Vietnam Journal of Mathematics **34**:4 (2006) 389–395

Vietnam Journal of MATHEMATICS
© VAST 2006

On the Approximability of Max-Cut

Le Cong Thanh

Institute of Mathematics, 18 Hoang Quoc Viet Road, 10307 Hanoi, Vietnam

Dedicated to Professor Do Long Van on the occasion of his 65th birthday

Received December 15, 2005

Abstract. We introduce the almost sure performance ratio of an approximation algorithm for a discrete optimization problem and consider it for the MAX-CUT problem. It is known that MAX-CUT cannot be solved by a polynomial time approximation algorithm with the ratio less than 1.0625 for all instances of the problem unless $P = NP$. The aim of this note is to show that MAX-CUT can be solved by a linear time approximation algorithm with the ratio less than $1 + \varepsilon$ (for any $\varepsilon > 0$) for almost every instance, and hence with the almost sure performance ratio 1.

2000 Mathematics Subject Classification: 68Q17.
Keywords: Approximation algorithm, absolute performance ratio, almost sure performance ratio.

1. Introduction, Terminology, Main Result

In certain problems called optimization problems we seek to find the optimal solution among a collection of candidate (feasible) solutions. If the optimization problem is NP-hard, then we have known that a polynomial time optimization algorithm cannot be found unless $P = NP$. A more reasonable goal is that of finding an approximation algorithm that runs in a polynomial time and that finds a solution that is "nearly" optimal may be good enough.

To formalize this approach, we settled on a general form for our guarantees, in terms of ratios, which was useful for comparison purpose and which seems to express nearness to optimality in a reasonable way. The terminology follows that in [1].

Let Π be an optimization problem with instance set D_Π. We will use $OPT(I)$ to denote the value of an optimal solution for an instance $I \in D_\Pi$. And let A be an approximation algorithm for Π. The value of the candidate solution found by A when applied to I will be denoted by $A(I)$.

If Π is a minimization problem (resp., maximization problem), and I is any instance in D_Π, then the *ratio* $R_A(I)$ of an approximation algorithm A on an instance I is defined by

$$R_A(I) = \frac{A(I)}{OPT(I)} \qquad \Big(\text{resp., } R_A(I) = \frac{OPT(I)}{A(I)}\Big).$$

The *absolute performance ratio* R_A of an approximation algorithm A for a problem Π is given by

$$R_A = \inf\{r \geq 1 : R_A(I) \leq r \text{ for all instances } I \in D_\Pi\}.$$

Notice that the absolute performance ratio is always a number greater than or equal to 1 and is as close to 1 as the candidate solution found by the approximation algorithm is close to the optimal solution. An approximation algorithm with the absolute performance ratio not greater than some positive integer α is called $\alpha-$*approximation algorithm*.

Notice also that performance guarantees for approximation algorithms are in their nature works-case bounds, and algorithms often behave significantly better in practice than their performance guarantees would suggest.

As an alternative to the"works-case" performance guarantee approach, one might therefore attempt to do performance analysis from an "average-case" point of view. Indeed, such analysis has a long history and has been performed primality through empirical studies.

Rather practically, we are interested in performance analysis from an "almost every-case" point of view, i.e., the analysis for "almost every instance" of the considered problem. We now present this conception for only optimization problems Π, whose instances with discrete structure, and for which the set D_Π^n of instances of size n ($n = 1, 2, \dots$) is finite and $|D_\Pi^n| \to \infty$ as $n \to \infty$. For example, if instances of Π are finite graphs then as D_Π^n one can choose the set of graphs with n vertices.

