Journal of Algorithms **34,** 203–221 (2000)
doi:10.1006/jagm.1999.1062, available online at http://www.idealibrary.com on IDEAL®

Greedily Finding a Dense Subgraph

Yuichi Asahiro

Department of Computer Science & Communication Engineering, Kyushu University, Fukuoka 812, Japan
E-mail: asahiro@csce.kyushu-u.ac.jp

Kazuo Iwama[1]

School of Informatics, Kyoto University, Kyoto 606-01, Japan
E-mail: iwama@kuis.kyoto-u.ac.jp

Hisao Tamaki[2]

Department of Computer Science, Meiji University, Kawasaki 214-71, Japan
E-mail: tamaki@cs.meiji.ac.jp

and

Takeshi Tokuyama

IBM Tokyo Research Laboratory, Yamato 242, Japan
E-mail: ttoku@trl.ibm.co.jp

Received August 20, 1996

Given an n-vertex graph with nonnegative edge weights and a positive integer $k \le n$, our goal is to find a k-vertex subgraph with the maximum weight. We study the following greedy algorithm for this problem: repeatedly remove a vertex with the minimum weighted-degree in the currently remaining graph, until exactly k vertices are left. We derive tight bounds on the worst case approximation ratio R of this greedy algorithm: $(1/2 + n/2k)^2 - O(n^{-1/3}) \le R \le (1/2 + n/2k)^2 + O(1/n)$ for k in the range $n/3 \le k \le n$ and $2(n/k - 1) - O(1/k) \le R \le 2(n/k - 1) + O(n/k^2)$ for $k < n/3$. For $k = n/2$, for example, these bounds are $9/4 \pm O(1/n)$, improving on naive lower and upper bounds of 2 and 4, respectively.

[1] Research supported in part by Science Research Grant 07458061, Ministry of Education, Japan.

[2] Main part of the work done while author was at IBM Tokyo Research Laboratory.

0196-6774/00 $35.00

The upper bound for general k compares well with currently the best (and much more complicated) approximation algorithm based on semidefinite programming.

1. INTRODUCTION

In the *maximum edge subgraph problem*, we are given an n-vertex graph with nonnegative edge weights and a positive integer $k \leq n$ and are asked to find a k-vertex subgraph with the maximum weight. Since the problem is trivial for $k \leq 2$, we assume $k \geq 3$ throughout the paper. Being a generalization of the maximum clique problem, this problem is obviously NP-hard. The decision problem version of the problem for unweighted graphs, in which one is asked if there is a k-vertex graph with more than m edges, is NP-complete even if m is restricted by $m \leq k^{1+\epsilon}$ [AI95]. Naturally, we seek efficient algorithms for approximate solutions [RRT94, KP93, AKK95, AI95, FS97, SW98]. The problem is related to facility dispersion [RRT94] and to the sparsest spanner problem [KP94]. It may also find application in data mining, where diverse methods of data clustering are being actively explored.

In the following discussions, the *approximation ratio* of a solution is OPT/A, where OPT is the weight of the optimal subgraph and A is the weight of the given solution. The (*worst case*) *approximation ratio* of an algorithm is the supremum, over all inputs (and, when the algorithm is nondeterministic, over all nondeterministic choices the algorithm makes), of the approximation ratio of the solution it produces. Ravi *et al.* [RRT94] studied the case where the weight function satisfies the triangle inequality (in the context of the facility dispersion problem) and showed that a greedy algorithm achieves the approximation ratio 4. They also gave an algorithm with the approximation ratio 2 in this special case. Kortsarz and Peleg [KP93] studied the problem with a general weight function and constructed an algorithm with approximation ratio $O(n^{7/18})$. Recently, Arora *et al.* [AKK95] have developed a framework for approximately solving dense instances of NP-hard problems and applied it to the maximum edge subgraph problem: the result is a PTAS (i.e., a polynomial time algorithm A_ϵ with approximation ratio $(1 + \epsilon)$, for each constant $\epsilon > 0$) for the special cases where the graph is unweighted and (1) the graph has $\Omega(n^2)$ edges and $k = \Omega(n)$ or (2) each vertex of the graph has degree $\Omega(n)$.

Yet more recently, after the first submission of the present paper, some authors developed algorithms based on semidefinite programming relaxations: the algorithm of Feige and Seltser [FS97] achieves an approximation ratio of roughly n/k and that of Srivastav and Wolf [SW98] slightly outperforms this ratio when k is around $n/3$.

The greedy algorithm of Ravi *et al.* [RRT94] starts from a heaviest edge and repeats adding a vertex to the current subgraph that maximizes the weight of the resulting new subgraph, until k vertices are chosen. Although this algorithm works well for a weight function with the triangle inequality, it fails miserably for a general weight function, as pointed out by Kortsarz and Peleg [KP93].

What we study in this paper is an alternative greedy algorithm which we call GREEDY: given G and a weight function,

1. Set $X = V(G)$.
2. If $|X| = k$ then output $G[X]$. Otherwise, find a vertex in X with the minimum weighted degree in $G[X]$ under the given weight function and remove it from X. Repeat Step 2.

Here, $G[X]$ denotes the subgraph of G induced by X. This algorithm can be viewed as a derandomization (using the method of conditional probability [Rag88]) of a randomized algorithm that picks a random set of k vertices. Since the expected weight of a random k-vertex subgraph of G is $\frac{k(k-1)}{n(n-1)}$ times the total weight of G (each edge of weight w contributes $\frac{k(k-1)}{n(n-1)}w$ to the expectation), we immediately have an $\frac{n(n-1)}{k(k-1)}$ upper bound on the approximation ratio, which is, for example, $4 + o(1)$ when $k = n/2$. On the other hand, when $k = n/2$, there is a particularly simple lower bound of $2 - o(1)$: an unweighted graph consisting of $n/4$ independent edges and a path of $n/2$ vertices. When GREEDY is applied to this graph, it may leave the $n/4$ independent edges as its solution while the optimal is the path with $n/2 - 1$ edges.

In what follows, we prove upper and lower bounds on the approximation ratio of GREEDY, which are tight up to an $o(1)$ additive term as long as $k = \omega(\sqrt{n})$.

THEOREM 1. *The worst-case approximation ratio R of GREEDY satisfies*

$$\left(\frac{1}{2} + \frac{n}{2k}\right)^2 - O(n^{-1/3}) \le R \le \left(\frac{1}{2} + \frac{n}{2k}\right)^2 + O(1/n) \qquad \text{for } n/3 < k \le n,$$

and

$$2\left(\frac{n}{k} - 1\right) - O(1/k) \le R \le 2\left(\frac{n}{k} - 1\right) + O(n/k^2) \qquad \text{for } k \le n/3.$$

For the special case $k = n/2$, the bounds are $9/4 \pm o(1)$ improving the naive bounds of 2 and 4. For $\sqrt{n} < k \leq n/3$, the upper bound improves the above naive analysis by a factor of $n/2k$.

