Topology Control in Cooperative Ad Hoc Wireless Networks

T. F. Neves [1]

Department of Computer Science
University of Brasilia
Brasilia, Brazil

J. L. Bordim[2]

Department of Computer Science
University of Brasilia
Brasilia, Brazil

Abstract

Cooperative communication (CC) is a technique that exploits spatial diversity allowing multiple nodes to cooperatively relay signals to the receiver so that it can combine the received signals to obtain the original message. CC can be combined with topology control to increase connectivity at the cost of a small increase in energy consumption. This work focuses on exploring CC to improve the connectivity with a sink node in ad hoc wireless networks. More precisely, this work proposes a new technique, named CoopSink, that combines CC and topology control techniques to increase connectivity to a sink node while ensuring energy-efficient routes. Simulation results show that connectivity and routing to the sink cost can be improved up to 6.8 and 2.3 times, respectively, when compared with other similar strategies.

Keywords: Topology Control, Wireless Networks, Network Protocols, Cooperative Communication.

1 Introduction

Ad hoc wireless networks are networks where the nodes can communicate with each other without resorting to a centralised infrastructure. These networks have a large number of civil and military applications, ranging from communication support in battlefield, search and rescue operations to object monitoring and tracking. One of the major challenges in ad hoc networks is to reduce energy consumption as these nodes are often powered by batteries [2]. As battery replacement may not be a feasible option during operation, alternatives to improve and optimize energy expenditure in wireless networks are of great interest. This facts have driven the quest for power saving strategies aiming to extend the network lifetime [25,5]. These energy

[1] Email: tfn.thiago@cic.unb.br

[2] Email:bordim@unb.br

saving proposals can be grouped in two main categories: (*i*) techniques that allow the nodes to alternate between active/idle operational modes; and (*ii*) techniques that allow the nodes to adjust their transmission power. Recent works in the first category can be found in [34,9,30]. Topology control strategies fall in the second category and have been largely explored in the literature [22,8]. Topology control consists in allowing wireless nodes to select a subset of neighbouring nodes and/or adjust the transmission power with the objective of reducing energy consumption while maintaining network connectivity [7,3,23].

In traditional multiple-hop wireless networks, intermediate nodes cooperate with each other to assist in the task of relying data packets from a source node to the desired destination. Note that this process occurs at the network layer. Cooperative communication (CC), on the other hand, is a physical layer technique that allows single antenna devices to benefit from some advantages of Multiple-Input Multiple-Output (MIMO) systems by exploring the spatial diversity [28]. This technique allows nodes to improve signal quality and transmission range. In CC, when a source node transmits a packet, a set of *helper nodes* in the vicinity of the source overhear the signal and, simultaneously, relay independent copies of the same signal to the destination node. The destination node then combines the received signals to obtain the original packet.

Recent works have explored CC with topology control techniques to reduce energy consumption [5]. Ways to increase network connectivity and improve network lifetime has been investigated [35,33]. However, to the best of our knowledge, no work so far explored CC to increase the connectivity to the sink in ad hoc wireless networks. Link failure due to battery depletion and node failure may prevent wireless nodes to reach the sink. In this context, CC can be explored to improve network connectivity and to allow the establishment of alternative routes to the sink node.

This work presents a new technique, named CoopSink, that combines CC and topology control in an ad hoc wireless network to increase connectivity to the sink while ensuring energy efficient routes. This proposal could be applied to the environment described in [4], where there is a sink node equipped with a large range radio for query broadcast and the nodes cooperate to overcome link failures and report information to the sink. This scenario is similar to that found in the Amazon Tall Tower Observatory (ATTO) project, where the objective is to position a high central tower in the middle of the Amazon forest and, with the help of smaller and strategically placed sensors, to obtain reliable estimates of sources of greenhouse gases like CO_2, CH_4 and N_20 [24]. The proposed technique has been evaluated through simulation and the results have confirmed that CoopSink is able to improve network connectivity and provide energy-efficient routes to the sink. More precisely, the simulation results show that connectivity and routing to the sink improved up to 6.8 and 2.3 times, respectively, as compared with other similar strategies.

The remaining of this paper is organised as following. Section 2 makes an overview of related works on topology control and cooperative communication. Section 3 describes the communication and network models and formalises the main problem addressed in this work. Section 4 describes the CoopSink protocol and Section 5 presents the simulation results that compares the proposed scheme with other similar and prominent strategies. Section 6 concludes the work.

2 Related Works

This section presents a brief review of the closely related works that explore topology control and cooperative communication in wireless networks.

2.1 Topology Control

Topology Control is a technique that alters the network topology based on some given conditions. For instance, topology control can be used to optimize network power consumption, reduce routing cost and the number of control messages, improve throughput or meet certain QoS requirements [5,15,11].

According to [5], topology control protocols can be classified as: (*i*) centralised; or (*ii*) distributed. Centralised protocols consider that global information is available, such as topological information, routing information, global memory status, and so on. However, even when global information is at hand, it has been proven that finding strongly connected topologies with minimum total energy consumption is a NP-complete problem [6]. Among centralised protocols, Ramanathan et al. [27] proposed alternatives to optimize network connectivity while improving network lifetime. Distributed protocols consider k-hop neighbouring information, where k is typically one or two. Li et al. [20] propose a cone-based algorithm for TC that aims to optimize energy consumption while maintaining network connectivity. To achieve this, each node adjusts its transmitting power to cover a number of neighbouring nodes, under the condition that they lay at most α degrees apart from each other. The authors show that a degree of $\alpha = \frac{5\pi}{6}$ is enough to preserve network connectivity. Several optimized solutions of the basic algorithm are also discussed and a beacon-based protocol is defined for topology maintenance. In a more recent work, Li et al. [21] proposed a Localised Minimum Spanning Tree (LMST) algorithm. The LMST works by having each node building a localised MST based on 1-hop neighbouring information. The final topology is constructed so that the maximum node degree is 6. Comprehensive surveys can be found in [22,14].

2.2 Cooperative Communication (CC)

Cooperative communication (CC) is a technique introduced by [19] and [26] that allow single antenna devices to explore characteristics of MIMO systems. In cooperative communication, a set of nodes transmit independent copies of the original signal. The intended receiver obtain independent versions of the transmitted signal which reduces the fading effect through multi-path propagation. In this communication model, each wireless node is assumed to transmit data and to act as a cooperative agent, relaying data from other users. CC was previously used in energy efficient broadcasting [1], constructing connected dominating sets [32], routing [17], among others applications.

CC techniques can be classified as *amplify-and-forward* and *decode-and-forward* [19]. In the former, a node that receives a noise version of the signal, amplify and relay this noisy version. The receiver then combines the information sent by both sender and relay nodes. When decode-and-forward is employed, a relay node must first decode the signal before retransmitting it. As the cost of a CC-link area usually higher than conventional links, ways to select suitable nodes is usually employed. Among the techniques used to identify the best set of relay nodes are received Signal-to-Noise Ratio (SNR) and/or remaining battery energy.

