Washington University in St. Louis

Washington University Open Scholarship

All Computer Science and Engineering
Research

Computer Science and Engineering

Report Number: WUCSE-2006-3

2006-01-01

An Improved Analysis for a Greedy Remote-Clique Algorithm using Factor-Revealing LPs

Benjamin E. Birnbaum and Kenneth J. Goldman

Given a complete graph with nonnegative edge weights satisfying the triangle inequality and a positive integer p , the remote-clique problem is to find a subset of p vertices having a maximum-weight induced subgraph. A simple greedy algorithm for the problem has been shown to have an approximation ratio of 4, but a tight example has not been provided. In this paper, we use the technique of factor revealing linear programs to prove that this algorithm actually achieves an approximation ratio of 2, matching the best-known ratio for the problem. The greedy algorithm's running time of $O(pn)$ makes it the... [Read complete abstract on page 2.](#)

Follow this and additional works at: https://openscholarship.wustl.edu/cse_research

Part of the [Computer Engineering Commons](#), and the [Computer Sciences Commons](#)

Recommended Citation

Birnbaum, Benjamin E. and Goldman, Kenneth J., "An Improved Analysis for a Greedy Remote-Clique Algorithm using Factor-Revealing LPs" Report Number: WUCSE-2006-3 (2006). *All Computer Science and Engineering Research*.

https://openscholarship.wustl.edu/cse_research/180

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

An Improved Analysis for a Greedy Remote-Clique Algorithm using Factor-Revealing LPs

Benjamin E. Birnbaum and Kenneth J. Goldman

Complete Abstract:

Given a complete graph with nonnegative edge weights satisfying the triangle inequality and a positive integer p , the remote-clique problem is to find a subset of p vertices having a maximum-weight induced subgraph. A simple greedy algorithm for the problem has been shown to have an approximation ratio of 4, but a tight example has not been provided. In this paper, we use the technique of factor revealing linear programs to prove that this algorithm actually achieves an approximation ratio of 2, matching the best-known ratio for the problem. The greedy algorithm's running time of $O(pn)$ makes it the fastest known 2-approximation for the remote-clique problem.

2006-3

An Improved Analysis for a Greedy Remote-Clique Algorithm using Factor-Revealing LPs

Authors: Benjamin E. Birnbaum, Kenneth J. Goldman

Corresponding Author: kjg@cse.wustl.edu

Abstract: Given a complete graph with nonnegative edge weights satisfying the triangle inequality and a positive integer p , the remote-clique problem is to find a subset of p vertices having a maximum-weight induced subgraph. A simple greedy algorithm for the problem has been shown to have an approximation ratio of 4, but a tight example has not been provided. In this paper, we use the technique of factor revealing linear programs to prove that this algorithm actually achieves an approximation ratio of 2, matching the best-known ratio for the problem. The greedy algorithm's running time of $O(pn)$ makes it the fastest known 2-approximation for the remote-clique problem.

Type of Report: Other

An Improved Analysis for a Greedy Remote-Clique Algorithm using Factor-Revealing LPs *

Benjamin E. Birnbaum
Kenneth J. Goldman

Department of Computer Science and Engineering
Washington University in St. Louis

January 10, 2006

Abstract

Given a complete graph with nonnegative edge weights satisfying the triangle inequality and a positive integer p , the *remote-clique* problem is to find a subset of p vertices having a maximum-weight induced subgraph. A simple greedy algorithm for the problem has been shown to have an approximation ratio of 4, but a tight example has not been provided. In this paper, we use the technique of *factor-revealing linear programs* to prove that this algorithm actually achieves an approximation ratio of 2, matching the best-known ratio for the problem. The greedy algorithm's running time of $O(pn)$ makes it the fastest known 2-approximation for the remote-clique problem.