The algorithm of Kortsarz and Peleg [KP93] uses a subprocedure called LARGE, which solves the problem with an approximation ratio $O(n/k)$, where the constant in the O notation is at least 4. The advantage of GREEDY lies partly in its smaller constant and mostly in its simplicity: procedure LARGE is a combination of several ideas including binary search, ''expanding kernels,'' and the method of conditional probability. Their algorithm also includes a subprocedure that solves the unweighted version of the problem with approximation ratio $O((n/k)^{2/3})$, but only when the number of edges in the optimal solution is larger than $\sqrt{k^5/n}$.

In combinatorial optimization problems in general, greedy heuristics are particularly attractive for their simplicity and it is important to determine how well they perform in terms of approximation. Our result adds to the interesting cases where tight bounds on the approximation ratio are obtained through nontrivial analysis. See Grötschel and Lovász [GL95] for other examples and general discussion on greedy heuristics.

Although we study the problem for weighted graphs, the special case of unweighted graphs may be of special interest. It is clear that our upper bounds apply to unweighted graphs. On the other hand, since the optimal solution can have at most $\binom{k}{2}$ edges, a trivial upper bound of $O(k)$ applies to any algorithm that outputs a subgraph with $\Omega(k)$ edges; it certainly applies to GREEDY since the solution of GREEDY can contain at most one isolated vertex (unless the original graph contains more than $n - k$ isolated vertices, in which case GREEDY finds the optimal solution). This upper bound is stronger than our general upper bound when $k \leq \sqrt{n}$. For $k \geq cn^{3/4}$ where c is some positive constant, however, our general upper bound is tight even when restricted to unweighted graphs, since a matching lower bound can be exhibited by an unweighted graph for k in this range. For $k \leq cn^{3/4}$, our lower bound example does use nonunit weights. Thus, we do not know whether our upper bound is tight for unweighted graphs for the range $\sqrt{n} \leq k \leq cn^{3/4}$.

The rest of the paper is organized as follows. Section 2 contains a few definitions and notation. In Section 3 we describe examples that establish the lower bounds in Theorem 1. Then in Section 4 we prove the upper bound. Although the two sections can be read independent of each other, the authors view them as an unsplittable pair: the upper bound proof would have never been discovered without the guidance of the lower bound examples and the understanding of one may give some insight into the other.

2. DEFINITIONS

In this paper, a graph is simple, i.e., without a self-loop or parallel edges, unless otherwise stated. For a graph G, we denote the vertex set by $V(G)$ and the edge set by $E(G)$, as usual. The subgraph of G induced by a vertex set $U \subseteq V(G)$ is denoted by $G[U]$.

A weighted graph is a pair (G, w), where G is a graph and w: $E(G) \to \mathrm{R}^+$ is a *weight function* that assigns a nonnegative real to each edge of G. The *weighted degree* or simply *degree* of a vertex v in the weighted graph (G, w) is $\sum w(e)$ where e ranges over all the edges incident to v. The *average degree* of a nonempty vertex set $U \subseteq V(G)$ in (G, w) is $\sum_{v \in U} d(v)/|U|$ where $d(v)$ denotes the degree of v in (G, w). The *average degree* of a weighted graph (G, w) is the average degree of $V(G)$ in (G, w). Clearly, the average degree of (G, w) is equal to $2w(G)/|V(G)|$, where $w(G)$ denotes the total weight of G. A weighted graph (G, w) is *d-regular* if every vertex of it is of weighted degree d. We sometimes regard an unweighted graph G as a weighted graph $(G, 1)$, where 1 denotes the weight function that assigns 1 to each edge, and apply the above definitions. Finally, if (G, w) is a weighted graph and H is a subgraph of G, we abuse the notation and write (H, w) to denote the weighted graph in which the weight function w is restricted to the edges of H.

3. LOWER BOUND EXAMPLES

The goal of this section is to exhibit examples that establish the lower bound in Theorem 1.

We first deal with the case $k \leq n/3$. Let K_n denote the complete weighted graph on n-vertices in which each edge is assigned a weight of 1 and $P_{n,d}$ a path on n vertices in which each edge is assigned a weight of d. Our lower bound example is the graph consisting of two connected components K_{n-k} and $P_{k,n-k-1}$. A possible execution of GREEDY on this weighted graph removes $P_{k,n-k-1}$ first and then peels off vertices of $V(K_{n-k})$ until it obtains K_k as the solution. Since the weight of $P_{k,n-k-1}$ is $(n-k-1)(k-1)$ and the weight of K_k is $k(k-1)/2$, the approximation ratio of this solution is $2(n/k-1) - O(1/k)$. This gives the lower bound in Theorem 1 for the case $k \leq n/3$. Note that, although this lower bound is valid as long as $k \leq n/2$, it is weaker than the bound in the theorem when $k > n/3$.

We now deal with the case $k > n/3$. Our construction in this case does not use nonunit weights; i.e., the graph we construct is an unweighted graph. We remark that the above construction for $k \leq n/3$ can also be

replaced by a nonweighted version if $k \geq cn^{3/4}$, where c is a sufficiently large constant, using similar ideas.

Fix k and n, with $n/3 < k$. We may assume that $k < n - n^{2/3}$, since otherwise the trivial lower bound of 1 can be expressed as $(1/2 + n/2k)^2 - O(n^{-1/3})$. For the following construction, it is convenient to assume that k has a divisor s in the range $n^{2/3} \leq s < 2n^{2/3}$. We will later show how to remove this assumption. Let $k = sd$ and let p be an integer in the range $1 \leq p \leq sk/n$; p is a parameter to be tuned later in order to obtain a best possible lower bound. Note that, from the assumption on the range of k, we have $p < s(1 - n^{-1/3}) \leq s - n^{1/3}$. Given these parameters k, n, s, d, and p, we define a graph G as follows.

The structure of our graph G is illustrated in Fig. 1. The vertex set $V(G)$ of G is a disjoint union of U and W, $|U| = k$. The subset U is going to be the optimal solution; i.e., $G[U]$ will have the largest number of edges among all the k-vertex subgraphs. Within $V(G)$ is a chain of subsets $X_0 \subset X_2 \subset \cdots \subset X_h \subseteq V(G)$, such that $|X_i| = s(d + i)$, where h will be determined later. The set of edges will be defined so that GREEDY can give $G[X_0]$ as its solution. Note $|X_0| = sd = k$. Let $U_i = X_i \cap U$ and $W_i = X_i \cap W$. Subsets X_i are chosen so that $|U_i| = p(d + i)$ and $|W_i| = (s - p)(d + i)$. In other words, X_i intersects with U and W at a fixed proportion p: $(s - p)$. The condition on p, $p \leq sk/n$, implies that this ratio $p/(s - p)$ is not greater than $k/(n - k) = |U|/|W|$. This means that the maximum possible value of h is determined from the size constraint on

