algorithms

MDPI

Article

Broadcasting in Stars of Cliques and Path-Connected Cliques †

Akash Ambashankar * and Hovhannes A. Harutyunyan *

Department of Computer Science and Software Engineering, Concordia University, Montreal, QC H3G 1M8, Canada

* Correspondence: akash.ambashankar@mail.concordia.ca (A.A.); haruty@cs.concordia.ca (H.A.H.)

† This article is a revised and expanded version of a paper presented at IWOCA 2024: Ambashankar, A.; Harutyunyan, H.A. Broadcasting in Stars of Cliques. In Proceedings of the International Workshop on Combinatorial Algorithms. Springer, 2024, pp. 485–496.

Abstract: Broadcasting is a fundamental information dissemination problem in a connected network where one node, referred to as the originator, must distribute a message to all other nodes through a series of calls along the network's links. Once informed, nodes assist the originator by forwarding the message to their neighbors. Determining the broadcast time for a node in an arbitrary network is NP-complete. While polynomial-time algorithms exist for specific network topologies, the problem remains open for many others. In this paper, we focus on addressing the broadcasting problem in network topologies represented by specialized clique-based structures. Specifically, we investigate the windmill graph $Wd_{k,l}$, which consists of k cliques of size l connected to a universal node, and extend our study to the star of cliques, a generalization of the windmill graph with cliques of arbitrary sizes. Our primary objective is to propose an efficient algorithm for determining the broadcast time of any node in an arbitrary star of cliques and to rigorously prove its optimality. Additionally, we broaden the scope by examining the broadcasting problem in path-connected cliques, a topology featuring k cliques of varying sizes sequentially connected along a path. For this structure, we develop a computationally efficient algorithm that leverages clique sizes and adjacency to optimize broadcast strategies, with broader implications for understanding communication in block graphs.

Keywords: broadcasting; algorithm; star of cliques; path-connected cliques; windmill graphs

Academic Editor: Adele Anna Rescigno

Received: 28 November 2024
Revised: 20 January 2025
Accepted: 22 January 2025
Published: 1 February 2025

Citation: Ambashankar, A.; Harutyunyan, H.A. Broadcasting in Stars of Cliques and Path-Connected Cliques. *Algorithms* **2025**, *18*, 76. https://doi.org/10.3390/a18020076

1. Introduction

Rapid advancements in technology, particularly the Internet and the proliferation of parallel and distributed computing, have revolutionized global communication by enabling seamless and instantaneous data exchange over vast distances. As networks become increasingly complex, the need for efficient, secure, and scalable communication protocols has grown critical, driving research into the design of optimal network architectures to meet the evolving demands of parallel and distributed systems.

This paper focuses on the field of broadcasting, which was originally introduced as a variant of the gossiping problem [1]. Broadcasting involves disseminating a message throughout a network, where informed nodes communicate the message to their uninformed neighbors through calls that occur within discrete time units. The problem is typically studied in the context of connected networks, with messages transmitted along the communication lines of the network. The process begins with a single informed node, referred to as the originator, which initiates the dissemination by informing other nodes. The broadcasting concludes when all nodes in the network have received the message.

During each time unit, newly informed nodes participate in the broadcasting process by informing their uninformed neighbors. The broadcasting process operates under the following constraints: (1) Time is divided into discrete units. (2) Within each time unit, a node can participate in at most one call. (3) Each call has a fixed duration of one time unit and occurs between two neighboring nodes. (4) Multiple calls can occur simultaneously during a single time unit, provided they involve distinct pairs of neighboring nodes.

A connected undirected graph $G = (V, E)$ is used to depict the network, where V represents the set of nodes (vertices) and E represents the communication lines (edges) connecting the nodes of the network. The originator vertex is typically labelled as u. The minimum time required to complete broadcasting in G from originator u is referred to as the broadcast time from originator u and is depicted by $b(u, G)$. Assuming each informed vertex informs one of its uninformed neighbours during a given time unit, the number of informed vertices can, at most, double in each step. This observation establishes the following lower bound on the broadcast time from a given originator vertex u: $b(u, G) \geq \lceil \log n \rceil$ where $n = |V|$. Note that all logarithms referenced in this paper are base 2. During each time unit, at least one new vertex receives the message. This observation supports the following upper bound on the broadcast time: $b(u, G) \leq n - 1$. The broadcast time of a graph is defined as the maximum broadcast time across all vertices in the graph: $b(G) = \max_{\forall u \in V} \{b(u, G)\}$. Note that another lower bound on $b(G)$ is the diameter of the graph, defined as the length of the shortest path between the most distanced nodes. The set of calls used to disseminate the message from a given originator vertex u to all other vertices of the graph is referred to as the *broadcast scheme* for vertex u. The broadcast scheme corresponds to the *broadcast tree*, which is a spanning tree rooted at the originator.

Note that the above upper and two lower bounds are tight for some specific graphs, while this is not the case in general. For example, in the binary hypercube, both lower bounds are tight. For the star graph, which has diameter 2, the upper bound is tight but both lower bounds are much less so.

Computing $b(u, G)$ and $b(G)$ for arbitrary graphs, given arbitrary originators, is an NP-complete problem [2]. This complexity persists even within restricted graph families, including bounded degree graphs [3] and 3-regular planar graphs [4,5]. Approximating the broadcast time problem within a factor of $\frac{57}{56} - \epsilon$ has been proven to be NP-Hard [6]. This result was subsequently improved, showing that the problem remains NP-Hard to approximate within a factor of $3 - \epsilon$. Additionally, an approximation algorithm was introduced, capable of building a broadcast scheme in $O\left(\frac{\log(|V|)}{\log\log(|V|)} b(G)\right)$ rounds [7].

The study of exact algorithms for specific graph families was initiated through the development of a linear-time algorithms for trees [2]. This was later extended to include exact algorithms for grids, tori [8], cube-connected cycles [9], and the shuffle exchange graph [10]. Further advancements addressed more complex graph families, such as fully connected trees [11], k-cacti with constant k [12], Harary-like graphs [13], and Unicyclic graphs [14]. For a comprehensive overview of broadcasting and related problems, we direct the reader to the survey papers [1,10,15,16] as well as more recent papers [12,17–27].

