Linear Time Algorithms for Generalized Edge Dominating Set Problems

André Berger and Ojas Parekh

Department of Mathematics and Computer Science,
Emory University, Atlanta, GA, 30322, USA
`aberge2@emory.edu, ojas@mathcs.emory.edu`

Abstract. In this paper we consider a generalization of the edge dominating set (EDS) problem, in which each edge e needs to be covered b_e times and refer to this as the b-EDS problem. We present an exact linear time primal dual algorithm for the weighted b-EDS problem on trees with $b_e \in \{0,1\}$, and our algorithm generates an optimal dual solution as well. We also present an exact linear time algorithm for the unweighted b-EDS problem on trees. For general graphs we exhibit a relationship between this problem and the maximum weight matching problem. We exploit this relationship to show that a known linear time $\frac{1}{2}$-approximation algorithm for the weighted matching problem is also a 2-approximation algorithm for the unweighted b-EDS problem on general graphs.

1 Introduction

Domination problems in graphs have been subject of many studies in graph theory, and have many applications in operations research, e.g. in resource allocation and network routing as well as in coding theory.

In this paper we consider a generalization of the *edge dominating set (EDS) problem*. Given a graph $G = (V, E)$, a function $b : E \to \mathbb{N}$ and a weight function $c : E \to \mathbb{Q}^+$, a *b-EDS* is a subset $F \subseteq E$ together with a multiplicity $m_e \in \mathbb{N}^+$ for each $e \in F$, so that each edge in E is adjacent to at least $b_e = b(e)$ edges in F, counting multiplicities. The *b-EDS problem* is then to find a b-EDS which minimizes $\sum_{e\in F} m_e$ in the unweighted and $\sum_{e\in F} c(e)\cdot m_e$ in the weighted case. The b-EDS problem generalizes the EDS problem in much the same way that the set multicover generalizes the set cover problem [17].

When $b_e = 1$ for all $e \in E$ this is the edge dominating set problem (EDS), which is one of the four natural covering problems in graphs: edge cover (cover V with elements from E), vertex cover (E with V), dominating set (V with V), and EDS (E with E). In fact weighted EDS is a common generalization of weighted edge cover and weighted vertex cover [1] and is equivalent to a restricted total covering problem in which $E \cup V$ must be covered by a minimum weight set of elements from $E \cup V$ [13].

The unweighted version of EDS is NP-complete even for planar and bipartite graphs of maximum degree 3 [18] as well as several other families of graphs [8]. However, there are also families of graphs for which the unweighted EDS problem is polynomial-time solvable [8, 16]. In particular, linear time algorithms for the unweighted version are known for trees [10] and block graphs [9].

Much less is known about the weighted version of the problem. Recently Fujito and Nagamochi [4] and Parekh [12] independently discovered a 2-approximation for the weighted EDS problem; the latter also showed that weighted EDS restricted to bipartite graphs is no easier to approximate than weighted vertex cover, which is MAX-SNP-hard [11] and is suspected to have no polynomial time approximation algorithm with approximation ratio asymptotically less than 2. Thus a 2-approximation for b-EDS may be the best we can hope for.

When $b_e \in \{0, 1\}$ for all $e \in E$ we call the resulting problem $\{0, 1\}$-EDS. The weighted version of $\{0, 1\}$-EDS is particularly interesting since it is equivalent to the generalization of weighted vertex cover in which in addition to single vertices, weights may also be assigned to pairs, $\{u, v\}$, of vertices (by adding an edge uv with $b_{uv} = 0$ if one does not already exist). This generalization may be used to model a limited economy of scale in existing applications of vertex cover: for a pair of vertices $\{u, v\}$ one may stipulate that selecting both u and v costs less than the sum of the individual costs of u and v.

Our main contributions are linear-time algorithms for three special cases of the b-EDS problem. To the best of our knowledge an exact linear time algorithm was not known for even the special case of weighted EDS on trees. Table 1 gives an overview of known results and new results from this paper.

In Section 2 we expose a relationship between the maximum weighted matching problem and the unweighted b-EDS problem; we use this relationship to analyze an algorithm of Preis [14] and show that it is also a linear time 2-approximation for the unweighted b-EDS problem. This generalizes a known relationship between maximal matchings and (unweighted) edge dominating sets.