This work considers cooperative communication employing *decode-and-forward* approach where the relay nodes are selected based on the SNR of the received signal. This technique requires each node to have a dedicated memory to store data packet and a signal processor that can estimate the SNR of each received packet as in [19].

2.3 Topology Control in Cooperative Ad Hoc Networks to Extend the Link Coverage

Few works in the literature have considered the use of topology control and CC to improve network coverage in ad hoc networks. The possibility of link coverage extension using CC was investigated in [18,31]. In this context, Cardei et al. [5] studied the use of topology control combined with CC with the objective of obtaining strongly connected topologies with minimum energy consumption. The authors showed that this problem is NP-complete and two localised and distributed algorithms were proposed. Both algorithms take as input the result of a traditional topology control algorithm (without CC). The first algorithm uses a distributed decision process where each node uses information of neighbours with at most two hops. The second algorithm iteratively assigns transmission power to the nodes, using one hop neighbour information.

Yu et al. [33] proposed a centralised topology control scheme aiming to increase network connectivity and reduce transmission power. To minimise the number of cooperative communication links (CC-links) and to reduce transmission cost, a polynomial and an exponential (but optimal) helper decision algorithms were proposed. Zhu et al. [35] consider the problem of selecting energy efficient paths when CC-links are used. The authors propose two topology control algorithms to build cooperative energy spanners in which the energy efficiency of individual paths are guaranteed. Both algorithms can be executed in a distributed or localised fashion. The work in [33] focus on maintaining network connectivity with the objective of minimising global energy consumption. Similarly, the work in [35] focused on reducing energy consumption by selecting efficient routes.

In ad hoc settings, link failure due to battery depletion and node failure may prevent wireless nodes to reach the desired destination. In this context, CC can be explored to improve network connectivity and to allow the establishment of more efficient routes. This work addresses this problem. To the best of our knowledge, this is the first work to explore the use of CC-links to improve network connectivity to a sink node in a wireless ad hoc networks.

3 Network Model and Problem Definition

This section first describes the CC model and the corresponding network model that are used in this work. In a second moment, the model is exemplified and the main problem of this work is formalised. The model defined in this section is similar to that used in [5,33,35].

3.1 CC model

Consider a wireless ad hoc network where each node v_i can adjust its transmission power P_i with values within the interval $[0, P_{MAX}]$. When $P_i = 0$, the node's radio is off and, when $P_i = P_{MAX}$, the node's radio operates with maximum power. In traditional cooperative communication models, a sender node v_i can directly communicate with a receiver node v_j only if the transmission power of v_i satisfies Equation 1.

$$P_i(d_{i,j})^{-\alpha} \geq \tau \qquad (0 \leq P_i \leq P_{MAX}), \tag{1}$$

where: α is the path loss exponent, usually between 2 and 4, and represents the rate of signal fading with increasing distance; $d_{i,j}$ is the Euclidian distance between v_i and v_j; and τ is the minimum Signal-to-Noise Ratio (SNR) received by v_j, so that v_j can decode the signal and obtain the original message.

CC takes advance of the physical layer design to combine partial signals to obtain complete information [28]. This way, a communication between the nodes v_i and v_j can be achieved with CC if v_i transmits its signal jointly with a set of *helper* nodes $H_{i,j}$ and the sum of its transmission power satisfies Equation 2. In CC, a helper node is a node that cooperatively retransmit the signal along with the transmitting node.

$$\sum_{v_k \in v_i \cup H_{i,j}} P_k(d_{k,j})^{-\alpha} \geq \tau \qquad (0 \leq P_i \leq P_{MAX}) \tag{2}$$

Figures 1a and 1b exemplify a scenario where CC could be used to increase connectivity in an ad hoc network. In Figure 1a, there are three nodes (v_1, v_2 e v_3), which are close to each other, and one distant node (v_4). The transmission radius, that is, the transmission power of v_1 allows it to reach nodes v_2 and v_3 directly. Node v_2 and v_3 have a single neighbouring node, in this case, node v_1. Node v_4 is outside radio range of the other nodes.

When CC is employed, node v_1 could select the nodes v_2 and v_3 as its helpers to transmit to v_4, that is, $H_{1,4} = \{v_2, v_3\}$. After selecting these nodes as helpers, v_1, in a first moment, transmit its data to v_2 and v_3. In a second moment, v_1 and its helpers transmit the same data to v_4, amplifying the v_1 transmission radius. If the combined SNR received in v_4 is greater than τ, the node is able to decode the signal and obtain the original data from v_1, as illustrated in Figure 1b. When node v_1 transmits its data for its helper nodes v_2 and v_3, in a first moment, v_4 also receives partial data from v_1. In some CC models, such partial data is used by v_4 in the signal decoding process, during the second moment of the CC. In this work, such partial data is ignored for simplicity (as in [33] and [35]). Physical layer techniques to implement CC can be find in [13].

3.2 Network model

Consider an ad hoc network with n nodes that are capable to receive and combine partial, received data, in agreement with the CC model. The network topology is modelled as a planar graph $G = (V, E)$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of wireless nodes and E is the set of communication edges. An edge $v_iv_j \in E$ symbolises that node v_i can transmit data to v_j directly and/or using CC. $N(v_i)$ is the direct neighbour set of v_i within its maximum transmission range R_{MAX}, for every $v_k \in$

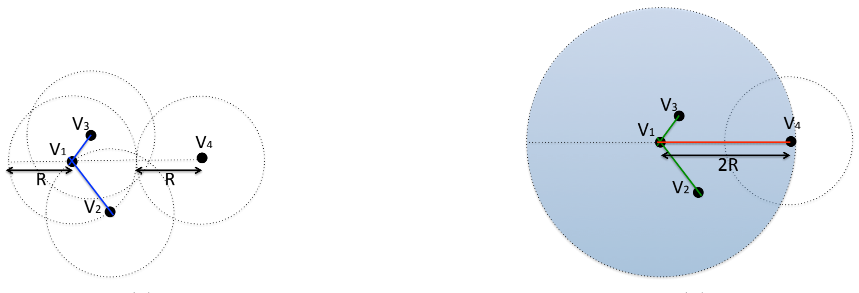

Fig. 1. (a) Scenario with three nodes (v_1, v_2 e v_3) within radio reach of each other and one distant node (v_4). (b) Node v_1 uses nodes v_2 and v_3 as helper nodes to increase radio range thus reaching node v_4.

$N(v_i)$, there is $P_i \leq P_{MAX}$ such that $P_i(d_{i,k})^{-\alpha} \geq \tau$, following Equation 1. In other words, v_i can directly communicate with its neighbours in $N(v_i)$. Each node $v_i \in V$ has an unique ID and knows its own location information. Node IDs and location information are exchanged among the nodes. Each node $v_i \in V$ has an unique radio and runs on battery power. Given the previous information, we define several important concepts, similar to those in [35].