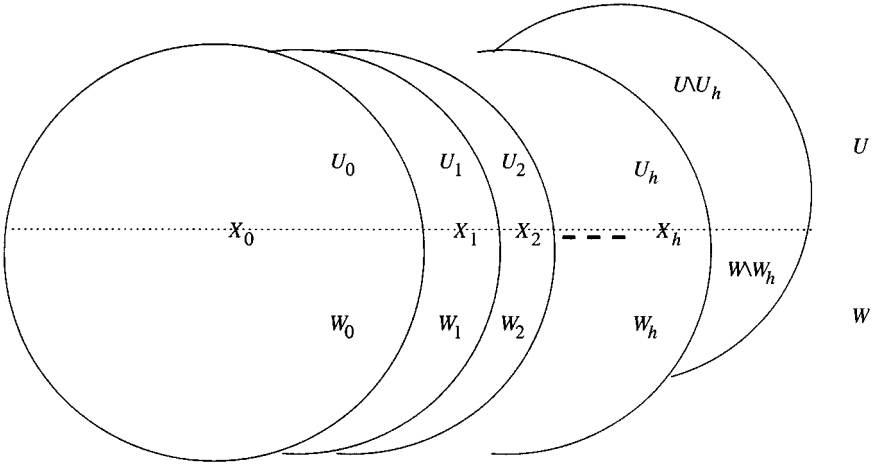

FIG. 1. Lower bound example: $|X_0| = |U| = k = sd$, $|X_i| = s(d + i)$, $|U_i| = p(d + i)$, and $|W_i| = (s - p)(d + i)$.

W: $(s-p)(d+h) \le n-k$. We set $h = \lfloor (n-k)/(s-p) \rfloor - d$. Our graph G is the union of the following subgraphs.

1. arbitrary $(d+1)$-regular graphs $G[U_0]$ and $G[W_0]$ (since $|U_0|$ and $|W_0|$ are multiples of d and hence even whenever $d+1$ is odd, such regular graphs always exist);

2. for each i, $1 \le i \le h$,

(a) an arbitrary bipartite graph between vertex sets U_{i-1} and $U_i \setminus U_{i-1}$ such that the degree of each vertex of U_{i-1} is 1 and the degree of each vertex of $U_i \setminus U_{i-1}$ is $|U_{i-1}|/p = d+i-1$,

(b) an arbitrary bipartite graph between vertex sets W_{i-1} and $W_i \setminus W_{i-1}$ such that the degree of each vertex of W_{i-1} is 1 and the degree of each vertex of $W_i \setminus W_{i-1}$ is $|W_{i-1}|/(s-p) = d+i-1$, and

(c) a cycle on $X_i \setminus X_{i-1}$, such that only two edges are between $U_i \setminus U_{i-1}$ and $W_i \setminus W_{i-1}$;

3. an arbitrary bipartite graph between vertex sets U_h and $U \setminus U_h$ such that the degree of each vertex in $U \setminus U_h$ is $d+h+1$.

It is easy to confirm that $G[U_i \cup W_i]$ is $(d+i+1)$-regular, for $0 \le i \le h$. A possible execution of GREEDY on G proceeds as follows. First, all the vertices in $W \setminus W_h$ which have degree 0 are removed. The minimum degree of the remaining graph is $d+h+1$ and next all the vertices of $U \setminus U_h$ are removed in sequence. This leaves us with the $(d+h+1)$-regular graph $G[X_h]$. The remaining execution consists of h phases, the jth of which removes all the vertices in $X_i \setminus X_{i-1}$, where $i = h-j+1$. To confirm that such an execution is indeed allowed, suppose we are left with X_i as the current set of vertices. Since the remaining graph is $(d+i+1)$-regular, we may start the next phase by removing a vertex from $X_i \setminus X_{i-1}$. Once this is done, we may always find a vertex of degree $d+i$ in $X_i \setminus X_{i-1}$ (or of degree $d+i-1$ at the last step of this phase), while the degree of each vertex in X_{i-1} remains at least $d+i$ during the phase. Thus, by induction, this execution of GREEDY leaves the vertex set X_0 as the solution.

Now we can compute the approximation ratio of this greedy solution. The number of edges in $G[X_0]$ is $k(d+1)/2$. The number of edges in $G[U]$ is calculated as follows. Let $x = |U_h|/k$. Since $G[U_h \cup W_h]$ is $(d+h+1)$-regular and there are only $2h$ edges between U_h and W_h, the number of edges in $G[U_h]$ is $|U_h|(d+h+1)/2 - h = xk(d+h+1)/2 - h$. We have additional $k(1-x)(d+h+1)$ edges between U_h and $U \setminus U_h$. Therefore, the total number of edges in $G[U]$ is $k(d+h+1)(1-x/2) - h$. Since $s(d+h) = |X_h| \le n$, we have $h \le n/s - d < 3k/s - d = 2d$ and

hence the approximation ratio R_0 of this greedy solution is

$$R_0 = \frac{k(d+h+1)(1-x/2) - h}{k(d+1)/2} \tag{1}$$

$$= \frac{(d+h+1)(2-x)}{d+1} - O(1/k). \tag{2}$$

We may represent the fraction $(d+h+1)/(d+1)$ in term of x as follows. We have

$$\begin{aligned}(d+h)s &= |X_h| \\ &= n - |U \setminus U_h| - |W \setminus W_h| \\ &> n - k(1-x) - s,\end{aligned}$$

where the last inequality follows from the choice of h that implies $|W \setminus W_h| \le s$, and hence

$$\begin{aligned}\frac{d+h+1}{d+1} &> \frac{n-k(1-x)}{s(d+1)} \\ &= \frac{n-k(1-x)}{k+s} \\ &= n/k - (1-x) - O(n^{-1/3}).\end{aligned}$$

Thus, we have $R_0 \ge (2-x)(n/k - 1 + x - O(n^{-1/3})) - O(1/n)$, or

$$R_0 \ge (2-x)(n/k - 1 + x) - O(n^{-1/3}). \tag{3}$$

What remains to be done is to maximize the right-hand side choosing an appropriate value of p (that determines x). Set $r = n/k$. The quadratic function $F_r(x) = (2-x)(r-1+x)$, $1 < r < 3$, has its maximum $(r+1)^2/4$ at $x = (3-r)/2$. Therefore, the right-hand side of inequality (3) as a function of x attains its maximum $(1/2 + n/2k)^2 - O(n^{-1/3})$ at $x = x_0 = (3 - n/k)/2 + O(n^{-1/3})$. The following lemma establishes that we can make x sufficiently close to x_0; more specifically $|x - x_0| \le O(n^{-1/3})$, by choosing p appropriately. Thus, our construction gives the lower bound of $(1/2 + n/2k)^2 - O(n^{-1/3})$ as claimed in Theorem 1.

