Approximation Algorithms for Minimum Time Broadcast

(Extended abstract)

Guy Kortsarz * David Peleg* †

Abstract

This paper deals with the problem of broadcasting in minimum time. Approximation algorithms are developed for arbitrary graphs, as well as for several restricted graph classes.

1 Introduction

This work concerns efficient algorithms for broadcast in a communication network. The network is modeled by a connected graph $G = (V, E)$ consisting of a set V of vertices, $V = \{v_1, .., v_n\}$ representing the processors, and a set E of edges, $E = \{e_1, ..., e_m\}$ representing the communication lines between the processors. This work assumes the *telephone* communication model (cf. [HHL88]). In this model messages are exchanged during *calls* placed over edges of the network. A round is a series of calls carried out simultaneously. Each round is assumed to require one unit of time, so round t begins at time $t-1$ and ends at time t. A vertex may participate in at most one call during a given round, however there are no limitations on the amount of information that can be exchanged during a given call. At a given round, if a call is placed over an edge e, we say that e is *active* in this round, else it is *idle*. The rule governing the activation of edges at each round is called a *schedule*.

While our communication model is very general, and allows arbitrarily complex communication patterns in the system, it is usually beneficial to define some basic *communication primitives,* which are simple communication procedures from which more involved protocols can be constructed. One important primitive that was studied extensively in the telephone model is *broadcasting* (cf. [HHL88]). A *broadcasting* problem refers to the process where a distinguished vertex originates a message M, that has to become known to all other processors. The efficiency of a broadcast scheme is usually measured by the number of time units it takes to complete the broadcast. Given a scheme S for broadcasting in a graph G, denote the broadcasting

*Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot 76100, Israel.

†Supported in part by an Allon Fellowship, by a Bantrell Fellowship and by a Walter and Elise Haas Career Development Award.

time from v using S by $b(v, G, S)$. Define $b(v, G)$, the ***broadcast time*** of a vertex v in G, as the minimum time for broadcasting a message originated at v in G, i.e., $b(v, G) = \min_S\{b(v, G, S)$. We denote it simply by $b(v)$ when the context is clear. We denote $b(G) = \max_v\{b(v, G)\}$.

For any connected graph G of n vertices and originator u, $b(u) \geq \lceil \log n \rceil$, since in each time unit the number of informed vertices can at most double. Another simple lower bound for $b(v)$ for an arbitrary v is $b(v) \geq Diam(v)$, since a vertex may only send information to a neighboring vertex at each round. An example of a graph for which $b(G) = \lceil \log n \rceil$ is K_n, the complete graph of n vertices.

In any connected graph G, a broadcast from a vertex u determines a spanning tree rooted at u. The parent of a vertex v is the vertex w that transmitted the message to v. Clearly, one may assume that such a vertex is unique. Even when using an arbitrary spanning tree, it is clear that at each step the set of informed vertices grows by at least one. Thus for each network G, $b(G) \leq n - 1$. We can not always improve upon this result. For example, in S_n, the star of n vertices the broadcast time is $b(S_n) = n - 1$.

Given a network $G = (V, E)$ and an originator u, the ***Minimum Broadcast Time*** (MBT) problem is to broadcast the message from u to the rest of the vertices, in $b(u)$ time units. This problem too has received considerable attention in the literature. For example, Slater, Cockayne and Hedetniemi study broadcasting in trees [SCH81], and Farley and Hedetniemi study broadcasting in grid graphs [FH78]. For a comprehensive survey on the subject of gossiping and broadcasting see [HHL88].

Note that the MBT problem in general graphs is NP-complete (cf. [GJ79]) and thus it is unlikely to find an exact solution for it. In this paper we consider approximation schemes for broadcast, namely, algorithms that do not give an exact solution, but rather give a solution that is not "too far" from the optimum. More formally, we call a scheme A for broadcasting on a family of graphs $\mathcal{F}$ a $k-$ *approximation* scheme if for every $G \in \mathcal{F}$ and vertex $v \in V$, $b(v, G, A) \leq k \cdot b(v)$. We say that a scheme S has a (k, k')-approximation ratio if $b(v, G, S) \leq k \cdot b(v) + k'$. A ratio is ***$k$-additive*** if it is an $(O(1), k)$ ratio. We give approximation schemes for broadcast on several networks classes, and analyze their approximation ratio. In particular, we give an approximation algorithm for general graphs with $O(\sqrt{n})$-additive ratio. For ***chordal*** and $O(1)$-separable graphs we have a scheme with ratio $O(\log n)$. We also give an efficient approximation scheme for broadcasting in a special family of graphs called ***trees of cliques***, and some tools for broadcasting in a bounded-face planar graph.

