Yet Another Algorithm for Dense Max Cut: Go Greedy

Claire Mathieu* Warren Schudy†

Abstract

We study dense instances of MaxCut and its generalizations. Following a long list of existing, diverse and often sophisticated approximation schemes, we propose taking the naïve greedy approach; we prove that when the vertices are considered in random order, our algorithms are still approximation schemes. Our algorithms may be simple, but the analysis is not. It relies on smoothing the vertices defining the partial cuts and on proving certain martingale properties. We also give a simple proof that dense problems have sample complexity $O(1/\epsilon^4)$, which improves the result of Alon, Fernandez de la Vega, Kannan, and Karpinski [1] by a $\log(1/\epsilon)$ factor. Like previous work, our results generalize to dense maximum constraint satisfaction problems.

1 Introduction and main results

The MaxCut problem strives to partition the vertices of an input graph into two parts to maximize the number of edges crossing the cut. It has been the object of intense research. The thread that concerns us in this paper deals with designing approximation schemes for dense graph instances of MaxCut, and some generalizations, see for example [1, 6, 3, 4, 5, 2]. There exist many algorithms, each with its own unique strong points, based on a combinatorial approach, or on spectral techniques, or on smoothed linear programming relaxations. This suggests that the problem is "easy" in some sense, and thus simple algorithms have a chance to be successful as well. Indeed, the combinatorial approach of [6] can be interpreted as some form of a "batched greedy" algorithm. Hence the question of whether a simple greedy algorithm would be sufficient.

The greedy algorithm for MaxCut considers vertices one by one in arbitrary order and places each of them on the left or right side of the cut, depending on the number of neighbors that are already placed on each side (breaking ties arbitrarily). It is well known that the greedy algorithm is a 2-approximation, and that this bound is tight if an adversary controls the input graph and the order in which the vertices are considered.

Here, we take advantage of the power of randomness by considering vertices *in random order*. We define three variants of the greedy algorithm for MaxCut. The first algorithm is closest to the "exhaustive search" techniques of previous papers, hence easiest to analyze.

Algorithm 1 Simplest-to-analyze greedy algorithm

- Take a sample S of $t_0 = 1/\epsilon^2$ vertices chosen uniformly at random without replacement.
- For each of the 2^{t_0} possible cuts of S:
  For each vertex v of $V \setminus S$ in random order,
  Place v on the side that maximizes the number of resulting crossing edges.
- Output the best cut found.

Algorithm 2 Fastest greedy algorithm

- Take a sample S of $t_1 = O(1/\epsilon^4)$ vertices chosen uniformly at random
- Find a near-optimal cut in the problem induced by the input on S, using Algorithm 1.
- For each vertex v of $V \setminus S$ in random order,
  Place v on the side that maximizes the number of resulting crossing edges.

THEOREM 1.1. *For any $\epsilon > 0$, Algorithm 1 has running time $O(n^2)2^{O(1/\epsilon^2)}$ and the expected value of the output is at least $OPT - O(\epsilon n^2)$.*

Thus, for dense instances, Algorithm 1 is a polynomial-time approximation scheme.

The second algorithm is basically the greedy algorithm with random order, except that we start with a good "hint" of a near-optimal solution, in the form of a near optimal solution in a constant size sample. Note that the running time separates into one term depending on ϵ but not on n (for solving the problem in the sample) and the other term depending on n but not on ϵ (for running the greedy algorithm on the rest of the graph.) Thus it is the fastest of our variants.

*Brown University – Computer Science. Part of this work was done while the first author was visiting Microsoft Research.

†Also Brown CS. Email: ws at cs.brown.edu

Algorithm 3 Most naïve greedy algorithm

- Repeat $2^{2^{\tilde{O}(poly(\epsilon))}}$ times
  For each vertex v of V in random order,
  Place v on the side that maximizes the number of resulting crossing edges.
- Output the best cut found

THEOREM 1.2. *For any $\epsilon > 0$, Algorithm 2 has running time $n^2 + 2^{O(1/\epsilon^2)}$ and the expected value of the output is at least $OPT - O(\epsilon n^2)$.*

Note that the use of Algorithm 1 in Algorithm 2 rather than an arbitrary approximation algorithm is crucial for our proof to work with a sample of size $O(1/\epsilon^4)$.

The third algorithm eliminates the preliminary random sample phase altogether; it is just the naïve greedy algorithm with random order, but it is repeated enough times to guarantee near-optimality. It is slowest but simplest in design and does not even really require prior knowledge of ϵ: one could just run it until satisfied with the quality of the current cut, with the reassuring knowledge that it will converge to a near-optimal value in polynomial time.

THEOREM 1.3. *For any $\epsilon > 0$, the expected value of the output of Algorithm 3 is at least $OPT - \epsilon n^r$.*

The proof of Theorem 1.3, omitted in this extended abstract, is a technical but relatively straightforward use of the cut norm results from [1], and a variant of the analysis of Algorithm 2.

With these results and techniques at hand, we can give a new and simple proof of the following result. The sample complexity is the number of vertices which must be looked at in order to reliably estimate the value of the maximum cut of the input graph.