This paper is a study of the broadcasting problem in graph families consisting of blocks which are cliques. We start with the windmill graph $Wd_{k,l}$, which is a graph with n vertices consisting of k cliques of size l and a universal vertex. This paper also introduces star of cliques, a generalization of the windmill graph where cliques can have arbitrary sizes. In addition, we analyze the topology of path-connected cliques, which consists of a sequence of cliques arranged along a path, where each pair of adjacent cliques is connected by a single edge.

The windmill graph $Wd_{k,2}$, also known as the Friendship graph, serves as a solution to Erdős et al.'s Friendship Theorem [28]. The family of graphs has also been studied

in the context of graceful and harmonious graph labeling, as well as in the movable antennae problem [29], with particular attention given to variations such as the Dutch windmill and the French windmill. We propose $O(1)$ algorithms for windmill graphs and path-connected cliques, and demonstrate how the latter serves as a model for optimizing message dissemination in topologies, such as block graphs, where connectivity is path-dependent. Further, we propose an $O(n \cdot \log n)$ algorithm to compute the broadcast time of any vertex in an arbitrary star of cliques and analyze the optimality of our approach.

We skip all proofs in Sections 2–4 since they are discussed in our conference paper [30]. However, we present the details of the algorithms for completeness.

2. Broadcasting in Windmill Graphs

We define $Wd_{k,l}$ to be a windmill graph containing n vertices distributed across k cliques $C_1, C_2, \ldots, C_k$, such that each clique has l vertices. Furthermore, Wd_{k_l} includes a universal vertex u, which is directly linked to all other vertices in the graph.

It can be observed that the broadcast time remains the same, regardless of the choice of the originator vertex, whether it is the universal vertex or a clique vertex. In both cases, after the first time unit, the only informed vertices will be the universal vertex and one clique vertex. Moreover, the broadcast time is unaffected by which specific clique vertex is informed first, as all cliques contain the same number of vertices. Therefore, for simplicity, we assume the universal vertex as the originator throughout this section.

Lemma 1. *Let $Wd_{k,l}$ be a windmill graph on n vertices containing k cliques $C_1, C_2, \ldots, C_k$ each of size l, and a universal vertex u. There exists an optimal broadcast scheme, wherein after informing one vertex in each of the $k-1$ cliques, the universal vertex will broadcast only to the last clique starting at time unit k.*

Algorithm 1 illustrates the procedure of broadcasting in a windmill graph from universal vertex u. Based on Lemma 1, broadcasting can be summarized as the universal vertex informing one vertex in each clique $C_1 \ldots C_{k-1}$. This phase lasts for $k-1$ rounds, following which the universal vertex broadcasts exclusively to clique C_k and the previously informed vertices contribute to the broadcasting process within their respective cliques. The latter phase requires another $\lceil \log(l+1) \rceil$ time units, leading to the following result for the overall broadcast time of the windmill graph:

$$b(G) = k - 1 + \lceil \log(l+1) \rceil$$

Algorithm 1 Broadcast algorithm for windmill graph from universal vertex u

Input Windmill graph $Wd_{k,l}$, originator vertex u.
Output A broadcast scheme for $Wd_{k,l}$.

1: **for** $i = 1 \rightarrow k$ **do**
2: u informs a vertex v_i in clique C_i
3: v_i broadcasts in C_i
4: **end for**
5: **while** C_k has an uninformed vertex **do**
6: u broadcasts in C_k
7: **end while**

3. Bounds on Broadcast Time of Star of Cliques

We define star of cliques G_k as a connected graph comprising n vertices distributed across k cliques $C_1, C_2, \ldots, C_k$, where each clique C_i contains l_i vertices for all $1 \leq i \leq k$. The sizes of the cliques are ordered such that $l_1 \geq l_2 \geq \cdots \geq l_k \geq 1$. Furthermore, the graph G_k includes a universal vertex u, which is directly connected to all other vertices in the

graph. Figure 1 illustrates an example of a star of cliques consisting of three cliques with sizes 10, 9, and 5.

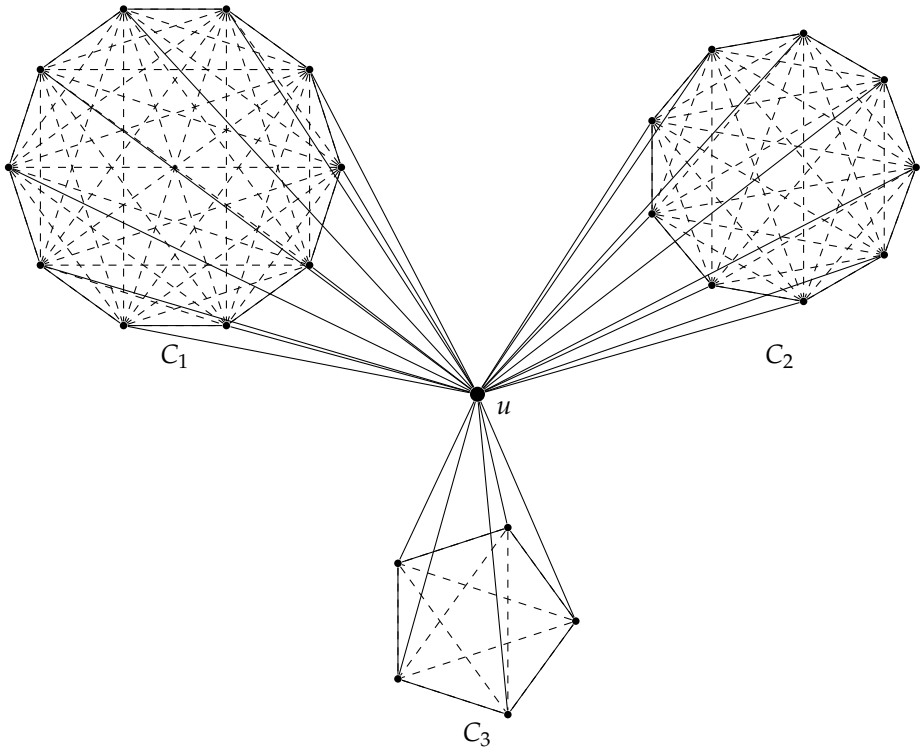

Figure 1. Star of cliques consisting of 3 cliques with sizes 10, 9, and 5.

