Heuristics for the Minimum Broadcast Time

Amaro de Sousa [1]

Instituto de Telecomunicações
Campus Universitário de Santiago
Aveiro, Portugal

Franco Robledo, Pablo Rodríguez-Bocca, Pablo Romero
Gabriela Gallo, Santiago Gutierrez [2]

Instituto de Matemática y Estadística
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay.

Abstract

The problem under study is the Minimum Broadcast Time (MBT). We are given an undirected connected graph and a singleton that owns a message. The goal is to broadcast this message as soon as possible, where the communication takes place between neighboring-nodes in a selective fashion and each forwarding takes one time-slot. Historically, the MBT finds applications in telephonic services; however, it serves as an inspirational problem for the design of current delay-tolerant forwarding schemes in modern communication systems like content delivery networks and peer-to-peer networks. The problem belongs to the class of $\mathcal{NP}$-Complete class. As a consequence, the literature offers heuristics, approximation algorithms and exact exponential-time solutions.

The contributions of this paper are two-fold. First, an efficient Integer Linear Programming (ILP) formulation for the problem is provided. Second, a competitive heuristic called *TreeBlock*, is developed. TreeBlock exploits the optimality for the MBT in tree-graphs, and the fact that arbitrary connected graphs accept a decomposition into a tree-block structure, where the building blocks are maximally biconnected subgraphs. A fair comparison between TreeBlock and previous heuristics highlights the effectiveness of our proposal.

Keywords: Minimum Broadcast Time, Computational Complexity, Integer Linear Programming, Heuristics.

1 Introduction

The identification of an ideal forwarding scheme is a challenging problem in telecommunications. The scientific literature offers fluid models for massive communication systems [1], tree-like or one-to-one forwarding schemes [23], the tit-for-tat communication paradigm for incentivess [21], among many others.

A fundamental fluid model for a complete peer-to-peer network was posed for the first time by Qiu and Srikant [31]. In this fluid model, the authors consider a complete network with identical users, where simultanous forwarding takes proportionally more time. The authors claim a counterintuitive result, where *the one-to-one is a good forwarding strategy.* Even though the authors study the service capacity of a file sharing peer-to-peer system, its formulation is general enough. They find a closed formula for the average waiting time following a one-to-one forwarding scheme when the population is a power of two. In [25], a formal proof that the one-to-one forwarding scheme achieves the minimum waiting time is included, when the population is a power of two. Furthermore, recent works confirm that the optimality holds for arbitrary (non-complete) graphs as well. Incidentally, the one-to-one forwarding scheme is open, in the sense that the neighboring selection strategy must be determined. This means that the optimal forwarding scheme is equivalent to a well-known historical problem, known as the *Minimum Broadcast Time*, or MBT for short [9]. It finds original applications in telephonic services. However, it serves as an inspirational problem for the design of current delay-tolerant forwarding schemes in modern communication systems like Content Delivery Networks (CDN) and Peer-to-Peer (P2P) [5] networks, or Cognitive Radio (CR) [22] networks.

A natural description of the problem is in terms of phone-calls (we can

[1] Email: `asou@ua.pt`
[2] Email: `(gabriela.gallo,santiago.gutierrez,frobledo,prbocca,promero)@fing.edu.uy`

identify forwarding or gossiping, depending on the context). Suppose that a member originates a message which is to be communicated to all other members of the network. This is to be accomplished as quickly as possible by a series of calls placed over lines of the network. In the MBT, there are three strict broadcasting rules:

i Each phone-call requires one time-slot,

ii a member can participate in only one call per slot, and

iii a member can only call a neighbor member.

The problem under study is the following: given a connected graph G and a originator vertex v_0, what is the minimum number of time-slots required to complete broadcasting starting from v_0 and following the previous broadcasting rules? Here the graph represents an communication network, where vertices V are the members, and the edges E are the possible communication links between them. Defining an originator vertex v_0, the *minimum broadcast time (MBT)*, denoted $b(v_0, G)$ is the minimal number of time-slots necessary to disseminate a message from v_0 to all the other vertices of G. Most works study the MBT of a graph, $b(G)$, as the worst time among all potential originator members in the graph G, i.e. $b(G) = max\{b(v_0, G)|v_0 \in V\}$. There must be at least one *forwarding strategy* that achieves $b(v_0, G)$, and it could be specified as a sequence of parallel calls that follows the broadcasting rules and informs all the nodes in the network G in $b(v_0, G)$ time-slots.

This article is organized as follows. The main body of related work is covered in Section 2. An Integer Linear Programming formulation is introduced in Section 2.1. TreeBlock heuristic is fully described in Section 3. A fair comparison with the most competitive heuristics is carried out in Section 4. Section 5 presents concluding remarks and trends for future work.

2 Minimum Broadcast Time and Related Work

Garey and Johnson include the MBT in the list of $\mathcal{NP}$-Complete problems [11]. They consider an arbitrary starting set V_0 with possible more than a singleton, subject to the three previous broadcasting rules. Observe that a completion time in $T = 1$ time-slot exists if and only if V_0 accepts a perfect matching in bipartite graphs, which accepts a polynomial-time algorithm [7]. However, the case $T = 2$ accepts a reduction from 3-dimensional matching [27].

The MBT of a complete graph $G = K_n$ is clearly $b(K_n) = \lceil log_2(n) \rceil$, yet K_n is not minimal with this property. The minimum number of links to achieve $b(K_n)$ is an inverse design problem with valuable progress. Indeed, when $n = 2^m$ for some natural m, the hypercube Q_m has $m2^{m-1}$ links and the same broadast time is $b(Q_m) = m = log_2(n)$. Hence, the hypercube is the most economical broadcasting structure when the population is a power of two.

The optimum forwarding scheme is available only for specific families of graphs. The optimality in trees is recursively achieved in a greedy-like manner [27]. The idea is a bottom-to-top process, choosing members with maximum delay. 2D Grid graphs accept an optimal forwarding scheme as well [17]. If we are given an $R \times C$ grid, the broadcast time is precisely the diameter $D = R - 1 + C - 1 = R + C - 2$. The forwarding is intuitive, as the reader can appreciate starting from a corner and ending in the opposite one. Several other particular graph families are studied, to mention: cube-connected cycles [20], Butterfly and DeBruijn networks [18], and Harary networks [3].

