# Optimal broadcast domination in polynomial time*

Pinar Heggernes† Daniel Lokshtanov†

**Abstract**

Broadcast domination was introduced by Erwin in 2002, and it is a variant of the standard dominating set problem, such that different vertices can be assigned different domination powers. Broadcast domination assigns an integer power $f(v) \geq 0$ to each vertex $v$ of a given graph, such that every vertex of the graph is within distance $f(v)$ from some vertex $v$ having $f(v) \geq 1$. The optimal broadcast domination problem seeks to minimize the sum of the powers assigned to the vertices of the graph. Since the presentation of this problem its computational complexity has been open, and the general belief has been that it might be NP-hard. In this paper, we show that optimal broadcast domination is actually in $\mathcal{P}$, and we give a polynomial time algorithm for solving the problem on arbitrary graphs, using a non standard approach.

## 1 Introduction

A *dominating set* in a graph is a subset of the vertices of the graph, such that every vertex of the graph either belongs to the dominating set or has a neighbor in the dominating set. A vertex outside of the dominating set is said to be *dominated* by one of its neighbors in the dominating set. The standard optimal domination problem seeks to find a dominating set of minimum cardinality. Since the introduction of this problem [2, 13], many domination related graph parameters have been introduced and studied, and domination in graphs is one of the most well known and widely studied subjects within graph algorithms [7, 8].

The standard dominating set problem can be seen as to represent a set of cities having broadcast stations, where every city can hear a broadcast station placed in it or in a neighboring city [12]. In 2002 Erwin [5] introduced the *broadcast domination* problem, which is more realistic in the sense that the various broadcast stations are allowed to transmit at different powers. FM radio stations are distinguished both by their transmission frequency

*A preliminary version of this work was presented at WG 2005 [9].

†Department of Informatics, University of Bergen, N-5020 Bergen, Norway. Emails: pinar.heggernes@ii.uib.no, daniel.lokshtanov@student.uib.no.

and by their ERP (Effective Radiated Power). A transmitter with a higher ERP can transmit further, but it is more expensive to build and to operate. Consequently, the optimal broadcast domination problem asks to compute an integer valued power function $f$ on the vertices, such that every vertex of the graph is at distance at most $f(v)$ from some vertex $v$ having $f(v) \geq 1$, and the sum of the powers are minimized.

Since the introduction of this problem, its computational complexity has been open [4, 11]. The standard optimal domination problem is NP-hard [6], and so are some variants that might resemble broadcast domination: optimal $r$-domination asks for a dominating set of minimum cardinality where every vertex of the graph is within distance $r$ from some vertex of the dominating set for a given $r$ [10, 14], and the $(k, r)$-center problem asks to find an $r$-dominating set containing at most $k$ vertices, where one parameter is given and the other is to be minimized [1, 6]. Since most of the interesting domination problems are NP-hard on general graphs, this gave some indication that optimal broadcast domination might also be NP-hard for general graphs. Following this, in 2003 Blair et al. gave polynomial time algorithms for optimal broadcast domination of trees, interval graphs, and series-parallel graphs [3].

In this paper, we show that, quite surprisingly, optimal broadcast domination is in $\mathcal{P}$. We first prove that every graph has an optimal broadcast domination in which the subsets of vertices dominated by the same vertex are ordered in a path or a cycle. Using this, we give a polynomial time algorithm for computing optimal broadcast dominations of arbitrary graphs. Our algorithm computes minimum weight paths in an auxiliary graph, and thus differs from standard methods of proving polynomial time bounds, like reductions to 2-SAT or 2-dimensional matching.

This paper is organized as follows. In the next section, we give the necessary background. In Section 3, we prove the necessary results on the structure of optimal broadcast dominations. In Section 4, we use this result to develop a polynomial time algorithm for all graphs. We conclude with a few remarks in Section 5.

## 2 Definitions and terminology

In this paper we work with unweighted, undirected, connected, and simple graphs as input graphs to our problem. Let $G = (V, E)$ be a graph with $|V| = n$ and $|E| = m$. For any vertex $v \in V$, the *neighborhood* of $v$ is the set $N_G(v) = \{u \mid uv \in E\}$. Similarly, for any set $S \subseteq V$, $N_G(S) = \cup_{v \in S} N(v) - S$. The *degree* of a vertex is $|N_G(v)|$. We let $G(S)$ denote the subgraph of $G$ induced by $S$.

The *distance* between two vertices $u$ and $v$ in $G$, denoted by $d_G(u, v)$, is the minimum number of edges on a path between $u$ and $v$. The *eccentricity*

of a vertex $v$, denoted by $e(v)$, is the largest distance from $v$ to to any vertex of $G$. The *radius* of $G$, denoted by $rad(G)$, is smallest eccentricity in $G$. The *diameter* of $G$, denoted by $diam(G)$, is the largest distance between any pair of vertices in $G$.

A function $f : V \to \{0, 1, \cdots, diam(G)\}$ is a *broadcast* on $G$. The set of *broadcast dominators* defined by $f$ is the set $V_f = \{v \in V \mid f(v) \geq 1\}$. A broadcast is *dominating* if for every vertex $u \in V$ there is a vertex $v \in V_f$ such that $d(u, v) \leq f(v)$. In this case $f$ is also called a *broadcast domination*. The *cost* of a broadcast $f$ incurred by a set $S \subseteq V$ is $c_f(S) = \sum_{v \in S} f(v)$. Thus, $c_f(V)$ is the total cost incurred by broadcast function $f$ on $G$.

For a vertex $v \in V$ and an integer $p \geq 1$, we define the *ball* $B_G(v, p)$ to be the set of vertices that are at distance $\leq p$ from $v$ in $G$. Thus $B_G(v, f(v))$ is the set of all vertices that are dominated by $v$ (including $v$ itself) if $f(v) \geq 1$. We will omit the subscript $G$ in the notation for balls, since a ball will always refer to the input graph $G$. A broadcast domination $f$ on $G$ is *efficient* if $B(u, f(u)) \cap B(v, f(v)) = \emptyset$ for all pairs of distinct vertices $u, v \in V$.

For an efficient broadcast domination $f$ on $G$, we define the *domination graph* $G_f = (V_f,\ \{uv \mid N_G(B(u, f(u))) \cap B(v, f(v)) \neq \emptyset\})$. Hence the domination graph can be seen as a modification of $G$ in which every ball $B(v, f(v))$ is contracted to the single vertex $v$ (with weight $f(v)$), and neighborhoods are preserved. Since $G$ is connected and $f$ is dominating, $G_f$ is always connected. An example is given in Figure 1.

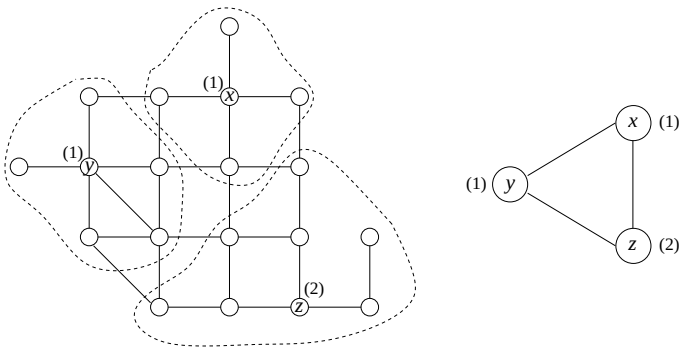


Figure 1: On the left hand side, a graph $G$ with an efficient broadcast domination $f$ is shown. For vertices $v$ with $f(v) \geq 1$, the broadcast powers $f(v)$ are shown in parentheses, and the dashed curves indicate the balls $B(v, f(v))$. For all other vertices $w$, $f(w) = 0$. On the right hand side, the corresponding domination graph $G_f$ is given, and the weight of each vertex is shown in parentheses.