Given a property Q, we shall say that *almost every instance* of Π has property Q if

$$\lim_{n\to\infty} (d_Q(n)/|D_\Pi^n|) = 1,$$

where $d_Q(n)$ is the number of instances $I \in D_\Pi^n$ having property Q. Then we define the *almost sure performance ratio* R_A^{as} of an approximation algorithm A by

$$R_A^{as} = \inf\{r \geq 1 : R_A(I) \leq r \text{ for almost every instance } I \in D_\Pi\}.$$

This note deals with the approximability of the NP-complete MAX-CUT problem which is defined as follows: *Given a simple loopless undirected graph $G = (V, E)$ with the vertex set V, we wish to find a separation of the set V into two disjoint subsets U and $\overline{U} = V \setminus U$ with the maximum number of edges passing between U and $\overline{U}$.* In the 1970s Johnson [4] gave a simple $2-$approximation algorithm for the MAX-CUT problem. This one is interesting because it stood unimproved for a long time. Furthermore, by applying semidefinite programming to the MAX-CUT problem Goemans and Williamson [2] introduced a $1.138-$approximation algorithm. This was the first improvement and it appeared in 1995. However, using a NP-completeness of the MAX-CUT problem. Håstad [3] has shown that if $P \neq NP$, then no polynomial time approximation algorithm A for the MAX-CUT problem can have the absolute performance ratio $R_A < 17/16 = 1.0625$. Therefore the MAX-CUT problem cannot be solved by a polynomial time approximation algorithm A with the ratio $R_A(G) < 1.0625$ for all instances (graphs) G.

As the main theorem of the present work we will prove the following somewhat more practical result.

Theorem 1. *The MAX-CUT problem can be solved by a linear time approximation algorithm ES with the ratio $R_{ES}(G) < 1+\varepsilon$ for almost every instance G and for any $\varepsilon > 0$, and hence with the almost sure performance ratio $R_{ES}^{as} = 1$.*

The proof of this result is given in Sec. 3 and is based on some estimations (obtained in Sec. 2) of the cardinality of cuts for almost every graph.

2. Estimations of the Cardinality of Cuts

We shall consider in this note only finite simple loopless undirected graphs. We write $\mathcal{G}_n$ for the set of all graphs with the vertex set V of n elements:

$$\mathcal{G}_n = \{G_i \mid V(G_i) = V;\ i = 1, 2, \ldots, p\},$$

where for simplicity of notation we put $p = 2^{\binom{n}{2}}$.

Let G be a graph of $\mathcal{G}_n$. Given a subset U of m vertices of V such that $1 \leq m \leq n/2$. Denote by $C_U(G)$ the set of edges passing between U and $\overline{U} = V \setminus U$ of G; such a set of edges is called a *cut associated with the separation* $V = U \cup \overline{U}$ or shortly a $U-$*cut* of G.

Thus, for complete graph K_n of $\mathcal{G}_n$, the $U-$cut $C_U(K_n)$ is the set of all $m(n-m)$ possible edges between U and $\overline{U}$. For this set we write:

$$C_U(K_n) = \{e_j \mid j = 1, 2, \ldots, q\},$$

where $q = m(n-m)$.

Throughout the note we use the following notations:

$c_U(G)$ - the cardinality of the $U-$cut $C_U(G)$ of G;

$c_U(n)$ - the mean value of $c_U(G)$ over $\mathcal{G}_n$, i.e., $c_U(n) = \frac{1}{p}\sum_{i=1}^{p} c_U(G_i)$;

$c(G)$ - the cardinality of a maximum cut of G, i.e., the maximum value of $c_U(G)$ when U ranges over all nonempty subsets of V.

The aim of this section is to estimate $c_U(G)$ and $c(G)$ for almost every graph G. This is based on the following lemmas.

Lemma 1. *For any subset U of m vertices of V $(|V| = n)$ we have*

$$c_U(n) = \frac{m(n-m)}{2}.$$

Proof. For every graph $G_i \in \mathcal{G}_n$ and every edge $e_j \in C_U(K_n)$ we define a variable $x(G_i, e_j)$ as follows:

$$x(G_i, e_j) = \begin{cases} 1 & \text{if } e_j \in C_U(G_i), \\ 0 & \text{if } e_j \notin C_U(G_i), \end{cases}$$

where $1 \le i \le p$ and $1 \le j \le q$. Then we have

$$\begin{aligned} c_U(n) &= \frac{1}{p}\sum_{i=1}^{p} c_U(G_i) \\ &= \frac{1}{p}\sum_{i=1}^{p}\sum_{j=1}^{q} x(G_i, e_j) \\ &= \frac{1}{p}\sum_{j=1}^{q}\sum_{i=1}^{p} x(G_i, e_j) \\ &= \frac{1}{p}\sum_{j=1}^{q} g(e_j), \end{aligned}$$

where $g(e_j)$ is the number of graphs $G \in \mathcal{G}_n$ such that $e_j \in C_U(G)$.