In the general case, for an arbitrary graph G and an arbitrary originator vertex, Elkin et. al. [8] present an algorithm to obtain a forwarding strategy that found the best theoretical upper bound knowns [26], with $O(\frac{\log(|V|)}{\log\log(|V|)}b(G))$ time-slots. Several extensions to the original problem was studied. We will not cover them here, please read [24] and [13] for a review in MBT and related problems.

The literature offers several metaheuristics and approximation algorithms for the MBT, see for instance: [8], [19], [10], [2], [15], [14] and [6].

2.1 Exact Formulation for the MBT

In this section, an exact Integer Linear Programming formulation for the MBT is introduced. Let $G = (V, E)$ be a connected graph, where $V = \{0, 1, \ldots, n-1\}$ is the node-set. Denote $V(i)$ to the set of neighboring nodes of i. For simplicity, we choose $v_0 = 0$ as the source-node.

Recall that in every time-slot, a node i that owns the message can forward it to a single member of its neighbor-set $V(i)$. Furthermore, this node did not receive the message yet. The broadcast time is the number of time-slots required for the reachability of the message to the whole set V. The goal is to minimize the broadcast time.

In order to find an efficient ILP formulation, we consider an upper-bound T for the minimum completion time. Consider the set of binary variables x_{ij}^t

such that, when equal to one, mean that the message is transmitted from node i to node j during time-slot t. In addition, we use an integer variable z whose value represents the broadcast time (measured in time-slots).

The Integer Linear Programming (ILP) model for the MBT is expressed as follows:

$$\begin{array}{l}\text{Minimize } z\\ \text{Subject to:}\end{array} \tag{1}$$

$$\sum_{j\in V(i)} x_{ij}^{1} = \begin{cases} 1\,, i=0 \\ 0\,, i\neq 0 \end{cases} \qquad , i = 0, ..., n-1 \tag{2}$$

$$\sum_{j\in V(i)} \sum_{t=1}^{T} x_{ji}^{t} = 1 \qquad , i = 1, ..., n-1 \tag{3}$$

$$\sum_{j\in V(i)} x_{ij}^{t} \leq 1 \qquad \begin{array}{l}, i = 0, ..., n-1\\ , t = 2, ..., T\end{array} \tag{4}$$

$$\sum_{\tau=1}^{t-1} \sum_{k\in V(i)\setminus\{j\}} x_{ki}^{\tau} \geq x_{ij}^{t} \qquad \begin{array}{l}, (i,j)\in A : i\neq 0\\ , t = 2, ..., T\end{array} \tag{5}$$

$$\sum_{t=1}^{T} t\cdot x_{ij}^{t} \leq z \qquad , (i,j)\in A \tag{6}$$

$$x_{ij}^{t} \in \{0,1\} \qquad \begin{array}{l}, (i,j)\in A\\ , t = 1, ..., T\end{array} \tag{7}$$

$$z\in\mathbb{N} \tag{8}$$

The objective function 1 is the broadcast time. Constraints 2 state that the message is transmitted only once by the source-node 0 during the first time-slot $t = 1$. Constraints 3 guarantee that the message is received by each node exactly once from some neighboring-node. Constraints 4 guarantee that on each time slot $t = 2, ..., T$ (the case of $t = 1$ is already constrained in 2) a node can only transmit the message to at most one neighboring node. Constraints 5 guarantee that the message is transmitted from node i (except when it is the source node 0) to node j on time slot t only if the message has been received by node i on a previous time slot from another neighboring node of i (*i.e.*, not from node j). Constraints 6 guarantee that the completion time is not lower than the time slot of any message transmission. Finally, constraints 7–8 are the variable domain constraints.

The runtime required to solve the optimization problem by using a standard solver is significantly shortened with the introduction of the following improvements. First, consider the minimum number of hops of any path in G from source node 0 to node i given by h_i (this can be easily obtained by a standard shortest path algorithm). For a given number of hops h, consider set N_h composed by all nodes such that $h_i = h$.

Then, a node $i \in N_h$ cannot transmit the message on any of the time slots $t = 1, ..., h_i$. So, for each node $i \in N_h$, we can set to 0 all variables x_{ij}^t, for $t = 1, ..., h$ (resulting, in practice, in a reduction of the number of variables). Moreover, a node $i \in N_h$ can transmit at most one message on time slot $t = h + 1$, and, generalizing, at most π messages on time slots $t = h + 1 + \pi$. So, we can define the following valid inequalities:

$$\sum_{i \in N_h} \sum_{j \in V(i)} x_{ij}^{h+1+\pi} \leq 1 + \pi \qquad \begin{array}{l} , h = 1, ..., T-1 \\ , \pi = 0, ..., T-h+1 \end{array} \tag{9}$$

Our computational tests have showed that constraints 9 do not improve significantly the model when $\pi > 0$ and, for this reason, the computational results were obtained by adding inequalities 9 only for $\pi = 0$.

Finally, we can reduce even further the number of variables in the following way. Consider d as the degree of source node 0. Without any lack of generality, we can say that if the d neighboring nodes are used by source node 0 to transmit the message, than, these transmissions will happen in the first d time slots. So, we can set to 0 all variables x_{0j}^t, for $(0, j) \in A$ and $t = d + 1, \ldots, T$.

3 Heuristic for the MBT

In this section we introduce a novel heuristic for the MBT. The key idea is a decomposition of the graph into two-node connected blocks, and an efficient routing in the tree that is defined by these blocks, inspired by the optimum routing in trees. Recall that the MBT accepts an optimal forwarding scheme in trees. This optimal routing is called MakespanTree here.

A node $v \in V(G)$ is called a *cut-point* if $G - v$ has more components than

G. A block is a maximal subgraph of G that has no cut-points. Clearly, if G is connected and has no cut-points, then G is a block. Every connected graph G accepts a tree-block descomposition, where each block B_i is an induced subgraph with no cut-points, and every cut-point $v \in V(G)$ is incident to the blocks that include it. Formally, given an arbitrary connected graph $G = (V, E)$, the tree-block graph is $B(G) = (B \cup U, E')$, where $U = \{v_1, \dots, v_r\}$ is the set of cut-points in G, $B = \{B_1, \dots, B_s\}$ the set of blocks of G, and $E' \subseteq B \times U$ consists of links between v_i and B_j whenever the cut-point belongs to B_j. By its definition, tree-block is a tree, where some nodes are blocks. Observe that if G has no cut-points, then $B(G)$ has a single isolated node, and it is a tree as well. Additionally, in a non-trivial tree, there exists some *leaf-blocks*, that are incident with a single cut-point. The reader is invited to consult [30] for further details.