The *optimal broadcast domination problem* on a given graph $G$ asks to compute a broadcast domination on $G$ with the minimum cost. We will denote this minimum cost by $\gamma_b(G)$. Note that if $f$ is an optimal broadcast

domination on $G = (V, E)$, then $c_f(V) \leq rad(G)$ since one can always choose a vertex $v$ of smallest eccentricity and dominate all other vertices with $f(v) = e(v) = rad(G)$. If $c_f(V) = rad(G) = f(v)$ for a single vertex $v$ in $G$, then $f$ is called a *radial* broadcast domination. For our purposes, we also need to define the minimum cost of a broadcast domination $f$ on $G$ such that $G_f$ is a simple path. Thus we let $\gamma_{bp}(G) = \min\{c_f(V) \mid f$ is a broadcast domination on $G$, and $G_f$ is a path$\}$.

# 3 The structure of an optimal broadcast domination

In [4], Dunbar et al. show that every graph has an optimal broadcast domination that is efficient. In particular, the following lemma is implicit from the proof of this result.

**Lemma 3.1** (Dunbar et al. [4]) *For any non efficient broadcast domination $f$ on a graph $G = (V, E)$, there is an efficient broadcast domination $f'$ on $G$ such that $|V_{f'}| < |V_f|$ and $c_{f'}(V) = c_f(V)$.*

We now add the following results.

**Lemma 3.2** *Let $f$ be an efficient broadcast domination on $G = (V, E)$. If the domination graph $G_f$ has a vertex of degree $> 2$, then there is an efficient broadcast domination $f'$ on $G$ such that $|V_{f'}| < |V_f|$ and $c_{f'}(V) = c_f(V)$.*

**Proof.** Let $v$ be a vertex with degree $> 2$ in $G_f$, and let $x, y$, and $z$ be three of the neighbors of $v$ in $G_f$. By the way the domination graph $G_f$ is defined, $v, x, y$, and $z$ are also vertices in $G$, and they all have broadcast powers $\geq 1$ in $f$. Since $f$ is efficient, $d_G(v, x) = f(v) + f(x) + 1$. Similarly, $d_G(v, y) = f(v) + f(y) + 1$ and $d_G(v, z) = f(v) + f(z) + 1$. Assume without loss of generality that $f(x) \leq f(y) \leq f(z)$.

If $f(x) + f(y) > f(z)$ then we construct a new broadcast $f'$ on $G$ with $f'(u) = f(u)$ for all vertices $u \in V \setminus \{v, x, y, z\}$. Furthermore, we let $f'(v) = f(v)+f(x)+f(y)+f(z)$, and $f'(x) = f'(y) = f'(z) = 0$. The new broadcast $f'$ is dominating since every vertex that was previously dominated by one of $v, x, y$, or $z$ is now dominated by $v$. To see this, let $u$ be any vertex that was dominated by $x, y$, or $z$ in $f$. Thus $d_G(v, u) \leq f(v) + 2f(z) + 1$ by our assumptions. Since $f'(v) > f(v) + 2f(z)$, vertex $u$ is now dominated by $v$ in $f'$. The cost of $f'$ is the same as that of $f$, and the number of broadcast dominators in $f'$ is smaller.

Let now $f(x) + f(y) \leq f(z)$. As we mentioned above, there is a path $P$ in $G$ between $v$ and $z$ of length $f(v) + f(z) + 1$. Let $w$ be a vertex on $P$ such that the number of edges between $w$ and $z$ on $P$ is $f(v) + f(x) + f(y)$. Since $f$ is efficient, $f(w) = 0$. We construct a new broadcast $f'$ on $G$ such

that $f'(u) = f(u)$ for all vertices $u \in V \setminus \{v, w, x, y, z\}$. Furthermore, we let $f'(w) = f(v) + f(x) + f(y) + f(z)$, and $f'(v) = f'(x) = f'(y) = f'(z) = 0$. By the way $d_G(z, w)$ is defined, any vertex that was dominated by $z$ or $v$ in $f$ is now dominated by $w$, since $d_G(v, w) < f(z)$. Let $u$ be a vertex that was dominated by $y$ in $f$. The distance between $u$ and $w$ in $G$ is $\leq 2f(y)+2f(v)+f(z)+2-f(v)-f(x)-f(y) = f(y)+f(v)+f(z)+2-f(x) \leq f(y) + f(v) + f(z) + f(x) = f'(w)$. Thus $u$ is now dominated by $w$. The same is true for any vertex that was dominated by $x$ in $f$ since we assumed that $f(x) \leq f(y)$. Thus $f'$ is a broadcast domination. Clearly, the costs of $f'$ and $f$ are the same, and $f'$ has fewer broadcast dominators.

Thus we have shown how to compute a new broadcast domination $f'$ as desired. If $f'$ is not efficient, then by Lemma 3.1 there exists an efficient broadcast domination with the same cost and fewer broadcast dominators, so the lemma follows. ∎

We are now ready to state the main result of this section, on which our algorithm will be based.

**Theorem 3.3** *For any graph $G$, there is an efficient optimal broadcast domination $f$ on $G$ such that the domination graph $G_f$ is either a path or a cycle.*

**Proof.** Let $f$ be any efficient optimal broadcast domination on $G = (V, E)$. If $G_f$ has a vertex of degree $> 2$ then by Lemma 3.2, an efficient broadcast domination $f'$ on $G$ with $|V_{f'}| < |V_f|$ and $c_{f'}(V) = c_f(V)$ exists. The proofs of both Lemmas 3.1 and 3.2 are constructive, so we know how to obtain $f'$. As long as there are vertices of degree $> 2$ in the domination graph, this process can be repeated. Since we always obtain a new domination graph with a strictly smaller number of vertices, the process has to stop after less than $n$ steps. Since domination graphs are connected, the theorem follows. ∎

Note that a path can be a single edge or a single vertex. If $G_f$ is a single vertex then $f$ is a radial broadcast.

**Corollary 3.4** *For any graph $G = (V, E)$, there is an efficient optimal broadcast domination $f$ on $G$ such that removing the vertices of $B(v, f(v))$ from $G$ results in at most two connected components, for every $v \in V_f$.*

**Proof.** Since there is always an efficient optimal broadcast domination $f$ on $G$ such that the balls $B(v, f(v))$ with $v \in V_f$ are ordered in a path or a cycle by Theorem 3.3, it suffices to observe that $B(v, f(v))$ induces a connected subgraph in $G$ for each $v \in V_f$. ∎

**Corollary 3.5** *For any graph $G = (V, E)$, there is an efficient optimal broadcast domination $f$ on $G$ such that a vertex $x \in V_f$ satisfies the following: $G' = G(V \setminus B(x, f(x)))$ either is empty, or is connected and has the property $\gamma_b(G') = \gamma_{bp}(G')$.*

**Proof.** By Theorem 3.3, let $f$ be an efficient optimal broadcast domination of $G$ such that $G_f$ is a path or a cycle. Let $x$ be any vertex of $G_f$ if $G_f$ is a cycle, any of the two endpoints of $G_f$ if $G_f$ is a path with at least two vertices, or $G_f$ itself if $G_f$ is a single vertex. Let $f'(v) = f(v)$ for all $v \in V \setminus \{x\}$. Since $f$ is efficient on $G$, $f'$ is an efficient dominating broadcast on $G'$, and $G'_{f'}$ is the result of removing $x$ from $G_f$. Thus $G'_{f'}$ is a path or empty. In addition $f'$ must be an optimal broadcast domination on $G'$, because otherwise $f$ cannot be optimal on $G$. ∎

# 4 Computing an optimal broadcast domination

By Theorem 3.3 we know that an efficient optimal broadcast $f$ on $G$ must exist such that $G_f$ is a path or a cycle. We will first give an algorithm for finding a broadcast domination $f$ with $c_f(V) \leq \gamma_{bp}(G)$.