1 Introduction

As data-security becomes increasingly important, techniques are being developed to ensure that essential services can survive in spite of a limited number of arbitrary faults, including communication failures, crash failures, software errors, and malicious attacks. One such technique, the CLBFT algorithm [1], uses $3f + 1$ replicated data servers to guarantee correct performance when up to f of the servers are exhibiting faulty behavior. However, this guarantee relies on the faults being *independent*, a condition only achieved when the servers are diverse in terms of variables such as system architecture, software, physical location, and the network administrator. Therefore, in choosing a set of $3f + 1$ replicas from a large group of potential servers, it is important to choose a set of replicas that are as different from each other as possible.

*This work was supported by the National Science Foundation under CISE Educational Innovation grant 0305954.

We can model this situation as a graph problem. For each potential server, create a vector of attributes based on these variables and define the distance between two servers to be the *Hamming distance* between the vectors (i.e. the number of times corresponding attributes differ.) For each server, create a vertex in a complete graph. Set the edge weight between two vertices to be the distance between their corresponding servers. Finding a subset of servers that is most diverse now becomes the problem of finding a subset of $3f + 1$ vertices having an induced subgraph with maximum average edge weight (or equivalently maximum total edge weight).

The CLBFT example provides one of several motivations for the so-called *dispersion* problems, which involve finding subsets of vertices that are in some way as distant from each other as possible. (See [2] for other motivating examples.) Maximizing the average weight of the induced subgraph is the measure of dispersion addressed here, but it is possible to measure dispersion in a number of other ways, including the minimum weight edge and the minimum weight spanning tree of the induced subgraph. As in [2], we call the problem of maximizing the average weight the *remote-clique* problem, but it has also been called *maxisum dispersion* [5] and *max-avg facility dispersion* [7].

We define the problem more formally as follows. Let $G = (V, E)$ be a complete graph with the weight for edge $(v_1, v_2) \in E$ given by $w(v_1, v_2)$. (Define $w(v, v) = 0$ for all $v \in V$.) The edge lengths are nonnegative and satisfy the triangle inequality: for all $v_1, v_2, v_3 \in V$, $w(v_1, v_2) + w(v_2, v_3) \geq w(v_1, v_3)$.¹ For a given integer parameter p , such that $1 \leq p \leq |V|$, the *remote-clique problem* is to find a subset $V' \subseteq V$ such that $|V'| = p$ and the average edge weight in V' , $2/(p(p-1)) \cdot \sum_{v_1, v_2 \in V'} w(v_1, v_2)$, is maximized. This problem can be shown to be NP-hard by an easy reduction from CLIQUE.

It has been shown that a simple greedy algorithm with running time $O(n^2)$ always finds a solution that has an average edge weight no less than $\frac{1}{4}$ times optimal [7]. An example is provided in which the algorithm finds a solution that is $\frac{1}{2}$ optimal, but the question of whether a tighter bound for the algorithm can be proved remained open. In another paper, it is proved that a more complicated algorithm with running time $O(n^2 + p^2 \log p)$ achieves an approximation ratio of 2 [3]. However, a tight approximation ratio for the simple greedy algorithm has never been proved. In this paper, we prove that in fact the simple greedy algorithm has an approximation ratio of 2. We also reduce its running time to $O(pn)$, thus providing a fast and easy to implement algorithm for the remote-clique problem with the best known approximation ratio.

To prove an approximation ratio of 2, we use the technique of *factor-revealing linear programs* [4, 6], which is a simple generalization of a method often used to provide bounds for approximation algorithms. Consider a maximization (minimization) problem P . A typical analysis of an approximation algorithm ALG for P proceeds by using the behavior of ALG and the structure of P to generate a number of inequalities. These inequalities are then combined to provide a bound on the value of the solution obtained by ALG to that of an optimal solution. Often, this can be done algebraically, but not always. A more general way

¹It can easily be shown that the Hamming distance defined earlier does indeed satisfy the triangle inequality.

of obtaining a bound is to view the process as an optimization problem Q in its own right, in which an adversary tries to minimize (maximize) the value of ALG's solution to P subject to the constraints given by the generated inequalities. The optimal solution to Q is then a bound on the performance of ALG. If Q can be formulated as a linear program, then this is a *factor-revealing LP*. The simplicity of this technique makes it applicable to many problems, but in most cases it does not seem to be the easiest way to provide a bound. However, there are some algorithms, including the one in [4] and the greedy algorithm examined here, in which it is the only known technique to provide a tight bound.