Lemma 2. *Let G_k be a star of cliques and the universal vertex u be the originator. Then, $b(u, G_k) \geq \max_{1 \leq i \leq k} \{i - 1 + \lceil \log (l_i + 1) \rceil\}$.*

Lemma 3. *Let G_k be a star of cliques and the universal vertex u be the originator. Then, $b(u, G_k) \leq \max_{1 \leq i \leq k} \{i + \lceil \log l_i \rceil\}$.*

Lemma 4. *Let G_k be a star of cliques where the originator is the universal vertex u. Then, the difference between the upper and lower bounds of $b(u, G_k)$ is at most 1.*

Remark 1. *From Lemma 4, we know $b(u, G_k)$ has two possible values. Thus, Algorithm 2 is a 1-additive approximation algorithm.*

Algorithm 2 Broadcast algorithm for star of cliques from the universal vertex u

Input Star of cliques G_k ($l_1 \geq l_2 \geq \cdots \geq l_k \geq 1$).
Output A broadcast scheme for G_k with time $\max_{1 \leq i \leq k} \{i + \lceil \log l_i \rceil\}$.

1: **for** $i = 1 \rightarrow k$ **do**
2: u informs a vertex v_i in clique C_i
3: v_i broadcasts in C_i
4: **end for**

4. Decision Algorithm for the Broadcast Time of Star of Cliques

This section introduces a decision algorithm for stars of cliques, which employs a greedy approach to determine whether broadcasting from the universal vertex can be completed within the upper bound established in Lemma 3. Specifically, the algorithm aims to identify a broadcast scheme that completes in at most $t = t' - 1$ time units, where

$t' = \max_{1 \le i \le k} \{i + \lceil \log l_i \rceil\}$ represents the upper bound on the broadcast time. Subsequently, we show how to compute the broadcast time of a stars of cliques graph originating from an arbitrary vertex using the decision algorithm.

Define $CS[1 \dots k]$ as an array of binary strings, with each string representing a clique in G_k. For all $1 \le i \le k$, the base-10 value of the binary string $CS[i]$ corresponds to l_i (i.e., the number of vertices in clique C_i). Note that the cliques are sorted in descending order such that $l_1 \ge l_2 \ge \cdots \ge l_k$. Each binary string $CS[i]$ contains t bits, with shorter strings left-padded with zeros. Each bit $CS[i][j]$ ($1 \le j \le t$) specifies the action to be performed by the universal vertex at time unit j: if $CS[i][j] = 1$, u must inform a vertex in C_i; if $CS[i][j] = 0$, no action is needed.

These binary strings can be used to construct an individual broadcast scheme from the universal vertex for their corresponding cliques, leveraging the following property: for any such binary string $CS[i]$, if a bit at index j is 1 (i.e., $CS[i][j] = 1$), broadcasting to 2^{t-j} vertices can be completed by time unit t without any further intervention from the universal vertex. This process can only be initiated if the universal vertex informs a new vertex in clique C_i at time unit j. This property holds because any integer can be represented as a sum of distinct powers of 2. Specifically, $l_i = 2^{p_1} + 2^{p_2} + \ldots + 2^{p_c}$, where l_i denotes the number of vertices in C_i and $p_1, \ldots, p_c$ are integers. Hence, if the universal vertex can broadcast to a new vertex in C_i at each time unit j where $CS[i][j] = 1$, C_i will be fully informed no later than time unit t.

It is noteworthy that if u informs a vertex in C_i at a time unit where $CS[i][j] = 0$, then C_i does not require any further action, even if subsequent bits in $CS[i]$ are set to 1. This is because $2^{t-j} > 2^{t-j-1} + 2^{t-j-2} + \ldots + 2^1 + 2^0$ (e.g., $1000 > 0111$).

4.1. Broadcasting from the Universal Vertex u

Algorithm 3, StarOfCliquesBroadcast(CS, t), determines whether broadcasting in a given star of cliques from the universal vertex u can be completed within t time units. The algorithm returns *True* if the broadcast is feasible within the given time frame and *False* otherwise.

Definition 1. ***Unaddressed Vertices****: When a vertex in some clique C_p is informed at time j, this vertex becomes the root of a binomial tree B_{t-j} within C_p. The vertices in B_{t-j} are guaranteed to be informed at or before time t. All uninformed vertices in C_p that do not belong to such a binomial tree are called unaddressed vertices.*

In the algorithm StarOfCliquesBroadcast(CS, t), the first step is to check whether vertex u needs to inform an unaddressed vertex in C_1 during the current time unit, i.e., if $CS[1][1] = 1$. If this condition holds, it implies that $l_1 \ge 2^{t-j}$. In this case, u cannot inform any other clique in the current time unit, as doing so would prevent C_1 from completing its broadcast. This is because $l_1 \ge 2^{t-j} > 2^{t-j+1} + 2^{t-j+2} + \ldots + 2^1 + 2^0$, and the deficit in vertices cannot be compensated. However, if $CS[2][1] = 1$ simultaneously, a conflict arises, and the algorithm returns *False*. It is important to note that it is sufficient to check only $CS[2]$ for conflicts, as the array is sorted in descending order, and all subsequent cliques $C_3 \dots C_k$ are no larger than C_2.

On the other hand, if $CS[1][1] \neq 1$, this indicates that no clique requires broadcasting during the current time unit. Rather than leaving u idle, the algorithm opts to broadcast to C_1 anyway, ensuring that C_1 no longer needs to be informed in the future, which may help avoid conflicts in subsequent iterations. In either scenario, the algorithm proceeds by informing an unaddressed vertex in C_1.