Although we formulate our statements in the telephone model, virtually all our results for broadcast hold also for the message-passing model, assuming that a processor may send at most one fixed size message in each time stamp. Since as far as the broadcast operation is concerned, these two models are equivalent in power.

Several generalizations of the MBT problem appear in the literature. In [Far80], Farley suggests to reconsider the assumption that a vertex may call only neighboring vertices. Farley defines two possible variants of the model using long distance calls, the ***open path*** and the ***open line*** models. In the open path model, communication is carried along vertex disjoint paths. At each round, an informed member v may call a noninformed vertex u on an (arbitrarily long) path, adding u to the set of informed vertices. Two paths corresponding to two different pairs must be vertex disjoint.

We denote the time needed to complete the broadcast from a distinguished vertex v in the graph G in the open path model, by $b_{op}(G,v)$ (or $b_{op}(v)$). We also denote $b_{op}(G) = \max_{v \in V}\{b_{op}(v)\}$. Note that $b_{op}(G) \geq \log n$, however, we can not argue that $b_{op}(G) \geq Diam(G)$. We call the problem of broadcasting from a vertex v in $b_{op}(G)$ time units, OMBT. In this paper we give an approximation algorithm for OMBT with ratio $O(\log n / \log\log n)$. (The open line model is similar, except that the paths used in a communication round need only be *edge* disjoint. The problem of approximating broadcast in this model is essentially solved up to logarithmic factors, since as shown in [Far80] the open line model enables broadcast from an arbitrary vertex in $\lceil \log n \rceil$ time units.)

Returning to the telephone model, another generalization we consider is to assume that the set V_0 of informed vertices at the beginning of the run, need not consist of a single vertex, but can be an arbitrary subset of V. Denote this problem by SMBT, and denote the time needed to broadcast from V_0 by $b(V_0, G)$ or $b(V_0)$. As mentioned in [GJ79], SMBT is NP-complete even for $k = 4$ where k is the bound on the time for completing the broadcast.

In order to approximate the MBT problem, we define in the sequel a problem called *Bipartite Edge Scheduling* (BES) which will be shown to be related to MBT. We give a *pseudo polynomial* solution to BES, and use it in virtually all our approximations for MBT.

2 Preliminaries

We denote the maximal degree of a vertex in a network $G = (V, E)$, by $\Delta(G)$ (or simply Δ, if the context is clear). Throughout the paper we denote the number of vertices of a graph G by n, and the number of edges by m. We denote by Z^+ the set of nonnegative integers. Given a graph $G = (V, E)$ and two vertices $v, w \in V$, we denote the number of edges in a shortest path between v and w by $dist(v, w)$. We denote $Diam(v) = \max_w\{dist(v, w)\}$. The diameter of the graph G is $\max_v\{Diam(v)\}$.

A *cluster* in a graph G is a subset V' of the vertices such that the subgraph induced by V' is *connected*. Two clusters V', V'' are said to be disjoint if $V' \cap V'' = \emptyset$. Two disjoint clusters C_1 and C_2 are said to be independent if there is no edge connecting a vertex of C_1 to a vertex of C_2.

Definition 2.1 *Let $G = (V, E)$ be a graph and let $S = \{v_1, \ldots, v_k\} \subset V$ be a subset of the vertices. A subtree $T = (V_1, E_1)$ of G rooted at a distinguished vertex v is a* shortest path tree leading from v to S *iff $S \subseteq V_1$, each path from v to v_i in T is a shortest path in G, and every leaf of T belongs to S. Denote a SPT leading from a vertex v to a set S by $SPT(v, S)$.*

We now state the definition of a *control graph* of a subset $V' \subseteq v$, in a graph $G = (V, E)$. This definition will be useful in most of our approximation algorithms.