2 Analysis of Greedy Augment

The algorithm we analyze, called GREEDY AUGMENT, maintains a set of vertices T , which it initializes with an arbitrary vertex in V . At each step in the algorithm, it augments T with a vertex $v \notin T$ that maximizes $\sum_{v' \in T} w(v', v)$. When $|T| = p$, GREEDY AUGMENT returns the set T . This algorithm is only slightly different from the algorithm examined in [7]: it starts with an arbitrary vertex rather than starting with two vertices connected by a maximum weight edge. However, starting with an arbitrary vertex turns out to be sufficient to guarantee an approximation ratio of 2 and, as shown in Algorithm 1, makes it possible to implement the algorithm in $O(pn)$ time instead of the $O(n^2)$ time necessary to find the maximum weight edge.

For an instance $(G = (V, E), p)$ of the remote-clique problem, let OPT be the average edge weight in an optimal solution. To prove that GREEDY AUGMENT achieves an approximation ratio of 2, we will prove that at each augmenting step, there exists a vertex $v \notin T$ that can be added to T , such that the average weight of the new edges introduced is at least $\frac{1}{2}OPT$.

Lemma 1. *At each step in the algorithm, there is a vertex $v \notin T$ such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2}OPT$.*

Proof. Deferred. □

With this fact, it is straightforward to prove that GREEDY AUGMENT achieves an approximation ratio of 2.

Theorem 2. *The average weight of the edges in the solution returned by GREEDY AUGMENT is at least $\frac{1}{2}OPT$.*

Proof. By Lemma 1, it is easy to establish by induction on $|T|$ that at every step in the algorithm, $2/(|T|(|T| - 1)) \cdot \sum_{v_1, v_2 \in T} w(v_1, v_2) \geq \frac{1}{2}OPT$. In other words, at every step, the average edge weight in T is at least one-half the average edge weight of an optimal solution. Thus, when the algorithm terminates, T is a set of vertices of size p with an average edge weight of at least $\frac{1}{2}OPT$. □

To prove Lemma 1, we begin by defining some notation. Consider an intermediate state of GREEDY AUGMENT. The set of vertices chosen so far is T , and let S be the set of vertices

Algorithm 1 GREEDY AUGMENT

```
{assume vertices in  $V$  are arbitrarily labeled  $v_1, \dots, v_{|V|}$ }
for  $i = 1$  to  $|V|$  do
   $weight(v_i) \leftarrow 0$  { $weight$  is an array keeping track of current distance to each vertex}
end for
 $T \leftarrow \{v_1\}$ 
 $last \leftarrow v_1$  {last vertex added to  $T$ }
while  $|T| < p$  do
  for  $i = 1$  to  $|V|$  do
     $weight(v_i) \leftarrow weight(v_i) + w(v_i, last)$ 
  end for
   $maxweight \leftarrow -\infty$ 
  for  $i = 1$  to  $|V|$  do
    if  $v_i \notin T$  and  $weight(v_i) > maxweight$  then
       $maxweightv \leftarrow v_i$  {maximum weight vertex}
       $maxweight \leftarrow weight(v_i)$ 
    end if
  end for
   $T \leftarrow T \cup \{maxweightv\}$ 
   $last \leftarrow maxweightv$ 
end while
return  $T$ 
```

in an optimal solution. Let $u = |S \cap T|$, $t = |T - S|$, and $s = |S - T|$. Arbitrarily label the vertices in $S \cap T$ as $a_1, a_2, \dots, a_u$, the vertices in $T - S$ as $b_1, b_2, \dots, b_t$, and the vertices in $S - T$ as $c_1, c_2, \dots, c_s$. Note that either u or t may be equal to zero, but since $1 \leq |T| < |S|$, $u + t \geq 1$ and $t < s$. This situation is illustrated in Figure 1.