Definition 1 *(Direct edge): A direct edge $\overline{v_iv_j}$ is an edge in E representing that node v_i can transmit data to node v_j directly, that is, P_i is such that v_i can achieve v_j when $P_i \leq P_{MAX}$. A solid horizontal line over the nodes denote a direct edge.*

Definition 2 *(Helper node set): $H_{i,j}$ symbolises the set of helper nodes of v_i in a co-operative communication with v_j. It is assumed that all helper nodes need to be direct neighbours of v_i, that is, $H_{i,j} \subseteq N(v_i)$, where $N(v_i)$ is the set of all direct neighbours of v_i. In other words, all the elements in $N(v_i)$ are helper nodes candidates.*

Definition 3 *(CC edge): A CC edge $\widetilde{v_iv_j}$ is an edge of E that represents that node v_i can transmit data to v_j cooperatively by using a set of helper nodes $H_{i,j}$. A wavy horizontal line is used to denote a CC edge.*

Definition 4 *(Helper edge): A helper edge is an edge from v_i to one of its helper nodes in $H_{i,j}$. For example, in Figure 1b, node v_1 uses the nodes v_2 and v_3 as helper nodes to create a CC edge between v_1 and v_4, that is, $H_{1,4} = \{v_2, v_3\}$, this way, the edges $\overline{v_1v_2}$ and $\overline{v_1v_3}$ are considered helper edges.*

Definition 5 *(Network topology): The union of all direct edges and CC edges are $\overline{E}$ and $\widetilde{E}$, respectively. Similarly, the direct communication graph and the CC communication graph are denoted as $\overline{G} = (V, \overline{E})$ and $\widetilde{G} = (V, \widetilde{E})$, respectively. Note that $E = \overline{E} \bigcup \widetilde{E}$. Following the notation, note that, if $v_iv_j \in E$, then: $v_iv_j = \overline{v_iv_j}$ if v_iv_j is a direct edge; and $v_iv_j = \widetilde{v_iv_j}$ if v_iv_j is a CC edge.*

Definition 6 *(Direct edge weight): The weight of a direct edge $\overline{v_iv_j}$ is defined as:*

$$w(\overline{v_iv_j}) = \tau d_{i,j}^{\alpha}.$$

Definition 7 *(CC edge weight): The weight of a CC edge $\widetilde{v_iv_j}$ is defined as:*

$$w(\widetilde{v_iv_j}) = w_d(H_{i,j}) + (|H_{i,j}| + 1)w_{CC}(H_{i,j}),$$

where:

- $|H_{i,j}|$: *is the number of elements in* $H_{i,j}$*;*
- $w_d(H_{i,j}) = \left(\frac{\tau}{(\max_{v_k \in H_{i,j}} d_{i,k})^{-\alpha}}\right)$*: is the minimum energy consumption of node* v_i *to communicate with the most distant node in* $H_{i,j}$*;*
- $w_{CC}(H_{i,j}) = \left(\frac{\tau}{\sum_{v_k \in v_i \bigcup H_{i,j}} (d_{k,j})^{-\alpha}}\right)$*: is the minimum energy consumption of node* v_i *to communicate with* v_j*, jointly transmitting with its helpers in* $H_{i,j}$

Note that, according to Equations 1 and 2, the following relation must be true to exist an CC edge:

$$\max\left(w_d(H_{i,j}), w_{CC}(H_{i,j})\right) \leq P_{MAX}$$

In a CC from v_i to v_j, the node v_i must, in a first moment, send its data to its helper nodes in $H_{i,j}$ and, in a second moment, node v_i and its helpers must simultaneously send the same data to v_j. This way, the weight of the CC edge consists in the sum of the communication costs of these two moments. $w_d(H_{i,j})$ is the cost of the first moment while $w_{CC}(H_{i,j})$ is the individual node cost to transmit a data with CC, that is, it must be multiplied by $(|H_{i,j}|+1)$, that is, the number of nodes that are involved in the CC between v_i and v_j. In this work, the CC model is simplified assuming that the transmission power of v_i and its helper nodes are the same. Furthermore, it is only considered the power consumption in each sender node.

Definition 8 *(Directional path cost): Given a source node* v_i *and a destination node* v_j *in a graph* $G = (V, E)$*, there is a directional path between* v_i *and* v_j *if, and only if, there is a sequence of vertex:*

$$v_i, v_{i_1}, v_{i_2}, ..., v_{i_{k-1}}, v_{i_k}, v_j \in V,$$

such that:

$$(v_i v_{i_1}), (v_{i_1} v_{i_2}), ..., (v_{i_{k-1}} v_{i_k}), (v_{i_k} v_j) \in E.$$

The directional path cost between v_i *and* v_j *in a graph* G *is defined as:*

$$\pi_G(v_i, v_j) =$$

$$w(v_i v_{i_1}) + w(v_{i_1} v_{i_2}) + ... + w(v_{i_{k-1}} v_{i_k}) + w(v_{i_k} v_j).$$

That is, the sum of the weight of all the edges, both direct edges and CC edges, that belong to the path between v_i *and* v_j*. The cost of the shortest path in* G *between* v_i *and* v_j *is defined as:*

$$\min(\pi_G(v_i, v_j)).$$

Definition 9 *(ESF - Energy Stretch Factor): Let* $G' = (V, E')$ *be a subgraph of* $G = (V, E)$*,* $E' \subseteq E$*,* G' *e* G *are connected graphs. The ESF of a pair of nodes* v_i *and* v_j *in* G' *with respect to the same nodes in* G *is defined as:*

$$\rho_G^{G'}(v_i, v_j) = \frac{\min(\pi_{G'}(v_i, v_j))}{\min(\pi_G(v_i, v_j))},$$

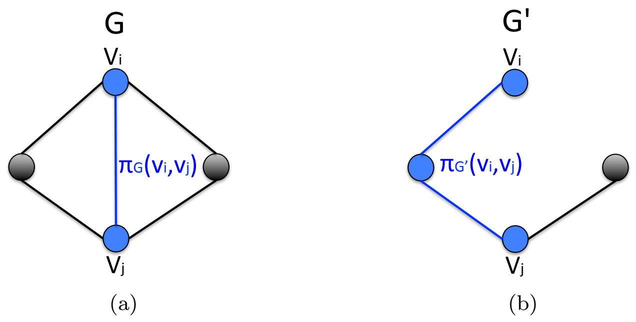

Fig. 2. (a) Example of graph $G = (V, E)$. (b) Example of graph $G' = (V, E')$, $E' \subseteq E$. $\pi_G(v_i, v_j)$ is the path cost from v_i to v_j in graph G. The ESF of a pair of nodes $v_i, v_j \in V$ of G' with respect to G is equal to $\rho_G^{G'}(v_i, v_j) = \frac{\min(\pi_{G'}(v_i,v_j))}{\min(\pi_G(v_i,v_j))}$.

Consider the ESF of G' with respect to G as:

$$\rho_G^{G'} = \max_{v_i, v_j \in V} \rho_G^{G'}(v_i, v_j).$$

That is, the greatest ESF between any pair of nodes in V in G' with respect to G. Figure 2 illustrates this concept.