Definition 2.2 *Suppose that the clusters formed when extracting V' from the graph G are $\{C_1, \ldots, C_k\}$. The* control graph *of V' in G is a bipartite graph $D_{V',G} = (V_1, V_2, A)$, where $V_1 = V'$, $V_2 = \{C_1, \ldots, C_k\}$ and A contains an edge (v, C_i) iff there is an edge between v and some vertex of C_i in G.*

3 The Bipartite Edge Scheduling Problem

The *Bipartite Edge Scheduling* (BES) problem is a basic tool we use in our approximation algorithms for MBT. In order to describe the BES problem we need some preliminary definitions. Let $G = (V_1, V_2, A)$ be a bipartite graph, where the vertices of V_1 know a message that has to be broadcasted to the vertices of V_2. I.e., the initial set of informed vertices is V_1. We call each vertex in V_1 a *server* and each vertex in V_2 a *customer*.

The problem imposes an additional requirement. Suppose that each customer $v_2 \in V_2$ has a task t_{v_2} to perform, and the *length* of the task (i.e., the time it takes to complete it) depends upon which vertex of V_1 transmits the message to v_2. I.e., for each edge $e = (v_1, v_2) \in A$ there is a weight $w(e) \in Z^+$, such that if v_1 transmites the message to v_2 then it takes $w(e)$ time units for v_2 to complete the task t_{v_2}, starting from the arrival time of the message (and of course, no vertex of V_2 can start performing its task before it is informed). It is required to minimize the completion time of the entire process, namely, the time by which every vertex in V_2 completes its job.

Thus a solution must determine a function $F : V_2 \longrightarrow V_1$, such that $F(v_2) = v_1$ means that v_2 receives the message from v_1. Clearly, if $F(v_2) = v_1$ then $(v_1, v_2) \in A$ must hold. If $F(v_2) = v_1$ we say that v_1 *controls* (or *dominates*) v_2. Suppose that v controls $u_1, ..., u_k$. Denote $e_i^v = (v, u_i)$ and without loss of generality assume that the vertices are ordered so that $w(e_1^v) \geq w(e_2^v) \geq \ldots \geq w(e_k^v)$. Intuitively, it is logical for v to send the message first to u_1, then to u_2, etc. For a given function F, denote

Definition 3.1 *The* weight *of a function F is* $w(F) = \max_v\{\max_i\{i + w(e_i^v)\}\}$

We are now ready to state the BES problem. Given a bipartite graph $G = (V_1, V_2, A)$ with no isolated vertices, and a weight $w(e) \in Z^+$ for every edge $e \in A$, determine a function $F : V_2 \longrightarrow V_1$ whose weight $w(F)$ is minimal. We call this function F the *minimum control function* for G (or just the minimal function).

It is important to note that in all the applications given in this paper to the *BES* problem, the weights satisfy $\max_e\{w(e)\} \leq n$. Thus in order to use this problem as a basic auxiliary tool for the study of MBT, a *pseudo polynomial* solution will suffice.

A special variant of the *BES* problem arises when for each $v_2 \in V_2$ the weights of the edges entering v_2 are identical. In this case, we might as well associate the weight with v_2 itself. We call this variant of the problem the *Bipartite Vertex Schedule* (BVS) problem. If *all* the weights are identical (thus without loss of generality all the weights are 0), a solution only needs to minimize the maximal number of vertices dominated by a single vertex, i.e., minimize the size of the largest inverse image of F, $\max_v |\{u : F(u) = v\}|$.

We now present a pseudo polynomial solution to the BES problem. Given a positive integer j we check if there exists a corresponding function of weight j. The solution method for this problem involves flow techniques . The original graph is modified as follows. Create a source vertex s and a sink vertex t. For a function of weight j to be feasible, a server v can not dominate a customer u such that $w(v, u) \geq j$. Assume that w_v is the maximal weight that is less than or equal to $j - 1$ of an edge incident to $v \in V_1$. Duplicate v into $w_v + 1$ different copies and arrange the copies in an arbitrary order

$v_1, \ldots, v_{w_v+1}$. For v_1, the first copy of v, create a directed edge (s, v_1), with capacity $j - w_v$, and create a directed edge (v_1, u) with capacity 1, from v_1 to every customer $u \in V_2$ such that $(v_1, u) \in E$. For v_i the i'th copy of v, $i \geq 2$, create a directed edge (s, v_i) with capacity 1, and a directed edge (v_i, u) with capacity 1 to all the customers u such that $(v, u) \in E$ and $w(v, u) \leq w_v - i + 1$. Finally create for each customer $u \in V_2$ a directed edge (u, t) with capacity 1. Call the resulting graph $F_{j,G}$.