Algorithm 3 Decision algorithm for broadcasting in star of cliques from the universal vertex u in t time units. (*Note:* Arrays are indexed starting from 1)

Input $CS[1 \dots k]$: Cliques in G_k as descending sorted binary strings, t: target broadcast time.
Output *True* if $b(u, G_k) = t$; *False* otherwise.

```
1:  procedure StarOfCliquesBroadcast(CS, t)
2:      if CS[1][1] == "1" then
3:          if CS.length > 1 and CS[2][1] == "1" then
4:              return False
5:          end if
6:      else
7:          s ← 1 + 0...0   (t − 1 zeros)
8:      end if
9:      if t == 1 then
10:         return True
11:     else
12:         for i ← 1 to k do
13:             CS[i] ← substring(CS[i], 2, t)
14:         end for
15:         CS.sort()
16:         return StarOfCliquesBroadcast(CS, t − 1)
17:     end if
18: end procedure
```

The next step involves recursively calling StarOfCliquesBroadcast(CS, $t-1$) for the next iteration. Here, $CS[i]$ is updated to $CS[i][2 \dots t]$, meaning the first bit, which corresponds to the now-elapsed time unit, is removed from each binary string. The algorithm continues this process until one of two conditions is met: either two cliques require broadcasting in the same time unit, resulting in the algorithm returning *False*, or the final time unit is reached, in which case the algorithm returns *True*.

Complexity Analysis

The StarOfCliquesBroadcast algorithm performs several operations whose complexities contribute to the overall runtime. Step 7 takes $O(t)$ time to update a string if the corresponding clique is informed early. In Step 13, deleting the first bit of the k strings incurs $O(k \cdot t)$ time. Since only the first element of the array is out of place in Step 15, the sorted array can be obtained through a single iteration of insertion sort in $O(k)$ time. Thus, a single iteration of the algorithm requires $O(k \cdot t)$ time.

The algorithm is executed t times, leading to a cumulative complexity of $O(k \cdot t^2)$ time. In addition, computing the upper bound t' requires $O(k \cdot \log k)$ time, while generating the binary strings for all cliques takes $O(k \cdot \log l_1)$ time.

The maximum possible value of t is $k + \lceil \log l_1 \rceil$, and since $k, l_1 \leq n$, where $n = l_1 + l_2 + \ldots + l_k$ represents the total number of vertices in the graph, it follows that $t, k = O(n)$. Consequently, the overall time complexity of the algorithm is $O(n^3)$.

4.2. Algorithm Correctness

Lemma 5 (Greedy Choice). *Let G_k be a star of cliques where the originator is the universal vertex u and $l_1 \geq l_2 \geq \cdots \geq l_k \geq 1$, where l_i is the number of vertices in clique C_i (excluding vertex u) for all $1 \leq i \leq k$. Then, there is an optimal broadcast scheme where the largest clique C_1 is informed in the first time unit.*

Lemma 6 (Optimal Substructure). *Let G_k be a star of cliques where the originator is the universal vertex u and the set $S_j = \{C_1^j, \ldots, C_k^j\}$ be the unaddressed vertices in cliques $C_1, \ldots, C_k$,*

respectively, at time unit j. Then, an ordering of S_j, which is an optimal broadcast scheme from the universal vertex u, must also include an optimal broadcast scheme for S_{j+1}.

Consider the star of cliques illustrated in Figure 1, which consists of three cliques with sizes 10, 9, and 5, respectively, and a universal vertex denoted by u. Based on Lemmata 2 and 3, the broadcast time $b(u, G_k)$ has a lower bound of five time units and an upper bound of six time units. By applying the procedure described in Algorithm 3, it is possible to design a broadcast scheme that achieves the lower bound. This can be carried out by prioritizing the early notification of C_1, which allows the universal vertex to allocate additional broadcasts to C_2 and C_3. Specifically, vertex u broadcasts to C_1 at time 1, to C_2 at times 2 and 4, and to C_3 at times 3 and 5, ensuring that all vertices in the graph are informed within the lower bound of the five time units.

4.3. Bounds on Broadcast Time from an Arbitrary Originator

Consider G_k as a star of cliques with a universal vertex u, and let v be an arbitrary originator such that $v \neq u$ and $v \in C_i$, where $1 \leq i \leq k$. Consider the broadcast scheme for an arbitrary originator presented in Algorithm 4. In an optimal broadcast scheme, v will transmit the message to u at time unit 1, as u is directly connected to every vertex in G_k. If $v \in C_1$, then $b(v, G_k) = b(u, G_k)$, because, after the first time unit, the two informed vertices in G_k are v and u. Similarly, if $C_i \neq C_1$ but $l_i = l_1$, it follows that $b(v, G_k) = b(u, G_k)$ since the broadcast scheme can interchange the roles of C_i and C_1.

For all other cases, the following bounds apply.

Lemma 7. *Let G_k be a star of cliques where the originator $v \in C_i$, for $1 < i \leq k$ and $|C_i| < |C_1|$. Then, $b(v, G_k) \geq b(u, G_k \setminus C_i) + 1$.*

Lemma 8. *Let G_k be a star of cliques where the originator $v \in C_i$, for $1 < i \leq k$ and $|C_i| < |C_1|$. Then, $b(v, G_k) \leq b(u, G_k \setminus C_i) + 1$.*

Algorithm 4 Broadcast algorithm for star of cliques from an arbitrary originator v

Input Star of cliques G_k, originator vertex v
Output A broadcast scheme for G_k from originator v with time $b(v, G_k) \leq b(u, G_k \setminus C_i) + 1$ where u is the universal vertex.

1: v informs u at time 1
2: u broadcasts to $G_k \setminus C_i$ according to Algorithm 3
3: v broadcasts to C_i

Remark 2. *From Lemmata 7 and 8, we can see that when $v \neq u$ and $v \in C_i$, $b(v, G_k) = b(u, G_k \setminus C_i) + 1$ for all $1 < i \leq k$ if $l_1 > l_i$. For all other cases, $b(v, G_k) = b(u, G_k)$.*

4.4. Improved Decision Algorithm for the Broadcast Time of Star of Cliques

Algorithm 3 can compute $b(u, G_k)$ for an arbitrary star of cliques in $O(n^3)$ time. However, in Algorithm 5, we present a more efficient approach with a time complexity of $O(n \cdot \log n)$ for calculating $b(u, G_k)$ in an arbitrary star of cliques.