THEOREM 1.4. *MaxCut has sample complexity $t_1 = O(1/\epsilon^4)$ in the following sense: if S is a random sample of t_1 vertices chosen uniformly at random, and OPT_S is the value of the maximum cut of the graph induced by G on S, then $\mathbf{E}[OPT_S]/t_1^2 = OPT/n^2 \pm O(\epsilon)$.*

Our proof of Theorem 1.4 improves the result of Alon, Fernandez de la Vega, Kannan and Karpinski [1] by a $\log(1/\epsilon)$ factor.

We can extend Theorems 1.1, 1.2 and 1.4 to dense Max-r-CSP problems (see extended version for details.) It should also be possible to extend these theorems to Chernoff-like tail bounds as well.

We analyze Algorithm 1 in §2 and Algorithm 2 in §3. We prove Theorem 1.4 in §4.

Although we find such algorithms appealingly simple, the difficulty of the problem has not entirely disappeared. In this paper, it has merely been shifted from the algorithmic design side to the algorithmic analysis side, and thus the proof is quite involved.

The key difficulty with the proof is viewing *the sequence of vertex placements chosen by the algorithm as a sample of something* at every timestep. Unfortunately, in our algorithms the initial seed sample is soon drowned out by the later greedily added vertices, so we need to interpret the greedily added vertices as a sample of something. Our solution is to define, at every time t, a fictitious cut which is a fractional extension of the current cut, and in which every vertex v not yet placed by the algorithm is placed randomly (i.e. fractionally) by considering how v would have been placed if it had been selected for placement at a random time $\tau < t$. This is quite different from the choice made in [6], in which the analog of the fictitious cut is obtained by placing vertices not yet considered by their algorithm according to the optimal cut. We use martingale arguments to show that the actual partial cut and its fictitious extension behave similarly, and therefore the greedy decisions in the algorithm do not hurt the cost of the fictitious cut much.

2 Analyzing Algorithm 1

2.1 Definitions and notations

The problem. Each vertex can take on 2 possible values $i \in [1,2]$, corresponding to the two sides of the cut. The goal is to find an cut of the vertices, among the 2^n possible cuts, so as to maximize the number of crossing edges. We will actually view the problem as trying to *minimize* the number of *non-crossing* edges. We describe a cut by an $2n$-dimensional vector x, where $x_{ui} \in \{0,1\}$ equals 1 if and only if the variable associated to vertex u has value equal to i. We can then write the objective function z that needs to be minimized as:

$$z(x) = \sum_{\substack{1 \le u_1, u_2 \le n, \\ 1 \le i_1, i_2 \le k}} a_{u_1,i_1,u_2,i_2} x_{u_1 i_1} x_{u_2 i_2}, \qquad (2.1)$$

where $A = (a_{u_1,i_1,u_2,i_2})$ is an $2n$-dimensional array (a matrix), symmetric under permutation of the 2 indices (u_j, i_j)'s, such that $a_{u_1 i_1 u_2 i_2} = 1/2$ whenever $i_1 = i_2$ and $\{u_1, u_2\} \in E$ and 0 otherwise. It will also be useful to write $z(x) = A(x,x)$, using the bilinear function

$$A(x^{(1)}, x^{(2)}) = \sum_{\substack{1 \le u_1, u_2 \le n, \\ 1 \le i_1, i_2 \le 2}} a_{u_1,i_1,u_2,i_2} x^{(1)}_{u_1 i_1} x^{(2)}_{u_2 i_2}. \qquad (2.2)$$

The algorithm. The analysis will focus on the run when the sample S is assigned in the same way as

OPT. For the sake of analysis pretend that the sample is chosen one vertex at a time rather than batched. Let "time t" denote the instant after the first t vertices are considered by the algorithm. At any time $t \in [0, n]$, Algorithm 1 defines a partial cut x^t:

$$x^t_{ui} = \begin{cases} 1 & \text{if vertex } u \text{ has been given value } i \text{ by time } t \\ 0 & \text{if vertex } u \text{ has been given some value} \neq i \\ & \text{by time } t \\ 0 & \text{if vertex } u \text{ has not yet been given a value} \\ & \text{by time } t \end{cases}$$

Thus x^0 is uniformly equal to 0, and x^n describes the output of the algorithm. We find it convenient to define x^t_v to be a 2-dimensional vector with components x^t_{vi}, and similarly for the other $2n$-dimensional vectors.

Let r_t denote the t^{th} vertex considered by the algorithm. Let $x^* = (x^*_{ui})$ be the optimal cut. In the case when $t \leq t_0$, we are in the initial phase and the vertex $u = r_t$ considered by the algorithm at time t is placed according to the optimal cut, so that the 2-dimensional vector describing the placement of vertex u is $x^t_u = x^*_u$. In the case when $t > t_0$, the algorithm decides in a greedy fashion: let $b(x)$ be the $2n$-dimensional vector of partial derivatives of $z(x)$; since z is bilinear and $A(x^t - x^{t-1},\ x^t - x^{t-1}) = 0$, $b_{ui}(x)$ is the increase of $z(x)$ when we increase x_{ui} by 1. Then, by definition of greedy, $x^t_{r_t} = x^{t-1}_{r_t} + g^t_{r_t}$, where

$$g^t_{vi} = \begin{cases} x^*_{vi} & \text{if } t \leq t_0 \\ 1 & \text{if } t > t_0 \text{ and } i = \arg\min_j b_{vj}(x^{t-1}) \\ 0 & \text{otherwise} \end{cases}$$

Our analysis computes x^t_u and $b_u(x^t)$ inductively.