In Section 3 we show that the weighted b-EDS is solvable in polynomial time on trees. We also present exact linear-time algorithms which solve the unweighted b-EDS problem, and the weighted $\{0, 1\}$-EDS problem on trees. The latter is a primal dual algorithm which also generates an optimal dual solution. If the costs c_e are integral for all $e \in E$, then the dual solution is integral as well and is a maximum size set of edges such that each edge e has at most c_e edges adjacent to it. This problem is a common generalization of the maximum independent set problem and the maximum strong matching problem. An exact linear time algorithm for the latter on trees is known [2].

Table 1: Approximation ratios for variants of the EDS problem (* denotes a linear time algorithm)

	unweighted EDS	weighted EDS	unweighted b-EDS	weighted b-EDS
general graphs	2^*	2 [4, 12]	2^* (Cor. 2)	8/3 [13]
bipartite graphs	”	”	”	2 [13]
trees	1^* [10]	1^* (Thm. 3)	1^* (Thm. 2)	1 (Thm. 1)

Notation We will use the following notation for a simple undirected graph $G = (V, E)$. The neighbors of a vertex $v \in V$ are denoted by $\delta(v) = \{u \in V : \exists uv \in E\}$. The edges incident upon the vertex $v \in V$ are denoted by $N(v)$ and the set of edges adjacent with an edge $e \in E$ plus the edge e itself is denoted by $N(e)$, i.e. $N(uv) = N(u) \cup N(v)$ for any $uv \in E$.

2 The unweighted *b*-EDS problem for general graphs

For general graphs, the weighted EDS problem admits a $2\frac{1}{10}$-approximation based on a natural linear program relaxation of the problem, whose integrality gap is also $2\frac{1}{10}$ [1]. A corresponding linear relaxation for the weighted b-EDS problem yields an $\frac{8}{3}$-approximation for general graphs and a 2-approximation for bipartite graphs [13]. This relaxation can be strengthened to yield a 2-approximation for weighted EDS; however, the corresponding strengthening fails to deliver a 2-approximation for weighted b-EDS [4, 12].

The unweighted EDS and b-EDS problems, however, can be approximated more easily due to their relation to matching problems. Harary's book [6] demonstrates that there always exists a minimum cardinality EDS which is also a maximal matching. Since any maximal matching is also an EDS, the minimum cardinality maximal matching and the unweighted EDS problems are equivalent; in contrast Fujito [3] showed that the minimum weighted maximal matching problem is much more difficult than weighted EDS. Any maximal matching in a graph has size at least one half times the size of a maximum matching. Therefore, finding any maximal matching, which can be easily done in linear time, yields a 2-approximation for the unweighted EDS problem.

An issue with extending the relationship described above to the minimum unweighted b-EDS problem is that a maximal matching is not necessarily a feasible b-EDS. Using the resemblance of the maximum weight matching problem to the dual of a natural linear formulation for b-EDS, we exhibit a connection between weighted matchings and b-edge dominating sets that generalizes the relationship between maximal matchings and edge dominating sets. Given a matching M and a weight vector $b \in \mathbb{N}^{|E|}$, let $b|_M \in \mathbb{N}^{|E|}$ denote the vector which for each component $e \in M$ has value b_e, and has value 0 for all other components.

Lemma 1. *For any matching M and any weight vector $b \in \mathbb{N}^{|E|}$, $\sum_{e \in M} b_e$ is at most twice the cost of a minimum size (counting multiplicities) b-EDS.*

Proof. Consider the following pair of dual LP's, LP 1 being the linear programming relaxation for the unweighted b-EDS problem and LP 2 being the relaxation for the weighted strong matching problem.

LP 1: Min $\mathbb{1} \cdot x$, subject to	LP 2: Max $b \cdot y$, subject to
$x(N(e)) \geq b_e$ for all $e \in E$	$y(N(e)) \leq 1$ for all $e \in E$
$x_e \geq 0$ for all $e \in E$	$y_e \geq 0$ for all $e \in E$

We call P and R the sets of feasible fractional solutions of LP 1 and LP 2, respectively. By setting the dual variables to $y_e = 1/2$ for each $e \in M$, and to $y_e = 0$ for each $e \in E \setminus M$, we obtain such a feasible solution $y \in R$ to LP 2, since each edge $e \in E$ can be adjacent with at most two edges from M. Using duality we have

$$b(M)/2 = b \cdot y \leq Max_{y \in R} b \cdot y \leq Min_{x \in P} \mathbb{1} \cdot x \leq OPT.$$

Hence the weight $b(M)$ of the solution we return is at most $2 \cdot OPT$. □

Corollary 1. *For any matching M and any weight vector $b \in \mathbb{N}^{|E|}$, if $b|_M$ is a feasible b-EDS then it is a 2-approximate unweighted b-EDS.*

Corollary 1 motivates the following definition: we say a matching M is b-feasible if $b|_M$ is a feasible b-EDS. Thus any matching algorithm that returns a b-feasible matching M is a 2-approximation for the unweighted b-EDS problem. Before presenting a linear time algorithm, we present a very simple $O(|E| log |V|)$ 2-approximation for unweighted b-EDS.