The two main building blocks of our algorithm are Functions $BroadcastTree$ and $BroadcastBlock$. Essentially, $BroadcastTree$ serves as an *interblock* forwarding in the tree-block structure, while $BroadcastBlock$ serves as an *intra-block* forwarding. A full forwarding strategy is obtained by a cooperation of both functions.

Algorithm 1 $F = TreeBlock(G, v_0)$

```
  $U \leftarrow CutPoints(G)$
  $U_0 \leftarrow \{v_0\} \cup U$
  $d \leftarrow Distances(U_0)$
  $T \leftarrow MST(U_0, d)$
  $B \leftarrow Blocks(G)$
  $L \leftarrow LeafBlocks(B)$
  for all $B_i \in L$ do
    $z_i \leftarrow B_i \cap CutPoints(G)$
    $e_i \leftarrow max_{v \in B_i}\{d(w_i, v)\}$
    $w(z_i, B_i) \leftarrow e_i$
  end for
  $T \leftarrow T \cup \{(z_i, B_i)\}_{i=1,\dots,|L|}$
  $F \leftarrow BroadcastTree(T, v_0)$
  for all $B_i \in B$ do
    $F_i \leftarrow BroadcastBlock(B_i, F(B_i))$
    $F \leftarrow F \cup F_i$
  end for
  return $F$
```

A step-by-step description of the algorithm $TreeBlock$ follows. All the cut-points U of G are determined in Line 1. The source-node is added to this special set in U_0 (Line 2). A complete weighted graph is built in Lines 3, where the corresponding weights are the distance between every pair of nodes in U_0. A minimum spanning tree T is found using Kruskal algorithm in Line 4. This tree serves as the *skeleton* of the interblock forwarding scheme. The weights in the links represent a *delay* between different cut-points. Observe that there is an additional delay with leaf-blocks, that should be added to reach all nodes in the graph. Therefore, in Lines 5 and 6, the blocks and leaf-blocks from G are found. Recall that a leaf-node has a single cut-point. In the block of Lines 7-11, the eccentricity e_i from the cut-point $z_i \in B_i$ is found for any leaf-block $B_i \in L$. This additional latency is considered by a tree-augmentation, in Line 12. The result is that the new nodes z_i are leaf-nodes in T, and the corresponding links have weights e_i. An inter-block forwarding scheme F is performed using Function $BroadcastTree$ with source-node v_0 in the tree T. Function $BroadcastTree$ returns the optimum forwarding scheme in this weighted tree. Finally, the intra-block forwarding is performed using Function $BroadcastBlock$ (see the last for-loop; Lines 14-17).

In the following paragraphs we provide details of Functions $BroadcastTree$ and $BroadcastBlock$ respectively.

Algorithm 2 $F = BroadcastTree(T, v_0)$

```
1: $H \leftarrow FindHeight(T, v_0)$
2: for all $v \in T : height(v) = 0$ do
3:     $l(v) \leftarrow 0$
4: end for
5: for $h = 1$ to $H$ do
6:     $T(h) = \{v \in T : height(v) = h\}$
7:     for all $v \in T(h)$ do
8:         $(d_1, \dots, d_r) \leftarrow SortChildren(v)$
9:         $l(v) \leftarrow max_{1 \leq i \leq r}\{d_i + i\}$
10:    end for
11: end for
12: $F \leftarrow LargestLabel(v_0)$
13: return $F$
```

The height H of the tree with respect to the source-node as root is found in Line 1. In the block of Lines 2-4, all leaf-nodes are labelled with $l(v) = 0$. This label represent the *priority* to forward this node (the default label to leaves mean that the forwarding order in leaf-nodes is indifferent). All the remaining nodes are labelled during the for-loop of Lines 5-11. We iteratively consider different heights, represented by the node-subset $T(h) \subseteq T$ (see Line 6). Consider an arbitrary node $v \in T(h)$, with r children. These children are ordered from the largest label d_1 to the lowest label d_r (Line 8), and the label of v is found in Line 9. It is worth to remark how this labelling process works: the message is transmitted to node with the most stringent global timing constraint (i.e., the largest label) first. This greedy-like forwarding scheme (Line 12) achieves optimality in trees, and it is returned in Line 13.

Algorithm 3 $F_i = BroadcastBlock(B_i, v'_1, v'_2, \ldots, v'_k)$

```
1: $T \leftarrow MST(v'_1, B_i)$
2: for $i = 1$ to $k$ do
3:     $w_i \leftarrow \{|B_i|/i\}$
4:     $w(u_i, v'_i) \leftarrow w_i$
5:     $T \leftarrow T \cup w(u_i, v'_i)$
6: end for
7: $F_i \leftarrow BroadcastTree(T, v'_1)$
8: return  $F_i$
```

Let us finally consider *BroadcastBlock*. It receives a block B_i and the expected forwarding order $F(B_i)$ of all its k cut-points, to know, $v'_1, v'_2, \ldots, v'_k$. Therefore, node v'_1 is the root-node. In Line 1, Kruskal algorithm is applied into B_i to find the minimum spanning tree, rooted at v'_1. In order to force the correct forwarding order in the cut-points, the weights in their incident links are modified as follows. The incident link to v'_i in T will be assigned a weight $w_i = |B_i|/i$ (Lines 3-4). The resulting tree is updated in Line 5. Finally, Function *BroadcastTree* is applied into this tree in order to define an intra-block forwarding scheme (Line 7), and the result is returned in Line 8.

4 Experimental Analysis

In this section, we evaluate the performance of our heuristic for the MBT. The problem was previously studied for well-known topologies, therefore we first compare our heuristic with the known results of the problem for these graphs. Later, we explore the performance of our heuristic over a large dataset of synthetic graphs, following the Watts-Strogatz small-world model [29], that it is an acceptable model for real communication networks [28]. For this dataset, whenever it is possible, we compare results with an IPL solver.

4.1 Particular Topologies

Following previous studies, we will present the results of our heuristic for five well-known topologies: *Lattice*, *De Bruijn*,*Hypercube*, *Harary*, and *Chord Harary*. Tables 1-5 show the number of nodes (N), the optimal result according to the ILP formulation (ILP), and the $TreeBlock$ heuristic result (TB). Other columns in tables are specific of the particular topology.