It is easy to see that, for every edge $e_j, 1 \le j \le q$,

$$g(e_j) = 2^{\binom{n}{2}-1}.$$

Hence

$$c_U(n) = \frac{q}{p}.2^{\binom{n}{2}-1} = \frac{m(n-m)}{2},$$

and the lemma is proved. ∎

Let $\xi_{U,n}$ be a random variable taking the value ℓ with the probability $H(\ell)/|\mathcal{G}_n|$, where $H(\ell)$ is the number of graphs $G \in \mathcal{G}_n$ such that $c_U(G) = \ell$. Denote by $E\xi_{U,n}$ the expectation and by $Var\xi_{U,n}$ the variance of $\xi_{U,n}$. Then by Lemma 1

$$E\xi_{U,n} = c_U(n) = \frac{m(n-m)}{2}.$$

Lemma 2. *For any subset U of m vertices of V we have*

$$Var\xi_{U,n} = \frac{m(n-m)}{4}.$$

Proof. By definition,

$$Var\xi_{U,n} = E\xi_{U,n}^2 - (E\xi_{U,n})^2.$$

To calculate $E\xi_{U,n}^2$, now for every $G_i \in \mathcal{G}_n$ and every pair $(e_j, e_k) \in C_U(K_n) \times C_U(K_n)$ we define a variable $x(G_i, e_j, e_k)$ as follows:

$$x(G_i, e_j, e_k) = \begin{cases} 1 & \text{if both } e_j, e_k \in C_U(G_i), \\ 0 & \text{otherwise}, \end{cases}$$

where $1 \le i \le p$ and $1 \le j, k \le q$. Then we have

$$\begin{aligned}
E\xi^2_{U,n} &= \frac{1}{p}\sum_{i=1}^{p}\big(c_U(G_i)\big)^2 \\
&= \frac{1}{p}\sum_{i=1}^{p}\Big(\sum_{j=1}^{q} x(G_i,e_j)\Big)^2 \\
&= \frac{1}{p}\sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{q} x(G_i,e_j,e_k) \\
&= \frac{1}{p}\sum_{j=1}^{q}\sum_{k=1}^{q}\sum_{i=1}^{p} x(G_i,e_j,e_k) \\
&= \frac{1}{p}\sum_{j=1}^{q}\sum_{k=1}^{q} g(e_j,e_k),
\end{aligned}$$

where $g(e_j,e_k)$ is the number of graphs $G \in \mathcal{G}_n$ such that both $e_j, e_k \in C_U(G)$.

It is obvious that for every pair (e_j,e_k) of $C_U(K_n)\times C_U(K_n)$

$$g(e_j,e_k) = \begin{cases} 2^{\binom{n}{2}-2} & \text{if } e_j \neq e_k, \\ 2^{\binom{n}{2}-1} & \text{if } e_j = e_k. \end{cases}$$

Hence

$$\begin{aligned}
E\xi^2_{U,n} &= \frac{1}{p}\big[(q^2-q).2^{\binom{n}{2}-2} + q.2^{\binom{n}{2}-1}\big] \\
&= \frac{q^2-q}{4} + \frac{q}{2} \\
&= \frac{q^2}{4} + \frac{q}{4}.
\end{aligned}$$

Since $q = m(n-m)/2$ and $E\xi_{U,n} = m(n-m)/2$ we have

$$E\xi^2_{U,n} = (E\xi_{U,n})^2 + \frac{m(n-m)}{4}.$$

Thus

$$Var\xi_{U,n} = E\xi^2_{U,n} - (E\xi_{U,n})^2 = \frac{m(n-m)}{4},$$

as claimed. ∎

Theorem 2. *For almost every graph G and for any nonempty subset U of the vertex set $V(G)$ of G, the number $c_U(G)$ of edges between U and $\overline{U} = V(G)\setminus U$ of G satisfies*

$$\frac{m(n-m)}{2} - \frac{\sqrt{m(n-m)}}{4}\log_2 n < c_U(G) < \frac{m(n-m)}{2} + \frac{\sqrt{m(n-m)}}{4}\log_2 n,$$

where $n = |V(G)|$ and $m = |U|$.

