ \forall u \in V}} \left| \sum_{u \in S, v \in \overline{S}} (1 - 2A_{uv}) \langle x_u, x_v \rangle \right|$$

From a Chernoff bound, the first term is $o(k\sqrt{n})$. To bound the second term, we apply Grothendiek's inequality to instead consider a maximum over $2n$ ± 1 variables. This reduces this expression to a sum of ± 1 Bernoulli random variables over the boundary of S, where a Chernoff bound followed by a union bound over all 2^{2n} choices of variables gives the desired inequality with high probability. This implies the second term in Lemma 1 is within $O(k\sqrt{n})$ of $4k(n-k)$, giving Lemma 3.

Lemma 3. For the optimal solution to the crude SDP, taking a uniform expectation, $\mathbb{E}_{u,v \in S}(||u^* - v^*||^2) = O(nk^{-3/2})$.

example

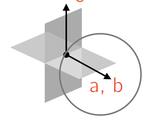
We illustrate the algorithm on an extremely simple graph drawn from the semi-random distribution. Although we proved only asymptotic guarantees, this highlights the method which can be generalized to larger graphs.

First, we construct the crude semi-definite program, dividing the objective function by 2. The number of constraints should be roughly cubic in the number of vertices, from the Triangle inequality constraints.

$$\begin{aligned} \min \ ||x_a - x_b||^2 + ||x_b - x_c||^2 + ||x_a - x_c||^2 \\ ||x_a|| = 1, \ ||x_b|| = 1, \ ||x_c|| = 1 \\ \langle x_a, x_b \rangle \ge 0, \ \langle x_b, x_c \rangle \ge 0, \ \langle x_a, x_c \rangle \ge 0 \\ \langle x_a, x_b \rangle = 0, \ \langle x_b, x_c \rangle = 0 \end{aligned}$$

The ellipsoid method guarantees finding an optimum to within arbitrary precision in polynomial time. Applying this algorithm yields a vector assignment of the vertices like the one below.

$$\begin{cases} x_a = (1,0,0) \\ x_b = (1,0,0) \\ x_c = (0,1,0) \end{cases}$$



Here, the optimum is achieved for any assignment where a and b correspond to the same vector orthogonal to c. Drawing balls around each vector, we immediately get the lists $\{a, b\}, \{a, b\}, \{c\}$. No additional vectors can be added to each of these lists so we can simply return these as our candidate solution: one of these is the large planted independent set.

The intuition behind our proof is to show for sufficiently large graphs, regardless of adversarial choices, the vectors in S will similarly be clustered together or close to one another, so a large proportion will be a ball around some vertex. The remaining vertices can then be added with a single pass.

references

(1) R. M. Karp. Reducibility among combinatorial problems, In *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher (eds.), Plenum Press, New York, 1972, pp. 85-103.

(2) D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, pp. 681-690, 2006.

(3) U. Feige and J. Kilian. Heuristics for semirandom graph problems. *Journal of Computer and System Sciences, 63*(4) pp. 639-671, 2001.

(4) A. Kolla, K. Makarychev, and K. Makarychev. How to play unique games against a semi-random adversary. In *Proceedings of the Fifty-Second Annual IEEE Symposium on Foundations of Computer Science*. pp. 443-452, 2011.

(5) K. Makarychev, Y. Makarychev, and A. Vijayaraghavan. Approximation algorithms for semi-random graph partitioning problems. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, pp. 367-384, 2012.