We break the proof of Lemma 1 into five cases based on the values of u and t and state the proof for each case as its own lemma. Factor-revealing linear programs seem to be necessary to prove the first three cases, but the last two can be proved using more direct arguments. Lemma 1 states that there exists a vertex that we can augment T to for which the average length of the new edges is at least $\frac{1}{2}\text{OPT}$. For each of these five cases, we show that in fact the set $S - T$ contains such a vertex. We start with the most general (and hardest) case, when $u \geq 2$ and $t \geq 1$.

Lemma 3. *If $u \geq 2$ and $t \geq 1$, then set $S - T$ contains a vertex v such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2}\text{OPT}$.*

Proof. To prove this lemma, we show that

$$\sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) + \sum_{j=1}^t \sum_{\ell=1}^s w(b_j, c_\ell) \geq \frac{s(u+t)}{2} \text{OPT} \quad (1)$$

Figure 1: An intermediate state of GREEDY AUGMENT.

If this is true, then by rearranging terms we have

$$\sum_{\ell=1}^s \left(\sum_{h=1}^u w(a_h, c_\ell) + \sum_{j=1}^t w(b_j, c_\ell) \right) \geq \frac{s(u+t)}{2} \text{OPT}$$

which means that there must exist an ℓ such that $1 \leq \ell \leq s$ and

$$\begin{aligned} \sum_{h=1}^u w(a_h, c_\ell) + \sum_{j=1}^t w(b_j, c_\ell) &\geq \frac{u+t}{2} \text{OPT} \\ \sum_{v' \in T} w(v', c_\ell) &\geq \frac{|T|}{2} \text{OPT} \end{aligned}$$

Hence c_ℓ is the vertex v that satisfies the statement of the lemma. To prove equation (1), we first make the following observations based on the triangle inequality:

$$w(a_h, c_\ell) + w(a_h, c_m) - w(c_\ell, c_m) \geq 0 \quad 1 \leq h \leq u, 1 \leq \ell < m \leq s \quad (2)$$

$$w(b_j, c_\ell) + w(b_j, c_m) - w(c_\ell, c_m) \geq 0 \quad 1 \leq j \leq t, 1 \leq \ell < m \leq s \quad (3)$$

$$w(a_h, c_\ell) + w(a_i, c_\ell) - w(a_h, a_i) \geq 0 \quad 1 \leq h < i \leq u, 1 \leq \ell \leq s \quad (4)$$

Also, since S is an optimal solution, the sum of the edge weights in S must be at least $\binom{p}{2} \text{OPT} = \frac{1}{2}(u+s)(u+s-1) \text{OPT}$. Therefore,

$$\sum_{h=1}^u \sum_{i=h+1}^u w(a_h, a_i) + \sum_{\ell=1}^s \sum_{m=\ell+1}^s w(c_\ell, c_m) + \sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) \geq \frac{(u+s)(u+s-1)}{2} \text{OPT} \quad (5)$$

It does not seem to be easy to directly manipulate inequalities (2), (3), (4), and (5), to yield (1), but we can prove (1) by viewing these inequalities as the constraints of a factor-revealing linear program. In particular, consider an adversary trying to minimize

$$\sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) + \sum_{j=1}^t \sum_{\ell=1}^s w(b_j, c_\ell) \quad (6)$$

subject to constraints (2), (3), (4), and (5). If we can show that the optimal value of this linear program is no less than $\frac{1}{2}s(u+t)\text{OPT}$, then we will have proved (1). Since the value of any feasible dual solution lower bounds the optimal value of the primal, we can prove (1) by finding a feasible dual solution with value $\frac{1}{2}s(u+t)\text{OPT}$. The dual linear program is