Since there are exactly $|V_2|$ edges entering t, and each of them is of capacity 1, the maximal flow can not exceed $|V_2|$. We now claim the following:

Lemma 3.2 *Given a BES problem P on a given graph G and the corresponding construction $F_{j,G}$, the maximal flow is $|V_2|$ iff there exists a function F such that $w(F) \leq j$.*

The minimum weighted function can now be found by using binary search. function (whose weight is j_1). The flow computation is performed for at most a polynomial number of times. Note however that a vertex may be duplicated for a number of times that can equal the maximal number in the input. To summarize, we have established the following.

Theorem 3.3 *There exists a pseudo polynomial algorithm for the BES problem.*

Corollary 3.4 *There exists a pseudo polynomial algorithm for the BVS problem.*

Given a BES instance with a graph $G = (V_1, V_2, E)$, and a weight function w, we give a general pseudo polynomial procedure to solve the scheduling problem.

Algorithm 3.5

1. Compute a minimal function F.
2. Suppose that $v \in V_1$ dominates the vertices $v_1, \ldots, v_k \in V_2$, and without loss of generality assume that $t_{v_i} \geq t_{v_{i+1}}$, for each i.
3. Every vertex v sends the message to the vertices v_i in increasing order of indices.
4. If a vertex $v' \in V_2$ gets the message from v then it finishes its task in $w(v, v')$ time units starting from the arrival time.

Fact 3.6 *The scheduling process terminates in $w(F)$ time units.*

4 Approximating Broadcast in General Graphs

In this section we give an approximation scheme for broadcasting in general graphs, both in the telephone model and in the open path model.

4.1 Approximating MBT

In this section we consider approximation schemes for broadcasting in general graphs, which guarantee a ratio of $(2, O(\sqrt{n}))$. The method used for the approximation is based on dividing the set of vertices into clusters of size $\lceil\sqrt{n}\rceil$, and broadcasting separately on those clusters. This scheme is based upon the following lemma. (The proofs of this and several subsequent lemmas are omitted from this abstract, and are deferred to the full paper.)

Lemma 4.1 *The set of vertices of any graph $G = (V, E)$ can be (polynomialy) decomposed into two sets of clusters $\mathcal{A}$ and $\mathcal{B}$, such that $|\mathcal{A}| \leq \sqrt{n}$, the clusters in $\mathcal{A} \cup \mathcal{B}$ are pairwise disjoint, $(\bigcup \mathcal{A}) \cup (\bigcup \mathcal{B}) = V$, the size of each cluster $C' \in \mathcal{A}$ is $|C'| = \lceil\sqrt{n}\rceil$, the size of each cluster $C' \in \mathcal{B}$ is bounded by $|C'| \leq \sqrt{n}$, and the clusters in $\mathcal{B}$ are pairwise independent.*

Let $G = (V, E)$ be a graph and $V' \subset V$ subset of the vertices. Form the control graph of V', $D_{V',G} = (V_1, V_2, A)$, in G as in Def 2.2. Let the weight of each edge be 0. In this scenario we claim:

Lemma 4.2 *Let F be a minimum control function for $D_{V',G}$. Then $w(F) \leq b(V', G)$.*

In the next lemma we use a shortest paths tree $SPT(v, S)$ rooted at a vertex v and leading to a set S of vertices (see Def. 2.1). Note that it is easy to construct such a tree in time polynomial in $|E|$ using a shortest path tree algorithm; simply construct a shortest path tree T spanning all the graph vertices, and iteratively exclude from it each leaf not belonging to S, until no such leaf exists. Recall that given a tree $T = (V_1, E_1)$ and a vertex $v \in V_1$ it is easy to compute the optimal scheme for broadcasting on T from v (cf. [SCH81]). Let us call the optimal scheme for broadcasting in a tree the *OT scheme*. By Using the OT scheme on $SPT(v, V')$ we can show

Lemma 4.3 *Transmitting a message from a vertex v to a subset $V' \subseteq V, |V'| = l$ of the graph, can be performed in no more than $l - 1 + Diam(v)$ time units.*

We are now ready to approximate broadcast on general graphs.

Algorithm 4.4 *Approximating broadcast in general graphs*
Input: a graph $G = (V, E)$ and a distinguished vertex $v \in V$.