Proof. Applying Chebyshev's inequality for the variable $\xi_{U,n}$ we have

$$Prob\Big(|\xi_{U,n} - E\xi_{U,n}| \geq t\Big) \leq \frac{Var\xi_{U,n}}{t^2}$$

for any real $t > 0$. Choose $t = \frac{\sqrt{m(n-m)}}{4}\log_2 n$. Then by Lemmas 1 and 2 we obtain

$$Prob\left(\left|c_U(G) - \frac{m(n-m)}{2}\right| \geq \frac{\sqrt{m(n-m)}}{4}\log_2 n\right) \leq \frac{4}{\log_2^2 n} \to 0$$

as $n \to \infty$. This means that for almost every graph G

$$\left|c_U(G) - \frac{m(n-m)}{2}\right| < \frac{\sqrt{m(n-m)}}{4}\log_2 n,$$

implying the assertion of the theorem. ∎

Theorem 3. *For almost every graph G the cardinality $c(G)$ of a maximum cut of G satisfies*

$$\frac{n^2}{8} - \frac{n}{8}\log_2 n < c(G) < \frac{n^2}{8} + \frac{n}{8}\log_2 n,$$

where n is the number of vertices of G.

Proof. By definition of $c(G)$ and by Theorem 2 we have

$$\begin{aligned} c(G) &= \max_U c_U(G) \\ &< \max_m \left\{\frac{m(n-m)}{2} + \frac{\sqrt{m(n-m)}}{4}\log_2 n\right\} \\ &< \frac{n^2}{8} + \frac{n}{8}\log_2 n. \end{aligned}$$

In order to find a lower bound of $c(G)$ we choose a subset U_0 of V such that $|U_0)| = \lfloor n/2 \rfloor$, where $\lfloor n/2 \rfloor$ is the greatest integer not greater than $n/2$. Then applying Theorem 2 for the subset U_0 we obtain

$$c(G) \geq c_{U_0}(G) > \frac{n^2}{8} - \frac{n}{8}\log_2 n.$$

Thus the proof is complete. ∎

3. Proof of Theorem 1

To prove the theorem, we now give an approximation algorithm for the MAX-CUT problem and analyse its performance ratios. Our algorithm is very simple as follows: Equitably separate the vertex set $V(G)$ of a given graph G into two disjoint subsets V_1 and $\overline{V_1} = V(G) \setminus V_1$, i.e., $\big||V_1| - |\overline{V_1}|\big| \leq 1$. Therefore the algorithm is denoted by ES.

It is easy to see that the algorithm ES runs in linear time. The analysis of the performance ratios of ES is based on Theorems 4 and 5 as follows:

Since, for any graph G,

$$ES(G) = c_{V_1}(G)$$

and

$$OPT(G) = c(G).$$

Hence, for almost every graph G, by Theorems 2 and 3 we have

$$ES(G) > \frac{n^2}{8} - \frac{n}{8}\log_2 n$$

and

$$OPT(G) < \frac{n^2}{8} + \frac{n}{8}\log_2 n,$$

where n is the number of vertices of G.

Thus the ratio $R_{ES}(G)$ of the algorithm ES for almost every graph G is bounded by

$$R_{ES}(G) = \frac{OPT(G)}{ES(G)} < 1 + \frac{3\log_2 n}{n},$$

and hence the almost sure performance ratio of ES is

$$R_{ES}^{as} = 1.$$

This completes the proof of Theorem 1. ∎

Notice that the algorithm ES have the absolute performance ratio $R_{ES} = \infty$.

References

1. M. R. Garey and D. S. Johnson, *Computers and Intractability - A Guide to Theory of NP-completeness*, W. H. Freeman, San Fransisco, 1979.
2. M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfibility problems using semidefinite programming, *J. ACM* **42** (1995) 1115–1145.
3. J. Håstad, *Some Optimal Inapproximability Results*, Proc. 29th Ann. ACM Symp. on Theory of Computing, 1997, pp. 1–10.
4. D. S. Johnson, Approximation algorithms for combinatorial problems, *J. Comput. System Sci.* **9** (1974) 256–278.