$$\begin{aligned}
& \text{maximize} && \frac{(u+s)(u+s-1)}{2} \text{OPT} \cdot z \\
& \text{subject to} && -\sum_{\ell=1}^s y_{hil} + z \leq 0 && 1 \leq h < i \leq u \\
& && -\sum_{h=1}^u w_{h\ell m} - \sum_{j=1}^t x_{j\ell m} + z \leq 0 && 1 \leq \ell < m \leq s \quad (7) \\
& && \sum_{n=\ell+1}^s w_{h\ell n} + \sum_{n=1}^{\ell-1} w_{hnl} + \sum_{n=h+1}^u y_{hnl} + \sum_{n=1}^{h-1} y_{nh\ell} + z \leq 1 && 1 \leq h \leq u, 1 \leq \ell \leq s \\
& && \sum_{n=\ell+1}^s x_{j\ell n} + \sum_{n=1}^{\ell-1} x_{jnl} \leq 1 && 1 \leq j \leq t, 1 \leq \ell \leq s
\end{aligned}$$

where $w_{h\ell m}$ corresponds to (2), $x_{j\ell m}$ corresponds to (3), y_{hil} corresponds to (4), and z corresponds to (5). It can easily be verified that the following dual solution is feasible.

$$\begin{aligned}
w'_{h\ell m} &= \frac{s-t}{(u+s)(s-1)} && 1 \leq h \leq u, 1 \leq \ell < m \leq s \\
x'_{j\ell m} &= \frac{1}{s-1} && 1 \leq j \leq t, 1 \leq \ell < m \leq s \\
y'_{hil} &= \frac{u+t}{(u+s)(u+s-1)} && 1 \leq h < i \leq u, 1 \leq \ell \leq s \\
z' &= \frac{s(u+t)}{(u+s)(u+s-1)}
\end{aligned}$$

The only constraint that is not trivial to verify is (7), but some straightforward algebraic manipulation shows that the left-hand side is equal to

$$-\frac{su(u+t)}{(u+s)(u+s-1)(s-1)}$$

which is no greater than 0 since $s \geq 2$ when $t \geq 1$. The value of this dual solution is

$$\frac{(u+s)(u+s-1)}{2} \text{OPT} \cdot z' = \frac{s(u+t)}{2}$$

which implies that the optimal value of the primal is at least $\frac{1}{2}s(u+t)$ and hence implies inequality (1), thus proving the lemma. $\square$

The next case that we consider is when $u = 1$ and $t \geq 1$. We need to consider this separately because in this case, we do not have constraint (4) (or dual variables of the form y_{hil}). However, the proof for this case is very similar.

Lemma 4. *If $u = 1$ and $t \geq 1$, then set $S - T$ contains a vertex v such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2} \text{OPT}$.*

Proof. As in Lemma 3, it is sufficient to prove that

$$\sum_{\ell=1}^s w(a_1, c_\ell) + \sum_{j=1}^t \sum_{\ell=1}^s w(b_j, c_\ell) \geq \frac{s(t+1)}{2} \text{OPT}$$

Again, we prove this by lower-bounding the optimal solution to the following linear program, in which the constraints are derived by the triangle inequality and the optimality of S .

$$\begin{aligned} & \text{minimize} && \sum_{\ell=1}^s w(a_1, c_\ell) + \sum_{j=1}^t \sum_{\ell=1}^s w(b_j, c_\ell) \\ & \text{subject to} && w(a_1, c_\ell) + w(a_1, c_m) - w(c_\ell, c_m) \geq 0 && 1 \leq \ell < m \leq s \\ & && w(b_j, c_\ell) + w(b_j, c_m) - w(c_\ell, c_m) \geq 0 && 1 \leq j \leq t, 1 \leq \ell < m \leq s \\ & && \sum_{\ell=1}^s w(a_1, c_\ell) + \sum_{\ell=1}^s \sum_{m=\ell+1}^s w(c_\ell, c_m) \geq \frac{s(s+1)}{2} \text{OPT} \end{aligned}$$