1. Decompose the vertices of V into two sets of clusters $\mathcal{A}$ and $\mathcal{B}$ as in Lemma 4.1.
2. Choose for each cluster C in $\mathcal{A}$ a single representative vertex v_C. Let R denote the set of representatives, $R = \{v_C \mid C \in \mathcal{A}\}$.
3. Transmit the message from v to all the vertices of R by choosing an arbitrary tree $SPT(v, R)$ leading from v to R, and applying the OT scheme to the tree.
4. Choose for each cluster $C \in \mathcal{A}$ an arbitrary spanning tree rooted at its representative v_C, and broadcast (in parallel) in the clusters of $\mathcal{A}$ according to the OT scheme.
5. Construct the bipartite control graph $D_{V',G}$ where $V' = (\bigcup \mathcal{A}) \cup \{v\}$. Compute a minimum control function F. Assume that a vertex v' dominates clusters $C_1, .., C_k \in \mathcal{B}$. Choose for each C_i an arbitrary vertex $v_i \in C_i$ connected to v' and deliver the message from v' to $v_1, .., v_k$ (in arbitrary order). This is done in parallel for all the dominating vertices of V'.

6. Choose for each cluster in $\mathcal{B}$ an arbitrary spanning tree rooted at an informed vertex and transmit the message in parallel to all the vertices in the clusters of $\mathcal{B}$ using the OT scheme in each cluster.

Analyzing the time required by Alg. 4.4, we get the following theorem.

Theorem 4.5 *The broadcast time of* Alg. 4.4 *from a vertex v in a graph G, is bounded by $3 \cdot \sqrt{n} + Diam(v) + b(v)$ time units.* ∎

4.2 Broadcasting in the Open Path Model

Let us turn to the open path communication model. Algorithm 4.4 can be generalized to give a good approximation scheme for the open path broadcasting problem. It is easy to see that Lemma 4.2 holds even in the open path model.

Lemma 4.6 *Let T be a tree rooted at v, with up to k leaves. Then it is possible to broadcast a message from the root v to all the vertices of the tree in the open path model, in no more than $2 \cdot k + \log n$ time units.*

This discussion motivates the following algorithm. Our approach for the approximation for OMBT, is based on the following idea. First we define sets of representatives, $\{R_1, \ldots, R_f\}$, where $R_1 = V$, $|R_f| \leq \log n$ and $f \leq \log n / \log\log n$. To each set R_j and vertex $v \in R_j$ there is a corresponding tree T_j^v, containing exactly $\lceil \log n \rceil$ vertices of R_{j-1}. The trees corresponding to different vertices in R_j are *vertex disjoint*. The main algorithm operates in f stages. First it informs the vertex set R_f. Then it proceeds to inform the sets R_j in reverse order of indices, i.e., at the end of stage i of the main algorithm, the message is known by the set R_{f-i+1}, and the goal of the next stage is for R_{f-i+1} to inform R_{f-i}. We next give Procedure $Choose-Rep$, that chooses the sets R_i of representatives, and the corresponding trees $T_i^v, v \in R_i$. After that, we give the main algorithm that uses $Choose-Rep$ to approximate OMBT.

Throughout the execution of procedure Choose-Rep we extract trees from G. The set of remaining vertices, i.e., the set of vertices not extracted yet from G, will be denoted by V'. The set of clusters in the graph induced by V' will be denoted by $\mathcal{C}$.

Algorithm 4.7 $Choose-Rep$
Input: A graph $G = (V, E)$ and a distinguished vertex $v \in V$.

1. $V' \leftarrow V$, $R_1 \leftarrow V, i \leftarrow 1, \mathcal{C} \leftarrow \{V\}$.
2. **repeat**
   (a) $R_{i+1} \leftarrow \emptyset$.
   (b) **while** $\mathcal{C} \neq \emptyset$ **do:**
      i. Choose cluster $C \in \mathcal{C}$. Select $\lceil \log n \rceil$ vertices in $C \cap R_i$ arbitrarily, except that if $v \in C$, take v as one of them. Let $v_1, \ldots, v_{\lceil \log n \rceil}$, such that we set $v_1 = v$ if $v \in C$. Select in C a subtree $T_i^{v_1}$ leading from v_1 to $\{v_2, \ldots, v_{\lceil \log n \rceil}\}$.
      ii. Extract $T_i^{v_1}$ from C, and set $V' \leftarrow V' \setminus T_i^{v_1}$.

iii. Set $C \leftarrow C \cup \{B : B$ is a connected component of $C \setminus T^{v_1}, |B \cap R_i| > \lceil \log n \rceil\}$.

end-while

(c) For every tree $T_i^{v_1}$ obtained in stage (b), add its root v_1 to R_{i+1}.