Definition 10 *(CE-t-S - Cooperative Energy t-Spanner): Let $G' = (V, E')$ be a subgraph of $G = (V, E)$, $E' \subseteq E$, G and G' are connected graphs. G' is a CE-t-S with respect to G if its ESF is less than a constant t, that is:*

$$\rho_G^{G'} \leq t.$$

Definition 11 *(COE-t-S - Cooperative Oriented Energy t-Spanner): COE-t-S is a specific case of the CE-t-S. Let $G' = (V, E')$ be a subgraph of $G = (V, E)$, $E' \subseteq E$, G' and G are not necessarily connected (unlike the COE-t-S). Consider a particular node $v_o \in V$ and constant t. G' is a COE-t-S with respect to G and v_o if:*

$$\max_{v_i \in V} \mu(v_i, v_o) \leq t,$$

where:

$$\mu(v_i, v_o) = \begin{cases} \max \rho_{G_P}^{G'_P}(v_i, v_o), & \textit{if there is a path from } v_i \textit{ to } v_o; \\ 0, & \textit{otherwise.} \end{cases}$$

3.3 Computing the CC cost

To exemplify the described model, consider the graph represented by Figure 1a. This is a direct graph $\overline{G} = (V, \overline{E})$, where $V = \{v_1, v_2, v_3, v_4\}$ and $\overline{E} = \{\overline{v_1v_2}, \overline{v_1v_3}, \overline{v_2v_1}, \overline{v_3v_1}\}$. Consider, just for exemplification, that $\alpha = 1$, $\tau = 1$ and that $R_{max} = R$. Note that $\alpha = 1$ is not practical in real situations, but it makes the calculation easier in an example. Thus, we can compute the maximum transmission power of each node $v_i \in V$ based on Equation 1:

$$P_{MAX}(d_{MAX})^{-\alpha} = \tau,$$
$$P_{MAX}(R)^{-1} = 1,$$
$$P_{MAX} = R.$$

Note that each node $v_i \in V$ has a maximum transmission power of R with the provided settings. In $\overline{G}$, nodes v_2 and v_3 are neighbours of node v_1, that is, $N(v_1) = \{v_2, v_3\}$. This way, node v_1 could select any non-empty subset of $N(v_1)$ as its helper nodes to create a CC edge from v_1 to v_4, that is $H_{1,4} \subseteq N(v_1)$, following Definition 2. The new graph with the CC edge that comes from v_1 to v_4 is illustrated in Figure 1b.

Consider that, for the proposed example, $d_{12} = R$, $d_{13} = \frac{R}{2}$, $d_{14} = 2R$, $d_{24} = 1.73R$ and $d_{34} = 1.80$. From Equation 2, we can compute the transmission power that node v_1 needs to achieve node v_4 using nodes v_2 and v_3 as helper nodes:

$$\sum_{v_k \in v_1 \cup \{v_2, v_3\}} P_k(d_{k,4})^{-1} \geq 1,$$
$$= P_1(d_{14})^{-1} + P_2(d_{24})^{-1} + P_3(d_{3,4})^{-1} \geq 1,$$
$$= P_1 \geq \frac{1}{\frac{1}{2R} + \frac{1}{1{,}73R} + \frac{1}{1{,}80R}},$$
$$= P_1 \geq 0.612R.$$

Note that all the involved nodes (v_1, v_2 e v_3) operate with the same transmission power, that is, $P_1 = P_2 = P_3$. Thus, the minimum transmission power for v_1 to achieve v_4 with CC is equal to $P_1 = 0.612R$. This transmission power is bellow the previous seen maximum transmission power that is equal to $P_{MAX} = R$. To compute the CC edge weight for the edge $\widetilde{v_1v_4}$ we use Definition 7:

$$w(\widetilde{v_1v_4}) = \frac{1}{R^{-1}} + \frac{(|H_{1,4}| + 1)}{\sum_{v_k \in v_1 \bigcup \{v_2, v_3\}} (d_{k4})^{-1}}$$
$$= R + (|H_{1,4}| + 1) * 0.612R,$$
$$= 2.836R.$$

Note that the weight of the CC edge $\widetilde{v_1v_4}$, using nodes v_2 and v_3 as helpers, is equal to $w(\widetilde{v_1v_4}) = 2.836R$. Likewise the previous example, we use Equation 2 to compute the transmission power that node v_1 needs to achieve v_4 using just node v_2 as a helper node and using just node v_3 as a helper node. Table 1 summarises all the calculus made for this example.

Analysing the $w(\widetilde{v_1v_4})$ column in table, one can verify that the weight of the CC edge $\widetilde{v_1v_4}$ is lower using just node v_3 as a helper node. This happens because, as can be observed from Definition 7, the weight of a CC edge is the sum of the cost to send a data packet from the source node to its helpers and the cost to transmit collaboratively with its helpers. As, in the example, node v_2 is farther away from from v_1 than v_3, respectively distances of R and $\frac{R}{2}$. Hence, node v_1 spends more power to reach node v_2 than to reach v_3. So, using just node v_3 as its helper, allows for grether battery savings. The problem of efficient helper set selection in CC is a challenging issue and has been addressed in other works [12,29].

Table 1
Weight of the CC edge $\widetilde{v_1v_4}$ (graph in Figure 1b) for different helper node sets $H_{1,4}$.

$H_{1,4}$	P_1	$w(\widetilde{v_1v_4})$
$\{v_2, v_3\}$	$0.612R$	$2.836R$
$\{v_2\}$	$0.928R$	$2.855R$
$\{v_3\}$	$0.947R$	$2.395R$

3.4 Problem formulation

Consider an ad hoc network, where the nodes are scattered in a fixed area and there is an unique sink in the network border. The sink node is responsible for collecting and requesting information from the other nodes. The nodes are fixed, that is, no mobility is assumed, and the underline network graph can get disjoint due to link failures, battery depletion and so on. This work's proposal consists in exploring CC communication to improve network connectivity while keeping energy expenditure as low as possible. Using the defined notation, given a network topology $\overline{G} = (V, \overline{E})$, where $v_o \in V$ is a sink node, the goal of this work is to propose a technique that uses CC to create a graph G and a subgraph G', $G' \subseteq G$, such that G' is a COE-t-S with respect to G and the node v_o, following the Definition 11. In the following section, this work's proposal is described.

4 Proposal

In this section, it is described the *CoopSink* (short for Cooperative Sink), a topology control technique for cooperative ad hoc networks. This technique aims to improve node connectivity to the sink while the energy consumption of the routes to the sink are minimised. In a first moment, a greedy heuristic for the helper set selection problem is described, then the CoopSink technique is described as a sequence of four steps. The formulation and algorithms of this section are based on the definitions of Section 3 and it is assumed a central computing unit.

4.1 Greedy Helper Set Selection Algorithm (GHSS)

Algorithm 1 describes a greedy heuristic, proposed by [33] and adapted for this work. This heuristic is used to select the most efficient helper nodes for CC and can be used in both directional and bi-directional graphs. The algorithm's inputs are: a transmitter node v_i; a set $N(v_i)$ of v_i's neighbours; and a receiver node v_j. The output is the set of helper nodes $H_{i,j}$, such that the weigh of the CC-links $\widetilde{v_iv_j}$ is minimised (although optimal solutions are not guaranteed). Consider the function prototype $GreedyHelperSetSelection(v_i, N(v_i), v_j)$ to refer to Algorithm 1. In the following, the algorithm and the gaining function is described.