The dual of this linear program is

$$\begin{aligned} & \text{maximize} && \frac{s(s+1)}{2} \text{OPT} \cdot z \\ & \text{subject to} && -x_{\ell m} - \sum_{j=1}^t y_{j\ell m} + z \leq 0 && 1 \leq \ell < m \leq s \\ & && \sum_{n=\ell+1}^s x_{\ell n} + \sum_{n=1}^{\ell-1} x_{n\ell} + z \leq 1 && 1 \leq \ell \leq s \\ & && \sum_{n=\ell+1}^s y_{j\ell n} + \sum_{n=1}^{\ell-1} y_{jn\ell} \leq 1 && 1 \leq j \leq t, 1 \leq \ell \leq s \end{aligned}$$

We conclude the proof of this lemma by providing the following dual solution, which can easily be shown to be feasible and to have the value $\frac{1}{2}s(t+1)\text{OPT}$.

$$\begin{aligned} x'_{\ell m} &= \frac{s-t}{(s+1)(s-1)} && 1 \leq \ell < m \leq s \\ y'_{j\ell m} &= \frac{1}{s-1} && 1 \leq j \leq t, 1 \leq \ell < m \leq s \\ z' &= \frac{t+1}{s+1} \end{aligned}$$

□

The next case we consider is when $u \geq 2$ and $t = 0$. Although the factor-revealing LP is slightly different, the proof for this case is very similar to the other proofs presented so far.

Lemma 5. *If $u \geq 2$ and $t = 0$, then set $S-T$ contains a vertex v such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2} \text{OPT}$.*

Proof. Again, it is sufficient to prove that

$$\sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) \geq \frac{su}{2} \text{OPT}$$

which we do by lower-bounding the optimal solution to the following linear program.

$$\begin{aligned} & \text{minimize} && \sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) \\ & \text{subject to} && w(a_h, c_\ell) + w(a_h, c_m) - w(c_\ell, c_m) \geq 0 && 1 \leq h \leq u, 1 \leq \ell < m \leq s \\ & && w(a_h, c_\ell) + w(a_i, c_\ell) - w(a_h, a_i) \geq 0 && 1 \leq h < i \leq u, 1 \leq \ell \leq s \\ & && \sum_{h=1}^u \sum_{i=h+1}^u w(a_h, a_i) + \sum_{\ell=1}^s \sum_{m=\ell+1}^s w(c_\ell, c_m) + \sum_{h=1}^u \sum_{\ell=1}^s w(a_h, c_\ell) \geq \frac{(u+s)(u+s-1)}{2} \text{OPT} \end{aligned}$$

The dual of this linear program is

$$\begin{aligned} & \text{maximize} && \frac{(u+s)(u+s-1)}{2} \text{OPT} \cdot z \\ & \text{subject to} && -\sum_{\ell=1}^s y_{hil} + z \leq 0 && 1 \leq h < i \leq u \\ & && -\sum_{h=1}^u x_{h\ell m} + z \leq 0 && 1 \leq \ell < m \leq s \\ & && \sum_{n=\ell+1}^s x_{h\ell n} + \sum_{n=1}^{\ell-1} x_{hnl} + \sum_{n=h+1}^u y_{hnl} + \sum_{n=1}^{h-1} y_{nh\ell} + z \leq 1 && 1 \leq h \leq u, 1 \leq \ell \leq s \end{aligned}$$

The following dual solution is feasible and has value $\frac{1}{2}su\text{OPT}$, thus concluding the proof of the lemma.

$$\begin{aligned} x'_{h\ell m} &= \frac{s}{(u+s)(u+s-1)} && 1 \leq h \leq u, 1 \leq \ell < m \leq s \\ y'_{hil} &= \frac{u}{(u+s)(u+s-1)} && 1 \leq h < i \leq u, 1 \leq \ell \leq s \\ z' &= \frac{su}{(u+s)(u+s-1)} \end{aligned}$$

□

The final two cases are simple enough that it is not necessary to use factor-revealing linear programs. We begin with the case when $u = 0$. The idea of this proof is also found in [7].