LEMMA 2. *For each y, $0 < y < 1$, there is an integer value of p such that the value of x corresponding to p satisfies $|x - y| \le O(n^{-1/3})$.*

Proof. Recall from the definition of h and x that x depends on p as $x(p) = p\lfloor (n-k)/(s-p) \rfloor /k$. If we put $f(p) = p(n-k)/k(s-p)$ ignoring the floor operation, we have $|f(p) - x(p)| \le p/k \le O(n^{-1/3})$. There-

fore, it suffices to show that, for each y, $0 < y < 1$, there is an integer p such that $|f(p) - y| \leq O(n^{-1/3})$. Since $f(t)$, for t real, is continuous and increasing in the range $0 \leq t \leq ks/n$ with $f(0) = 0$ and $f(ks/n) = 1$, it suffices to show that $f(p) - f(p-1) \leq O(n^{-1/3})$ for every $1 \leq p \leq ks/n$. This quantity,

$$f(p) - f(p-1) = \frac{n-k}{k}\left(\frac{p}{s-p} - \frac{p-1}{s-p+1}\right)$$
$$= \frac{(n-k)s}{k(s-p)(s-p+1)},$$

is increasing in p in the range $1 \leq p \leq ks/n$ and hence its maximum is

$$\frac{(n-k)s}{k(s-ks/n)(s-ks/n+1)} = \frac{n}{ks(1-k/n+1/s)}.$$

From our assumptions that $k > n/3$, $k < n - n^{2/3}$ (hence $1 - k/n > n^{-1/3}$), and $s \geq n^{2/3}$, we see that this value is $O(n^{-1/3})$. ∎

Finally, we need to remove the assumption that k is a multiple of some integer s in the range $n^{2/3} \leq s < 2n^{2/3}$. Fixing n, call k *simple* if it satisfies this assumption. For simple k, let $\mathsf{OPT}(k)$ denote the number of edges of $G[U]$ in the above construction and $\mathsf{GREEDY}(k)$ denote the number of edges in $G[X_0]$. For k that is not simple, we use the above construction G for k', where k' is the largest simple integer such that $k' < k$. Note that $k' \geq k - O(n^{2/3}) = k(1 - O(n^{-1/3}))$. The execution of GREEDY on this graph proceeds exactly as before, except that it stops when k vertices (instead of k' vertices) are left. The heaviest k-vertex subgraph of G has at least $\mathsf{OPT}(k')$ edges while the greedy solution contains at most $\mathsf{GREEDY}(k') + O(n)$ edges since the maximum degree of G is $O(n^{1/3})$. Therefore, the approximation ratio of this greedy solution is at least $\mathsf{OPT}(k')/(\mathsf{GREEDY}(k') + O(n)) = \mathsf{OPT}(k')/\mathsf{GREEDY}(k') - O(n^{-1/3})$, since $\mathsf{GREEDY}(k') = \Theta(n^{4/3})$ and $\mathsf{OPT}(k') = \Theta(n^{4/3})$. But $\mathsf{OPT}(k')/\mathsf{GREEDY}(k') = (1/2 + n/2k')^2 - O(n^{-1/3}) = (1/2 + n/2k)^2 - O(n^{-1/3})$.

4. UPPER BOUND

In this section, we give an upper bound that matches the lower bound construction of the previous section. An (n, k)-*instance* or an *execution instance*, when unambiguous, is a triple (G, w, σ), where (G, w) is a

weighted graph with $|V(G)| = n$ and σ is a sequence of distinct $n - k$ vertices of G. When we refer to an (n, k)-instance $(G, w, (v_n, \ldots, v_{k+1}))$, we will use V_i to represent $V(G)\backslash\{v_{i+1}, \ldots, v_n\}$, for $k < i \leq n$. We call an (n, k)-instance $(G, w, (v_n, \ldots, v_{k+1}))$ *greedy* if GREEDY may legally remove the vertices $v_n, \ldots, v_{k+1}$ in this order; i.e., v_i is the vertex of $(G[V_i], w)$ with the minimum weighted degree.

For a very technical reason, we compare the weight of the greedy solution to the weight of a solution which is not necessarily optimal. We say that a set U of k vertices of a weighted graph (G, w) is *quasioptimal* with respect to an execution instance $I = (G, w(v_n, \ldots, v_{k+1}))$ if $w(G[U])$ is the maximum under the constraint that $U \cap V_k \neq \emptyset$ and $U \neq V_k$. Let U be an optimal set of k vertices of (G, w). Replacing a vertex of the minimum degree in $(G[U], w)$ (whose degree is at most $2w(G[U])/k$) with some appropriate vertex, we obtain a vertex set U' with weight at least $w(G[U])(1 - 2/k)$ such that $U' \cap V_k \neq \emptyset$ and $U' \neq V_k$. Therefore, the weight of a quasioptimal solution is at least $(1 - 2/k)$ times the optimal. In the following, instead of bounding the worst case approximation ratio R, we bound the worst case ratio R' of a quasioptimal solution to a greedy solution. Since $R \leq R'(1 + 8/k)$ (as we are assuming $k \geq 3$), the inaccuracy incurred is absorbed by the $O(\ )$ error terms in the upper bounds of Theorem 1.

The strategy of our upper bound proof is as follows. Given an arbitrary greedy execution instance we transform the instance (by which we roughly mean changing the weight function w) into a certain canonical form without decreasing the approximation ratio and then analyze this canonical form to obtain the upper bound. The objects we are going to manipulate are in fact slightly abstract versions of execution instances, which we call *scenarios*, rather than execution instances themselves. Before formally defining scenarios, let us first discuss some underlying ideas. As soon as we try to change the weight function in an execution instance in such a way that greediness is preserved, we notice that it is a tricky process: simply moving some weight from one edge to another can destroy the greediness of the instance in several ways. So, let us consider relaxing the greediness condition somewhat to make transformations easier.

We call an (n, k)-instance $(G, w, (v_n, \ldots, v_{k+1}))$ *semigreedy* if the weighted degree of v_i in $G[V_i]$ is not greater than the average weighted degree of $(G[V_i], w)$ for $k < i \leq n$. Note that any greedy execution instance is semigreedy and therefore an upper bound on the approximation ratios of semigreedy instances will be an upper bound for greedy instances. The advantages of dealing with semigreedy instances is that the actual weight function w is not important in deciding if the given instance is semigreedy or not. For such a check, we only need the sequence $h(v_n), h(v_{n-1}), \ldots, h(v_{k+1})$, where $h(v_i)$ is the weighted degree of v_i in

$(G[V_i], w)$, and the average degree $\hat{h}$ of $(G[V_k], w)$. This is because the average degree $\hat{h}_i$ of $(G[V_i], w)$ is determined by $i\hat{h}_i = k\hat{h} + 2\sum_{k<j\le i} h(v_j)$, for $k < i \le n$. If we dealt with a structure in which the weight function w is replaced by h and $\hat{h}$, then preserving semigreediness in transformations would be easier than preserving greediness with explicit weight assignments. Unfortunately, by doing so we would lose information necessary to determine the approximation ratio of an execution instance: we would know the weight of the greedy solution but not the optimal (or quasioptimal) solution. To keep track of the weight of the quasioptimal solution U, we need to divide the weighed degree $h(v_i)$ into two parts: weights going into U and those going into $V(G)\setminus U$. These considerations lead to the following definition of scenarios.