The fictitious cut. Instead of analyzing x^t directly, the analysis will instead focus on the following auxiliary variables. Let $S^t = \{r_1, r_2 \cdots r_t\}$ be the vertices placed up to time t. We extrapolate the partial cut x^t into a vector $\hat{x}^t$ that is a complete (fractional) cut by examining, for each $v \notin S^t$, how v would have been placed if it had been placed at time $\tau \leq t$, and taking the average over all past times τ:

$$\hat{x}^t_v = \begin{cases} x^t_v & \text{if } v \in S^t \\ (1/t)\sum_{\tau=1}^t g^\tau_v & \text{if } v \notin S^t. \end{cases}$$

2.2 Proof of Theorem 1.1 In the first lemma, we bound the increase in the value of the fictitious cut when going from time $t-1$ to time t, and the bound uses the derivative of the objective function.

LEMMA 2.1. *For every t, we have $z(\hat{x}^t) - z(\hat{x}^{t-1}) \leq (\hat{x}^t - \hat{x}^{t-1}) \cdot b(\hat{x}^{t-1}) + 4n^2/t^2$.*

Proof. Note that $b(x)_{u_1 i_1} = 2\sum_{u_2, i_2} a_{u_1 i_1 u_2 i_2} x_{u_2 i_2}$, so that $A(x^{(1)},\ x^{(2)}) = (1/2)x^{(1)} \cdot b(x^{(2)})$. Let $\hat{s}^t = \hat{x}^t - \hat{x}^{t-1}$. By multilinearity and symmetry of A,

$$\begin{aligned} z(\hat{x}^t) &= A(\hat{x}^{t-1} + \hat{s}^t,\ \hat{x}^{t-1} + \hat{s}^t) \\ &= A(\hat{x}^{t-1},\ \hat{x}^{t-1}) + \hat{s}^t \cdot b(\hat{x}^{t-1}) + A(\hat{s}^t,\ \hat{s}^t). \end{aligned}$$

A short calculation proves the following expression.

$$\hat{s}^t_u = \left(g^t_u - \hat{x}^{t-1}_u\right) \begin{cases} 1 & \text{if } u = r_t \text{ is being} \\ & \text{placed at time } t \\ (1/t) & \text{if } u \notin S^t \text{ has} \\ & \text{not yet been placed} \\ 0 & \text{if } u \in S^{t-1} \text{ has} \\ & \text{already been placed} \end{cases} \tag{2.3}$$

Since both g^t_u and $\hat{x}^{t-1}_u$ have ℓ_1 norm equal to 1, we obtain that the ℓ_1-norm of $\hat{s}^t$ is $\sum_u |\hat{s}^t_u| \leq 2 + \frac{2}{t}(n-t) = 2\frac{n}{t}$. Then

$$|A(\hat{s}^t, \hat{s}^t)|_1 \leq |A|_\infty |\hat{s}^t|_1^2 \leq 4\frac{n^2}{t^2}.$$

Replacing $A(\hat{x}^{t-1}, \hat{x}^{t-1})$ by $z(\hat{x}^{t-1})$ concludes the proof.

In the second lemma, we bound the expectation of the increase, and relate it to the difference between the values of the fictitious cut and of the (scaled) true cut, when evaluated by the derivative function b.

LEMMA 2.2. *For every t, the expected value of $z(\hat{x}^t) - z(\hat{x}^{t-1})$ is less than or equal to*

$$4n^2/t^2 + 2\frac{n}{t(n-t+1)}\mathbf{E}\left[|b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1})|_1\right].$$

Proof. First apply Lemma 2.1. Then, write:

$$\begin{aligned} &\hat{s}^t \cdot b(\hat{x}^{t-1}) \\ &= \hat{s}^t \cdot b(\frac{n}{t}x^{t-1}) + \hat{s}^t \cdot (b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1})) \\ &= (\frac{n}{t})\hat{s}^t \cdot b(x^{t-1}) + \hat{s}^t \cdot (b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1})) \\ &\leq \hat{s}^t \cdot (b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1})), \end{aligned}$$

where the equality follows from the fact that the function $b(y)$ is linear. To justify the last inequality, recall that the greedy choice g^t_u minimizes $y \cdot b_u(x^{t-1})$ for y with $|y|_1 = 1$ and $y_{ui} \in [0, 1]$. Going back to (2.3), we see that $\hat{s}^t \cdot b(x^{t-1})$ is less than or equal to 0. This is where the greedy definition of the algorithm comes into the analysis.