Proposition 1. *The greedy matching algorithm that repeatedly selects the edge of greatest cost that maintains a matching is b-feasible.*

Proof. The greedy algorithm also satisfies Lemma 2 and thus the proof is the same as the proof of Lemma 3 below. □

It is not difficult to see that any matching of maximum weight with respect to b is b-feasible. Lemma 1 also implies that any feasible b-EDS has size at least $\frac{1}{2}$ the cost of a maximum weight matching with respect to b, thus if a b-EDS algorithm always returns a b-EDS that is a matching when copies of an edge are removed, then the algorithm is a $\frac{1}{2}$-approximation for the maximum weight matching problem.

Preis [14] gave a linear time algorithm, which, given a weighted graph, computes a maximal matching with weight at least one half times the weight of any matching. The algorithm incrementally adds edges to a matching M. A vertex u is called free (w.r.t. to M), if u is not incident with any edge in M. We will use the following lemma, which gives a necessary condition for an edge to be added to the matching, to show that Preis's algorithm always generates a b-feasible matching.

Lemma 2 ([14, Lemma 3]). *If an edge uv is added to M during the algorithm, then u and v are free and neither u nor v are adjacent to a free vertex with an edge of higher weight than the weight of the edge uv.*

Lemma 3. *Preis's algorithm always generates a b-feasible matching.*

Proof. First, note that each edge $e \in M$ is covered b_e times by itself. Since M is a maximal matching, any edge $e = uv$ in $E \setminus M$ must be adjacent to some edge $f = vw$ in M. If $b_f \geq b_e$, then e is covered. Otherwise, Lemma 2 says that u was not a free vertex at the time when f was added to M. But then there must be an edge $tu \in M$, which was added to M before f, i.e. t, u and v were free vertices at the time tu was considered. Using Lemma 2 again, we must have $b_{tu} \geq b_e$. $\square$

Corollary 2. *The unweighted b-EDS problem on general graphs can be 2-approximated in linear time.*

3 The *b*-EDS problem for trees

3.1 The general case

Many problems which are hard to solve optimally or even approximate for general graphs become a lot easier when restricted to a small family of graphs. The same is true for the weighted b-EDS problem when we restrict the possible inputs to trees.

In this section we will show that the weighted b-EDS problem on trees can be solved optimally as a linear program. A square matrix A is called *totally unimodular* if every square sub-matrix of A has determinant +1, -1 or 0. Totally unimodular matrices play an important role in linear programming due to the following lemma.

Lemma 4 ([7, 15]). *If $A \in \mathbb{Z}^{mxn}$ is a totally unimodular matrix and $b \in \mathbb{Z}^m$, then every extreme point of the polyhedron $\{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\} \neq \emptyset$ has integer coordinates.*

Ghouila-Houri [5] gave the following sufficient condition for a matrix to be totally unimodular.

Lemma 5. *A matrix $A \in \{1, 0, -1\}^{nxm}$ is totally unimodular if for every $J \subseteq \{1, \ldots, n\}$ there exist a partition $J = J_1 \cup J_2$ such that for any $1 \leq j \leq m$ it holds that $|\sum_{i \in J_1} a_{ij} - \sum_{i \in J_2} a_{ij}| \leq 1$.*

For completeness we include a proof for the fact that the constraint matrix for the EDS problem for any tree is totally unimodular.