4.1.1 Algorithm general description

Algorithm 1 has three main steps:

- **Step 1 (lines 1-11)**: this step consists on sorting the neighbour nodes from v_i ($N(v_i)$) according to a gain heuristic. The greater the gain that a node in $N(v_i)$ brings, according to the heuristic, up ahead it will be positioned in the vector B. The following functions are used in this step: the function $head(X)$ returns the first element from a vector or a set X; the function $remove(X, Y)$ removes the element Y from the set X; the function $SortDescending(X)$ receives the vector X and returns the same vector but sorted in descending order; while function $indexTerms(X)$, where $X = [\frac{b_{k_1}}{c_{k_1}}, \frac{b_{k_2}}{c_{k_2}}, ..., \frac{b_{k_{|N(v_i)|}}}{c_{k_{|N(v_i)|}}}]$, returns the vector $Y = [v_{k_1}, v_{k_2}, ..., v_{k_{|N(v_i)|}}]$. The gain function is described in section 4.1.2.
- **Step 2 (lines 12-21)**: This step consists on adding the elements from vector C to the helper set $H_{i,j}$ so that v_i and its helpers should have the minimum transmission power to create the CC-link $\widetilde{v_i v_j}$, according to Equation 2. If, after adding all the elements in C, the coupled power from these nodes is not enough to create the CC-link $\widetilde{v_i v_j}$, an error message is returned (lines 16-18);
- **Step 3 (lines 22-31)**: This step consists on adding the maximum number of helper nodes to the set $H_{i,j}$ so that the inclusion of the next node does not increase the CC-link weight. In the case of the inclusion of a CC-link increases the CC-link weight or all the elements in C were already tested, the function returns the set $H_{i,j}$ (lines 23-24). The function $WeightCC(v_i v_j, \Omega)$, where $v_i v_j \in E_t$ and $\Omega \subseteq V$, returns the CC-link weight of the link $v_i v_j$ using the nodes in Ω as helper nodes.

4.1.2 Gain function

The gain function in Algorithm 1 (Step 1) is used to sort the nodes in $N(v_i)$ in order of greater gain in terms of energy consumption to create a CC-link $\widetilde{v_i v_j}$. Consider, for each node $v_k \in N(v_i)$, the following values:

- $b_k \leftarrow \frac{\tau}{(d_{i,j})^{-\alpha}} - \frac{\tau}{\sum_{v_l \in \{v_i, v_k\}} (d_{l,j})^{-\alpha}}$: the amount of power that node v_i can save if it adds node v_k as a helper. Note that the first element in the subtraction is the cost for v_i directly communicate with v_j and the second element of the subtraction is the power for v_i cooperatively communicate with v_j using node v_k as a helper;
- $c_k = \frac{\tau}{(d_{i,k})^{-\alpha}}$: The energy cost for v_i to directly communicate with the helper v_k.

As the objective here is to reduce the weight of the CC-link, the values for b_k, that represents the gain, should be maximised while the values of c_k, that represents the cost to communicate with the helper node, should be minimised. This way, the gain function considers the ratio $\frac{b_k}{c_k}$ as a metric that indicates which node brings greater gain, that is, if $\frac{b_k}{c_k}$ is maximum for $k = h$, so the inclusion of the helper node v_h will bring a greater gain than other helper nodes.

4.2 CoopSink: Proposal Description

This section describes the steps of CoopSink technique. The CoopSink consists on the execution in sequence of the following four steps:

- **Step 1**: Create a topology graph, where each node creates as much edges as its maximum transmission power allows;

Algorithm 1 : $GreedyHelperSetSelection(v_i, N(v_i), v_j)$

```
Require: $v_i, N(v_i), v_j$
Ensure: $H_{i,j}$;
1:  # (Step 1)
2:  Set: $A \leftarrow N(v_i)$;
3:  vector: $B \leftarrow []$, $C \leftarrow []$;
4:  while $(v_k \leftarrow head(A)) \neq \emptyset$ do
5:      $b_k \leftarrow \frac{\tau}{(d_{i,j})^{-\alpha}} - \frac{\tau}{\sum_{v_l \in \{v_i, v_k\}} (d_{l,j})^{-\alpha}}$;
6:      $c_k = \frac{\tau}{(d_{i,k})^{-\alpha}}$;
7:      $B \leftarrow [B, \frac{b_k}{c_k}]$;
8:      $remove(A, v_k)$;
9:  end while
10: $B \leftarrow SortDescending(B)$;
11: $C \leftarrow indexTerms(B)$
12: # (Step 2)
13: $k \leftarrow 0$, $\Omega \leftarrow v_i$;
14: while $(\sum_{v_k \in \Omega} P_{MAX}(d_{k,j})^{-\alpha}) < \tau$ do
15:     $k \leftarrow k + 1$;
16:     if $k > |C|$ then
17:         return error;
18:     end if
19:     $H_{i,j} \leftarrow H_{i,j} \cup A[k]$;
20:     $\Omega \leftarrow \Omega \cup H_{i,j}$
21: end while
22: # (Step 3)
23: while $(k \leq |N(v_i)|)$ do
24:     if $(k = |N(v_i)|)$ or
        $(WeightCC(\widetilde{v_i v_j}, \Omega) < WeightCC(\widetilde{v_i v_j}, \Omega \cup C[k+1]))$ then
25:         return $H_{i,j}$;
26:     else
27:         $k \leftarrow k + 1$;
28:         $H_{i,j} \leftarrow H_{i,j} \cup C[k]$;
29:         $\Omega \leftarrow \Omega \cup H_{i,j}$;
30:     end if
31: end while
32: return $H_{i,j}$;
```

- **Step 2**: Use CC to create as much CC-links as possible in the network;
- **Step 3** From previous step's graph, use topology control to cut edges such that energy efficient routes to the sink are kept;
- **Step 4**: Adjust the transmission power of the nodes, to minimise the energy consumptions but maintaining the network connectivity.

The following sub-sections better describe each of previous steps.

4.2.1 Step 1: Construction of graph $\overline{G_P}$

Algorithm 2 describes the CoopSink's first step. This step consists on creating all the possible direct edges in a graph topology, since all nodes operates with its maximum transmission power P_{MAX}. Algorithm 2 takes as input: the node set $V = \{v_1, v_2, ..., v_n\}$ and its information in the map; the maximum transmission power P_{MAX}. As output, we have the direct graph $\overline{G_P}$. Figures 3a and 3b show an input graph and the computed $\overline{G}$ graph. In Figure 3a we have a 500x500m area with $n = 50$ nodes no edges, that is, $\overline{E} = \emptyset$. Figure 3b illustrates the resulting graph when nodes create edges based on its maximum transmission power. In this example, $P_{MAX} = 4900$ and $R_{MAX} = 70$. Note that the resulting graph is not necessarily connected.