## 4.1 Optimal broadcast domination of $G$ when $\gamma_b(G) = \gamma_{bp}(G)$

In this subsection, we want to find an efficient broadcast domination of minimum cost over all broadcast dominations $f$ on $G = (V, E)$ such that $G_f$ is a path. Our approach will be as follows: for each vertex $u$ of $G$, we will compute a new graph $\mathcal{G}_u$, and use this to find the best possible broadcast domination $f$ such that $G_f$ is a path and $u$ belongs to a ball corresponding to one of the endpoints of $G_f$. We will repeat this process for every $u$ in $G$, and choose at the end the best $f$ ever computed.

Given a vertex $u \in V$, we define a directed graph $\mathcal{G}_u$ with weights assigned to its vertices as follows: For each $v \in V$ and each $p \in \{1, ..., rad(G)\}$, there is a vertex $(v, p)$ in $\mathcal{G}_u$ if and only if one of the following is true:

- $G(V \setminus B(v, p))$ is connected or empty, and $u \in B(v, p)$
- $G(V \setminus B(v, p))$ has at most two connected components, and $u \notin B(v, p)$.

Thus $\mathcal{G}_u$ has a total of at most $n \cdot rad(G)$ vertices. Following Corollaries 3.4 and 3.5, each vertex $(v, p)$ represents the situation that $f(v) = p$ in the broadcast domination $f$ that we are aiming to compute. We define the *weight* of each vertex $(v, p)$ to be $p$.

The role of $u$ is to define the "left" endpoint of the path that we will compute. All edges will be directed from "left" to "right". We partition the vertex set of $\mathcal{G}_u$ into four subsets:

- $A_u = \{(v, p) \mid G(V \setminus B(v, p))$ is connected and $u \in B(v, p)\}$
- $B_u = \{(v, p) \mid G(V \setminus B(v, p))$ has two connected components$\}$
- $C_u = \{(v, p) \mid G(V \setminus B(v, p))$ is connected and $u \notin B(v, p)\}$

- $D_u = \{(v,p) \mid B(v,p) = V\}$

For each vertex $(v,p)$, let $L_u(v,p)$ be the connected component of $G(V \setminus B(v,p))$ that contains $u$ (i.e., the component to the "left" of $B(v,p)$), and left $R_u(v,p)$ be the connected component of $G(V \setminus B(v,p))$ that does not contain $u$ (i.e., the component to the "right" of $B(v,p)$). Thus $L_u(v,p) = \emptyset$ for every $(v,p) \in A_u \cup D_u$, and $R_u(v,p) = \emptyset$ for every $(v,p) \in C_u \cup D_u$.

The edges of $\mathcal{G}_u$ are directed and defined as follows: A directed edge $(v,p) \to (w,q)$ is an edge of $\mathcal{G}_u$ if and only if all of the following three conditions are satisfied:

- $B(v,p) \cap B(w,q) = \emptyset$ in $G$
- $R_u(v,p) \neq \emptyset$ and $L_u(w,q) \neq \emptyset$
- $(N_G(B(w,q)) \cap L_u(w,q)) \subseteq B(v,p)$ and $(N_G(B(v,p)) \cap R_u(v,p)) \subseteq B(w,q)$ in $G$.

To restate the last requirement in plain text: $B(v,p)$ must contain all neighbors of $B(w,q)$ in $L_u(w,q)$, and $B(w,q)$ must contain all neighbors of $B(v,p)$ in $R_u(v,p)$.

By the way we have defined the edges of $\mathcal{G}_u$, all vertices belonging to $A_u$ have indegree 0 and all vertices belonging to $C_u$ have outdegree 0. Hence, any path in $\mathcal{G}_u$ can contain at most one vertex from $A_u$ (which must be the starting point of the path) and at most one vertex from $C_u$ (which must be the ending point of the path). The vertices of $D_u$ are isolated, and every vertex of $D_u$ defines a radial broadcast domination on its own.

In the rest of this section we show how we can use $\mathcal{G}_u$ to compute an efficient broadcast domination of minimum cost over all broadcast dominations $f$ on $G = (V,E)$ such that $G_f$ is a path. First, let us justify the intuition that edges in $\mathcal{G}_u$ go from left to right.

**Lemma 4.1** *Let $G$ be an arbitrary graph, and let $u$ be a vertex of $G$. Every edge in $\mathcal{G}_u$ goes from left to right, i.e., if $(v,p) \to (w,q)$ is an edge in $\mathcal{G}_u$ then $B(w,q) \subseteq R_u(v,p)$ and $B(v,p) \subseteq L_u(w,q)$.*

**Proof.** By the last two requirements in the definition of edges in $\mathcal{G}_u$, the set $B(w,q) \cap R_u(v,p)$ is nonempty. Let $x$ be a vertex in this set, and assume for contradiction that $y$ is a vertex that belongs to $B(w,q)$, but not to $R_u(v,p)$. As $B(w,q)$ induces a connected subgraph in $G$, there is a path between $x$ and $y$ in $G$ containing only vertices belonging to $B(w,q)$. On this path let $a$ be the last vertex belonging to $R_u(v,p)$, and $b$ be the next vertex on the path after $a$. As there are no edges from $R_u(v,p)$ to $L_u(v,p)$ we know that $b$ must be in $B(v,p)$, contradicting that $B(v,p) \cap B(w,q) = \emptyset$, thus proving that $B(w,q) \subseteq R_u(v,p)$. The next claim $B(v,p) \subseteq L_u(w,q)$ is proved using an identical argument. ∎

We are now ready to show that any directed path in $\mathcal{G}_u$ from $A_u \cup D_u$ to $C_u \cup D_u$ corresponds to a broadcast domination. The following observation follows directly from the definition of edges in $\mathcal{G}_u$

**Observation 4.2** *Given a graph $G$, and a vertex $u$ in $G$, let $P = (v_1, p_1)$, $(v_2, p_2), ..., (v_k, p_k)$ be a path in $\mathcal{G}_u$ with $(v_1, p_1) \in A_u \cup D_u$ and $(v_k, p_k) \in C_u \cup D_u$. If $P$ has length at least 2, then $N_G(B(v_1, p_1)) \subseteq B(v_2, p_2)$, $N_G(B(v_k, p_k)) \subseteq B(v_{k-1}, p_{k-1})$, and $N_G(B(v_i, p_i)) \subseteq B(v_{i-1}, p_{i-1}) \cup B(v_{i+1}, p_{i+1})$, for $1 < i < k$.*

For each path $P = (v_1, p_1), (v_2, p_2), ..., (v_k, p_k)$ in $\mathcal{G}_u$ with $(v_1, p_1) \in A_u \cup D_u$ and $(v_k, p_k) \in C_u \cup D_u$, let $f_P$ be the following broadcast on $G$: For every $(v_i, p_i) \in P$, $f_P(v_i) = p_i$, and $f_P(v) = 0$ for every other vertex $v$.