(d) $i \leftarrow i + 1$.

3. **until** $|R_i| \leq \lceil \log n \rceil$.

We now proceed to define the main algorithm. Throughout the algorithm we maintain a set R of informed vertices that equals R_j for some j. The point is that j decreases by one in each iteration, thus at last $R = R_1 = V$.

Algorithm 4.8 *Approximation algorithm for OMBT*

Input: A graph $G = (V, E)$ and a distinguished vertex $v \in V$.

1. Apply procedure *Choose* – *Rep* on G and v. Assume that the sets of representatives are $\{R_1, \ldots, R_f\}$.
2. Choose an arbitrary tree leading from v to the other vertices of R_f, and inform all the vertices in R_f using the scheme suggested in the proof of Lemma 4.6.
3. $R \leftarrow R_f, i \leftarrow f$.
4. **repeat**
   (a) Each vertex $u \in R_i$ informs (in parallel) all the vertices in T_i^u using the scheme suggested in the proof of Lemma 4.6.
   (b) Let G_i' denote, $G_i' = G \setminus \bigcup_{u \in R_i} T_i^u$. Let $C_1^i, \ldots, C_s^i$ denote the clusters in the graph induced by G_i'.
   (c) The vertices $\bigcup T_i^u$ inform a vertex v_j in C_j^i, for each $1 \leq j \leq s$, using a minimum control function, as in step 5 of Alg. 4.4.
   (d) Choose for each j, a tree TL_j leading from v_j to the vertices in $R_{i-1} \cap C_j^i$.
   (e) The vertices v_j inform the vertices of TL_j using the scheme of Lemma 4.6.
   (f) $R \leftarrow R_{i-1}, i \leftarrow i - 1$.
5. **until** $i = 1$.

It is clear from the algorithm that when stage 4(e) of Alg. 4.8 is completed, all the vertices of R_{i-1} know the message. It follows that at the end all the vertices are informed.

Theorem 4.9 Algorithm 4.8 *is an* $O(\log n / \log\log n)$ *approximation scheme for OMBT.*

The method of Alg. 4.8 can be used to deal with the MBT problem as well. However, at each stage, broadcasting in a tree T may take $O(\log n + h(T))$ time units, where $h(T)$ is the height of T. Since the diameter of a subcluster of a graph G may largely increase, this may not be a good approximation scheme in the worst case.

However, it is instructive to consider the behavior of Alg. 4.8 on random inputs. Let us consider a random graph $G \in G_{n,p}$. The graph consists of n vertices, where for each pair of vertices $v, w \in V$, the edge $(v, w) \in E$ is drawn with probability p, where p is constant, $0 < p < 1$. For such graphs the scheme of Alg. 4.4 yields only an $O(\frac{\sqrt{n}}{\log n})$ approximation ratio. In contrast, Alg. 4.8 is an $O(\log n / \log\log n)$ approximation scheme for random graphs, with high probability.

5 Separator Based Strategies for Broadcasting

This section examines the idea of using the separability property of a graph in order to achieve fast approximation schemes for broadcasting.

Definition 5.1 *Let $\varphi(n)$ be a nondecreasing function, and let y and ρ be fixed numbers such that $0 < \rho < 1$.*

1. *A graph $G = (V, E)$,* has a *$\langle \rho, y\rangle$ − separator if exists a set $S \subset V$ such that the removal of S leaves no connected component of size greater than $\rho \cdot n$, and $|S| \leq y$.*
2. *A graph $G = (V, E)$ is $\langle \rho, \varphi(n)\rangle$ − separable if every vertex-induced subgraph $G' \subset G$ of n' vertices has a $\langle \rho, \varphi(n')\rangle$ separator.*

Given a $\langle \rho, \varphi(n)\rangle$−separable graph, denote the corresponding separator of every subgraph G' by $sep(G')$.