An (n, k)-*scenario* is a 7-tuple $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$, where V is a set of n vertices, U is a subset of V with $|U| = k$, σ is a sequence of distinct $n - k$ vertices of V that contains at least one vertex of U but not all the vertices of U, $\hat{f}$ and $\hat{g}$ are nonnegative reals, and f, g are mappings from the set of vertices of σ to the set of nonnegative reals.

The following notation will be used freely whenever we refer to an (n, k)-scenario $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$. We let $\sigma = (v_n, v_{n-1}, \ldots, v_{k+1})$, $V_i = V\setminus\{v_{i+1}, \ldots, v_n\}$, and $W = V\setminus U$. We set $s_i = |V_i \cap U|$ and $t_i = |V_i \cap W|$, for $k \le i \le n$, and $h(v_i) = f(v_i) + g(v_i)$ for $k < i \le n$.

Given an (n, k)-instance $I = (G, w, \sigma)$ and $U \subseteq V(G)$ with $|U| = k$, we say that an (n, k)-scenario $S = (V', U', \sigma', \hat{f}, \hat{g}, f, g)$ corresponds to the pair (I, U) if all of the following conditions hold.

1. $V' = V(G)$, $U' = U$, and $\sigma' = \sigma$;
2. $f(v_i)$, for $k < i \le n$, is the total weight of the edges between v_i and $V_{i-1} \cap U$ under weight function w;
3. $g(v_i)$, for $k < i \le n$, is the total weight of the edges between v_i and $V_{i-1} \cap W$ under w;
4. $\hat{f}$ is the average degree of $V_k \cap U$ in $(G[V_k], w)$;
5. $\hat{g}$ is the average degree of $V_k \cap W$ in $(G[V_k], w)$.

Note that 2 and 3 imply that $h(v_i) = f(v_i) + g(v_i)$ is the weighted degree of v_i in $(G[V_i], w)$. We say that an (n, k)-scenario S *corresponds* to an execution instance I if S corresponds to (I, U) for some set of vertices U.

Given an (n, k)-scenario $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$, define $\hat{f}_i$ and $\hat{g}_i$, $k \le i \le n$ by

$$s_i\hat{f}_i = s_k\hat{f} + \sum_{k<j\le i} f(v_j) + \sum_{k<j\le i,\, v_j\in U} h(v_j),$$

and

$$t_i\hat{g}_i = t_k\hat{g} + \sum_{k<j\le i} g(v_j) + \sum_{k<j\le i,\, v_j\in W} h(v_j).$$

In particular, we have $\hat{f}_k = \hat{f}$ and $\hat{g}_k = \hat{g}$. Note that the restriction on σ in the definition of scenarios implies that both $V_k \cap U$ and $V_k \cap W$ are nonempty and hence $s_i > 0$ and $t_i > 0$ for $k \le i \le n$. The purpose of our earlier decision to compare the greedy solution to a quasioptimal solution is to eliminate the exceptional cases where $s_i = 0$ or $t_i = 0$. We remark that the awkwardness of this rather artificial arrangement can be avoided if one is willing to go through tedious details arising from such exceptional cases.

It is easy to verify the following proposition.

PROPOSITION 3. *If scenario $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$ corresponds to (I, U) where $I = (G, w, \sigma)$, then for each i, $k < i \le n$, $\hat{f}_i$ is the average degree of $V_i \cap U$ in $(G[V_i], w)$ and $\hat{g}_i$ is the average degree of $V_i \cap W$ in $(G[V_i], w)$.*

In view of this proposition, we call $\hat{f}_i$ ($\hat{g}_i$, resp.) the average degree of $V_i \cap U$ ($V_i \cap W$, resp) in scenario S, even when S does not explicitly correspond to an execution instance and hence no explicit weight function is involved.

We say that (n, k)-scenario S is *valid* if the following inequality is satisfied for $k < i \le n$:

$$h(v_i) \le \min\left(\hat{f}_i, \hat{g}_i\right). \tag{4}$$

LEMMA 4. *Let $I = (G, w, \sigma)$ be a greedy (n, k)-instance and U a quasioptimal k-vertex set with respect to I. Then, there is a unique (n, k)-scenario S that corresponds to (I, U) and, moreover, S is valid.*

Proof. Each component of the scenario $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$ is uniquely determined from (I, U) by each condition in the definition of correspondence. Note that the condition on σ in the definition of a scenario is satisfied because U is quasioptimal and hence $U \cap V_k \ne \emptyset$ and $U \ne V_k$. It remains to show that S is valid. Since S corresponds to I, $h(v_i)$ is the weighted degree of v_i in $(G[V_i], w)$ and, because I is greedy, $h(v_i)$ is not greater than the weighted degree of v in $(G[V_i], w)$ for any vertex $v \in V_i$. Therefore, $h(v_i)$ is not greater than the average degree of any vertex set in $(G[V_i], w)$; in particular, it is not greater than the average degree $\hat{f}_i$ of $V_i \cap U$ or the average degree $\hat{g}_i$ of $V_i \cap W$. ∎

Define the *value* of an (n, k)-scenario S by

$$\mathrm{val}(S) = \frac{s_k \hat{f} + 2\Sigma_{k < i \le n,\, v_i \in U} f(v_i)}{s_k \hat{f} + t_k \hat{g}}.$$

LEMMA 5. *Let $I = (G, w, \sigma)$ be a greedy (n, k)-instance and U a quasioptimal solution for (G, w). If S is the (n, k)-scenario that corresponds to (I, U), then* $\mathrm{val}(S) \ge w(G[U])/w(G[V_k])$.

Proof. The numerator in the definition of $\mathrm{val}(S)$ upper-bounds $2w(G[U])$ while the denominator equals $2w(G[V_k])$. ∎

Thus, to upper-bound the approximation ratio of the greedy algorithm (with respect to a quasioptimal solution), it suffices to upper-bound $\mathrm{val}(S)$ over all valid scenarios S. In what follows, we define several transformations of scenarios that bring a given valid scenario into a certain canonical form without decreasing the value of the scenario. Then, we analyze canonical scenarios to upper-bound their values. Defining the transformation directly on execution instances would involve global changes to the weight function; the local transformations below are made possible by our use of more general objects—scenarios—that abstract away actual edge weights.