**Lemma 4.3** *Let $G$ be an arbitrary graph and let $u$ be a vertex of $G$. For every path $P$ in $\mathcal{G}_u$ from $A_u \cup D_u$ to $C_u \cup D_u$, $f_P$ is a broadcast domination on $G$.*

**Proof.** $P$ is an isolated vertex if and only if it contains a vertex from $D_u$. In this case the lemma follows trivially, as $D_u = \{(v, p) \mid B(v, p) = V\}$.

Let $P = (v_1, p_1)$, $(v_2, p_2), ..., (v_k, p_k)$, and let $S = \bigcup_{j=1}^{k} B(v_j, p_j)$. We show that $S = V$. Assume for contradiction that $x \in V$ but $x \notin S$. Since $G$ is connected, there is a path from $x$ to $v_1$. Let $z$ be the first vertex on this path that is in $S$, and let $y$ be the vertex on the path before $z$. Let $j$ be such that $z \in B(v_j, p_j)$. Then $y \in N_G(B(v_j, p_j))$. By Observation 4.2, $y \in B(v_2, p_2)$ if $j = 1$, $y \in B(v_{k-1}, p_{k-1})$ if $j = k$, and $y \in B(v_{j-1}, p_{j-1}) \cup B(v_{j+1}, p_{j+1})$ otherwise, in any case contradicting that $z$ is the first vertex in $S$ on the path from $x$ to $v_1$. ∎

Now we want to prove that for every efficient broadcast domination $f$ on $G$, where $G_f$ is a path, there is a vertex $u$ in $G$, such that $f$ corresponds to a directed path in $\mathcal{G}_u$ that starts in $A_u \cup D_u$ and ends in $C_u \cup D_u$.

**Lemma 4.4** *Let $f$ be an efficient broadcast domination on $G$, such that $G_f = u_1, u_2, ..., u_k$ is a path. If $k = 1$ then $(u_1, f(u_1)) \in D_{u_1}$. If $k \geq 2$ then $(u_1, f(u_1)) \in A_{u_1}$, $(u_k, f(u_k)) \in C_{u_1}$, and $(u_i, f(u_i)) \in B_{u_1}$, for $1 < i < k$.*

**Proof.** Assume that $k = 1$. As $f$ is a broadcast domination, $B(u_1, f(u_1)) = V$, so $(u_1, f(u_1)) \in D_{u_1}$. Assume now that $k > 1$. Then $B(u_1, f(u_1))$ contains $u_1$. $G(V \setminus B(u_1, f(u_1)))$ is connected, as $B(u_i, f(u_i))$ is connected for every $i$, and there is an edge between every consecutive pair of balls. This gives us $(u_1, f(u_1)) \in A_{u_1}$. By the same argument $G(V \setminus B(u_k, f(u_k)))$ is connected. Also, $B(u_k, f(u_k))$ does not contain $u_1$, since $B(u_1, f(u_1))$ does, and $f$ is efficient. We conclude that $(u_k, f(u_k)) \in C_{u_1}$. For $1 < i < k$, $B(u_i, f(u_i))$ does not contain $u_1$, by the efficiency of $f$. $G(V \setminus$

$B(u_i, f(u_i)))$ has two connected components, namely $\bigcup_{j=1}^{i-1} B(u_j, f(u_j))$ and $\bigcup_{j=i+1}^{k} B(u_j, f(u_j))$. Each of those components is connected because of the same argument as above, and there are no edges between them, as that would have yielded an edge between some $u_s$ and $u_t$ in $G_f$, with $s < i < t$, which would contradict that $G_f$ is a path. That means that $1 < i < k$ implies $(u_i, f(u_i)) \in B_{u_1}$ and the proof is complete. ∎

¿From the proof of Lemma 4.4 we can see that $L_i = \bigcup_{j=1}^{i-1} B(u_j, f(u_j))$ and $R_i = \bigcup_{j=i+1}^{k} B(u_j, f(u_j))$ are the connected components of $G(V \setminus B(u_i, f(u_i)))$, for $1 < i < k$. Now, obviously $L_i$ contains $u_1$ and $R_i$ does not, so we conclude that $L_i = L_{u_1}(u_i, f(u_i))$ and $R_i = R_{u_1}(u_i, f(u_i))$, for $1 < i < k$.

**Lemma 4.5** *Let $f$ be an efficient broadcast domination on $G$, such that $G_f = u_1, u_2, ..., u_k$ is a path. In $\mathcal{G}_{u_1}$ there is an edge between $(u_i, f(u_i))$ and $(u_{i+1}, f(u_{i+1}))$ for $1 \leq i < k$.*

**Proof.** $B(u_i, f(u_i)) \cap B(u_{i+1}, f(u_{i+1})) = \emptyset$ by the efficiency of $f$. $R_{u_1}(u_i, f(u_i)) = R_i$, and $R_i \neq \emptyset$, since $i < k$. $L_{u_1}(u_{i+1}, f(u_{i+1})) = L_{i+1}$, and $L_{i+1} \neq \emptyset$, since $i + 1 > 1$. For $s > i + 1$, $N_G(B(u_i, f(u_i))) \cap B(u_s, f(u_s)) = \emptyset$ because there are no edges in $G_f$ between $u_i$ and $u_s$. Thus $N_G(B(u_i, f(u_i))) \cap R_i$ is a subset of $B(u_{i+1}, f(u_{i+1}))$. The proof that $N_G(B(u_{i+1}, f(u_{i+1}))) \cap L_{i+1}$ is a subset of $B(u_i, f(u_i))$ is identical. ∎

**Corollary 4.6** *Let $f$ be an efficient broadcast domination on $G$, such that $G_f = u_1, u_2, ..., u_k$ is a path. Then $(u_1, f(u_1)), (u_2, f(u_2)), ...(u_k, f(u_k))$ is a directed path in $\mathcal{G}_{u_1}$, starting in $A_{u_1} \cup D_{u_1}$ and ending in $C_{u_1} \cup D_{u_1}$*

**Proof.** We have $(u_1, f(u_1)) \in A_{u_1} \cup D_{u_1}$, $(u_k, f(u_k)) \in C_{u_1} \cup D_{u_1}$, and $(u_i, f(u_i)) \in B_{u_1}$, for $1 < i < k$ by Lemma 4.4. By Lemma 4.5 there is an edge between each consecutive pair of vertices. ∎

Now the idea is to find a directed path $P_u$ in $\mathcal{G}_u$ from a vertex of $A_u \cup D_u$ to a vertex of $C_u \cup D_u$ such that the sum of the weights of the vertices of $P_u$ (including the endpoints) is minimized. Let us call this sum $W(P_u)$. Then we will compute $\mathcal{G}_u$ for each vertex $u$ in $G$, and repeat this process, and at the end choose a path with the minimum total weight. Our algorithm for the path case is as follows:

**Algorithm** Minimum Path Broadcast Domination - MPBD
**Input:** A graph $G = (V, E)$.
**Output:** An efficient broadcast domination function $f$ on $G$ with $c_f(V) \leq \gamma_{bp}(G)$.
**begin**
  **for** each vertex $v$ in $G$ **do**
    $f(v) = 0$;
  Let $P$ be a dummy path with $W(P) = rad(G) + 1$;

**for** each vertex $u$ in $G$ **do**
Compute $\mathcal{G}_u$ with vertex sets $A_u$, $B_u$, $C_u$, and $D_u$;
Find a minimum weight path $P_u$ starting in a vertex of $A_u \cup D_u$ and ending in a vertex of $C_u \cup D_u$;
**if** $W(P_u) < W(P)$ **then**
$P = P_u$;
**end-for**
**for** each vertex $(v, p)$ on $P$ **do**
$f(v) = p$;
**end**

**Theorem 4.7** *Given a graph $G = (V, E)$, Algorithm MPBD computes an efficient broadcast domination $f$ on $G$ such that $c_f(V) \leq \gamma_{bp}(G)$.*

**Proof.** Let $f'$ be a broadcast domination on $G$ with cost $\gamma_{bp}(G)$ and $G_{f'}$ a path. Corollary 4.6 assures us that $G_{f'}$ corresponds to a path $P'$ with $W(P') = \gamma_{bp}(G)$ in $\mathcal{G}_u$ for some vertex $u$ of $G$. We compute a minimum weight path $P$ in $\mathcal{G}_u$ over all $u \in V$. Thus $W(P) \leq W(P') \leq \gamma_{bp}(G)$. By Lemma 4.3 $P$ corresponds to a broadcast domination $f$ with $c_f(V) = W(P) \leq \gamma_{bp}(G)$. ∎

**Corollary 4.8** *Let $G$ be a graph such that $\gamma_b(G) = \gamma_{bp}(G)$. Then Algorithm MPBD computes an efficient optimal broadcast domination on $G$.*

In a straight forward implementation of Algorithm MPBD, building $\mathcal{G}_u$ is the most time-consuming part. As $\mathcal{G}_u$ has potentially $O(n^2)$ vertices, it might have $O(n^4)$ edges. When we build $\mathcal{G}_u$ we have to check each of these potential edges for the properties of edges in $\mathcal{G}_u$. To check for the property "$B(v, p) \cap B(w, q) = \emptyset$ in $G$" we can use two breadth first searches, using $O(n + m) = O(n^2)$ time, and it is easy to see that the other properties can be checked within the same time bound. Finding a minimum weight path can be done in $O(n^4 \log n^4)$ time using a simple modification of Dijkstra's algorithm. As we have to do this for every vertex $u$ of $G$, we can conclude that the running time of a straight forward implementation of Algorithm MPBD is $O(n^7)$. In the next subsection, we improve this running time to $O(n^4)$.

## 4.2 Improving the running time of Algorithm MPBD

In order to improve the running time, we are going to show that the number of edges of $\mathcal{G}_u$ is actually at most $n^3$, and that $\mathcal{G}_u$ is acyclic. In addition, we show that the choice of $u$ does not affect $\mathcal{G}_u$ extensively, so a substantial amount of pre-computation can be done outside the outer loop over vertices $u$.

**Lemma 4.9** *Let $G$ be a graph on $n$ vertices, and let $u$ be any vertex of $G$. The graph $\mathcal{G}_u$ has $O(n^3)$ edges.*

**Proof.** We will need a series of claims for the proof of this lemma.

**Claim 4.10** *The statements $d_G(u,v) \leq p+q$ and $B(u,p) \cap B(v,q) \neq \emptyset$ are equivalent.*

**Proof.** Assume $x \in B(u,p)$ and $x \in B(v,q)$. Then $d_G(u,v) \leq d_G(u,x) + d_G(v,x) \leq p+q$. In the other direction, assume $d_G(u,v) \leq p+q$ and let $x$ be the vertex on a shortest path from $u$ to $v$ in $G$ having $d_G(u,x) = p$. As $x$ lies on a shortest path from $u$ to $v$, $d_G(x,v) = d_G(u,v) - d_G(u,x) \leq p+q-p = q$. Thus $x \in B(u,p)$ and $x \in B(v,q)$ ∎

**Claim 4.11** *Let $x$ be a vertex in $N_G(B(u,p))$. Then $x \in B(v,q)$ implies $d_G(u,v) \leq p+q+1$.*

**Proof.** Observe that $N_G(B(u,p)) \cap B(u,p) = B(u,p+1)$. Now $x \in B(u,p+1)$ so $B(u,p+1) \cap B(v,q)$ contains $x$ and is nonempty. By Claim 4.10 $d_G(u,v) \leq p+1+q$. ∎

**Claim 4.12** *If $(v,p) \rightarrow (w,q)$ is an edge in $\mathcal{G}_u$ then $d_G(v,w) = p+q+1$.*

**Proof.** By the requirements for an edge in $\mathcal{G}_u$, $B(v,p) \cap B(w,q) = \emptyset$ and $N_G(B(v,p)) \cap L(v,p)$ is nonempty and contained in $B(w,q)$. The first requirement implies $d_G(v,w) > p+q$ by Claim 4.10. The second implies $d_G(v,w) \leq p+q+1$ by Claim 4.11. ∎

This shows that the number of edges in $\mathcal{G}_u$ is at most $n^3$, because given $v$, $w$, and $p$, we know that $q$ must be $d_G(v,w) - p - 1$ if there is to be a possibility for an edge between $(v,p)$ and $(w,q)$ in $\mathcal{G}_u$. Thus the proof of Lemma 4.9 is complete. ∎

Now, we show that $\mathcal{G}_u$ is acyclic.

**Lemma 4.13** *Let $G$ be a graph, and let $u$ be any vertex of $G$. The graph $\mathcal{G}_u$ is acyclic.*

**Proof.** Observe first that a cycle in $\mathcal{G}_u$ can only contain vertices from $B_u$. This is because the vertices of $D_u$ are isolated, the vertices of $A_u$ have indegree 0 and the vertices of $C_u$ have outdegree 0.

Observe next that, for an edge $(v,p) \rightarrow (w,q)$ in $\mathcal{G}_u$ with both $(v,p)$ and $(w,q)$ in $B_u$, we have $d_G(u,w) \geq d_G(u,v) - p + 1 + q$. The argument for this is as follows: $u$ is neither in $B(v,p)$ nor in $B(w,q)$. Every path from $u$ to $w$ must pass through $B(v,p)$, so it must also pass through some edge $x \rightarrow y$ with $x$ in $B(v,p)$ and $y$ in $B(w,q)$. Now $d_G(u,w) = d_G(u,x) + 1 + d_G(y,w)$, but $d_G(u,x) \geq d_G(u,v) - d_G(x,v)$ by the triangle inequality, while $d_G(x,v) = p$ and $d_G(y,w) = q$ by the second requirement for edges in $\mathcal{G}_u$.