As a particular lattice topology, we studied the 2D grid graphs. In this case, a problem instance is defined by the number of rows R and number of columns C of the grid, and by the source node defined as (r, c), where $1 \leq r \leq R$ is the row number and $1 \leq c \leq C$ is the column number of the source node. Following [17], when source node is $(1, 1)$ then the broadcast time is precisely the diameter $D = R-1+C-1 = R+C-2$. Table 1 summirizes the results for 16 random 2D grid instances. The heuristic $TreeBlock$ reaches the optimum result in all of them.

In [12], Frank Harary introduced the Harary graph, noted $H_{k,n}$. It is a k-connected graph on n vertices which have a degree of at least k and $kn2$ edges. The structure of $H_{k,n}$ depends on the parities of k and n.

The algorithm analyzed in [4] ensures to find the exact result of broadcasting time for half of the cases. The results are mostly optimal in the instances studied with the TB heuristic; in the other cases they turn out to be very close, being a timeslot the difference.

Chord-Harary graph is a variation of the Harary graph, $CH_{k,n}$, introduced in [3]. This topology is proposed by having managed to find an algorithm where low values of broadcasting time can be reached. It was also shown that it reaches the optimum for a subset of these graphs, where k maintains a certain relation with n. Table ref tab: chordharary shows that the results obtained by the TB heuristic are still competitive, reaching the exact result in some cases, for this topology. The chosen instances did not take into account the relationship between the parameters for which the algorithm of cite bhabak2017 achieves optimality precisely in order to observe how it behaves in generality.

Finally, following what is presented in [16] in Table 4 and Table 5 it can be observed that the TB algorithm achieves competitive results with the previous heuristics for the analyzed cases .

Having tested different topologies it is found that the ILP formula, although it is not capable of solving all the instances, reaches the optimal values for many cases. On the other hand, the TB heuristic was developed without specializing in any type of graph in particular and with focus on graphs with more than one biconnex component. However, it achieves acceptable results, with the gap between TB and the heuristics presented very small in almost all cases.

Table 1
Lattice

R	C	r	c	N	ILP	TB
7	10	1	1	70	15	15
7	10	2	3	70	12	12
7	10	4	1	70	13	13
7	10	4	5	70	9	9
9	15	1	1	135	22	22
9	15	3	4	135	17	17
9	15	5	1	135	19	19
9	15	5	8	135	13	13
11	20	1	1	220	29	29
11	20	3	5	220	23	23
11	20	6	1	220	25	25
11	20	6	10	220	16	16
13	30	1	1	390	41	41
13	30	4	8	390	31	31
13	30	7	1	390	36	36
13	30	7	15	390	22	22

Table 2
Harary. Broadcast scheme S. [3]

N	k	n	ILP	TB
17	2	17	9	9
17	3	17	5	5
17	5	17	5	5
17	6	17	5	5
17	7	17	5	5
30	2	30	15	15
30	3	30	9	9
30	8	30	5	6
30	9	30	5	6
30	10	30	5	6
50	2	50	25	25
50	3	50	14	14
50	11	50	-	7
50	20	50	-	8
50	21	50	-	7
100	2	100	50	50

Table 3
Chord-Harary. Broadcast Scheme B. [3]

N	k	n	ILP	TB
10	3	10	4	4
10	5	10	4	4
30	3	30	9	9
30	5	30	5	6
30	7	30	5	6
50	5	50	6	7
50	7	50	-	7
50	9	50	-	7
50	11	50	-	6
100	7	100	7	8
100	11	100	-	8
100	13	100	-	8
300	7	300	-	15
300	11	300	-	10
300	17	300	-	10

Table 4
De Bruijn. RH, TBA, NTBA [16]

N	m	n	RH	TBA	NTBA	ILP	TB
8	2	3	4	4	4	3	4
16	2	4	5	5	5	4	5
32	2	5	7	6	7	6	7
64	2	6	8	8	8	7	8
128	2	7	9	9	10	-	10
256	2	8	11	11	12	10	12
512	2	9	12	12	13	-	14
1024	2	10	14	14	15	-	15
2048	2	11	15	15	17	-	17
4096	2	12	17	17	19	-	19
8192	2	13	18	18	20	-	21

Table 5
Hypercube. C, TBA, NTBA [16].

N	l	dim	O	C	TBA	NTBA	TB
16	2	4	4	-	-	-	4
32	2	5	5	5	5	5	5
64	2	6	6	7	6	7	6
128	2	7	7	8	7	9	7
256	2	8	8	9	9	11	8
512	2	9	9	10	10	14	9
1024	2	10	10	12	11	15	10
2048	2	11	11	-	12	18	11
4096	2	12	12	-	13	20	12

4.2 Large Dataset of Watts-Strogatz Graphs

An analysis was carried out on small-world networks for different number of nodes, and with different levels of connectivity, in order to be able to identify if it performs for graphs of more significant sizes and closer to reality.

In small world instances, we expect to compare quality and execution time.

Watts and Strogatz identified that many real networks have high levels of clustering, but small distances between most nodes. And it is particulary true in communication networks, where our problem applies. In order to create a network graph with both of these properties, Watts and Strogatz creates a graph model that begings with a lattice structure, and then randomly rewires a small percentage of the edges (with an independent probability p, and with attention to avoid the construction of loops and multi-edges). As expected, this model has high level of clustering and low average distances between nodes.