5.1 Broadcasting Scheme for a Graph with Arbitrary Separator

In order to develop a separator-based broadcasting scheme, we first need to generalize Lemma 4.2. Suppose that a graph G contains a set V' of informed vertices. Denote the clusters created when extracting V' from the graph by $C_1, ..., C_k$. Choose for each C_i an arbitrary nonempty subset $C'_i \subset C_i$. We can use the fact that in broadcasting it is needed to inform the vertices of C'_i to achieve a lower bound on the best possible time for broadcasting. We use the technique developed for solving BES problems. Let us first define a BES instance $B' = BES(V', \{C'_1, \ldots, C'_k\})$ as follows.

1. Form the control graph $D_{V',G}$ of V' in G.
2. Put weights on the edges as follows. For an arbitrary vertex $v \in V'$ connected to a vertex in C_i, choose a vertex $v' \in C_i$ connected to v that is closest to the set C'_i. Attach a weight $d^v_{C_i} \equiv dist(v', C'_i)$, to the edge (v, C_i).

Lemma 5.2 *If F is a minimal control function for B', then $w(F) \leq b(V', G)$.*

Note, that we can use an algorithm similar to Alg. 3.5 to establish:

Fact 5.3 *In the above scenario, it is possible to inform at least one vertex in C'_i, for every i, in no more than $w(F)$ time units.*

It is possible to use Lemma 5.2 in order to construct schemes for broadcasting from a distinguished vertex v in a graph with a "small" separator. Let G be a $\langle \rho, \varphi(n)\rangle$-separable graph. Throughout the run, the set V' will denote the set of already informed vertices.

Algorithm 5.4

1. $V' \leftarrow \{v\}$.

2. Construct a separator $sep(G)$ for G.
3. Build an arbitrary tree $SPT(v, sep(G)) = (V_1, E_1)$ rooted at v and leading to the members of $sep(G)$. Broadcast the message to the vertices of the tree using the OT scheme.
4. $V' \leftarrow V' \cup V_1$.
5. **Repeat**
   (a) Assume the clusters formed when extracting V' from the graph are $C_1, ..., C_k$. Each C_i has a separator $sep(C_i) = C_1^i \cup C_2^i \cup \ldots C_{l_i}^i$, where $C_1^i, C_2^i, \ldots, C_{l_i}^i$ are $sep(C_i)$'s connected components.
   repeat
      i. For each i pick the lowest index $j(i)$ that wasn't chosen yet (for i).
      ii. Build the instance $B' = BES(V', \{C_{j(i)}^i : 1 \leq i \leq k\})$ of the BES problem, as described.
      iii. Send the message to at least one vertex of $C_{j(i)}^i$ for every i and j, using a minimal function F and the scheme suggested in Fact 5.3.

   until the C_j^i's clusters are exhausted for every i and j.
   (b) For every i and j, broadcast the message within C_j^i using the best known scheme for C_j^i. (If no good known scheme exists for the kind of graph C_j^i is, broadcast using an arbitrary tree.)
   (c) $V' \leftarrow V' \cup sep(C_1) \cup \ldots \cup sep(C_k)$.

   Until $V' = V$.

It is easy to see that when Alg. 5.4 terminates, all the vertices in V are informed.

Theorem 5.5 Alg 5.4 *terminates the broadcast process in* $O(\log n) \cdot \varphi(n) \cdot b(v)$ *time units.*

Further, it can be shown that if we can assure that every separator $sep(C)$ is connected and $b(sep(C)) \leq k$ for some integer $k < \varphi(n)$, then the bound is improved to

$$O(\log n) \cdot (b(v) + k). \tag{1}$$

5.2 Applications

In this subsection we give some examples for graph families for which Algorithm 5.4 can be applied. The first example is the one of *chordal graphs*. The following theorem is shown in [GRE84] regarding chordal graphs.

Theorem 5.6 [GRE84] *Any n-vertex chordal graph G contains a (polynomialy computable) maximal clique C such that if the vertices in C are deleted from G, every connected component in the graph induced by any of the remaining vertices is of size at most n/2.*

From Theorem 5.6 and Eq. (1) it follows that:

Corollary 5.7 *There exists a polynomial* $2 \cdot \log n + 1$ *approximation scheme for broadcasting on chordal graphs.* ∎

A second example is the family of a c-separable graphs, consisting of graphs for which $\varphi(n) = c$ for some constant c. It follows from Theorem 5.5 that Alg. 5.4 is an $O(\log n)$ approximation scheme for broadcasting in such graphs. Note that *k-outerplanar* graphs are $O(k)$-separable, and that *outerplanar* graphs are $O(1)$-separable. Thus, as a specific cases we haves:

Theorem 5.8 *There exist an* $O(k \cdot \log n)$ *approximation algorithm for broadcasting in a* k*-outerplanar graph, for a fixed* k.