Now, assume for the sake of contradiction that we have a cycle $(c_1, p_1), (c_2, p_2), \ldots, (c_k, p_k), (c_1, p_1)$ in $\mathcal{G}_u$. Then, by the above argument, $d_G(u, c_1) \geq d_G(u, c_k) - p_k + 1 + p_1 \geq d_G(u, c_{k-1}) - p_{k-1} + 1 + p_k - p_k + 1 + p_1 \geq d_G(u, c_{k-1}) - p_{k-1} + 1 + p_1 \geq \ldots \geq d_G(u, c_1) - p_1 + 1 + p_1 = d_G(u, c_1) + 1$, which is an obvious contradiction. ∎

As $\mathcal{G}_u$ is acyclic, it is possible to find a minimum weight path from $A_u \cup D_u$ to $C_u \cup D_u$ in $O(n^3)$ time using topological sort and dynamic programming.

In order to move workload outside the outer loop, we can observe that the sets $A_u$, $B_u$, $C_u$ and $D_u$ do not change so much when we change $u$. First we can notice that the set $D_u$ does not depend on the choice of $u$ at all; thus we rename this set as $D$, independent of $u$. Now we define the following sets that are independent of $u$:

- $A = \{(v, p) \mid G(V \setminus B(v, p))$ is connected$\}$
- $B = \{(v, p) \mid G(V \setminus B(v, p))$ has two connected components$\}$.

These sets can be computed by the following algorithm:

**Algorithm** Compute $A$, $B$, and $D$
**Input:** A graph $G = (V, E)$.
**Output:** The sets $A$, $B$, and $D$
**begin**
    $A = \emptyset$; $B = \emptyset$; $D = \emptyset$;
    **for** each vertex $u$ in $V(G)$ **do**
        **for** each integer $k$ in $\{1, ..., rad(G)\}$ **do**
            $V' = \emptyset$;
            **for** each vertex $v$ in $V$ **do**
                **if** $d_G(u, v) > k$ **then**
                    $V' = V' \cup \{v\}$;
            Compute $c$, the number of connected components in $G(V')$;
            **if** $V' = \emptyset$ **then**
                $D = D \cup \{(u, k)\}$
            **else if** $c = 1$ **then**
                $A = A \cup \{(u, k)\}$
            **else if** $c = 2$ **then**
                $B = B \cup \{(u, k)\}$;
        **end-for**
**end**

Now, obviously $A_u \cup C_u = A$ and $B_u \subseteq B$ for any $u$. This means that, if the sets $A$ and $B$ are pre-computed, and we want to determine $A_u$ and $C_u$, we only need to iterate through every element $(v, p)$ in $A$ and determine whether to put it in $A_u$ or in $C_u$. That is done by checking whether $u \in B(v, p)$, something that can be done in constant time by checking whether $d_G(u, v) \leq p$, assuming that the distance between every pair of vertices in $G$ is pre-computed. To compute $B_u$ we iterate through $(v, p) \in B$, and assert that $d_G(u, v) > p$ in order for $(v, p)$ to be in $B_u$.

Now that we have improved the running time for finding the vertices of $\mathcal{G}_u$, we wish to accelerate the computation of the edges. In order to achieve that we need to be able to check for membership in $A_u$, $B_u$, and $C_u$ in constant time. To do this we construct an array $Member$, and the following algorithm explains this construction.

**Algorithm** Compute $A_u$, $B_u$ and $C_u$, and $Member$
**Input:** A graph $G = (V, E)$, a vertex $u$ and the sets $A$ and $B$.
**Output:** The sets $A_u$, $B_u$, $C_u$, and $Member$
**begin**
    $A_u = \emptyset$; $B_u = \emptyset$; $C_u = \emptyset$;
    **for** every $v \in V$ and every $k \in \{1, ..., rad(G)\}$ **do**
        $Member(v, k) = \emptyset$;
    **for** each element $(v, k)$ in $A$ **do**
        **if** $d_G(u, v) \leq k$ **then**
            $A_u = A_u \cup \{(v, k)\}$;
            $Member(v, k) = A_u$;
        **else**
            $C_u = C_u \cup \{(v, k)\}$;
            $Member(v, k) = C_u$;
        **end-if**
    **for** each element $(v, k)$ in $B$ **do**
        **if** $d_G(u, v) > k$ **then**
            $B_u = B_u \cup \{(v, k)\}$;
            $Member(v, k) = B_u$;
        **end-if**
**end**

After computing the vertices of $\mathcal{G}_u$ we want to compute the edges. Recall the three conditions given on page 7 for $(v, p) \rightarrow (w, q)$ to be an edge in $\mathcal{G}_u$. We can find the edges of $\mathcal{G}_u$ by trying all possible combinations of $v$, $p$, and $w$, letting $q = d_G(v, w) - p - 1$ (see Claim 4.12), and checking the conditions for an edge from $(v, p)$ to $(w, q)$. But before we do that, we need to verify that $(v, p)$ and $(w, q)$ indeed are vertices of $\mathcal{G}_u$. This is easily done using the $Member$ array.

The first condition of being an edge is fulfilled by our choice of $q$, as $d_G(v, w) = p + q + 1 > p + q$. The second condition can be tested using the following observation

**Observation 4.14** *Let $(v, p)$ be a vertex of $\mathcal{G}_u$. Then $R_u(v, p) \neq \emptyset \Leftrightarrow (v, p) \in A_u \cup B_u$, and $L_u(v, p) \neq \emptyset \Leftrightarrow (v, p) \in B_u \cup C_u$.*

Since we can use the $Member$ array to test membership in $A_u$, $B_u$ and $C_u$ in constant time, we can use Observation 4.14 to test for the second condition in constant time. Now, notice that we check whether $(v, p)$ and $(w, q)$ are vertices of $\mathcal{G}_u$ twice, first before testing for the first condition, and

then when testing the second. This is clearly unnecessary, so the first of these tests can be omitted.

Now we want to speed up the testing of the third condition of an edge. In order to do that, we will often need to know whether a pair of vertices lie in the same connected component in a given graph $G'$ on the form $G' = G(V \setminus B(u,p))$ where $(u,p)$ is a vertex of $\mathcal{G}_u$. By the definition of vertices in $\mathcal{G}_u$ such a graph has at most two connected components. Therefore we pre-compute a three dimensional array *Component* that assigns a 1 or a 0 to each vertex of $G(V \setminus B(u,p))$. Thus, for a vertex $(u,p)$ in $V(\mathcal{G}_u)$, $Component(u,p,v) = Component(u,p,w)$ if and only if $v$ and $w$ lie in the same connected component of $G(V \setminus B(u,p))$. This array can trivially be computed in $O(n^4)$ time by performing a depth first search in $G(V \setminus B(u,p))$ for every vertex $(u,p)$ in $\mathcal{G}_u$.

**Algorithm** Compute *Component*
**Input:** A graph $G = (V,E)$ and the set $B$
**Output:** The array *Component*
**begin**
  **for** every pair of vertices $u, v \in V$ and every $p \in \{1, ..., rad(G)\}$ **do**
    $Component(u,p,v) = 0$;
  **for** every $(u,p)$ in $B$ **do**
    $V' = \emptyset$;
    **for** each vertex $v$ in $V$ **do**
      **if** $d_G(u,v) > p$ **then**
        $V' = V' \cup \{v\}$;
    **if** $V' \neq \emptyset$ **then**
      $w$ = first element of $V'$;
      Compute $C$, the connected component of $G(V')$ containing $w$;
      **for** each vertex $v$ in $V(C)$ **do**
        $Component(u,p,v) = 1$;
    **end-if**
  **end-for**
**end**

In the following discussion, we will say that a pair of vertices of $\mathcal{G}_u$ form an *edge candidate* if $G(V \setminus B(v,p))$ has a nonempty component $C_v$ and $G(V \setminus B(w,p))$ has a nonempty component $C_w$ such that $(N_G(B(w,q)) \cap C_w) \subseteq B(v,p)$ and $(N_G(B(v,p)) \cap C_v) \subseteq B(w,q)$ in $G$.