Fig. 1. Instances Distribution

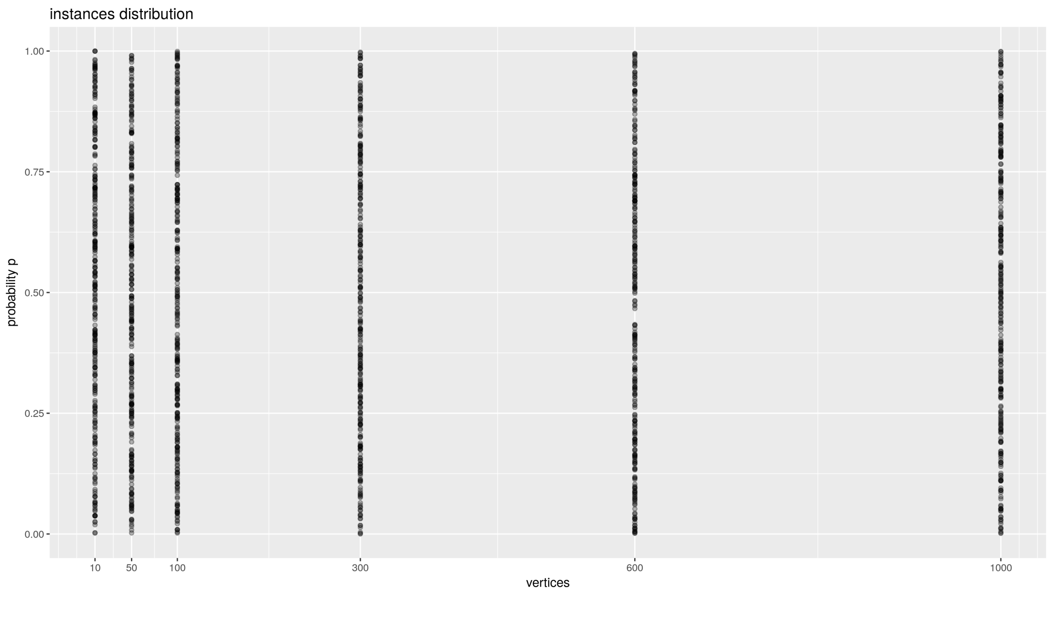
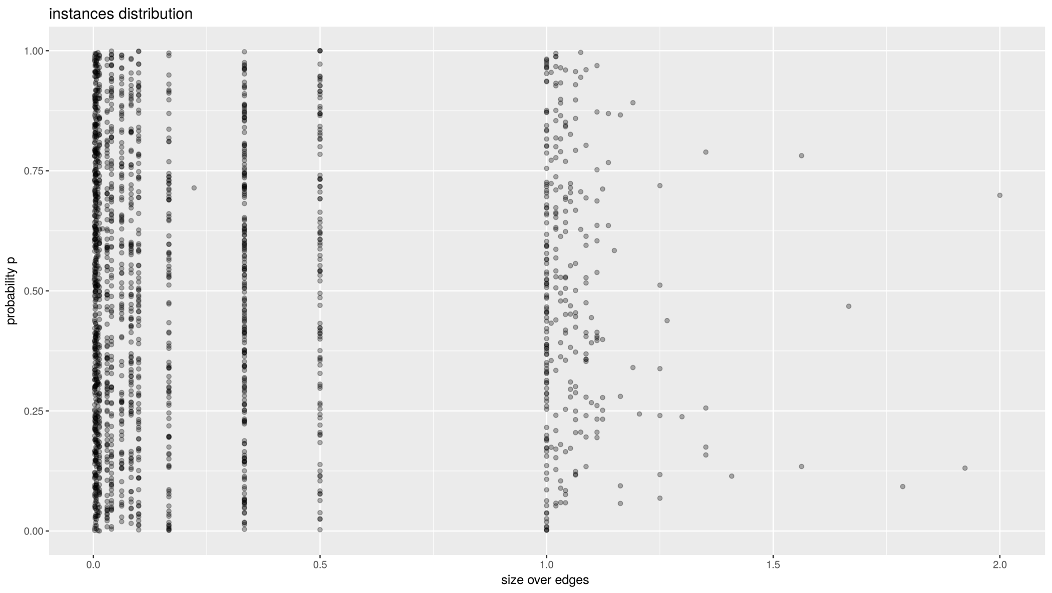


Fig. 2. Instances Distribution considering edges



The figure 1 shows the distribution of instances generated based on the probability of rewirng p and the number of nodes, clarifying that the choice of p was made uniformly independent.

In order to prove the deviation of the heuristic by altering the size; Besides having generated instances with different numbers of nodes, a second variable called *neighboorhood* was used for some cases, which directly impacts the number of edges of the graphs, increasing connectivity between nodes. In figure 2 the distribution is observed taking into account the probability and also the relationship between the number of nodes and the amount of edges.

For each instance, both the constructed heuristic and the ILP formula were executed in order to make a comparison of the results.

Regarding the processing times, it is evidenced in the figure 3 that for many cases it was not possible to throw a solution using the ILP formula, mostly for large instances. This is due to the complexity that the algorithm supposes, consequence of the complexity of the problem.

However, in cases where ILP found an optimal solution, it did so in less time than the heuristic. But this is not relevant since: 1. ILP is run on CPLEX, a highly optimized code commercial package while the heuristic is a research non-optimized code. 2. CPLEX is run on a very high performing server while heuristic is run on home-computers.

Likewise, it is observed in the figures 4 and 5 that the execution times of the heuristic are also strongly related to the size of the instance, both by the number of nodes, as well as by the number of edges. However, it is visualized that p does not seem to affect at least the times.