4.2.2 Step 2: Construction of graphs $\widetilde{G}$ and G

Algorithm 3 describes the CoopSink's second step. This step consists on creating all the possible CC edges taking as input the graph $\overline{G} = (V, \overline{E})$ from the previous step and returning the graph G, which has both direct and CC edges. As described in Subsection 2.3, the efficient helper set selection in CC is a challenging problem, since it is computationally costly. Therefore, heuristics can be used to handle this problem. In this work we consider the greedy heuristic proposed in [33]. This heuristic, which has the $GreedyHelperSetSelection(v_i, v_j)$ prototype, receive as input: the source node v_i; and the destination node v_j. The output is the helper set $H_{i,j}$. As an example, consider the graphs represented in Figures 3b and 3c. The graph in Figure 3b is the output graph from Step 1 and the input graph of Step 2. Graph in Figure 3c is the output from Step 2. In these graphs, direct edges $\overline{v_i v_j} \in E$ are shown in blue, CC edges $\widetilde{v_i v_j} \in E$ are shown in red and helper edges $H_{i,j}$ are shown in green. Note that, in this example, a large number of CC edges were created, however, these are direct edges, that is, the existence of $\widetilde{v_i v_j} \in E$ does not imply $\widetilde{v_j v_i} \in E$.

4.3 Step 3: Removing edges from G

Algorithm 4 describes the CoopSink's third step. This step consists on creating a graph G' such that the graph is a COE-t-S with respect to G and a node $v_o \in V$. It receives as input: a graph G; a constant t; and a node $v_o \in V$. It returns a graph G'. Figures 3c and 3d illustrate what happens in this step. The graph in Figure 3c is a graph with several direct and CC edges and is one of the inputs of Step 3. The other inputs are, in this example, $t = 1.4$ and v_o on location $(0, 0)$ in the cartesian plane. The resulting graph from Step 3, illustrated in Figure 3d, is a COE-t-S with respect to the graph in Figure 3c. The resulting graph is a direct graph oriented to the node v_o, that is, all the connected nodes have direct routes to v_o. Note that previously unconnected nodes on Figure 3b now have a path to the sink node v_o. One may note that some nodes area still disconnected in the final stage. Figure 3d shows a node that failed in establishing CC links to its neighbouring nodes.

Algorithm 2 : $CoopSinkStep1(V, P_{MAX})$

```
Require: $V$ e $P_{MAX}$;
Ensure: $\overline{G}$;
1:  $\overline{G} = (V, \emptyset)$;
2:  for $(v_i, v_j \in V)$ do
3:    if $(P_{MAX}(d_{ij})^{-\alpha} \geq \tau)$ then
4:      Add $\overline{v_i v_j}$ to $\overline{G_P}$;
5:    end if
6:  end for
```

Algorithm 3 : $CoopSinkStep2(\overline{G})$

```
Require: $\overline{G}$;
Ensure: $G$;
1:  $\widetilde{G} = (V, \emptyset)$;
2:  for $(v_i, v_j \in V)$ do
3:    $H_{ij} \leftarrow GreedyHelperSetSelection(v_i, v_j)$;
4:    if $(w_{CC}(H_{i,j}) \leq P_{MAX})$ then
5:      Add $\widetilde{v_i v_j}$ to $\widetilde{G}$;
6:      if $(\overline{v_i v_j} \in \overline{G}$ and $w(\overline{v_i v_j}) > w(\widetilde{v_i v_j}))$ then
7:        Remove $\overline{v_i v_j}$ from $\overline{G}$;
8:      end if
9:    end if
10: end for
11: $G = \overline{G} + \widetilde{G}$;
```

4.3.1 Step 4: Adjusting the transmission power

Algorithm 5 describes the CoopSink's last step. This step consists on adjusting the transmission power of all the nodes in the input graph G' such that the COE-t-S property is maintained and the energy consumption is minimised.

In the following section, the CoopSink technique is compared with other techniques from the literature.

5 Simulation Results

In the previous section, it was proposed a new technique for topology control in cooperative ad hoc networks, which focus on increasing connectivity while maintaining efficient routes to the sink. In this section, it is used simulation to compare the proposal to others in the literature. The following techniques were chosen for comparison: CoopBridges [33] and MST-Kruskal [16]. CoopBridges starts from the direct graph $\overline{G}$ (output from CoopSink Step 1) and splits all connect components into clusters. Creates as much bi-direction CC edges as possible within clusters. Finally, uses a distributed MST algorithm to remove inter-clusters edges and intra-clusters latter. MST-Kruskal grows a minimal spanning tree (MST) one edge at a time by finding an edge that connects two trees in a forest of growing MSTs, receives as input the graph $\overline{G}$.

Initially, it was our intention to compare the proposal also with the Greedy-

Algorithm 4 : *CoopSinkStep*3(G, t, v_o)

```
Require: $G = (V, E)$;
Ensure: $G' = (V, E')$;
1: $G' \leftarrow G$;
2: $k \leftarrow 1$;
3: vector $A \leftarrow$ receives all the edges in $E$ ordered by weight;
4: while $(k \leq |B|)$ do
5:    $v_iv_j \leftarrow A[k]$;
6:    $G'_P \leftarrow G'_P - v_iv_j$;
7:    if $(\max_{v_i \in V} \mu(v_i, v_o) \leq t)$ then
8:       # The deletion of edge $v_iv_j$ will harm the ESF;
9:       $G'_P \leftarrow G'_P + v_iv_j$;
10:      if $(v_iv_j = \widetilde{v_iv_j})$ then
11:         Remove all the edges from $v_i$ to all the nodes in $H_{i,j}$ from B;
12:      end if
13:   end if
14:   $k \leftarrow k + 1$;
15: end while
```

Algorithm 5 : *CoopSinkStep*4(G')

```
Require: $G' = (V, E')$;
1: for $v_i \in V$ do
2:    $a \leftarrow \max_{\overline{v_iv_j} \in G'_P} w_d(H_{i,j})$;
3:    $b \leftarrow \max_{\widetilde{v_iv_j} \in G'_P} w_{CC}(H_{i,j})$;
4:    $P_i = \max\{a, b\}$;
5: end for
```

(Add/Del)-Link technique [35], however, this technique assumes that the topology created, after adding the CC edges, would be necessarily connected. This hypothesis is not considered in both CoopSink and CoopBridges. Here, a resulting topology is the topology created after the application of a topology control algorithm. To evaluate the CoopSink performance, it is considered the following metrics:

- M1 - Connectivity to the sink: The percentage of nodes that have a route to the sink in the resulting topology;
- M2 - Average transmission power: The average of transmission power assigned to each node in the resulting topology;
- M3: ESF to the sink: Consists on creating a subgraph $G' = (V, E')$ from $G = (V, E)$, $E' \subseteq E$, and to verify the value for the constant t when it is considered that G' is a COE-t-S with relation to G;
- M4: Number of CC edges: The number of CC edges in the resulting topology.