We say that a valid scenario S is *saturated* if the equality holds in inequality (4), for $k < i \le n$. Our first transformation, called saturate, converts an arbitrary valid scenario S into a saturated one. Fixing other components of S, it successively redefines one of $f(v_i)$ or $g(v_i)$, for $i = k + 1$ up to n, increasing its value by the minimum amount needed to make the equality hold in (4). More specifically, it increases $f(v_i)$ and keeps $g(v_i)$ fixed if $v_i \in W$; it increases $g(v_i)$ and keeps $f(v_i)$ fixed if $v_i \in U$. If the amount of increase (of $f(v_i)$ or $g(v_i)$) is Δ, then, in either of the above two cases, each of $s_i \hat{f}_i$ and $t_i \hat{g}_i$ is increased by Δ. Since $s_i + t_i = i > k \ge 3$, we have either $s_i \ge 2$ or $t_i \ge 2$. In the former case, the increase of $\hat{f}_i$ is at most $\Delta/2$; in the latter case the increase of $\hat{g}_i$ is at most $\Delta/2$. Therefore, with an appropriate value of Δ, we can make $h(v_i)$ equal to $\min(\hat{f}_i, \hat{g}_i)$. It is clear that the result of saturate is a valid and saturated scenario. Moreover, this transformation never decreases the value of S, because the denominator is fixed and the numerator may only increase.

The purpose of transformation saturate is to make a scenario *monotone*: we call scenario S monotone if $h(v_{k+1}) \le h(v_{k+2}) \le \cdots \le h(v_n)$.

PROPOSITION 6. *If S is valid and saturated then S is monotone.*

Proof. Since S is saturated, it suffices to show that $\min(\hat{f}_{i+1}, \hat{g}_{i+1}) \ge \min(\hat{f}_i, \hat{g}_i)$ for $k \le i < n$. Fix i, $k \le i < n$, and suppose $v_{i+1} \in U$. Since

$t_{i+1}\hat{g}_{i+1} \geq t_i\hat{g}_i$ and $t_{i+1} = t_i$, we have $\hat{g}_{i+1} \geq \hat{g}_i$. Therefore, it suffices to show that $\hat{f}_{i+1} \geq \min(\hat{f}_i, \hat{g}_i)$, i.e., $\hat{f}_{i+1} \geq \hat{f}_i$ or $\hat{f}_{i+1} \geq \hat{g}_i$. First note that $s_{i+1}\hat{f}_{i+1} \geq s_i\hat{f}_i + h(v_{i+1})$, since $v_{i+1} \in U$. If we set $y = (s_i\hat{f}_i + h(v_{i+1}))/s_{i+1}$ (hence $\hat{f}_{i+1} \geq y$) then y is a convex combination of $\hat{f}_i$ and $h(v_{i+1})$ since $s_{i+1} = s_i + 1$. Therefore, either $h(v_{i+1}) \geq y \geq \hat{f}_i$ or $h(v_{i+1}) < y < \hat{f}_i$. The first case trivially implies $\hat{f}_{i+1} \geq \hat{f}_i$. In the second case, we must have $h(v_{i+1}) = \hat{g}_{i+1}$, since S is saturated, and hence $\hat{f}_{i+1} \geq y > \hat{g}_{i+1} \geq \hat{g}_i$. To deal with the case $v_i \in W$, swap the roles of f and U with those of g and W. ∎

Our two other transformations are defined on valid monotone scenarios. As we will see, they keep $h(v_i)$ unchanged for every v_i and hence preserve monotonicity. Note, however, that they do not necessarily preserve saturatedness—it is not important since saturation is only a mean for achieving monotonicity.

Call scenario S *lean* if there exists some i, $k < i \leq n$, such that $\hat{f}_i \leq h(v_n)$. Our second transformation, called squeeze, converts an arbitrary monotone valid scenario S into a lean one: fixing other components of S, it decreases $\hat{f}$ by the minimum amount needed to make S lean. Note here that $\hat{f}_i$ for each i is an increasing function of $\hat{f}$ and setting $\hat{f} = 0$ certainly makes $\hat{f}_k \leq h(v_n)$ hold. Therefore, transformation squeeze is always well defined. Transformation squeeze does not change $h(v_i)$ or $\hat{g}_i$ for any i and does not make any $\hat{f}_i$ smaller than $h(v_n)$. And since $h(v_n) \geq h(v_i)$ due to the monotonicity of S, the result of squeeze remains valid. Moreover, since $\hat{f}$ is common in the numerator and the denominator of val(S), squeeze does not decrease val(S) provided val(S) ≥ 1 originally (which we can always assume for our upper-bounding purposes).

Our last transformation, which is parameterized by integers j and l, $k < j < l \leq n$, and is called exchange(j, l), applies to lean, monotone valid scenarios. Loosely speaking, this transformation corresponds to moving some weight from the edges between v_j and U to the edges between v_j and W while moving back the same amount of weight from the edges between v_l and W to the edges between v_l and U. See Fig. 2. More formally suppose a monotone valid scenario S is given that is lean. Let $m(S)$ denote the largest integer m such that $\hat{f}_m \leq h(v_n)$. Transformation exchange(j, l) is applicable to S if $m(S) < j < l \leq n$, $f(v_j) > 0$, $g(v_l) > 0$, and $v_l \in U$. Suppose these conditions are satisfied. Keeping fixed each of $h(v_j)$, $h(v_l)$, $f(v_j) + f(v_l)$, and hence $g(v_j) + g(v_l)$, the transformation decreases $f(v_j)$ (and hence $g(v_l)$) by the minimum amount such that one of the following happens: (a) $f(v_j) = 0$, (b) $g(v_l) = 0$, or (c) $\hat{f}_i \leq h(v_n)$ for some i, $j \leq i < l$. The result of this transformation is a valid scenario, because $\hat{g}_i$ for each i is never decreased and, although $\hat{f}_i$ may be

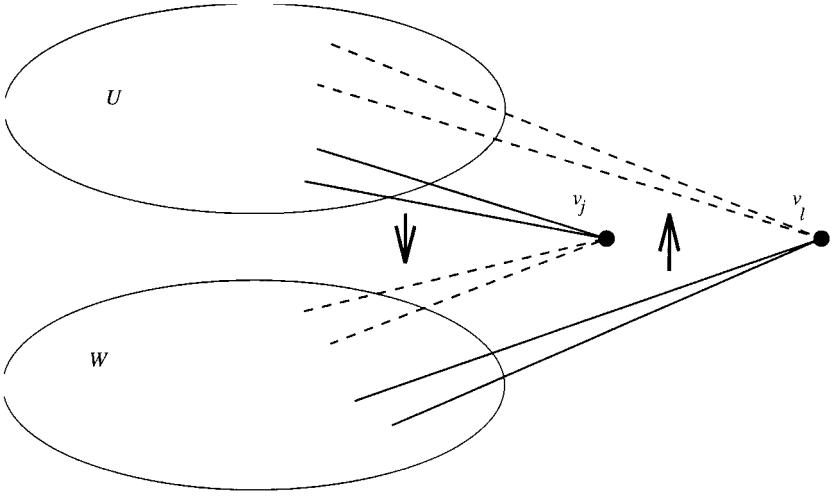

FIG. 2. Operation exchange(j, l).

decreased it may never become strictly smaller than $h(v_n) \geq h(v_i)$ due to condition (c). It is also clear that the scenario remains monotone (because h is not changed) and lean (because neither $\hat{f}$ nor $f(v_i)$, for any $k < i \leq m(S)$, is changed and hence $\hat{f}_i$ remains unchanged for every i in that range). Finally note that this transformation does not decrease val(S), because the denominator is fixed and the numerator is either unchanged (when both v_j and v_l are in U) or increased (when $v_j \in W$ and $v_l \in U$).