Lemma 6. *If $u = 0$, then set $S - T$ contains a vertex v such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2} \text{OPT}$.*

Proof. By the triangle inequality,

$$w(b_j, c_\ell) + w(b_j, c_m) \geq w(c_\ell, c_m) \quad 1 \leq j \leq t, 1 \leq \ell < m \leq s$$

Summing over all j , this becomes

$$\sum_{j=1}^t w(b_j, c_\ell) + \sum_{j=1}^t w(b_j, c_m) \geq t \cdot w(c_\ell, c_m) \quad 1 \leq \ell < m \leq s$$

Adding up each of the $\binom{s}{2}$ instances of this inequality yields

$$(s-1) \sum_{\ell=1}^s \sum_{j=1}^t w(b_j, c_\ell) \geq t \sum_{\ell=1}^s \sum_{m=\ell+1}^s w(c_\ell, c_m) = \frac{ts(s-1)}{2} \text{OPT}$$

where the equality follows from the optimality of S . But this implies that

$$\sum_{\ell=1}^s \sum_{j=1}^t w(b_j, c_\ell) \geq \frac{ts}{2} \text{OPT}$$

which is the same as inequality (1) when $u = 0$ and which thus implies the lemma. $\square$

The final case, when $t = 0$ and $u = 1$, follows from a simple contradiction argument.

Lemma 7. *If $u = 1$, and $t = 0$ then set $S - T$ contains a vertex v such that $\frac{1}{|T|} \sum_{v' \in T} w(v', v) \geq \frac{1}{2} \text{OPT}$.*

Proof. Since $|T| = 1$, we need only show that there exists a vertex c_ℓ , $1 \leq \ell \leq s$, such that $w(u_1, c_\ell) \geq \frac{1}{2} \text{OPT}$. Suppose by way of contradiction that there is no such vertex. Then by the triangle inequality, $w(c_\ell, c_m) < \text{OPT}$ for $1 \leq \ell < m \leq s$. But this contradicts the optimality of S , since it implies that every edge in S has a weight strictly less than OPT . Thus we conclude that set $S - T$ does indeed contain a vertex c_ℓ satisfying the statement of the lemma. $\square$

Lemmas 3, 4, 5, 6, and 7 together imply Lemma 1, which in turn implies Theorem 2, stating that GREEDY AUGMENT is a 2-approximation.

3 Conclusion

For the remote-clique problem, we have reexamined the algorithm GREEDY AUGMENT and shown that it achieves an approximation ratio of 2, matching the best approximation ratio known for the problem [3]. GREEDY AUGMENT’s running time of $O(pn)$ makes it the fastest known 2-approximation to remote-clique, and it also has the advantage of being very easy to implement, as shown in Algorithm 1. Finally, because it seems that the technique of factor-revealing LPs lends itself very naturally to remote-clique, it would be an interesting direction for future research to examine how it applies to other dispersion problems.

References

- [1] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *OSDI ’99: Proceedings of the third symposium on Operating systems design and implementation*, pages 173–186, Berkeley, CA, USA, 1999. USENIX Association.
- [2] Barun Chandra and Magnus M. Halldorsson. Approximation algorithms for dispersion problems. *J. Algorithms*, 38(2):438–465, 2001.
- [3] R. Hassin, S. Rubinstein, and A. Tamir. An approximation algorithm for maximum dispersion. *Operations Research Letters*, 21:133–137, 1997.
- [4] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *J. ACM*, 50(6):795–824, 2003.
- [5] M.J. Kuby. Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems. *Geographical Analysis*, 19(4):315–329, 1987.
- [6] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized on-line matching. In *FOCS ’05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 264–273, Washington, DC, USA, 2005. IEEE Computer Society.
- [7] S. S. Ravi, D. J. Rosencrantz, and G. K. Tayi. Heuristic and special case algorithms for dispersion problems. *Operations Research*, 42:299–310, 1994.