Lemma 6. *Let $T = (V, E)$ be a tree on n vertices. Let $A = (a_{ij})$ be the edge-edge adjacency matrix of T with 1's on the diagonal, i.e. $a_{ij} = 1$ for all $ij \in E$, $a_{ij} = 0$ for all $ij \notin E$ and $a_{ii} = 1$ for all $1 \leq i \leq n-1$, $1 \leq j \leq n-1$ and $i \neq j$. Then A is totally unimodular.*

Proof. According to Lemma 5 it is enough to show that we can partition any $E' \subseteq E$ into two sets of edges E_1 and E_2, so that for any $e \in E$ we have $|N(e) \cap E_1| - |N(e) \cap E_2| \in \{-1, 0, 1\}$. This is equivalent to say that for any

$E' \subseteq E$ there is a labeling $\pi : E \to \{1, 0, -1\}$ such that $\pi(e) = 0$ if and only if $e \in E \setminus E'$ and such that $\Pi(e) := |\sum_{f \in N(e)} \pi(f)| \leq 1$ for any $e \in E$.

We will prove by induction on $|V|$ that such a labeling exists for any $E' \subseteq E$ and that it also satisfies $|\Pi(v)| \leq 1$ where $\Pi(v) := \sum_{f \in N(v)} \pi(f)$ for any $v \in V$. Note that if such a labeling π exists, than $-\pi$ clearly also satisfies the condition.

The base case $|V| = 1$ is trivial. Let now $|V| > 2$ and let $v_0 \in V$ be an arbitrary vertex of T. Let $T_i = (V_i, E_i)$ $(1 \leq i \leq k)$ be the connected components of $T - \{v\}$, which are trees as well. Let v_i be the neighbor of v in T_i and $e_i = v_0 v_i$. For every $1 \leq i \leq k$ let π_i be a labeling of E_i with $\pi_i(e) = 0$ if and only if $e \in E_i \setminus E'$, which exist by the inductive hypothesis. They also satisfy $|\Pi_i(v_i)| \leq 1$ for every $1 \leq i \leq k$. If we set $\pi(e_i) = 0$ for $1 \leq i \leq k$ and $\pi(e) = \pi_i(e)$ whenever $e \in E_i$, we obtain a labeling $\pi : E \to \{1, 0, -1\}$ such that $\Pi(e_i) = |\Pi_i(v_i)| \leq 1$ and $\Pi(v_0) = 0$, i.e. π satisfies the claim. However, the edges incident with v_0 which are in E' are falsely labeled 0. We will show how to label these edges with 1 or -1 and maintain the desired properties.

We can assume w.l.o.g. that for some $0 \leq s \leq k$ we have that $\{e_1, \ldots, e_s\} \subseteq E'$ and that $\{e_{s+1}, \ldots, e_k\} \subseteq E \setminus E'$. Further assume that $|\Pi(v_i)| = 1$ for $1 \leq i \leq s_0$ and that $|\Pi(v_i)| = 0$ for $s_0 < i \leq s$ for some $0 \leq s_0 \leq s$.

We will switch the labelings π_i for $1 \leq i \leq s_0$ if necessary to obtain $\Pi(v_i) = 1$ for $1 \leq i \leq \lfloor s_0/2 \rfloor$ and $\Pi(v_i) = -1$ for $\lfloor s_0/2 \rfloor < i \leq s_0$. Then we define $\pi(e_i) = -1$ for $1 \leq i \leq \lfloor s_0/2 \rfloor$ and $\pi(e_i) = 1$ for $\lfloor s_0/2 \rfloor < i \leq s_0$. The edges $e_{s_0+1}, \ldots, e_s$ will be labeled with 1 and -1 so that we have $\Pi(v_0) \in \{0, 1\}$. The numbers of edges labeled 1 and -1, respectively, will depend on the parity of s and s_0 and differ by at most 1. If $\Pi(v_0) = 1$, then we also switch the labelings π_i of those trees T_i with $i > s$ for which $\Pi(v_i) = 1$ to $-\pi_i$ to ensure that $\Pi(e_i) \leq 1$ for those indices i.

It is easy to check that the conditions on π remain true for all edges and vertices of T. □

Using Lemma 4 and Lemma 6 we immediately have

Theorem 1. *The b-EDS problem on weighted trees can be solved optimally in strongly polynomial time.*

The algorithm to solve the b-EDS problem on trees relies on solving a linear program. However, we would prefer a combinatorial algorithm, ideally running in linear time. This is indeed possible if we restrict the trees to have either uniform weights (Section 3.2) or if we restrict ourselves to the $\{0, 1\}$-EDS problem on weighted trees.

3.2 The unweighted b-EDS problem for trees

A linear time algorithm for the unweighted EDS problem on trees was first given by Mitchell and Hedetniemi [10] and later simplified by Yannakakis and Gavril [18]. The unweighted b-EDS problem on trees can also be solved by an easy greedy algorithm in linear time. Call an edge e of a tree a *leaf edge* if it is incident with a leaf.