The algorithm utilizes base 10 digits, which are stored in a max heap. In each iteration, we extract the two largest cliques from the top of the heap. At each time unit j, where $1 \leq j \leq t$, the algorithm can inform 2^{t-j} vertices. If any clique, other than the first, requires at least 2^{t-j} vertices in the current time unit, the algorithm returns False. Otherwise, 2^{t-j} vertices from the first clique are informed. The first element in the heap is then updated to $l_1 - 2^{t-j}$. This process is repeated for t iterations.

Algorithm 5 Heap-based decision algorithm for broadcasting in star of cliques from the universal vertex u in t time units

Input *cliqueHeap*: G_k as a max heap of clique sizes, t: target broadcast time, j: current time unit $(1 \leq j \leq t)$.
Output *True* if $b(u, G_k) = t$; *False* otherwise.

```
1:  procedure StarOfCliquesHeapBroadcast(cliqueHeap, t, j)
2:      l1 ← cliqueHeap.extractMax()
3:      l2 ← cliqueHeap.peek()
4:      if l1 ≥ 2^(t−j) then
5:          if l2 ≥ 2^(t−j) then
6:              return False
7:          else
8:              cliqueHeap.insert(l1 − 2^(t−j))
9:          end if
10:     else
11:         cliqueHeap.insert(0)
12:     end if
13:     if j + 1 ≤ t then:
14:         return StarOfCliquesHeapBroadcast(cliqueHeap, t, j + 1)
15:     else
16:         return True
17:     end if
18: end procedure
```

Complexity Analysis

The algorithm requires $O(\log k)$ time to perform one iteration of all heap operations in Steps 2, 8, and 11, where k denotes the number of elements in the heap (i.e., the number of cliques in the graph). Consequently, performing t iterations, where each iteration corresponds to a time unit, incurs a total time complexity of $O(t \cdot \log k)$. Additionally, the preliminary operations of sorting the input and calculating the upper bound t' require $O(k \cdot \log k)$ time, while initializing the heap incurs $O(k)$ time.

It is evident that $t, k = O(n)$ as t is bounded by $k + \lceil \log l_1 \rceil$, and $k, l_1 \leq n$, where $n = l_1 + l_2 + \ldots + l_k$ denotes the total number of vertices in the graph. Therefore, the overall time complexity of the algorithm is $O(n \cdot \log n)$.

5. Broadcasting in Path-Connected Cliques

In this section, we study the broadcasting problem in path-connected cliques, a graph defined on n vertices and containing k cliques of arbitrary sizes $l_1, l_2, \ldots, l_k$, with neighboring cliques being connected by an edge. Path-connected cliques can be classified as block graphs since the edge connecting two neighboring cliques is also a clique K_2.

Consider a graph G_k consisting of a path P with vertices $u_1, u_2, \ldots, u_k$ (in the path order), and k pairwise vertex disjoint arbitrarily-sized cliques $C_1, C_2, \ldots, C_k$ which are attached to P with the following properties:

- $v_i \in V(C_i)$ for all $1 \leq i \leq k$;
- For all $1 \leq i \leq k$, $1 \leq j \leq k$, $i = j$, an arbitrary vertex $v \in C_i$, and an arbitrary vertex $w \in C_j$, $(v, w) \notin E(G_k)$.

We may denote such a graph as $G_k = (P, C_1, C_2, \ldots, C_k)$. For all $1 \leq i \leq k-1$ We define $t_i = \lceil \log l_i \rceil$ as the time units required for clique C_i to complete broadcasting without needing to inform C_{i+1}. We also define a_i to be the last possible time unit at which $u_i \in V(C_i)$ informs $u_{i+1} \in V(C_{i+1})$ such that the overall broadcasting is completed in minimum time. Figure 2 illustrates the general structure of path-connected cliques.

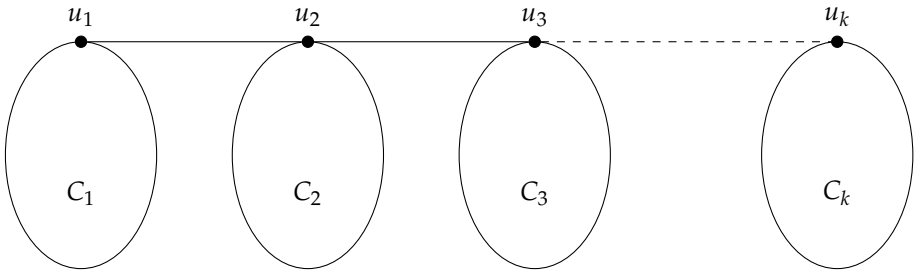

Figure 2. General Structure of the Path-Connected Cliques graph.

Given a graph $G = (P, G_1, G_2, \ldots, G_k)$, where $G_1, G_2, \ldots, G_k$ can be graphs with any structure and $P = \{u_1, u_2, \ldots, u_k\}$ is a path, a fixed-deadline message dissemination algorithm for general path-connected graphs has been presented [31]. In this work, we present an $O(1)$ method to calculate the broadcast time for an arbitrary path-connected cliques graph.

5.1. Broadcasting in Path-Connected Cliques When $k = 2$

Lemma 9. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. If $k = 2$, then $b(u_1, G_2) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, a_1 + \lceil \log l_2 \rceil\}$.*

Proof. In any broadcast scheme for originator u_1 in G_2, u_1 must inform u_2 at some time unit a_1 to ensure broadcasting is completed in both cliques. In an optimal scheme, u_1 waits until the very last moment to inform u_2 in such a way that the broadcasting process in C_2 completes either simultaneously with or before the broadcasting process in C_1. Additionally, u_1 tries to minimize the number of vertices in C_1 that it does not inform because it is busy informing u_2. Once u_2 is informed, C_2 can finish broadcasting in $\lceil \log l_2 \rceil$ time units. Thus, all vertices in C_2 will be informed by time unit $a_1 + \lceil \log l_2 \rceil$.