Fig. 3. Comparative: execution time versus instance size

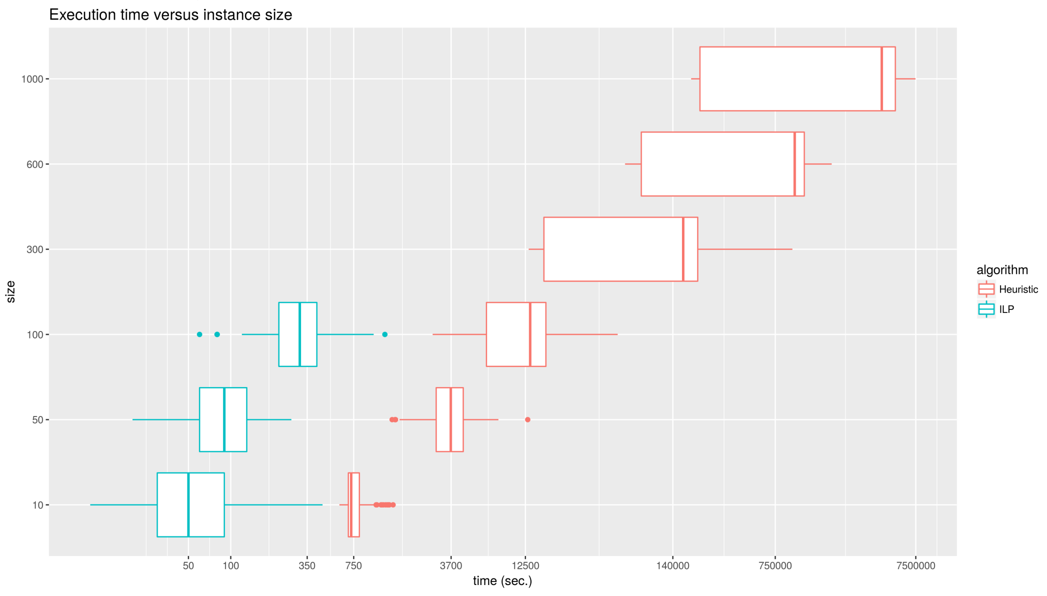


Fig. 4. Heuristic execution time versus instance size

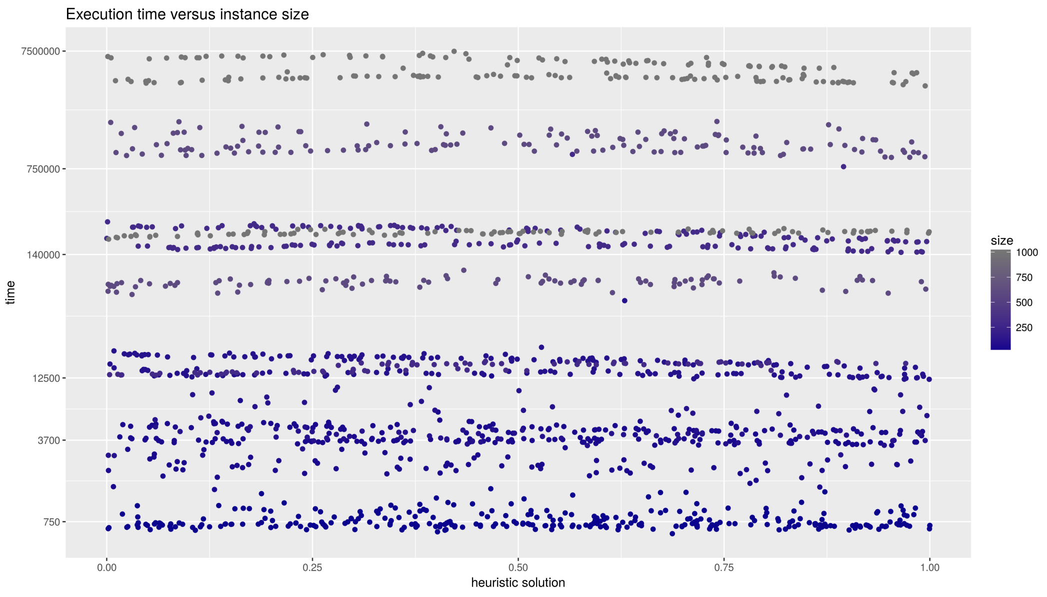
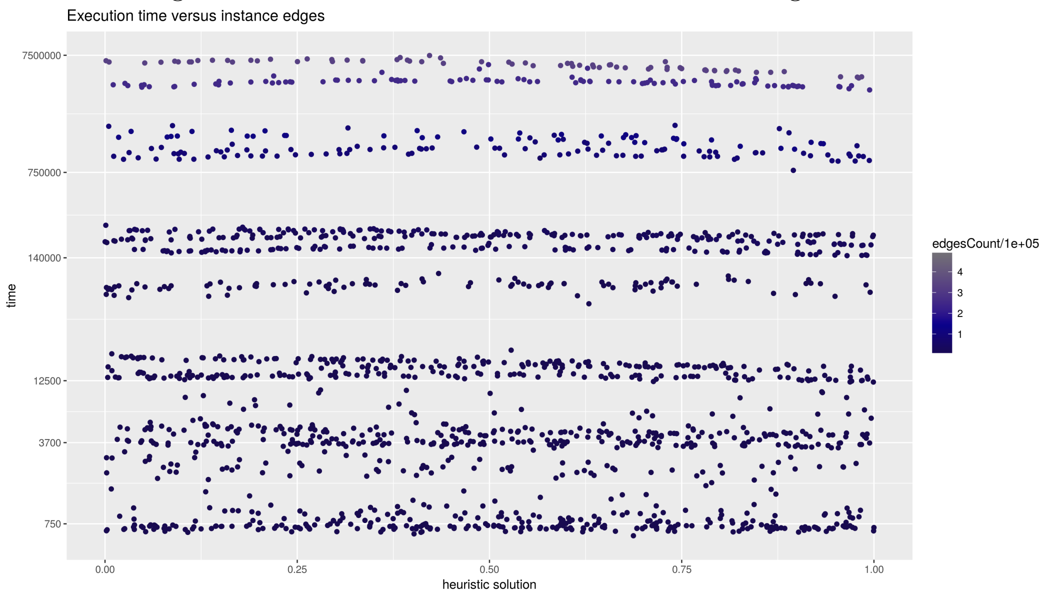
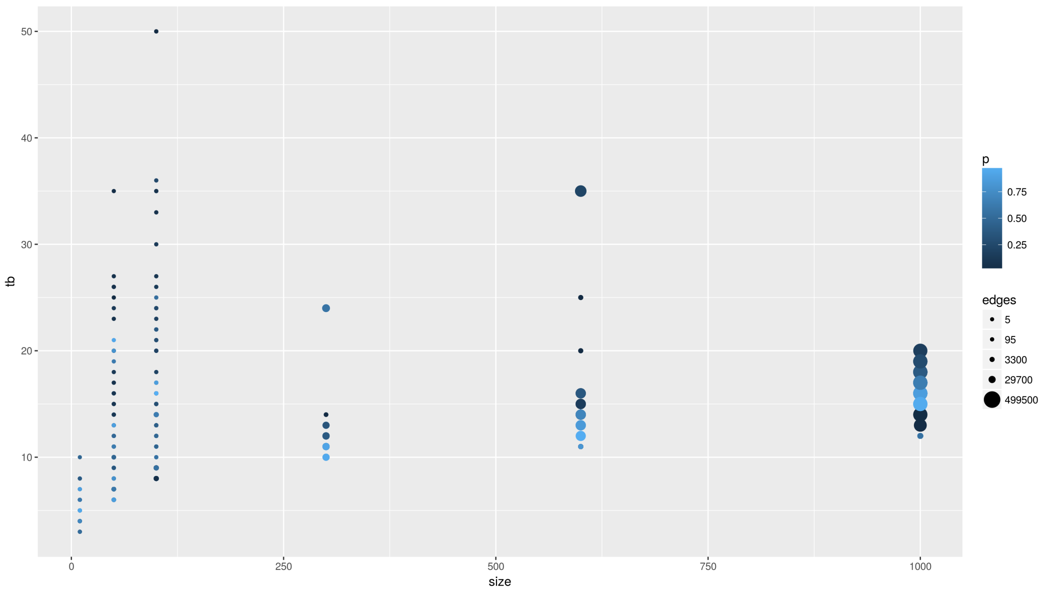