For any tree T there will always be an optimal solution to the b-EDS problem which does not use any leaf edges (unless T is a star), since any edge adjacent with a leaf edge covers at least those edges covered by the leaf edge.

Therefore we can recursively solve the problem by first finding a vertex v which is incident with exactly one non-leaf edge e, then setting the multiplicity of e to the maximum b-value of the leaf edges incident with v and finally removing those leaf edges and updating the b-values of e and those edges adjacent with e in the remaining tree.

Any optimal solution to the b-EDS problem on that updated tree plus the multiplicity of the edge e as determined before will give an optimal solution to the original instance. A formal proof of this fact is straightforward and we omit it in this abstract.

Theorem 2. *The unweighted b-EDS problem on trees can be solved optimally in linear time.*

3.3 The weighted $\{0,1\}$-EDS problem for trees

To the best of our knowledge no linear time algorithm for the weighted EDS problem on trees has appeared in the literature. Algorithm 1 is a linear time primal-dual algorithm which solves the weighted $\{0,1\}$-EDS problem on trees optimally in linear time. This problem generalizes the weighted b-vertex cover problem on trees as follows:

Lemma 7. *The weighted vertex cover problem for trees can be solved optimally in linear time by solving a weighted* $\{0,1\}$*-EDS instance on a tree in linear time.*

Proof. Let $T = (V,E)$, $c_v \in \mathbb{R}^+$ for all $v \in V$ be an instance of the weighted vertex cover problem for trees. We build a tree $T' = (V \cup V', E \cup E')$ with $V' = \{v' : v \in V\}$ and $E' = \{vv' : v \in V\}$, i.e. we add an extra edge incident with each vertex of T. We set $b_e = 1$ for every $e \in E$ and $b_e = 0$ for every $e \in E'$. Furthermore, we set $c'_{vv'} = c_v$ for all $v \in V$ and $c'_e = \infty$ for all $e \in E$. Then any b-EDS of T' of finite weight corresponds to a vertex cover of T having the same weight, and vice versa (an edge $vv' \in E'$ is in the b-EDS if and only if v is in the vertex cover). □

We now present our primal-dual algorithm for the weighted $\{0,1\}$-EDS problem for trees. In a nutshell the algorithm works as follows. We first pick some arbitrary vertex of the tree as the root. Then we determine an optimal dual solution by raising dual variables from the leaves up to the root, making at least one constraint of the dual problem tight whenever we raise a dual variable. Finally, we recover a primal solution from the root down to the leaves, which satisfies the complementary slackness conditions with the dual solution.

We denote by $\mathrm{d}_T(v,u)$ the (combinatorial) distance between v and u in T, i.e. the number of edges on the path between v and u in T. If T is rooted at v_0, then by denoting an edge by $e = vu$ we implicitly mean that v is closer to the root, i.e. $\mathrm{d}_T(v,v_0) = \mathrm{d}_T(u,v_0) - 1$. For a vertex $v \neq v_0$ $p(v)$ denotes the *parent* *of* v, i.e. the unique vertex on the path from v to v_0 which is adjacent to v. The set of children of v is denoted $\overline{\delta}(v) = \delta(v) \setminus \{p(v)\}$.