Given the history S^{t-1}, we can write:

$$\begin{aligned}
&\mathbf{E}\left[\hat{s}^t \cdot (b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1}))|S^{t-1}\right] \\
&= \sum_v \mathbf{E}\left[\hat{s}^t_v \cdot (b_v(\hat{x}^{t-1}) - b_v(\frac{n}{t}x^{t-1}))|S^{t-1}\right] \\
&= \sum_v \mathbf{E}\left[\hat{s}^t_v|S^{t-1}\right] \cdot (b_v(\hat{x}^{t-1}) - b_v(\frac{n}{t}x^{t-1})) \\
&\leq \sum_v |\mathbf{E}\left[\hat{s}^t_v|S^{t-1}\right]|_1 \cdot |b_v(\hat{x}^{t-1}) - b_v(\frac{n}{t}x^{t-1})|_1 \\
&\leq 2\frac{n}{t(n-t+1)}|b(\hat{x}^{t-1}) - b(\frac{n}{t}x^{t-1})|_1,
\end{aligned}$$

where the second equality follows from the fact that given S^{t-1}, $b(\hat{x}^{t-1})$ is fixed, the next inequality follows from general principles. As for the last inequality, fix S^{t-1} and a vertex u and consider evaluating $\mathbf{E}\left[\hat{s}^t_u|S^{t-1}\right]$. If $u \in S^{t-1}$, then $\hat{s}^t_u = 0$ and so the expectation is zero. Otherwise, by the random order assumption,

$$\mathbf{E}\left[|\hat{s}^t_u|_1|S^{t-1}\right] = \mathbf{E}\left[\frac{|\hat{s}^t|_1}{n-t+1}|S^{t-1}\right] \leq 2\frac{n}{t(n-t+1)}.$$

and the last inequality follows. This concludes the proof of Lemma 2.2.

The next lemma bounds the difference between the b-values of the fictitious cut and of the (scaled) true cut.

LEMMA 2.3. *For every t, we have $\mathbf{E}\left[|b(\hat{x}^t) - b(\frac{n}{t}x^t)|_1\right] = O(\sigma)$, where $\sigma = O\left(\frac{n^2}{\sqrt{t}}\sqrt{\frac{n-t}{n}}\right)$.*

The proof is based on a certain martingale property and deferred to the next section.

The next lemma relates the z-values of the fictitious cut at two different times.

LEMMA 2.4. *For every $t \geq t_0$ we have $\mathbf{E}\left[z(\hat{x}^t)\right] - \mathbf{E}\left[z(\hat{x}^{t_0})\right] = O(\epsilon n^2)$.*

Proof. Use Lemma 2.2 for every τ from t_0 to t, and sum; apply Lemma 2.3 to each term in the sum, and use $t_0 \geq 1/\epsilon^2$: after a short calculation we get:

$$\mathbf{E}\left[z(x^t) - z(\hat{x}^{t_0})\right] = n^2 O\left(\frac{1}{t_0} + \frac{1}{\sqrt{t_0}} + \frac{1}{\sqrt{t}}\right) = O(\epsilon n^2).$$

Proof. (of Theorem 1.1.) By definition of the fictitious cut, the output cut x^n is equal to $\hat{x}^n$. Apply Lemma 2.4 for $t = n$; and recall that by definition of x^*, $\hat{x}^{t_0}$ is the optimal cut.

2.3 Proof of Lemma 2.3 using martingales. In this section, we detail the proof of Lemma 2.3. Fix v. The following lemma is simple, yet central. This martingale is the key to all the future applications of the Azuma-Hoeffding inequality.

LEMMA 2.5. *$Z^t_v = \frac{t}{n-t}(\hat{x}^t_v - (n/t)x^t_v)$ is a martingale.*

Proof. If $v \in S^{t-1}$ then $\hat{x}^t_v = x^t_v = \hat{x}^{t-1}_v = x^{t-1}_v$. Let x be their common value. Then

$$\frac{t}{n-t}(\hat{x}^t_v - (n/t)x^t_v) = \frac{t-1}{n-t+1}(\hat{x}^{t-1}_v - (n/(t-1))x^{t-1}_v)$$

are both equal to $-x$ and the martingale statement holds in that case. If $v \notin S^{t-1}$, then $x^{t-1}_v = 0$; by the random order, with probability $1/(n-t+1)$ we have $v = r_t$ and $x^t_v = \hat{x}^t_v = g^t_v$; with the remaining probability $(n-t)/(n-t+1$, we have $x^t_v = 0$ and $\hat{x}^t_v = ((t-1)/t)\hat{x}^{t-1}_v + (1/t)g^t_v$, and so:

$$\begin{aligned}
&\mathbf{E}\left[\frac{t}{n-t}(\hat{x}^t_v - (n/t)x^t_v)|S^{t-1}\right] = \\
&\quad \frac{1}{n-t+1}(-g^t_v) + \\
&\quad \frac{n-t}{n-t+1} \cdot \frac{t}{n-t}\left[\frac{t-1}{t}\hat{x}^{t-1}_v + \frac{1}{t}g^t_v\right] \\
&= \frac{t-1}{n-t+1}\hat{x}^{t-1}_v,
\end{aligned}$$

which concludes the proof since $x^{t-1}_v = 0$ in that case.