The reason to consider M1 is to evaluate the connectivity improvements that CoopSink can bring to the network. M2 is justified once minimising energy consumption is a basic concern in ad hoc networks. M3 is used to verify the goodness of the routes connecting to the sink when compared with another proposals. M4 enumerates the average number of CC edges in the resulting topology. M4 is also

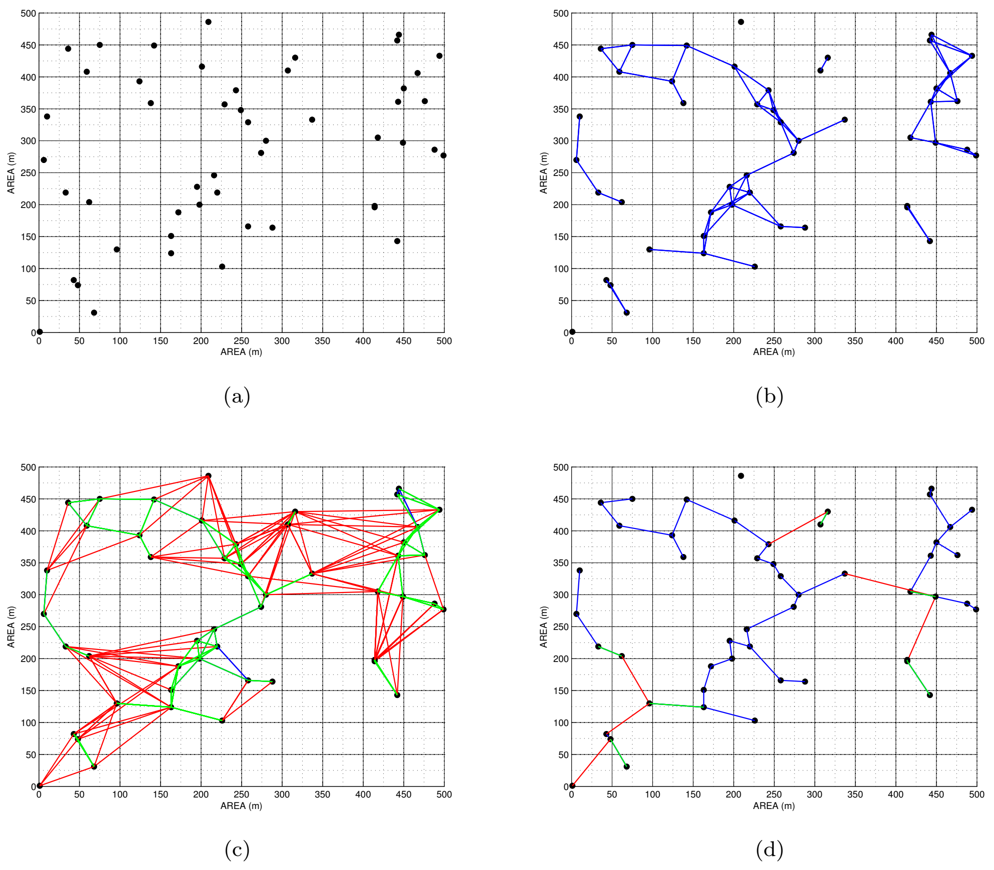

Fig. 3. (a) Network topology example with $n = 50$ nodes and $|E| = 0$. (b) Direct graph $\overline{G}$ created from (a) when nodes operate with maximum power P_{MAX}. (c) Graph G, created from the graph in (b), by adding all the possible CC edges. (d) COE-t-S created from (c).

useful to better understand the average energy cost, as CC edges consume more power than direct links.

In order to perform the evaluation, a simulation was developed in Matlab [10], following the steps described in Section 3 for CoopSink and the algorithm described in [33] for CoopBridges. The MST-Kruskal implementation is already present in Matlab. The CoopBridges implementation was validated with the original results. Similarly to other works [33,35], the simulation process takes the following parameters: $n = 10, 20, ..., 100$ nodes are randomly positioned in a 500x500m area; the sink node is always node v_1 at position $(0, 0)$; The PLE is equal to 2 $(\alpha = 2)$; $P_{MAX} = 4900$; SNR is equal to 1 $(\tau = 1)$; and $t = 1.0, 1.4, 1.8$ are the values for constant t. The simulation results have been drawn from an average of a 100 simulations. The compared proposals are: CoopBridges, MST, CoopSink-1.0, CoopSink-1.4 and CoopSink-1.8, where the last three are the CoopSink technique with t assuming values of 1.0, 1.4 and 1.8, respectively. The following subsections present and analyse the simulation results for each metric considered.

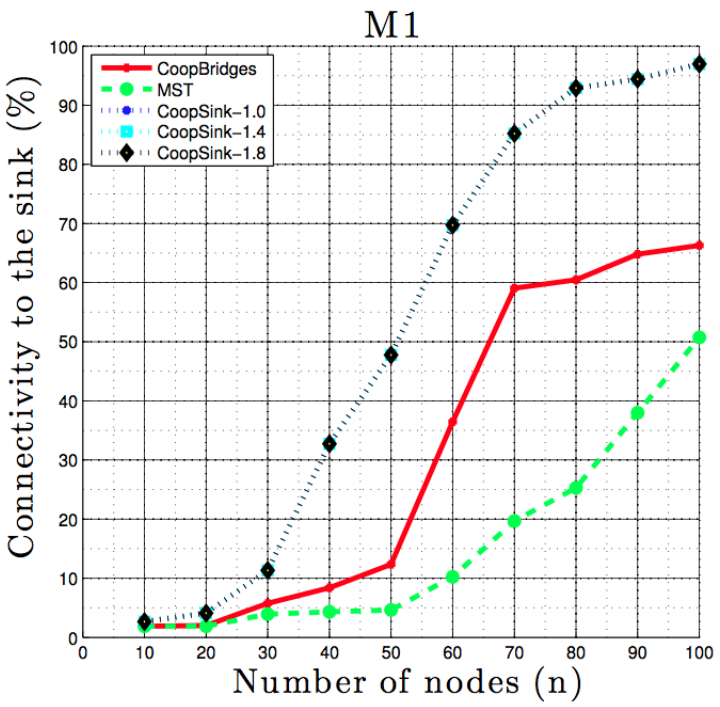

Fig. 4. Simulation results for M1.

5.1 M1: Connectivity to the sink

Figure 4 presents the simulation results for metric M1. The x-axis shows the number of network nodes and the y-axis shows the percentage of the nodes that have a route to the sink. The figure shows that the parameter t had no impact on the CoopSink connectivity to the sink. As can be seen in the figure, CoopSink allows for greater connectivity than CoopBridges and MST. Indeed, with 80 nodes, the connectivity to the sink using CoopSink exceeds 90%. The CoopSink's connectivity to the sink was up to 3.8 times better than CoopBridges and up to 6.8 times better than MST.

5.2 M2: Average transmission power

Figure 5 presents the simulation results for metric M2. The x-axis shows the number of network nodes and the y-axis shows the average transmission power for each node. As the MST does not create CC edges, the energy consumption in this technique is lower that that in CoopSink and CoopBridges. For this reason, the average transmission power of the nodes in CoopSink tend to be greater. However, using CoopSink-1.4 and CoopSink-1.8, the energy consumption is not very far from CoopBridges (up to 30% in the best case scenario). CoopSink, on the other hand, provides better connectivity. For smaller t values, the energy consumption tends to increase as the freedom to remove edges from the graph reduces.