**Observation 4.15** *If $(v,p) \rightarrow (w,q)$ is an edge in $\mathcal{G}_u$ then the pair $(v,p),(w,q)$ is an edge candidate.*

**Proof.** We let $C_v = R_u(v,p)$ and $C_w = L_u(w,q)$, and the proof follows. ∎

Now, we see that one can test whether a pair $(v,p)$, $(w,q)$ is an edge candidate outside the loop over all $u$'s. This means that we can build a three dimensional boolean array *Candidate* where an entry $Candidate(v,w,p)$

is set to true if and only if the pair $(v, p)$ and $(w, d_G(v, q) - p - 1)$ form an edge candidate. This array can be built in $O(n^4)$ time, as we can let $q = d_G(v, q) - p - 1$, scan all the neighbors of $B(v, p)$ in $G(V \setminus B(v, p))$ in the component labeled 0, and check whether they are contained in $B(w, q)$. We repeat the scan in the component labeled 1. Now we are done checking whether there exists a $C_v$ so that $(N_G(B(v, p)) \cap C_v)) \subseteq B(w, q)$. The other part of the requirement can be checked similarly.

**Algorithm** Compute *Candidate*
**Input:** A graph $G = (V, E)$ and the array *Component*
**Output:** The array *Candidate*
**begin**
  **for** every pair $v, w \in V$ and every $p \in \{1...rad(G)\}$ **do**
    $q = d_G(v, w) - p - 1$;
    $isCandidateV0 =$ **true**;
    $isCandidateV1 =$ **true**;
    $isCandidateW0 =$ **true**;
    $isCandidateW1 =$ **true**;
    **for** every $x$ in $V$ **do**
      **if** $d_G(v, x) = p + 1 \wedge Component(v, p, x) = 0 \wedge d_G(w, x) > q$ **then**
        $isCandidateV0 =$ **false**;
      **if** $d_G(v, x) = p + 1 \wedge Component(v, p, x) = 1 \wedge d_G(w, x) > q$ **then**
        $isCandidateV1 =$ **false**;
      **if** $d_G(w, x) = q + 1 \wedge Component(w, q, x) = 0 \wedge d_G(v, x) > p$ **then**
        $isCandidateW0 =$ **false**;
      **if** $d_G(w, x) = q + 1 \wedge Component(w, q, x) = 1 \wedge d_G(v, x) > p$ **then**
        $isCandidateW1 =$ **false**;
    **end-for**
    $Candidate(v, w, p) = (isCandidateV0 \vee isCandidateV1) \wedge (isCandidateW0 \vee isCandidateW1)$;
  **end-for**
**end**

The next lemma shows how this structure can be used.

**Lemma 4.16** *There is an edge $(v, p) \rightarrow (w, q)$ in $\mathcal{G}_u$ if and only if $(v, p)$ and $(w, q)$ are contained in $A_u \cup B_u \cup C_u$, fulfill the first two conditions for an edge in $\mathcal{G}_u$, form an edge candidate, and satisfy $w \in R_u(v, p) \wedge v \in L_u(w, q)$.*

**Proof.** If there is an edge from $(v, p)$ to $(w, q)$ then one can easily confirm that all requirements in the lemma are met. In the other direction, we see that the first two conditions for an edge are fulfilled. As for the third, we have that they form an edge candidate, so it holds to show that $C_v$ is in fact $R_u(v, p)$ while $C_w$ is $L_u(w, q)$. Assume for contradiction that $C_v$ is $L_u(v, p)$. Then $B(w, q)$ contains vertices both in $L_u(v, p)$ and $R_u(v, p)$ which is impossible as $B(w, q)$ is connected while $L_u(v, p)$ and $R_u(v, p)$ are separate components. This means that $C_v$ indeed must be $R_u(v, p)$. The proof that $C_w$ is $L_u(w, q)$ uses the same strategy and the lemma follows. ∎

In order to check for an edge in $\mathcal{G}_u$ one can use the requirements in Lemma 4.16. If we make use of our pre-computed arrays we can check whether a given pair $(v,p)$, $(w,q)$ fits these requirements in constant time. We are now ready to give the details of a more efficient version of Algorithm MPBD.

**Algorithm** Revised Minimum Path Broadcast Domination - RMPBD
**Input:** A graph $G = (V, E)$.
**Output:** An efficient broadcast domination function $f$ with $C_f(V) \leq \gamma_{bp}(G)$.
**begin**
Compute the distance matrix and the radius of $G$;
Compute $A$, $B$, and $D$;
Compute $Component$;
Compute $Candidate$;
**for** each vertex $v$ in $G$ **do**
$f(v) = 0$;
Let $P$ be a dummy path with $W(P) = rad(G) + 1$;
**for** each vertex $u$ in $G$ **do**
Compute $A_u$, $B_u$, $C_u$, and $Member$;
$E(\mathcal{G}_u) = \emptyset$;
**for** each pair $v, w \in V$ and each $p \in \{1, ..., d_G(v,w) - 3\}$ **do**
$q = d_G(v,w) - p - 1$;
**if not** $(Member(v,p) = A_u \vee Member(v,p) = B_u)$
$\wedge (Member(w,q) = B_u \vee Member(w,q) = C_u)$ **then continue**
**if not** $Candidate(v,w,p)$ **then continue**
**if** $d_G(u,v) - p \leq d_G(u,w) - q$ **then**
$E(\mathcal{G}_u) = E(\mathcal{G}_u) \cup ((v,p) \rightarrow (w,q))$
**else**
$E(\mathcal{G}_u) = E(\mathcal{G}_u) \cup ((w,q) \rightarrow (v,p))$;
**end-for**
Find a minimum weight path $P_u$ starting in a vertex of $A_u \cup D$ and ending in a vertex of $C_u \cup D$;
**if** $W(P_u) < W(P)$ **then**
$P = P_u$;
**end-for**
**for** each vertex $(v,p)$ on $P$ **do**
$f(v) = p$;
**end**

The correctness of this algorithm follows directly from Lemma 4.7, because this algorithm does exactly the same things as the one proposed in the previous section. The complexity analysis is deduced from the discussion above.

**Lemma 4.17** *The running time of Algorithm RMPBD on a graph $G$ with $n$ vertices is $O(n^4)$.*

**Proof.** ¿From our discussion above, it should be clear that all the pre-computation can be done in $O(n^4)$ time. In the loop over vertices $u$ we can

compute the sets $A_u$, $B_u$ and $C_u$ using $O(n^2)$ operations. The edges of $\mathcal{G}_u$ are found in $O(n^3)$ time, and a minimum weight path is found in $O(n^3)$ time as well. As these operations are inside the loop over $u$ they are repeated $O(n)$ times, yielding a time complexity of $O(n^4)$ for the whole algorithm. ∎

### 4.3 Optimal broadcast domination for all cases

Now we want to compute an optimal broadcast domination for any given graph $G$. Our approach will be as follows. Let $x$ be any vertex of $G$. For each $k$ between 1 and $rad(G)$ such that $G' = G(V \setminus B(x,k))$ is connected or empty, we run the minimum path broadcast domination algorithm RMPBD on $G'$. Our algorithm for the general case is given below.