LEMMA 2.6. *$B^t_{vi} = \frac{t}{n-t}(b_{vi}(\hat{x}^t) - b_{vi}(\frac{n}{t}x^t))$ is a martingale, with stepsize bounded by $4n/(n-t)$.*

Proof. The martingale statement follows from Lemma 2.5 by linearity: $b_{vi}(x) = \sum_{uj}\alpha_{uj}x_{uj}$. To bound its step-size, we note that $\alpha = \max|\alpha_{uj}| \leq 2$ (since $|A|_\infty \leq 1$), so that the step size is at most:

$$\left|\sum_{uj}\alpha_{uj}Z^t_{uj} - \sum_{uj}\alpha_{uj}Z^{t-1}_{uj}\right| \leq \alpha\sum_u |Z^t_u - Z^{t-1}_u|_1.$$

When $u \in S^{t-1}$, we have $|Z^t_u - Z^{t-1}_u|_1 = 0$. When $u = r_t$, we have

$$|Z^t_u - Z^{t-1}_u|_1 = \left|\frac{-g^t_v}{n-t+1} - \frac{(t-1)\hat{x}^{t-1}_v}{n-t+1}\right|_1 \leq \frac{t}{n-t+1}.$$

When $u \notin S^t$, we have

$$\begin{aligned}
|Z^t_u - Z^{t-1}_u|_1 &= \left|\frac{(t-1)\hat{x}^{t-1}_u + g^t_u}{n-t} - \frac{t-1}{n-t+1}\hat{x}^{t-1}_u\right|_1 \\
&\leq \frac{t-1}{(n-t)(n-t+1)} + \frac{1}{n-t} \\
&= \frac{n}{(n-t)(n-t+1)}.
\end{aligned}$$

Summing, the stepsize is at most $\alpha((t+n)/(n-t+1)) \leq 4n/(n-t)$.

We recall the Azuma-Hoeffding inequality.

THEOREM 2.1. *[Azuma-Hoeffding] Let $X_0, X_1, \ldots, X_t$ be a martingale such that $|X_k - X_{k-1}| \leq c_k$ for all k. Then, for all $\lambda > 0$,*

$$\Pr(|X_t - X_0| \geq \lambda) \leq 2e^{-\lambda^2/(2\sum_{k=1}^{t} c_k^2)}.$$

Proof. (of Lemma 2.3) From Lemma 2.6, $|b(\hat{x}^t) - b(\frac{n}{t}x^t)|_1$ is also a martingale with stepsize at most $4n/(n-t)$. Apply the Azuma-Hoeffding inequality:

$$\Pr\left(\left|b(\hat{x}^t) - b(\frac{n}{t}x^t)\right|_1 \geq \lambda\right) \leq e^{-\lambda^2/\sigma^2} \tag{2.4}$$

where $\sigma = O\left(\frac{n^2}{\sqrt{t}}\sqrt{\frac{n-t}{n}}\right)$. Finally, integrate over λ, using the fact that for a non-negative random variable X, $\mathbf{E}[X] = \int_{\lambda=0}^{\infty} \Pr(X \geq \lambda)\, d\lambda$.

3 Analyzing Algorithm 2

Instead of Algorithm 2, we will analyze a variant, Algorithm 4.

LEMMA 3.1. *With probability at least $1-\epsilon$, Algorithm 4 and Algorithm 2 give the same output.*

Proof. Consider a coupon collection problem with $N = 2^{t_0}$ coupons, one per cut of the first t_0 variables. Let $\kappa = N\ln(N/\epsilon) = 2^{t_0}(t_0 \ln 2 + \ln(1/\epsilon)) = 2^{O(t_0)}$ be the number of trials, one per cut in Y (excluding x^*). As long as Algorithm 4 collects all the coupons, it iterates over the same cuts of T as Algorithm 2 and hence returns the same result. Each coupon has probability $(1-1/N)^\kappa \leq e^{-\kappa/N} = \frac{\epsilon}{N}$ of not being collected, so a union bound shows that all are collected with probability at least $1-\epsilon$.

In the analysis of the previous section, the construction always started by a cut induced on the sample by the optimal cut x^*. Now we need a new notation: let $x^t_{(y)}$ denote the construction at time t, starting from the cut induced on the sample by the cut y (instead of by the optimal cut x^*). Let $\hat{x}^t_{(y)}$ denote the fictitious cut at time t, starting from cut y. Formally, replace x^* in the definition of g^t_{vi} with y.

LEMMA 3.2.

$$\mathbf{E}\left[\max_{y\in Y}\left|b(\frac{n}{t}x^t_{(y)}) - b(\hat{x}^t_{(y)})\right|_1\right] = \frac{1}{\epsilon}O(\sigma)$$

where $\sigma = O\left(\frac{n^2}{\sqrt{t}}\sqrt{\frac{n-t}{n}}\right)$.