Algorithm 1: $\{0,1\}$-EDS on weighted trees

```
   Input: A tree $T = (V, E)$, $c : E \to \mathbb{R}^+ \cup \{0\}$,
   $b : E \to \{0,1\}$ and a root $v_0 \in V$.
1. Set $K := \max_{v \in V} \mathrm{d}_T(v, v_0)$.
% *Construct the dual solution from the leaves to the root.*
2. FROM $i = K$ DOWNTO 0 DO
       FOR ALL $v \in V$ with $d_T(v, v_0) = i$ DO
           IF $v$ is a leaf THEN $y_v := c_{p(v)v}$
           ELSE
               $\underline{c} := \min_{u \in \delta(v)} c_{vu}$
               FOR EVERY $u \in \overline{\delta}(v)$ with $b_{vu} = 1$ DO
                  $y_{vu} := \min\{y_u, \underline{c}\}$
                  $\underline{c} := \underline{c} - y_{vu}$
               $\overline{y} := \sum_{u \in \overline{\delta}(v)} y_{vu}$
               $y_v := \min_{u \in \overline{\delta}(v)} (c_{vu} - \overline{y})$
               IF $v \neq v_0$ THEN $c_{p(v)v} := c_{p(v)v} - \overline{y}$
% *Construct the primal solution from the root to the leaves.*
% *$e \in E$ is 'tight', if $y(N(e)) = c_e$*
3. $F := \emptyset$
4. Whenever an edge $vu$ is added to $F$, set $x_v = 1$ and $x_u = 1$.
5. IF $y_{v_0 v} = 0$ for all $v \in \overline{\delta}(v_0)$ THEN add all tight edges incident with $v_0$
   to $F$ ELSE add one tight edge incident with $v_0$ to $F$.
6. FROM $i = 1$ to $K - 1$ DO
       FOR ALL $v \in V$ with $d_T(v, v_0) = i$ DO
           $e := p(v)v$
   Case 1 IF $y_e > 0$ and $x_e = 0$ and $x_{p(v)} = 0$ THEN add ONE arbitrary
          tight edge incident with $v$ to $F$
   Case 2 IF $y_{vu} = 0$ for all $u \in \overline{\delta}(v)$ and ($b_e = 0$ or $y_e = 0$) THEN add
          ALL tight edges incident with $v$ to $F$
   Case 3 IF $y_{vu} > 0$ for some $u \in \overline{\delta}(v)$ and $\Big((b_e = 0$ and $x_e = 0)$ or
          $y_e = 0\Big)$ add ONE arbitrary tight edge incident with $v$ to $F$
   Case 4 In the remaining cases no edges are added to the primal solu-
          tion $F$.
          % *Remark: If in any of the cases there is no tight edge incident
          upon $v$ then $F$ remains unchanged.*
7. RETURN $F$.
```

Theorem 3. *Algorithm 1 solves the weighted $\{0,1\}$-EDS problem on trees optimally in linear time.*

Proof. We will argue that both x and y are feasible solutions to the following LP's and that they satisfy complementary slackness and hence are optimal solutions. Here we let $D = \{e \in E : b_e = 1\}$.

LP 1: Min $c \cdot x$, subject to	LP 2: Max $\mathbb{1} \cdot y$, subject to
$x(N(e)) \geq 1$ for all $e \in D$	$y(N(e)) \leq c_e$ for all $e \in E$
$x_e \geq 0$ for all $e \in E$	$y_e \geq 0$ for all $e \in D$

First note that y is feasible for LP 2, i.e. $y \in R$. The variable y_v always contains the maximum value that any y-value of an edge incident with v can be increased by to maintain a feasible solution to LP 2. Using this and the fact that any positively set y-value for an edge vu is at most $c_{vu'}$, where $u' \in \delta(v)$, we see that y is a feasible solution to LP 2.

The primary solution x constructed in steps 5 and 6 is chosen so that it satisfies the complementary slackness conditions with y. First, only *tight* edges are chosen to be in the solution F, i.e. whenever $x_e > 0$ then $y(N(e)) = c_e$. Second, if $y_e > 0$ for some $e = p(v)v$ with $b_e = 1$, and v is considered in step 6, then we only add edges to F in Case 2 with the additional conditions x_e=0 and $x_{p(v)} = 0$. But this means no edge incident with $p(v)$ is already in F and we add at most one edge incident with v to F, i.e. $x(N(e)) \leq 1$. As we will show below x is a feasible solution to LP 1 and therefore $x(N(e)) = 1$. Thus x and y satisfy the complementary slackness conditions.

To show that x is a feasible solution for the primal LP let $e = vu \in D$ where $v = p(u)$ and assume to the contrary that for all $x(N(e)) = 0$. For now assume e is not incident with a leaf or with the root.

Let $f = p(v)v$ denote the *parent edge* of e. The *sibling edges* of e are all edges $vu' \in E$ with $u' \neq u$ and $u' \neq p(v)$; the *children edges* of e are all edges $uw \in E$ with $w \neq v$. We claim that neither any of the children edges and sibling edges of e nor e itself have a tight dual inequality. If one of them did, then it was not added to F during step 6 because f imposed a constraint on the complementary slackness condition, i.e. $f \in D$ and $y_f > 0$. However, we can only have $y_f > 0$ if none of the sibling edges of e and e itself were tight when y_f was considered to be increased in step 2. This means at least one of the children edges of e must be tight and if neither f nor e nor any of the sibling edges of e are in F, then this child edge of e must be in F, a contradiction to our assumption.