Let $t_1 = \lceil \log l_1 \rceil$ be the time unit at which broadcasting in C_1 would finish if u_1 does not inform u_2 at time unit a_1. If $2^{t_1} - l_1 < 2^{t_1 - a_1}$, then C_1 may require an extra time unit to complete broadcasting; since vertex u_1 informs u_2 instead of a vertex in C_1 at time unit a_1, $2^{t_1 - a_1}$ vertices could remain uninformed at time unit t_1. In any case, all vertices in C_1 will be informed by time unit $\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil$. Hence the broadcast time of G_2 from originator u_1 is given by $b(u_1, G_2) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, a_1 + \lceil \log l_2 \rceil\}$. A similar result can be shown for broadcasting from vertex u_2 in G_2. □

5.2. Broadcasting in Path-Connected Cliques from Path Vertices

Vertices in the path-connected cliques graph can be categorized as *path vertices* and *clique vertices*. Within path vertices, vertices u_1 and u_k are collectively referred to as *path-end vertices*, while all other path vertices are called *internal path vertices*.

Lemma 10. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. Let $H_i = (P_i, C_i, C_{i+1}, \ldots, C_k)$ be a subgraph of G_k where P_i consists of vertices $u_i, u_{i+1}, \ldots, u_k$ for all $1 \leq i \leq k$. If vertex u_1 is the originator, then $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$.*

Proof. In any broadcast scheme for originator u_i in subgraph H_i, u_i must inform u_{i+1} at some time unit a_i to ensure broadcasting is completed in clique C_i as well as subgraph H_{i+1}. In an optimal scheme, u_i waits until the very last moment to inform u_{i+1} in such a way that the broadcasting process in H_i completes either simultaneously with or before the broadcasting process in C_i. Additionally, u_i tries to minimize the number of vertices in C_i that it does not inform because it is busy informing u_{i+1}.

Consider the subgraph H_{k-2}. We know that vertex u_{k-2} informs u_{k-1} at time unit a_{k-2}. From Lemma 9, we also know that $b(u_{k-1}, H_{k-1}) = \max\{\lceil \log(l_{k-1} + 2^{t_{k-1}-a_{k-1}}) \rceil, a_{k-1} + \lceil \log l_k \rceil\}$. Once u_{k-1} is informed, broadcasting in H_{k-1} finishes in $b(u_{k-1}, H_{k-1})$ time units.

As seen in Lemma 9, all vertices in C_{k-2} will be informed by time unit $\lceil \log(l_{k-2} + 2^{t_{k-2}-a_{k-2}}) \rceil$. Hence, the broadcast time of H_{k-2} from originator u_{k-2} is given by $b(u_{k-2}, H_{k-2}) = \max\{\lceil \log(l_i + 2^{t_i-a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. In general, for all $1 \leq i \leq k$ the broadcast time of any subgraph H_i from originator u_i is given by $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i-a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. A similar result can be obtained when the originator is vertex u_k. □

Remark 3. *From Lemma 10 we know that* $b(u_1, H_1) = \max\{\lceil \log(l_1 + 2^{t_1-a_1}) \rceil, a_1 + b(u_2, H_2)\}$. *Also, it is evident that* $b(u_1, G_k) = b(u_1, H_1)$. *Hence,* $b(u_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1-a_1}) \rceil, a_1 + b(u_2, H_2)\}$ *where* $b(u_k, H_k) = \lceil \log l_k \rceil$. *Alternatively,* $b(u_1, G_k)$ *is also given by*

$$b(u_1, G_k) = \begin{cases} 1 + b(u_2, H_2) & \text{if } b(u_2, H_2) \geq \lceil \log l_1 \rceil \\ \lceil \log l_1 \rceil & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 \geq 2^{b(u_2, H_2)} \\ \lceil \log l_1 \rceil + 1 & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 < 2^{b(u_2, H_2)} \end{cases}$$

Lemma 11. *Let* $G_k = (P, C_1, C_2, \ldots, C_k)$ *be a graph of path-connected cliques. Then, for all* $2 \leq i \leq k-1$, $\max\{b(u_1, G_k), b(u_k, G_k)\} \geq b(u_i, G_k)$

Proof. For an originator vertex u_i, let H_l be the subgraph of G_k with all cliques to the left of C_i. Then, $H_l = (P_l, C_{i-1}, C_{i-2}, \ldots, C_1)$, where P_l contains vertices $u_{i-1}, u_{i-2}, \ldots, u_1$. Similarly, let H_r be the subgraph with all cliques to the right of C_i. Then, $H_r = (P_r, C_{i+1}, C_{i+2}, \ldots, C_k)$, where P_r contains vertices $u_{i+1}, u_{i+2}, \ldots, u_k$. Without loss of generality, assume $b(u_{i-1}, H_l) \geq b(u_{i+1}, H_r)$. Consider the following broadcast scheme:

1. Originator u_i informs vertex u_{i+1} at time 1.
2. Vertices u_i and u_{i+1} broadcast separately within $G_k \setminus H_r$ and H_r, respectively.

Since $b(u_{i-1}, H_l) \geq b(u_{i+1}, H_r)$, we have $b(u_i, G_k \setminus H_r) \geq b(u_{i+1}, H_r)$. Thus, $b(u_i, G_k) \leq 1 + b(u_i, G_k \setminus H_r)$

Consider the following broadcast scheme from originator u_k, which begins by informing path vertices $u_{k-1}, u_{k-2}, \ldots, u_i$ before informing any clique vertices. Once vertex u_i is informed it begins informing the subgraph $G_k \setminus H_r$. Then, $b(u_k, G_k) \geq k - i + b(u_i, G_k \setminus H_r)$ since $b(u_k, G_k) \geq b(u_k, G_k \setminus H_r) \geq$ minimum time required to inform u_i from $u_k + b(u_i, G_k \setminus H_r)$. By assumption, $k - i \geq 1$. Hence,

$$\begin{aligned} b(u_k, G_k) &\geq k - i + b(u_i, G_k \setminus H_r) \\ &\geq 1 + b(u_i, G_k \setminus H_r) \\ &\geq b(u_i, G_k) \end{aligned}$$

The same proof applies when $b(u_{i-1}, H_l) < b(u_{i+1}, H_r)$. □

According to Lemma 11, path-end vertices exhibit a larger broadcast time compared to other path vertices in any path-connected cliques graph. This behavior is illustrated in Figure 3, where the broadcast times of vertices u_1 and u_4 exceed those of the intermediate path vertices u_2 and u_3.