Proof. Similarly to (2.4), we have for every $y \in Y$:

$$\Pr\left(\left|b(\hat{x}^t_{(y)}) - b(\frac{n}{t}x^t_{(y)})\right|_1 \geq \lambda\right) \leq e^{-\lambda^2/\sigma^2}. \tag{3.5}$$

Algorithm 4 Variant algorithm for analysis

- Take a sample S of $t_1 = O(1/\epsilon^4)$ vertices chosen uniformly at random; Take a sample T of $t_0 = O(1/\epsilon^2)$ vertices of S chosen uniformly at random
- Let Y be a set of $2^{O(t_0)}$ random cuts of the n vertices, plus the optimal cut x^*.
- For each cut of T induced by cuts in Y:
  For each vertex v of $S \setminus T$ in random order,
  Place v on the side that maximizes the number of resulting crossing edges.
- Let $w \in Y$ be the cut yielding the best cut of S.
- For each vertex v of $V \setminus S$ in random order,
  Place v on the side that maximizes the number of resulting crossing edges.

The set Y has size $2^{O(t_0)} = e^{O(1/\epsilon^2)}$, so by a union bound,

$$\Pr\left(\max_{y\in Y}\left|b(\hat{x}^t_{(y)}) - b(\frac{n}{t}x^t_{(y)})\right|_1 \geq \lambda\right) \leq e^{O(1/\epsilon^2)-\lambda^2/\sigma^2}$$

Finally, integrate $\min(1, e^{O(1/\epsilon^2)-\lambda^2/\sigma^2})$ over λ.

LEMMA 3.3.

$$\mathbf{E}\left[\max_{y\in Y}|z(\frac{n}{t}x^t_{(y)}) - z(\hat{x}^t_{(y)})|\right] = \frac{1}{\epsilon}O(\sigma)$$

where $\sigma = O\left(\frac{n^2}{\sqrt{t}}\sqrt{\frac{n-t}{n}}\right)$.

The ideas of the proof are similar to those in Lemmas 2.3 and 3.2 but somewhat technical, so we defer the proof to §3.3.

3.1 Proof of Theorem 1.2 By definition of the fictitious cut, the output cut $x^n_{(w)}$ is equal to $\hat{x}^n_{(w)}$. As in the previous section, looking at the process from time t_1 to time n we write:

$$z(\hat{x}^n_{(w)}) = z(\hat{x}^{t_1}_{(w)}) + \sum_{t_1 \leq t \leq n} z(\hat{x}^t_{(w)}) - z(\hat{x}^{t-1}_{(w)}).$$

Taking expectations, by Lemma 2.2 (which is still valid here) we have:

$$\mathbf{E}\left[z(\hat{x}^t_{(w)}) - z(\hat{x}^{t-1}_{(w)})\right] \leq$$

$$4n^2/t^2 + 2\frac{n}{t(n-t+1)}\mathbf{E}\left[|b(\hat{x}^{t-1}_{(w)}) - b(\frac{n}{t}x^{t-1}_{(w)})|_1\right].$$

Now, note that we have

$$|b(\hat{x}^{t-1}_{(w)}) - b(\frac{n}{t}x^{t-1}_{(w)})|_1 \leq \max_{y\in Y} |b(\hat{x}^{t-1}_{(y)}) - b(\frac{n}{t}x^{t-1}_{(y)})|_1.$$

Thus we can use Lemma 3.2 to deal with this term. To analyze $z(\hat{x}^{t_1}_{(w)})$, we write:

$$z\left(\hat{x}^{t_1}_{(w)}\right) \leq z\left(\frac{n}{t_1}x^{t_1}_{(w)}\right) + \left|z\left(\hat{x}^{t_1}_{(w)}\right) - z\left(\frac{n}{t_1}x^{t_1}_{(w)}\right)\right|.$$

The second term can be bounded above by $\max_{y\in Y} |z(\hat{x}^{t_1}_{(y)}) - z(\frac{n}{t_1}x^{t_1}_{(y)})|$, so that we can apply Lemma 3.3 to bound it. As to the first term, since x^* is one of the cuts in Y, by definition of the algorithm we have

$$z\left(\frac{n}{t_1}x^{t_1}_{(w)}\right) \leq z\left(\frac{n}{t_1}x^{t_1}_{(x^*)}\right).$$

Now we can write

$$z\left(\frac{n}{t_1}x^{t_1}_{(x^*)}\right) \leq z\left(\hat{x}^{t_1}_{(x^*)}\right) + \left|z\left(\hat{x}^{t_1}_{(x^*)}\right) - z\left(\frac{n}{t_1}x^{t_1}_{(x^*)}\right)\right|.$$

Again, the second term can be bounded above by $\max_{y\in Y} |z(\hat{x}^{t_1}_{(y)}) - z(\frac{n}{t_1}x^{t_1}_{(y)})|$, and we can apply Lemma 3.3 to bound it. Now, by Lemma 2.4 for $t = t_1$, we have

$$\mathbf{E}\left[z(\hat{x}^{t_1}_{(x^*)})\right] - \mathbf{E}\left[z(\hat{x}^{t_0}_{(x^*)})\right] = O(\epsilon n^2).$$

Finally, as in the previous section, it holds that $\mathbf{E}\left[z(\hat{x}^{t_0}_{(x^*)})\right] = \text{OPT}$. Together, these bounds prove Theorem 1.2.