Hence none of the sibling edges and children edges of e and e itself are tight. Therefore, the only reason y_e was not increased any further must be that f was tight already after e was considered. Consequently, none of the sister edges of f which are in D nor the parent of f (if it has one) can have a positive y-value. This finally contradicts our assumption, since then we should have picked f for the primal solution in step 6 (Case 2).

We now consider the cases that e is incident with a leaf or with the root. If e is incident with a leaf, then certainly one of its sister edges or e itself must be tight. Hence for the parent edge f of e either $f \notin D$ or $y_f = 0$, hence at least one of the tight children edges of f must be in F and hence we again have a contradiction.

Finally, when e is incident with the root and neither e nor any of the other edges incident with the root are in F, then it must be that all edges incident with the root are not tight. But then, if e is not incident with a leaf at the same time, at least one of e's children must be tight and should be added to F during the algorithm.

Noting that each edge of the tree is considered at most three times in step 2 and at most twice in step 6, we conclude that the algorithm runs in $O(|E|) = O(|V|)$ time. $\square$

References

1. Robert Carr, Toshihiro Fujito, Goran Konjevod, and Ojas Parekh. A 2 1/10-approximation algorithm for a generalization of the weighted edge-dominating set problem. *J. Comb. Optim.*, 5:317–326, 2001.
2. Gerd Fricke and Renu Laskar. Strong matchings on trees. In *Proceedings of the Twenty-third Southeastern International Conference on Combinatorics, Graph Theory, and Computing (Boca Raton, FL, 1992)*, volume 89, pages 239–243, 1992.
3. Toshihiro Fujito. On approximability of the independent/connected edge dominating set problems. *Inform. Process. Lett.*, 79(6):261–266, 2001.
4. Toshihiro Fujito and Hiroshi Nagamochi. A 2-approximation algorithm for the minimum weight edge dominating set problem. *Discrete Appl. Math.*, 118(3):199–207, 2002.
5. Alain Ghouila-Houri. Caractérisation des matrices totalement unimodulaires. *C. R. Acad. Sci. Paris*, 254:1192–1194, 1962.
6. Frank Harary. *Graph Theory.* Addison-Wesley, Reading, MA, 1969.
7. Alan Jerome Hoffman and Joseph Bernard Kruskal. Integral boundary points of convex polyhedra. In *Linear inequalities and related systems*, Annals of Mathematics Studies, no. 38, pages 223–246. Princeton University Press, Princeton, N. J., 1956.
8. Joseph D. Horton and Kyriakos Kilakos. Minimum edge dominating sets. *SIAM J. Discrete Math.*, 6(3):375–387, 1993.
9. Shiow Fen Hwang and Gerard J. Chang. The edge domination problem. *Discuss. Math. Graph Theory*, 15(1):51–57, 1995.
10. Sandra L. Mitchell and Stephen T. Hedetniemi. Edge domination in trees. In *Proc. of the 8th Southeastern Conference on Combinatorics, Graph Theory and Computing (Louisiana State Univ., Baton Rouge, La., 1977)*, pages 489–509, 1977.
11. Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. System Sci.*, 43(3):425–440, 1991.
12. Ojas Parekh. Edge dominating and hypomatchable sets. In *Proceedings of the 13th Annual ACM-SIAM Symposium On Discrete Mathematics (SODA-02)*, pages 287–291, New York, 2002. ACM Press.
13. Ojas Parekh. *Polyhedral techniques for graphic covering problems.* PhD thesis, Carnegie Mellon University, 2002.

14. Robert Preis. Linear time $\frac{1}{2}$-approximation algorithm for maximum weighted matching in general graphs. In *STACS 99 (Trier)*, volume 1563 of *Lect. Notes in Comp. Sci.*, pages 259–269. Springer, Berlin, 1999.
15. Alexander Schrijver. *Combinatorial optimization. Polyhedra and efficiency.*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003.
16. Anand Srinivasan, K. Madhukar, P. Nagavamsi, C. Pandu Rangan, and Maw-Shang Chang. Edge domination on bipartite permutation graphs and cotriangulated graphs. *Inform. Process. Lett.*, 56(3):165–171, 1995.
17. Vijay Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.
18. Mihalis Yannakakis and Fanica Gavril. Edge dominating sets in graphs. *SIAM J. Appl. Math.*, 38(3):364–372, 1980.