Fig. 5. Heuristic execution time versus instance edges count



Considering groups of instances of equal size (nodes and edges), in the figure 6 we see a tendency to reach a higher broadcasting time when the probability of rewiring is low. This observation seems to be accurate considering the process of creation of the small-worlds, because with a low probability of rewiring, the graphs would be more similar to the initial lattice. In the worst case a ring, with broadcasting time equal to the number of nodes -1.

Fig. 6. Heuristic solution versus p and edges grouped by size



Finally, for the instances in which the ILP formula achieved an optimal result, a comparison can be made in terms of the difference in value achieved in each algorithm.

In the table 6 is shown for each group of instances, which was the range of optimal values found by the formula ILP and which by the heuristic. The column 'GAP' indicates what is the average difference between the solution achieved with the heuristic and the ILP formula. In addition, the column 'no GAP' indicates the number of instances for which the heuristic found the optimum.

In the table 7 you can see in what percentage the results of the heuristic approach the optimum. The 100-size segment stands out, where 87% of the instances were resolved with a precision of less than 10%. For the rest of the cases it is not minor to emphasize that the size of the set of tests for which it was possible for the two algorithms to find a solution is considerably smaller.

Table 6

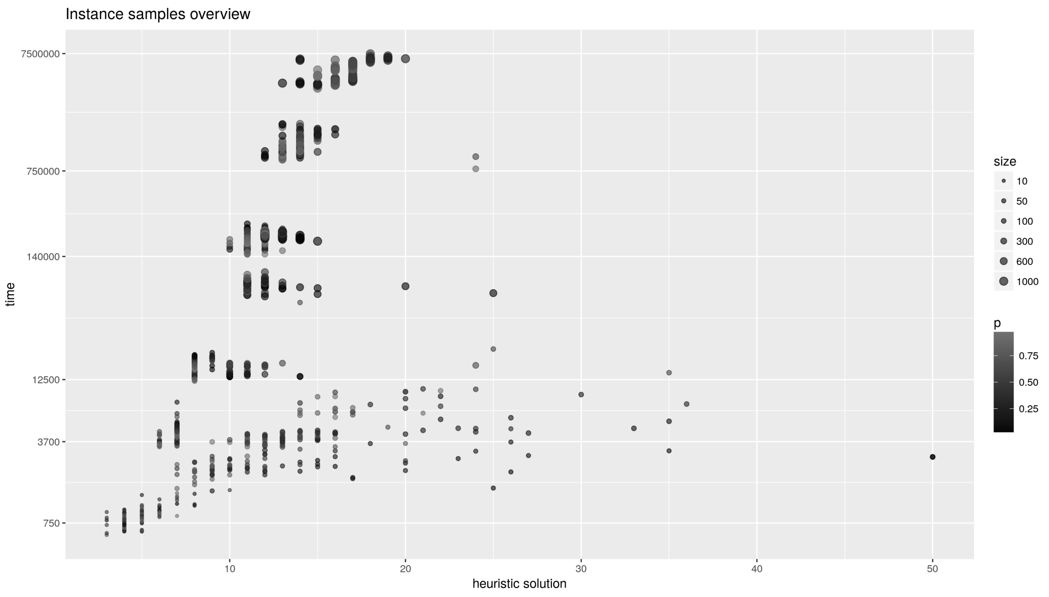
	size	edges	TB	ILP	GAP	no GAP	#instances
1	10	20.01	4.39 ± 0.91	4.22 ± 0.65	0.04 ± 0.12	263.00	301.00
2	50	46.52	12.85 ± 5.33	11.51 ± 5.27	0.14 ± 0.17	33.00	100.00
3	100	94.04	16.9 ± 7.4	14.46 ± 7.13	0.19 ± 0.21	22.00	100.00

Table 7

	size	edges	GAP ≤ 10%	GAP ≤ 25%	#instances
1	10	20.01	263.00	287.00	301.00
2	50	46.52	47.00	87.00	100.00
3	100	94.04	43.00	77.00	100.00

La figura 7 muestra la distribucin de los datos en terminos de tamao y probabilidad de rewiring en comparacion con los tiempos y resultados.

Fig. 7.



5 Conclusions and Trends for Future Work

This paper studies the problem Minimum Broadcast Time. Two algorithms are presented to find the result, one through an exact method and the other through a heuristic. The exact method, as could be anticipated, is interesting for the resolution of instances of smaller size, both in number of nodes and edges; throwing for these cases results in time quite small (less than two hours). For its part, the heuristic, being its intention to cover a larger universe of instances, was tested in larger sizes also yielding results in a reasonable time for up to 1000 nodes.

Regarding the deviation of the results achieved by the heuristic against the optimum; It is found that there is an important success in cases where the optimum is known. The margin of error exists and is considerable; it does not exceed 25 % in most cases. Likewise, in comparison to other known heuristics, focused mostly on specific topologies, the results are competitive.

In order to improve the accuracy of the results for the heuristic, it is expected to work on some improvements. First, work on the decision of each node to transmit to another block or continue in it. The assembly of the initial tree of each block does not yield the most convenient and does not take advantage of the maximum transmission capacity of each node for all cases. Improving this step of the algorithm would incur direct improvements to the solution found.

Regarding the data set used, we understand it opportune to generate a larger one, in order to confirm some trends. In addition, it would be interesting to be able to test the methods presented in other known topologies, to validate the behavior in them.

Acknowledgment

This work is partially supported by Project CSIC I+D 395 entitled *Sistemas Binarios Estocásticos Dinámicos.*

References

[1] Anulova, S., "Fluid Limit for Switching Closed Queueing Network with Two Multi-servers," Springer International Publishing, Cham, 2017 pp. 343–354.