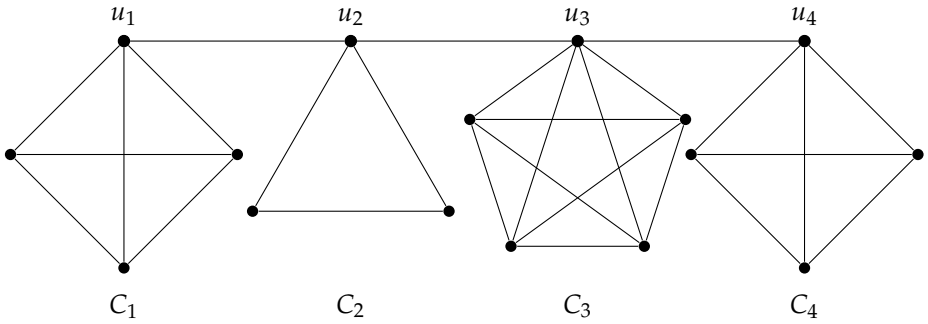

Figure 3. Path-connected cliques with $k = 4$ and 4 cliques of sizes 4, 3, 5, 4.

5.3. Broadcasting in Path-Connected Cliques from Clique Vertices

All non-path vertices within some clique C_i will have the same broadcast time; in any optimal scheme, a non-path originator v_i will inform the nearest path vertex u_i at time unit 1 since u_i can contribute to broadcasting within clique C_i as well as the path. Since all non-path vertices within some clique C_i are all connected to the same path vertex, they will all have the same broadcast time. Thus, to determine the broadcast time from an arbitrary originator, we must first identify the worst originator among all clique vertices.

Lemma 12. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. Then, for some vertex $v_1 \in C_1$, $b(v_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, 1 + a_1 + b(u_2, H_2)\}$ where $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$.*

Proof. Without the loss of generality, assuming that u_1 is the originator, v_1 is the first vertex to be informed in C_1. Then, if $a_1 \neq 1$, originator u_1 informs v_1 at time unit 1. In an optimal broadcast scheme, originator v_1 informs u_1 at time unit 1 since u_1 is a vertex in C_1 and a path vertex. For either originator, the only informed vertices in G_k after one time unit are v_1 and u_1. Hence, the remaining steps of broadcasting and the broadcast time are the same for both originators when $a_1 \neq 1$.

When $a_1 = 1$, by definition, originator u_1 informs u_2 at time unit 1. However, this is not possible when v_1 is the originator as v_1 informs u_1 at time unit 1, delaying broadcasting in H_2 by one time unit. Overall, broadcasting in H_2 completes at time unit $2 + b(u_2, H_2)$. Since $a_1 = 1$, we know that $b(u_2, H_2) \geq b(u_1, C_1)$. Broadcasting in C_1 finishes before H_2 due to the added delay in H_2. Thus, $b(v_1, G_k) = 2 + b(u_2, H_2)$ when $a_1 = 1$.

In general, the broadcast time of G_k from originator v_1 is given by $b(v_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, 1 + a_1 + b(u_2, H_2)\}$, where $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. Alternatively, $b(v_1, G_k)$ is also given by

$$b(v_1, G_k) = \begin{cases} 2 + b(u_2, H_2) & \text{if } b(u_2, H_2) \geq \lceil \log l_1 \rceil \\ \lceil \log l_1 \rceil & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 \geq 2^{b(u_2, H_2)} \\ \lceil \log l_1 \rceil + 1 & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 < 2^{b(u_2, H_2)} \end{cases}$$

A similar result can be shown for broadcasting in G_k from vertex $v_k \in C_k$. □

Remark 4. *From Lemma 12, we can see that $b(v_1, G_k) \geq b(u_1, G_k)$ by at most 1. Similarly, $b(v_k, G_k) \geq b(u_k, G_k)$. Thus, $b(G_k) = \max\{b(v_1, G_k), b(v_k, G_k)\}$.*

6. Conclusions

In this paper, we studied broadcasting in windmill graphs and stars of cliques, a generalization of windmill graphs with arbitrary clique sizes. Our most significant contribution is the development of an $O(n \cdot \log n)$ algorithm to determine the broadcast time of any orig-

inator in a star of cliques, proving its optimality. This result is foundational for designing efficient algorithms in other clique-based network topologies, such as block graphs.

We also examined path-connected cliques, proposing an algorithm that optimizes broadcast time based on clique size and adjacency, addressing challenges in path-dependent networks like hierarchical and block graphs.

While our work advances broadcasting algorithms for clique-centric topologies, it is limited to these structures. Future research could extend these methods to hybrid or generalized networks and explore scalability for large distributed systems.

Our contributions offer theoretical and practical insights, enhancing the efficiency of communication protocols in specialized network designs.

