

# CSCI2510 Approximation Algorithms

## Assignment 7

Fabio Vandin  
vandinf@cs.brown.edu

October 31, 2008

### Exercise 16.6

Consider the following randomized algorithm for the maximum cut problem:

MAXCUT( $G = (V, E)$ )  
1 Initialization:  $A \leftarrow \{v_1\}; B \leftarrow \{v_2\}$ ;  
2  $\forall v \in V \setminus \{v_1, v_2\}$  do:  
3 insert  $v$  in  $A$  or in  $B$  with equal probability.  
4 Output  $A$  and  $B$ .

where  $v_1, v_2$  are arbitrary vertices in  $V$ .

**Proposition 1.** *The expected size of the cut found is at least  $\text{OPT}/2$ .*

*Proof.* Given a pair of vertices  $(v_i, v_j)$ , with  $1 \leq i < j \leq n$ , let's define the random variable:

$$X_{i,j} = \begin{cases} 1, & \text{if } v_i \in A, v_j \in B \text{ or } v_i \in B, v_j \in A \\ 0, & \text{otherwise} \end{cases}$$

It is easy to see that  $\text{Prob}(X_{i,j} = 1) \geq 1/2, \forall i, j$  (in particular,  $X_{1,2} = 1$  with probability 1 for vertices  $v_1, v_2$  defined in the initialization, and  $X_{i,j} = 1$  with probability  $1/2$  for all the other pairs). Note that  $v_1, v_2$  are arbitrary vertices in  $V$ , not randomly defined vertices (the analysis with randomly defined vertices  $v_1, v_2$  is essentially the same).

Now define the random variable  $W$  that represents the size of the cut produced in output by the algorithm, and let  $e_{i,j} = 1$  if  $(v_i, v_j) \in E$ , and 0 otherwise. We have

$$W = \sum_{i < j} e_{i,j} X_{i,j}$$

thus

$$\begin{aligned}
E[W] &= E \left[ \sum_{i < j} e_{i,j} X_{i,j} \right] \\
&= \sum_{i < j} e_{i,j} E[X_{i,j}] \\
&\geq \frac{1}{2} \sum_{i < j} e_{i,j} \\
&= \frac{1}{2} \sum_{e=(v_i, v_j) \in E} e_{i,j} \geq \frac{1}{2} \text{OPT}.
\end{aligned}$$

□

Now we want to derive the derandomized version of the above algorithm via the method of conditional expectation. Let us denote with  $v_1, \dots, v_n$  the vertices in the order they are considered by the algorithm; we know that  $v_1$  is always inserted in  $A$ ,  $v_2$  is always inserted in  $B$ , and  $v_3, \dots, v_n$  are inserted in  $A$  or in  $B$  with the same probability. From the analysis above we have:

$$\frac{\text{OPT}}{2} \leq E[W] = \frac{1}{2} E[W | v_3 \text{ placed into } A] + \frac{1}{2} E[W | v_3 \text{ placed into } B]$$

and moreover we have that  $\forall i, 4 \leq i \leq n$ :

$$\begin{aligned}
E[W | v_3, \dots, v_{i-1} \text{ placed}] &= \frac{1}{2} E[W | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } A] \\
&+ \frac{1}{2} E[W | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } B].
\end{aligned}$$

From the inequalities above and using the same argument followed for MAXSAT, we can show that if we can decide in polynomial time if:

$$E[W | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } A] \geq E[W | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } B] \quad (1)$$

for all  $i$  ( $4 \leq i \leq n$ ), we can sequentially assign  $v_i$  to  $A$  or to  $B$  maximizing the conditional expected value of  $W$  after the assignment of  $v_i$ ; in this way we obtain a deterministic  $\frac{1}{2}$  factor approximation algorithm for MAXCUT:

$$\begin{aligned}
\frac{\text{OPT}}{2} &\leq E[W] \\
&\leq E[W | v_3 \text{ placed with the above strategy}] \\
&\leq E[W | v_3, v_4 \text{ placed with the above strategy}] \\
&\vdots \\
&\leq E[W | v_3, \dots, v_n \text{ placed with the above strategy}]
\end{aligned}$$

For a fixed vertex  $v_i$ , we show how condition (1) can be checked in polynomial time. Let's denote with  $C_i$  the number of edges running between  $A$  and  $B$  after the  $i$ -th vertex has been

assigned to  $A$  or  $B$ . For  $A \subset V$ , let us denote with  $d(v_i, A)$  the number of edges running between vertex  $v_i$  and set  $A$ , and let  $k$  the number of edges whose vertices are both in  $A$  or both in  $B$  before placing  $v_i$ . Now, if  $m = |E|$ , we have that the number of edges incident to  $\{v_{i+1}, \dots, v_n\}$  is  $\alpha_{i+1} = m - d(v_i, A) - d(v_i, B) - C_{i-1} - k$ .

Since each vertex in  $\{v_{i+1}, \dots, v_n\}$  will be placed in  $A$  or  $B$  with the same probability, each edge incident to at least one vertex in  $\{v_{i+1}, \dots, v_n\}$  has probability  $1/2$  to do not see both its vertices assigned to  $A$  or  $B$ . Then:

$$E[C_n | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } A] = C_{i-1} + d(v_i, B) + \frac{\alpha_{i+1}}{2}$$

$$E[C_n | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } B] = C_{i-1} + d(v_i, A) + \frac{\alpha_{i+1}}{2}$$

thus we have that:

$$E[C_n | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } A] \geq E[C_n | v_3, \dots, v_{i-1} \text{ placed, and } v_i \text{ placed into } B]$$

if and only if

$$d(v_i, B) \geq d(v_i, A)$$

that can be verified in polynomial time.

The derandomized algorithm for MAXCUT is thus the following:

MAXCUT( $G = (V, E)$ )

- 1 Initialization:  $A \leftarrow \{v_1\}; B \leftarrow \{v_2\};$
- 2  $\forall v \in V \setminus \{v_1, v_2\}$  do:
- 3 if  $d(v, A) \geq d(v, B)$  then  $B \leftarrow B \cup \{v\};$
- 4 else  $A \leftarrow A \cup \{v\};$
- 5 Output  $A$  and  $B$ .

that is precisely Algorithm 2.13 in Vazirani's book.

*Note:* for edge-weighted graphs with weight  $w_{i,j}$  on edge  $e = (v_i, v_j)$ , the algorithm and the analysis are the same defining  $e_{i,j} = w_{i,j}$  (with  $w_{i,j} = 0$  if  $(v_i, v_j) \notin E$ ) and denoting with  $d(v, A)$  the sum of the weights of the edges running from  $v$  to  $A$ , with  $C_i$  the weight of the cut after placing  $v_i$ , and with  $\alpha_{i+1}$  the sum of the weight of the edges incident to vertices  $\{v_{i+1}, \dots, v_n\}$ .