**Algorithm** Optimal Broadcast Domination - OBD
**Input:** A graph $G = (V, E)$.
**Output:** An optimal broadcast domination function $f$ on $G$.
**begin**
    $opt = rad(G) + 1$;
    **for** each vertex $x$ in $G$ **do**
        **for** $k = 1$ **to** $rad(G)$ **do**
            **if** $G' = G(V \setminus B(x,k))$ is connected or empty **then**
                $f =$ RMPBD$(G')$;
                **if** $c_f(V \setminus B(x,k)) + k < opt$ **then**
                    $opt = c_f(V \setminus B(x,k)) + k$;
                    $f(x) = k$;
                    **for** each vertex $v$ in $B(x,k) \setminus \{x\}$ **do**
                        $f(v) = 0$;
                **end-if**
            **end-if**
**end**

In this way, we consider all broadcast dominations $f$ whose corresponding domination graphs are paths or cycles. The advantage of this approach is its simplicity. The disadvantage is that we also consider many cases that do not correspond to a path or a cycle, which we could have detected with a longer and more involved algorithm. However, these unnecessary cases do not threaten the correctness of the algorithm, and detecting them does not decrease the asymptotic time bound.

**Theorem 4.18** *Algorithm OBD computes an optimal broadcast domination of any given graph.*

**Proof.** Let $G = (V, E)$ be the input graph. By Theorem 3.3 and Corollary 3.5, there is a vertex $x$ in $V$ and an integer $k \in [1, rad(G)]$ such that the graph $G' = G(V \setminus B(x,k))$ has an efficient optimal broadcast domination $f'$ where the domination graph $G'_{f'}$ is a path, and that $f'$ can be extended

to an optimal broadcast domination $f$ for $G$ with $f(x) = k$, $f(v) = 0$ for $v \in B(x,k)$ with $x \neq v$, and $f(v) = f'(v)$ for all other vertices $v$. Algorithm RMPBD computes an optimal broadcast domination of $G'$, and since Algorithm OBD tries all possibilities for $(x,k)$, the result follows. ∎

Note that although there is always an efficient optimal broadcast domination $f$ such that $G_f$ is a cycle or a path, there can of course exist other optimal broadcast dominations $f'$ with $c_{f'}(V) = c_f(V)$ such that $G_{f'}$ is not a path or a cycle, and such that $f'$ is not efficient. The optimal broadcast domination returned by algorithm OBD does not necessarily correspond to a path or a cycle, since we do not force the endpoints (or forbid the interior points) of the path for $G'$ to be neighbors of $B(x,k)$. Nor is the returned broadcast necessarily efficient, as some ball $B(v,p)$ might have an outreach outside of $G'$ and might overlap with $B(x,k)$.

**Theorem 4.19** *The running time of Algorithm OBD on a graph $G$ with $n$ vertices is $O(n^6)$.*

**Proof.** First, algorithm OBD finds the radius of $G$, which can be done in $O(n^3)$ time. For every iteration of the inner loop we find out whether $G'$ is connected, and call algorithm RMPBD. The first of these tasks can be done in $O(n+m) = O(n^2)$ time, while the second is done in $O(n^4)$ time by Lemma 4.17. The inner loop iterates $O(n^2)$ times, and the proof is complete. ∎

# 5 Concluding remarks

In this paper we have shown that the broadcast domination problem is solvable in polynomial time on all graphs. For further research, more efficient algorithms for this problem should be of interest. One could also look at generalizations and slightly different models. We outline a couple of them here.

The optimal broadcast domination problem studies the cost $c_f(V) = \sum_{v \in V} f(v)$ of a broadcast domination $f$ on a graph $G = (V,E)$. Other definitions of the cost of a broadcast may be appropriate depending on the application, since the cost of a broadcast can be different from the value of a broadcast. To be more precise, one could define a cost function $c(i)$, and let the total cost be $c_f(V) = \sum_{v \in V} c(f(v))$. Thus in our case $c(i) = i$ for all $i$. Our polynomial time algorithm can be used for all cost functions $c$, where $c(i) + c(j) \geq c(i+j)$ for all integers $i$ and $j \geq 0$. The reason for this is that the algorithm will work if the structural results hold. The proofs of these are all based upon that we in some cases can replace several weak dominators by one strong. If the cost function is sublinear, this replacement is even cheaper than in the linear case, and the proofs are still valid. If

we allow the cost function to be a part of the input the problem becomes NP-hard, because we can let $c(0) = 0$, $c(1) = 1$, and $c(i) > n$ for all $i > 1$, which gives a direct reduction from the standard dominating set problem. It seems that when the cost function gets "superlinear enough" the problem becomes NP-hard. It would be interesting to see how close one can tighten the gap between cost functions that make the problem polynomial and those that make it NP-hard.

One might also want to study weighted graphs. If we assign a weight to each edge and redefine the length of a path to be the sum of edge weights, we now get a natural generalization of the problem. However the algorithm for the unweighted case is not easily generalized to work for weighted graphs as well, because the efficiency result from [4] no longer applies.

# References


[1] J. Bar-Ilan, G. Kortsarz, and D. Peleg. How to allocate network centers. *J. Algorithms*, 15:385–415, 1993.

[2] C. Berge. *Theory of Graphs and its Applications.* Number 2 in Collection Universitaire de Mathematiques. Dunod, Paris, 1958.

[3] J. R. S. Blair, P. Heggernes, S. Horton, and F. Manne. Broadcast domination algorithms for interval graphs, series-parallel graphs, and trees. *Congressus Numerantium*, 169:55 – 77, 2004.

[4] J. E. Dunbar, D. J. Erwin, T. W. Haynes, S. M. Hedetniemi, and S. T. Hedetniemi. Broadcasts in graphs. 2004. Submitted to *Disc. Appl. Math.*

[5] D. J. Erwin. Dominating broadcasts in graphs. *Bull. Inst. Comb. Appl.*, 42:89–105, 2004.

[6] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W. H. Freeman and Co., 1978.

[7] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater. *Domination in Graphs: Advanced Topics.* Marcel Dekker, New York, 1998.

[8] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater. *Fundamentals of Domination in Graphs.* Marcel Dekker, New York, 1998.

[9] P. Heggernes and D. Lokshtanov. Optimal broadcast domination of arbitrary graphs in polynomial time. In *Proceedings of the 31stWorkshop on Graph Theoretic Concepts in Computer Science- WG 2005*, pages 187 – 198. Springer Verlag, 2005. LNCS 3787.

[10] M. A. Henning. Distance domination in graphs. In T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, editors, *Domination in Graphs: Advanced Topics*, pages 321–349. Marcel Dekker, New York, 1998.

[11] S. B. Horton, C. N. Meneses, A. Mukherjee, and M. E. Ulucakli. A computational study of the broadcast domination problem. Technical Report 2004-45, DIMACS Center for Discrete Mathematics and Theoretical Computer Science, 2004.

[12] C. L. Liu. *Introduction to Combinatorial Mathematics*. McGraw-Hill, New York, 1968.

[13] O. Ore. *Theory of Graphs*. Number 38 in American Mathematical Society Publications. AMS, Providence, 1962.

[14] P. J. Slater. $R$-domination in graphs. *J. Assoc. Comput. Mach.*, 23:446–450, 1976.