3.2 A probabilistic aside. For any $\sigma > 0$, let $C(\sigma)$ denote the set of random variables X such that for any $\lambda > 0$, $\Pr(X \geq \sigma + \lambda) \leq e^{-\lambda^2/\sigma^2}$.

This definition satisfies a number of easily verified properties.

LEMMA 3.4. *Let σ and α be positive constants and X and Y be random variables. Then:*

- *If $X \in C(\sigma)$ then $\alpha X \in C(\alpha\sigma)$.*
- *If $X \in C(\sigma)$ and $Y \leq X$ then $Y \in C(\sigma)$.*
- *The random variable with constant value σ is in $C(\sigma)$.*

Furthermore this class of random variables adds nicely:

LEMMA 3.5. *For any real-valued random variables X, Y, if $X \in C(\sigma_x)$ and $Y \in C(\sigma_y)$, then $X + Y \in C(\sigma_x + \sigma_y)$.*

Proof. [Proof sketch] The worst case is if both inequalities hold with equality. Using Chernoff bound techniques, it is sufficient to show that $\mathbf{E}\left[e^{\alpha(X+Y-\sigma_x-\sigma_y)}\right] \leq e^{-\alpha^2(\sigma_x+\sigma_y)^2}$. Clearly $\mathbf{E}\left[e^{\alpha(X+Y-\sigma_x-\sigma_y)}\right] = \mathbf{E}\left[e^{\alpha(X-\sigma_x)}e^{\alpha(Y-\sigma_y)}\right]$. The worst case for the expectation of a product given the distribution of the factors is when X and Y are maximally correlated, and hence $X/\sigma_x = Y/\sigma_y$.

3.3 Proof of Lemma 3.3 The following lemma plus a union bound suffices to prove Lemma 3.3.

LEMMA 3.6. *For a fixed y,*

$$\Pr\left(\left|z\left(\frac{n}{t}x^t_{(y)}\right) - z(\hat{x}^t_{(y)})\right| \geq \sigma + \lambda\right) \leq e^{-\lambda^2/\sigma^2}$$

where

$$\sigma = \frac{n^2}{\sqrt{t}} \cdot \sqrt{\frac{n-t}{n}}.$$

Proof. Fix $y \in Y$ and study $F_t = z(\frac{n}{t}x^t_{(y)}) - z(\hat{x}^t_{(y)})$. Define $\bar{x} = \frac{t}{n}\hat{x}^t_{(y)}$. By definition, multilinearity and symmetry:

$$F_t = \left(\frac{n}{t}\right)^2 (n-t) A\left(\frac{x^t - \bar{x}^t}{n-t},\ (x^t + \bar{x}^t)\right)$$

so it suffices to show $A\left((x-\bar{x})/(n-t),\ (x+\bar{x})\right) \in C(O(t^2/\sqrt{tn(n-t)}))$.

Let $D^t = A\left((x^t - \bar{x}^t)/(n-t),\ (x^t + \bar{x}^t)\right) - A\left((x^{t-1} - \bar{x}^{t-1})/(n-t+1),\ (x^{t-1} + \bar{x}^{t-1})\right)$. Therefore:

$$A\left((x^t - \bar{x}^t)/(n-t),\ (x^t + \bar{x}^t)\right) \tag{3.6}$$
$$= \sum_{\tau=1}^{t}(D^\tau - \mathbf{E}\left[D^\tau | S^{t-1}\right]) + \sum_{\tau=1}^{t}\mathbf{E}\left[D^\tau | S^{t-1}\right].$$

By Lemma 3.5, it is sufficient to show each of these two terms is separately in $C(O(t^2/\sqrt{tn(n-t)}))$.

The first term of (3.6) is a martingale with step size $O(t/(n-t))$, which leads to variance $O(t^3/n^2)$ for $t < n/2$ and $O(n^2/(n-t))$ for $t \geq n/2$. Azuma-Hoeffding and liberal use of big-oh shows the first term of (3.6) is in $C(t^2/\sqrt{tn(n-t)})$.