Corollary 5.9 *There exists a polynomial* $O(\log n)$ *approximation scheme for broadcasting on the family of outerplanar graphs.*

The third example is the well known family of series-parallel graphs. In [FJ90] it is shown that every series-parallel graph is $\langle 2/3, 2\rangle$-separable and the separator can be found in $O(n)$ time. Thus Alg. 5.4 is a polynomial $O(\log n)$ approximation algorithm for broadcasting on a series parallel graph.

Theorem 5.10 *There exists a polynomial* $O(\log n)$ *approximation algorithm for broadcasting on a series-parallel graph.*

The last example is of the family of *bounded face planar graphs.* The size of a face of a planar graph is the number of vertices in the face, counting multiple visits when traversing the boundary (cf. [Mil86]).

Theorem 5.11 [Mil86] *If* G *is an embedded planar graph with bounded face size then the graph is* $\langle 2/3, O(\sqrt{n})\rangle$*-separable, and the separator can be chosen to be a simple cycle, or a single vertex.*

Theorem 5.12 *There exists a polynomial* $O(n^{1/4}/\sqrt{\log n})$ *approximation algorithm for broadcasting in the family of bounded face planar graphs.*

6 Broadcasting in a Tree of Cliques

In this section we switch from general graphs to the other extreme, and give an approximation scheme for a special kind of graph family called *trees of cliques*, generalizing the family of trees.

Definition 6.1 *A graph* $G = (V, E)$ *is a* tree of cliques *(TOC) if*

1. *The vertex set* V *can be decomposed into a disjoint union of sets* $C_1, \ldots, C_k$ *such that each* C_i *induces a clique (i.e., a complete graph) in* G.
2. *The auxiliary graph* $T(G) = (\tilde{V}, \tilde{E})$ *whose vertices are* $\tilde{V} = \{C_1, \ldots, C_k\}$ *and whose edges are* $\tilde{E} = \{(C_i, C_j) \mid$ *there is an edge* $(v_i, v_j) \in E$, *for* $v_i \in C_i, v_j \in C_j\}$ *is a tree.*

To broadcast a message from a vertex v in a TOC, we use the following idea. In order to deliver the message between vertices of different cliques (i.e., from cliques to their clique children), we use the techniques developed for BVS problems. It follows that the total broadcast complexity spent while delivering message between cliques is bounded by $O(b(v))$. Since there is an efficient delivery method for message delivery in a clique, we have.

Theorem 6.2 *There is an* additive $\log n(\log n - \log\log n)$ *approximation algorithm for broadcasting in a* TOC. ∎

We then develop an alternative method for delivering the message inside the cliques. In this method, every vertex participates in the message delivery in its clique only for a small (fixed) number of rounds, and is thus free sooner to help in sending the message down the tree to its clique children. Using this method we establish (in the full paper) some improved bounds in restricted cases.

References

[Far80] A.M. Farley. Minimum time line broadcast network. *Networks*, 10:59–70, 1980.

[FH78] A. Farley and S. Hedetniemi. Broadcasting in grid graphs. *In Proc. Ninth Southeastern Conf. on Combin., Graph Theory and Comput.*, pages 275–288, 1978.

[FJ90] G. Frederickson and R. Janardan. Space-efficient message routing in c-decomposable networks. *SIAM J. on Computing*, 19:164–181, 1990.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H Freeman and Company, 1979.

[GRE84] J.R. Gilbert, D.J. Rose, and A. Edenbrandt. A separator theorem for chordal graphs. *SIAM J. Alg. and Disc. Meth.*, 5:306–313, 1984.

[HHL88] S. Hedetniemi, S. Hedetniemi, and A. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18:319–349, 1988.

[Mil86] G. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and System Sciences*, 32:265–279, 1986.

[SCH81] P.J. Slater, E.J. Cockayne, and T. Hedetniemi. Information dissemination in trees. *SIAM J. on Compu.*, 10, 1981.