[2] Beier, R., R. Beier, J. F. Sibeyn and J. F. Sibeyn, *A powerful heuristic for telephone gossiping* (2000).

[3] Bhabak, P., H. Harutyunyan and P. Kropf, *Efficient broadcasting algorithm in harary-like networks*, in: *2017 46th International Conference on Parallel Processing Workshops (ICPPW)*, 2017, pp. 162–170.

[4] Bhabak, P., H. A. Harutyunyan and S. Tanna, *Broadcasting in harary-like graphs*, pp. 1269–1276.
URL http://ieeexplore.ieee.org/document/7023754/

[5] Chuang, Y.-T., *Protecting against malicious and selective forwarding attacks for p2p search & retrieval system*, Peer-to-Peer Networking and Applications **10** (2017), pp. 1079–1100.

[6] Crescenzi, P., P. Fraigniaud, M. Halld?rsson, H. A. Harutyunyan, C. Pierucci, A. Pietracaprina and G. Pucci, *On the complexity of the shortest-path broadcast problem*, Discrete Applied Mathematics **199** (2016), pp. 101 – 109, sixth Workshop on Graph Classes, Optimization, and Width Parameters, Santorini, Greece, October 2013.
URL http://www.sciencedirect.com/science/article/pii/S0166218X15002267

[7] Edmonds, J., "Paths, Trees, and Flowers," Birkhäuser Boston, Boston, MA, 1987 pp. 361–379.

[8] Elkin, M. and G. Kortsarz, *A combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem*, SIAM Journal on Computing **35** (2005), pp. 672–689.

[9] Farley, A. M., *Broadcast time in communication networks*, SIAM Journal on Applied Mathematics **39** (1980), pp. 385–390.

[10] Fraigniaud, P. and S. Vial, *Approximation algorithms for broadcasting and gossiping*, J. Parallel Distrib. Comput. **43** (1997), pp. 47–55.
URL http://dx.doi.org/10.1006/jpdc.1997.1318

[11] Garey, M. R. and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," W. H. Freeman & Company, New York, NY, USA, 1979.

[12] Harary, F., *The maximum connectivity of a graph*, Proceedings of the National Academy of Sciences of the United States of America **48** (1962), pp. 1142–1146.
URL http://www.jstor.org/stable/71730

[13] Harutyunyan, H., A. Liestman, J. Peters and D. Richards, *Broadcasting and gossiping*, in: J. L. Gross, J. Yellen and P. Zhang, editors, *Handbook of Graph Theory, Second Edition*, Chapman & Hall/CRC, 2013, 2nd edition p. 18.

[14] Harutyunyan, H. A. and S. Kamali, *Efficient broadcast trees for weighted vertices*, Discrete Applied Mathematics **216** (2017), pp. 598 – 608, levon Khachatrian's Legacy in Extremal Combinatorics.
URL http://www.sciencedirect.com/science/article/pii/S0166218X16304358

[15] Harutyunyan, H. A. and B. Shao, *An efficient heuristic for broadcasting in networks*, Journal of Parallel and Distributed Computing **66** (2006), pp. 68 – 76.
URL http://www.sciencedirect.com/science/article/pii/S0743731505001681

[16] Harutyunyan, H. A. and W. Wang, *Broadcasting algorithm via shortest paths*, pp. 299–305.
URL http://ieeexplore.ieee.org/document/5695616/

[17] Hedetniemi, S. M., S. T. Hedetniemi and A. L. Liestman, *A survey of gossiping and broadcasting in communication networks*, Networks **18** (1988), pp. 319–349.
URL http://dx.doi.org/10.1002/net.3230180406

[18] Klasing, R., B. Monien, R. Peine and E. A. St?hr, *Broadcasting in butterfly and debruijn networks*, Discrete Applied Mathematics **53** (1994), pp. 183 – 197.
URL
http://www.sciencedirect.com/science/article/pii/0166218X94901848

[19] Kortsarz, G. and D. Peleg, *Approximation algorithms for minimum-time broadcast*, SIAM Journal on Discrete Mathematics **8** (1995), pp. 401–427.

[20] Liestman, A. L. and J. G. Peters, *Broadcast networks of bounded degree*, SIAM Journal on Discrete Mathematics **1** (1988), pp. 531–540.

[21] Liu, W., D. Peng, C. Lin, Z. Chen and J. Song, *Enhancing tit-for-tat for incentive in bittorrent networks*, Peer-to-Peer Networking and Applications **3** (2010), pp. 27–35.

[22] Liyana Arachchige, C. J., S. Venkatesan, R. Chandrasekaran and N. Mittal, "Minimal Time Broadcasting in Cognitive Radio Networks," Springer Berlin Heidelberg, Berlin, Heidelberg, 2011 pp. 364–375.

[23] Pazzi, R. W., A. Boukerche, R. E. Grande and L. Mokdad, *A clustered trail-based data dissemination protocol for improving the lifetime of duty cycle enabled wireless sensor networks*, Wirel. Netw. **23** (2017), pp. 177–192.

[24] Peleg, D., "Time-Efficient Broadcasting in Radio Networks: A Review," Springer Berlin Heidelberg, Berlin, Heidelberg, 2007 pp. 1–18.

[25] Romero, P., "Mathematical Analysis of Scheduling Policies in Peer-to-Peer video streaming networks," Ph.D. thesis, Universidad de la República, Montevideo, Uruguay (2012).

[26] Schindelhauer, C., *On the inapproximability of broadcasting time*, in: *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization*, APPROX '00 (2000), pp. 226–237.
URL http://dl.acm.org/citation.cfm?id=646688.702973

[27] Slater, P. J., E. J. Cockayne and S. T. Hedetniemi, *Information dissemination in trees*, SIAM Journal on Computing **10** (1981), pp. 692–701.

[28] Watts, D. J., “Six Degrees: The Science of a Connected Age,” W. W. Norton and Company, Inc., 2004, 1st edition.

[29] Watts, D. J. and S. H. Strogatz, *Collective dynamics of ’small-world’ networks*, Nature **393** (1998), pp. 440–442.

[30] West, D., “Introduction to Graph Theory,” Featured Titles for Graph Theory Series, Prentice Hall, 2001.

[31] Yang, X. and G. de Veciana, *Service Capacity of Peer to Peer Networks*, , **4**, 2004, pp. 2242–2252.