For the second term of (3.6), we write that D_t equals

$$\begin{aligned} &A(\frac{x^t - \bar{x}^t}{n-t} - \frac{x^{t-1} - \bar{x}^{t-1}}{n-t+1},\ x^{t-1} + \bar{x}^{t-1}) \\ &+ A(\frac{x^{t-1} - \bar{x}^{t-1}}{n-t+1},\ x^t + \bar{x}^t - x^{t-1} - \bar{x}^{t-1})) \\ &+ A(\frac{x^t - \bar{x}^t}{n-t} - \frac{x^{t-1} - \bar{x}^{t-1}}{n-t+1},\ x^t + \bar{x}^t - x^{t-1} - \bar{x}^{t-1}). \end{aligned} \tag{3.7}$$

The first term of (3.7) has expectation given history of zero by Lemma 2.5. The third term of (3.7) is bounded

by a negligible $t/(n-t)$. The second term can be rewritten as

$$\frac{t}{n(n-t)}(x^t + \bar{x}^t - x^{t-1} - \bar{x}^{t-1})(b(\hat{x}^t) - b(\frac{n}{t}x^t)).$$

A quick calculation shows

$$|\mathbf{E}\left[x^t + \bar{x}^t - x^{t-1} - \bar{x}^{t-1}|S^{t-1}\right]|_\infty \leq 1/(n-t)$$

and by (3.5), $\left|b(\hat{x}^t) - b(\frac{n}{t}x^t)\right|_1 \in C(O(n^2\sqrt{n-t/(tn)}))$ so the second term of (3.7) is in $C(O(\sqrt{nt/(n-t)^3}))$. For $t \leq n/2$ this sums to $O(t^{3/2}/n)$. For $t > n/2$, it sums to $O(t^2/\sqrt{tn(n-t)})$.

Now we use Lemma 3.6 to prove lemma 3.3. Union bound: $\Pr\left(\max_y |z(n/tx) - z(\hat{x})| \geq \sigma + \lambda\right) \leq e^{O(1/\epsilon^2)-\lambda^2/\sigma^2}$. Integration proves the lemma.

4 Proof of Theorem 1.4

With these techniques in hand, our proof of Theorem 1.4 [1] is actually quite simple. Recall [1] that the hard direction is showing that the subproblem isn't easier than the overall problem: $\mathbf{E}\left[OPT_{S^{t_1}}\right]/t_1^2 \geq OPT/n^2 - O(\epsilon)$. The easy direction is a consequence of Lemmas 2.4 and 3.3 (though much easier proofs exist). The key idea we use is that while Algorithm 1, as a subroutine of Algorithm 2, is solving the problem on the first t_1 vertices, it is also implicitly generating solutions for the whole graph $\hat{x}^{t_1}_{(y)}$.

By Theorem 1.1 the best cut found by Algorithm 1 for the outer sample S^{t_1} is a good approximation to the optimal cut for the outer sample:

$$\mathbf{E}\left[z(x^{t_1}_{(w)})\right]/t_1^2 \leq \mathbf{E}\left[OPT_{S^{t_1}}\right]/t_1^2 + O(\epsilon)$$

By Lemma 3.2, the cost of the best cut of the outer sample found by Algorithm 1 is approximately equal to the cost of the extrapolated solution $\hat{x}^{t_1}$:

$$\mathbf{E}\left[z(\hat{x}^{t_1}_{(w)})\right]/n^2 \leq \mathbf{E}\left[z(x^{t_1}_{(w)})\right]/t_1^2 + O(\epsilon)$$

Recall that $\hat{x}$ has every variable set to a convex combination of several values ($|\hat{x}_u| = 1$). One can make greedy changes to convert $\hat{x}$ into an integral solution, so by definition of OPT:

$$OPT/n^2 \leq \mathbf{E}\left[z(\hat{x}^{t_1}_{(w)})\right]/n^2$$

Adding inequalities together yields:

$$OPT/n^2 \leq \mathbf{E}\left[OPT_{S^{t_1}}\right]/t_1^2 + O(\epsilon).$$

5 Open problems

It is well-known that running greedy once is a 2-approximation for MaxCut on general graphs, and we just proved that running greedy many times is a $(1+\epsilon)$-approximation on dense graphs. Does repeating greedy on general graphs yield an approximation factor better than 2?

Previous algorithms for MaxCut have been extended to weighted instances when the weights define a metric. We conjecture that such instances can also be solved by an extension of our greedy algorithms.

What problems other than Max-r-CSP can be analyzed with our technique? It is easy to see that maximum separator can be solved with our techniques (albeit with a less efficient $t_0 = 1/\epsilon^4$), but many other possibilities exist for which PTASs are not already known. For example, consider the problem of finding a maximum cut with the condition that the fraction of the *edges* that are within the left side of the cut is the inverse of a prime (1/2, 1/3, 1/5, etc.). This problem is likely also solvable by our framework. Non-convex but smooth objectives should also be handled by our framework, such as finding a three-way cut that maximizes the square of the number of edges between piece 1 and 2 plus the number of edges between 2 and 3, should anyone care for such an objective function.

References

[1] Noga Alon, W. Fernandez de la Vega, Ravi Kannan, and Marek Karpinski. Random sampling and approximation of MAX-CSPs. *J. Comput. Syst. Sci.*, 67(2):212–243, 2003.

[2] Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It's all about regularity. In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 251–260, New York, NY, USA, 2006. ACM Press.

[3] Sanjeev Arora, David Karger, and Marek Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 284–293, 1995.

[4] W. Fernandez de la Vega. MAX-CUT has a randomized approximation scheme in dense graphs. *Random Struct. Algorithms*, 8(3):187–198, 1996.

[5] Alan M. Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.

[6] Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.