Author Contributions: Conceptualization, A.A. and H.A.H.; methodology A.A. and H.A.H.; investigation A.A.; writing—original draft A.A.; writing—review and editing A.A. and H.A.H.; supervision, H.A.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Hedetniemi, S.M.; Hedetniemi, S.T.; Liestman, A.L. A survey of gossiping and broadcasting in communication networks. *Networks* **1988**, *18*, 319–349. [CrossRef]
2. Slater, P.J.; Cockayne, E.J.; Hedetniemi, S.T. Information dissemination in trees. *SIAM J. Comput.* **1981**, *10*, 692–701. [CrossRef]
3. Dinneen, M.J. The complexity of broadcasting in bounded-degree networks. *arXiv* **1994**, arXiv:math/9411222 .
4. Jakoby, A.; Reischuk, R.; Schindelhauer, C. The complexity of broadcasting in planar and decomposable graphs. *Discret. Appl. Math.* **1998**, *83*, 179–206. [CrossRef]
5. Middendorf, M. Minimum broadcast time is NP-complete for 3-regular planar graphs and deadline 2. *Inf. Process. Lett.* **1993**, *46*, 281–287. [CrossRef]
6. Schindelhauer, C. On the Inapproximability of Broadcasting Time. In Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Saarbrücken, Germany, 5–8 September 2000; pp. 226–237.
7. Elkin, M.; Kortsarz, G. Combinatorial logarithmic approximation algorithm for directed telephone broadcast problem. In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, Montreal, QC, Canada, 19–21 May 2002; pp. 438–447.
8. Farley, A.M.; Hedetniemi, S.T. Broadcasting in grid graphs. In Proceedings of the 9th SE Conf. Combinatorics, Graph Theory, and Computing, Utilitas Mathematica, Boca Raton, FL, USA, 30 January–2 February 1978; pp. 275–288.
9. Liestman, A.L.; Peters, J.G. Broadcast networks of bounded degree. *SIAM J. Discret. Math.* **1988**, *1*, 531–540. [CrossRef]
10. Hromkovič, J.; Jeschke, C.D.; Monien, B. Optimal algorithms for dissemination of information in some interconnection networks. In Proceedings of the Mathematical Foundations of Computer Science 1990, Banská Bystrica, Czechoslovakia, 27–31 August 1990; Proceedings 15; Springer: Berlin/Heidelberg, Germany, 1990; pp. 337–346.
11. Gholami, S.; Harutyunyan, H.A.; Maraachlian, E. Optimal broadcasting in fully connected trees. *J. Interconnect. Netw.* **2023**, *23*, 2150037. [CrossRef]
12. Čevnik, M.; Žerovnik, J. Broadcasting on cactus graphs. *J. Comb. Optim.* **2017**, *33*, 292–316. [CrossRef]
13. Bhabak, P.; Harutyunyan, H.A.; Tanna, S. Broadcasting in harary-like graphs. In Proceedings of the 2014 IEEE 17th International Conference on Computational Science and Engineering, Chengdu, China, 19–21 December 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1269–1276.
14. Harutyunyan, H.A.; Maraachlian, E. On broadcasting in unicyclic graphs. *J. Comb. Optim.* **2008**, *16*, 307–322. [CrossRef]
15. Fraigniaud, P.; Lazard, E. Methods and problems of communication in usual networks. *Discret. Appl. Math.* **1994**, *53*, 79–133. [CrossRef]
16. Harutyunyan, H.A.; Liestman, A.L.; Peters, J.G.; Richards, D. Broadcasting and gossiping. In *Handbook of Graph Theory*; CRC: Boca Raton, FL, USA, 2013; pp. 1477–1494.
17. Albin, N.; Kottegoda, K.; Poggi-Corradini, P. Spanning tree modulus for secure broadcast games. *Networks* **2020**, *76*, 350–365. [CrossRef]

18. Averbuch, A.; Peeri, I.; Roditty, Y. New upper bound on m-time-relaxed k-broadcast graphs. *Networks* **2017**, *70*, 72–78. [CrossRef]
19. Averbuch, A.; Shabtai, R.H.; Roditty, Y. Efficient construction of broadcast graphs. *Discret. Appl. Math.* **2014**, *171*, 9–14. [CrossRef]
20. de Sousa, A.; Gallo, G.; Gutierrez, S.; Robledo, F.; Rodríguez-Bocca, P.; Romero, P. Heuristics for the minimum broadcast time. *Electron. Notes Discret. Math.* **2018**, *69*, 165–172. [CrossRef]
21. Sousa, A.D.; Gallo, G.; Gutiérrez, S.; Robledo, F.; Rodríguez-Bocca, P.; Romero, P. A tree-block decomposition-based heuristic for the minimum broadcast time. *Int. J. Metaheuristics* **2020**, *7*, 379–401. [CrossRef]
22. Fomin, F.V.; Fraigniaud, P.; Golovach, P.A. Parameterized complexity of broadcasting in graphs. *Theor. Comput. Sci.* **2024**, *997*, 114508. [CrossRef]
23. Hromkovič, J.; Jeschke, C.D.; Monien, B. Optimal algorithms for dissemination of information in some interconnection networks. *Algorithmica* **1993**, *10*, 24–40. [CrossRef]
24. Hromkovič, J.; Klasing, R.; Monien, B.; Peine, R. Dissemination of information in interconnection networks (broadcasting & gossiping). In *Combinatorial Network Theory*; Springer: New York, NY, USA, 1996; pp. 125–212.
25. Ivanova, M.; Haugland, D. Integer programming formulations for the shared multicast tree problem. *J. Comb. Optim.* **2019**, *38*, 927–956. [CrossRef]
26. Robledo, F.; Rodríguez-Bocca, P.; Romero, P. Optimal broadcast strategy in homogeneous point-to-point networks. In Proceedings of the Machine Learning, Optimization, and Data Science: 6th International Conference, LOD 2020, Siena, Italy, 19–23 July 2020; Revised Selected Papers, Part I 6; Springer: Berlin/Heidelberg, Germany, 2020; pp. 448–457.
27. Su, Y.H.; Lin, C.C.; Lee, D. Broadcasting in weighted trees under the postal model. *Theor. Comput. Sci.* **2016**, *621*, 73–81. [CrossRef]
28. Erdős, P.; Rényi, A.; Sós, V. On a problem of graph theory, Studia Sci. *Math. Hungar.* **1966**, *1*, 215–235.
29. Bermond, J.C. Graceful Graphs, Radio Antennae and French Windmills. In *Proceedings of the One Day Combinatorics Conference*; Research Notes in Mathematics, Vol. 34; Pitman: Milton Keynes, UK, 1978; pp. 18–37.
30. Ambashankar, A.; Harutyunyan, H.A. Broadcasting in Stars of Cliques. In *International Workshop on Combinatorial Algorithms*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 485–496.
31. Hovhannisyan, N. Exact and Factor Two Algorithms for Broadcast Time. Ph.D. Thesis, Concordia University, Montreal, QC, Canada, 2024.