We call a scenario S *canonical*, if it is valid, monotone, and lean and if, moreover, transformation exchange(j, l) is not applicable to S for any pair of indices j, l, i.e.,

(A) there is no pair (j, l), $m(S) < j < l \leq n$ such that $v_l \in U$, $f(v_j) > 0$, and $g(v_l) > 0$.

The following lemma summarizes the role of our transformations.

LEMMA 7. *For any valid* (n, k)*-scenario* S *such that* val$(S) \geq 1$, *there is a canonical scenario* S' *such that* val$(S') \geq$ val(S).

Proof. S can be made monotone by applying transformation saturate. Then, transformation squeeze can be applied to make it lean. Suppose S is already lean. We repeatedly apply transformation exchange(j, l) choosing the smallest possible j and largest possible l at each application, until S is canonical. This process terminates in a finite number of steps, because $m(S)$ does not decrease in this process and, during the steps in which $m(S)$ is fixed (and hence only conditions (a) and (b) in the definition of

exchange(j, l) happen) we can apply exchange(j, l) at most once for each fixed pair (j, l). Therefore, we eventually arrive at a canonical scenario. As we have observed, none of the transformations decreases the value of the scenario. ▮

The following property of a canonical scenario is crucial in our upper bound proof.

LEMMA 8. *Let* $S = (V, U, \sigma, \hat{f}, \hat{g}, f, g)$ *be a canonical scenario. Then* $g(v_i) = 0$ *for every* $v_i \in U$ *with* $m(S) + 2 \le i \le n$.

Proof. Let S be canonical and $m = m(S)$. Suppose that $g(v_i) > 0$ for some $v_i \in U$, $m + 2 \le i \le n$. Then, from condition (A), we have $f(v_{m+1}) = 0$. We consider two cases: $v_{m+1} \in U$ or not. Suppose first that $v_{m+1} \in U$. Then, since $f(v_{m+1}) = 0$, we have $s_{m+1}\hat{f}_{m+1} = s_m\hat{f}_m + h(v_{m+1})$ and, since $s_{m+1} = s_m + 1$, we have $h(v_{m+1}) = (s_m + 1)\hat{f}_{m+1} - s_m\hat{f}_m$. Since $\hat{f}_m \le h(v_n)$ and $\hat{f}_{m+1} > h(v_n)$ from the definition of $m(S)$, it must follow that $h(v_{m+1}) > h(v_n)$, a contradiction to the monotonicity of S. Suppose next that $v_{m+1} \in W$. Then it follows from $f(v_{m+1}) = 0$ that $s_{m+1}\hat{f}_{m+1} = s_m\hat{f}_m$ and hence, since $s_{m+1} = s_m$, $\hat{f}_{m+1} = \hat{f}_m$, contradicting the definition of $m(S)$. Therefore, there must be no i in the range $m + 2 \le i \le n$ such that $v_i \in U$ and $g(v_i) > 0$. ▮

We are ready to prove the main lemma for the upper bound result.

LEMMA 9. *For any canonical* (n, k)*-scenario* S, $\mathsf{val}(S) \le (1/2 + n/2k)^2 + O(1/n)$ *for* $n/3 \le k \le n$ *and* $\mathsf{val}(S) \le 2(n/k - 1) + O(n/k^2)$ *for* $k < n/3$.

Proof. Let $S = (G, U, \sigma, \hat{f}, \hat{g}, f, g)$ be a canonical (n, k)-scenario and let $m = m(S)$. To evaluate $\mathsf{val}(S)$, we split the numerator of $\mathsf{val}(S)$ in two parts: $N_1 = s_k\hat{f} + 2\Sigma_{k < i \le m,\, v_i \in U} f(v_i)$ and $N_2 = 2\Sigma_{m < i \le n,\, v_i \in U} f(v_i)$. The first part is bounded by

$$N_1 \le s_m\hat{f}_m \le s_m h(v_n) \tag{5}$$

and the second part by

$$N_2 \le 2(k - s_m)h(v_n). \tag{6}$$

Therefore, we have

$$\mathsf{val}(S) \le \frac{(2k - s_m)h(v_n)}{s_k\hat{f} + t_k\hat{g}}. \tag{7}$$

Let $n' = n$ if $m = n$ and $n' = n - k + s_{m+1}$ if $m < n$. Let $\hat{h} = (s_k \hat{f} + t_k \hat{g})/k$. We claim the following bound on $h(v_n)$.

$$h(v_n) \leq \frac{n' - 1}{k - 1} \hat{h} \tag{8}$$

To prove this bound, we first consider the easier case where $n' = n$. For $k \leq i \leq n$, let $\hat{h}_i = (s_i \hat{f}_i + t_i \hat{g}_i)/i$ denote the "average degree" of V_i in scenario S. Note here that $s_i + t_i = i$. Since $h(v_i) \leq \min(\hat{f}_i, \hat{g}_i)$ from the validity of S, we have $h(v_i) \leq \hat{h}_i$ for $k \leq i \leq n$. From the definition of $\hat{f}_i$ and $\hat{g}_i$, we have, for $k \leq i < n$, $(i + 1)\hat{h}_{i+1} = i\hat{h}_i + 2h(v_{i+1})$ and hence $(i + 1)\hat{h}_{i+1} \leq i\hat{h}_i + 2\hat{h}_{i+1}$ or $\hat{h}_{i+1}/i \leq \hat{h}_i/(i - 1)$. Therefore, we have $\hat{h}_n/(n - 1) \leq \hat{h}_k/(k - 1)$ by a simple induction. Since $h(v_n) \leq \hat{h}_n$, and $\hat{h}_k = \hat{h}$, inequality (8), with $n' = n$, holds.

Now suppose $m < n$ and hence $n' = n - k + s_{m+1}$. We consider an (n', k)-scenario $S' = (V', U', \sigma', \hat{f}, \hat{g}, f, g)$, where $U' = U \cap V_{m+1}$, $V' = U' \cup W$, and σ' is the sequence obtained from σ by striking out each v_i not in V'; remaining components are common with S except that functions f and g are restricted to the elements of σ'. For $k \leq i \leq n'$, let $j(i)$ denote the smallest integer j such that $|V' \cap V_j| = i$. Note that $\sigma' = (v_{j(n')}, v_{j(n'-1)}, \ldots, v_{j(k)})$ with this notation. In particular, $j(i) = i$ for $k \leq i \leq m + 1$.

We claim that S' is a valid scenario. We defer the proof and first argue that inequality (8) follows from this claim. Using the same argument as in the case $m(S) = n$, we obtain that

$$\hat{h}_{j(n')} \leq \frac{n' - 1}{k - 1} \hat{h}.$$

Since $j(n') > m$, the definition of $m = m(S)$ implies that $h(v_n) < \hat{f}_{j(n')}$. We also have $h(v_n) \leq \hat{g}_n$ from the validity of S. Furthermore, since $v_i \in U$ and $g(v_i) = 0$ for $j(n') < i \leq n$ (the latter being implied by Lemma 8), we have $\hat{g}_n = \hat{g}_{j(n')}$ and hence $h(v_n) \leq \hat{g}_{j(n')}$. Since $\hat{h}_{j(n')}$ is a weighted average of $\hat{f}_{j(n')}$ and $\hat{g}_{j(n')}$, we conclude that $h(v_n) \leq \hat{h}_{j(n')}$ and hence that inequality (8) holds.

It remains to show that S' is a valid scenario. Let $s'_i = |U' \cap V_{j(i)}|$ and $t'_i = |W \cap V_{j(i)}|$ and define $\hat{f}'_i$ and $\hat{g}'_i$ for $k \leq i \leq n'$ as $\hat{f}_i, \hat{g}_i$ are defined for S:

$$s'_i \hat{f}'_i = s_k \hat{f} + \sum_{k < l \leq j(i),\, v_l \in V'} f(v_l) + \sum_{k < l \leq j(i),\, v_l \in U'} h(v_l)$$

$$t'_i \hat{g}'_i = t_k \hat{g} + \sum_{k < l \leq j(i),\, v_l \in V'} g(v_l) + \sum_{k < l \leq j(i),\, v_l \in W} h(v_l).$$

Note that $s'_i = s_i$ for $k \le i \le m$ and $s'_i = s_{m+1}$ for $m+1 \le i \le n'$ while $t'_i = t_{j(i)}$ for the entire range $k \le i \le n'$.

We need to show that $h(v_{j(i)}) \le \min(\hat{f}'_i, \hat{g}'_i)$ for $k < i \le n'$. For $k < i \le m+1$, this follows from the validity of S, since $j(i) = i$, $\hat{f}'_i = \hat{f}_i$, and $\hat{g}'_i = \hat{g}_i$. Suppose $i > m+1$. Since $g(v_l) = 0$ for every $v_l \in V \setminus V'$ from Lemma 8, we have

$$\begin{aligned} t'_i \hat{g}'_i &= t_k \hat{g} + \sum_{k < l \le j(i),\, v_j \in V'} g(v_l) + \sum_{k < l \le j(i),\, v_l \in W} h(v_l) \\ &= t_k \hat{g} + \sum_{k < l \le j(i),\, v_j \in V} g(v_l) + \sum_{k < l \le j(i),\, v_l \in W} h(v_l) \\ &= t_{j(i)} \hat{g}_{j(i)} \end{aligned}$$

and hence $\hat{g}'_i = \hat{g}_{j(i)}$, since $t'_i = t_{j(i)}$. Therefore, from the validity of S, we have $h(v_{j(i)}) \le \hat{g}_{j(i)} = \hat{g}'_i$. To compare $h(v_{j(i)})$ with $\hat{f}'_i$, first note that $\hat{f}'_{m+1} = \hat{f}_{m+1} > h(v_n)$ from the definition of $m = m(S)$. Since $s'_i \hat{f}'_i \ge s_{m+1} \hat{f}'_{m+1}$ and hence $\hat{f}'_i \ge \hat{f}'_{m+1}$ (because $s'_i = s_{m+1}$) for $i > m$, it follows that $h(v_n) < \hat{f}'_i$ and hence $h(v_{j(i)}) < \hat{f}'_i$ since S is monotone. Therefore, we have $h(v_{j(i)}) \le \min(\hat{f}'_i, \hat{g}'_i)$ in the remaining range $m+1 < i \le n'$, establishing the validity of S'. This concludes the proof of inequality (8).

Combining inequalities (7) and (8) we have

$$\mathrm{val}(S) \le \frac{(2k - s_m)(n' - 1)}{k(k-1)}.$$

Recall that $n' = n - k + s_{m+1}$ if $m < n$ and $n' = n$ if $m = n$. In either case, we have $n' \le n - k + s_m + 1$ (since in the latter case $s_m = k$). Thus, we have $\mathrm{val}(S) \le ((2k - s_m)(n - k + s_m))/(k(k-1))$, or putting $x = s_m/k$ and $r = n/k$,

$$\begin{aligned} \mathrm{val}(S) &\le (2 - x)(r - 1 + x)(k/(k-1)) \\ &\le (2 - x)(r - 1 + x)(1 + 2/k) \\ &\le (2 - x)(r - 1 + x) + 4r/k \end{aligned}$$

arriving at the same quadratic function $F_r(x) = (2 - x)(r - 1 + x)$ as in Section 3. When $r < 3$, it attains its maximum $(r+1)^2/4$ at $x = (3 - r)/2$ and when $r \ge 3$, its maximum is $2(r - 1)$ at $x = 0$. This concludes the proof of Lemma 9. ∎

The upper bound in Theorem 1 now follows from the observation on quasioptimal solutions and Lemmas 5, 7, and 9.

We remark that some insight into the above upper bound proof may be obtained if the reader examines how the lower bound examples in Section 3 achieve equality or near-equality in the inequalities (5), (6), and (8).

ACKNOWLEDGMENT

The authors thank the anonymous referees for their valuable comments. They especially thank the second referee for pointing out some errors in the original submission.

REFERENCES

[AI95] Y. Asahiro and K. Iwama, Finding dense subgraphs, *in* "Proc. International Symposium on Algorithms and Computation '95," Lecture Notes in Computer Science, Vol. 1004, pp. 102–111, Springer-Verlag, Berlin, 1995.

[AKK95] S. Arora, D. Karger, and M. Karpinski, Polynomial time approximation schemes for dense instances of NP-hard problems, *in* "Proc. 27th ACM Symposium on Theory of Computing," pp. 284–293, 1995.

[FS97] U. Feige and M. Seltser, "On the Densest k-Subgraph Problem," Technical report, Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehobot, 1997.

[GL95] M. Grötschel and L. Lovász, Combinatorial optimization: A survey, *in* "Handbook of Combinatorics," North-Holland, Amsterdam, 1995.

[KP94] G. Kortsarz and D. Peleg, Generating sparse 2-spanners, *J. Algorithms* **17** (1994), 222–236.

[KP93] G. Kortsarz and D. Peleg, On choosing a dense subgraph, *in* "Proc. 34th IEEE Symposium on Foundations of Computer Science," pp. 692–701, 1993.

[Rag88] P. Raghavan, Probabilistic construction of deterministic algorithms: Approximating packing integer programs, *J. Comput. System Sci.* **37** (1988), 130–143.

[RRT94] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi, Heuristic and special case algorithms for dispersion problems, *Oper. Res.* **42** (1994), 299–310.

[SW98] A. Srivastav and K. Wolf, Finding dense subgraphs with semidefinite programming, *in* "Approximation Algorithms for Combinatorial Optimization," Lecture Notes in Computer Science, Vol. 1444, pp. 181–191, Springer-Verlag, Berlin, 1998.