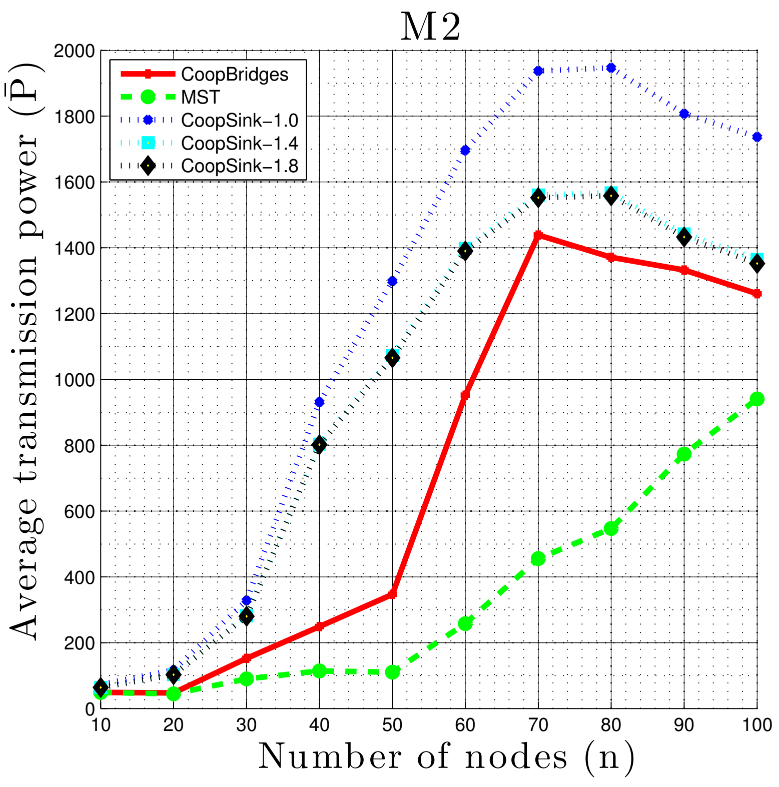

Fig. 5. Simulation results for M2.

5.3 M3: ESF to the sink

Figure 6 presents the simulation results for metric M3. As before, the x-axis shows the number of nodes while the y-axis presented the ESF values to the sink for each of the simulated techniques. As the ESF is measured comparing two graphs, the following graphs are used in the comparison:

- CoopSink-(1.0/1.4/1.8): Graph G' (output of CoopSink's Step 3) with graph G (output of CoopSink's Step 2);
- CoopBridges: Resulting graph from CoopBridges technique with the graph G (output of CoopSink's Step 2);
- MST: Resulting graph from MST with graph $\overline{G}$ (output of CoopSink's Step 1).

The results show that:

- The CoopSink variations obey the ESF to the sink, established during configuration. This means that the observed values for the ESF to the sink must be smaller or equal to 1.0 for CoopSink-1.0, smaller or equal to 1.4 for CoopSink-1.4 and smaller or equal to 1.8 for CoopSink-1.8.
- Both CoopBridges and MST have no commitment with the ESF to the sink. This mean that some of the routes to the sink can be quite inefficient in terms of energy consumption;
- CoopBridges obtained a ESF to the sink up to 2.34 times greater than CoopSink, while the MST obtained a ESF to the sink up to 2.29 greater. The greater the

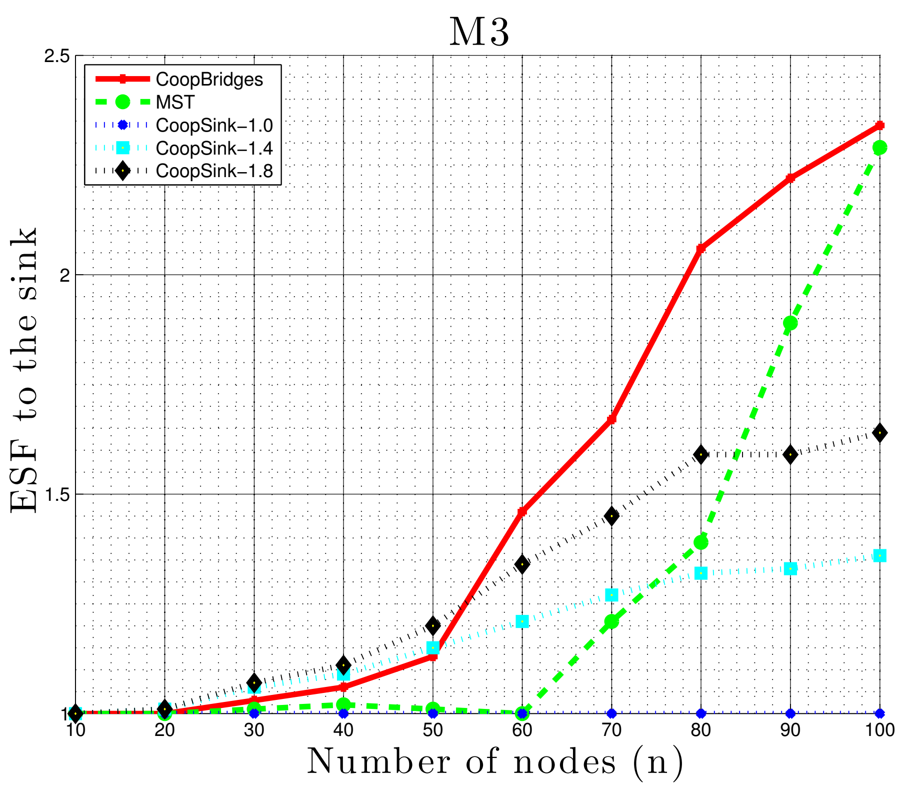

Fig. 6. Simulation results for M3.

ESF difference is, the less efficient the routes are. From the above, it can be verified that CoopSink presented routes up to 2.34 times better with relation to other proposals in the scenario considered in this work;

- Note that, for smaller values to constant t, we have very efficient routes in CoopSink at the price of a greater energy consumption, as can be observed in the results for metric M2. Clearly, striking a sensitive balance between these conflicting parameters is not an easy task and may depend on the application. In high throughput environments without severe energy restrictions, it would be desirable to keep $t \approx 1$, while in environments with energy restrictions, the t values should be higher.

5.4 M4: Number of CC edges

Figure 7 presents the simulation results for metric M4. The x-axis presents the number of network nodes while the y-axis shows the number of CC edges in the resulting topology. It should be noted that:

- CoopSink has more CC edges when the values for constant t are smaller. This happens because, with smaller values for t, greater is the restriction to remove edges in the CoopSink algorithm. These results are consistent with the results from metric M2, once the energy consumption for CoopSink-1.0 was substantially greater that CoopSink-1.4 and CoopSink-1.8;
- With less than 60 nodes, the CoopBridges has less CC edges than CoopSink-1.4 and CoopSink-1.8, and this number increases and surpass CoopSink later. This

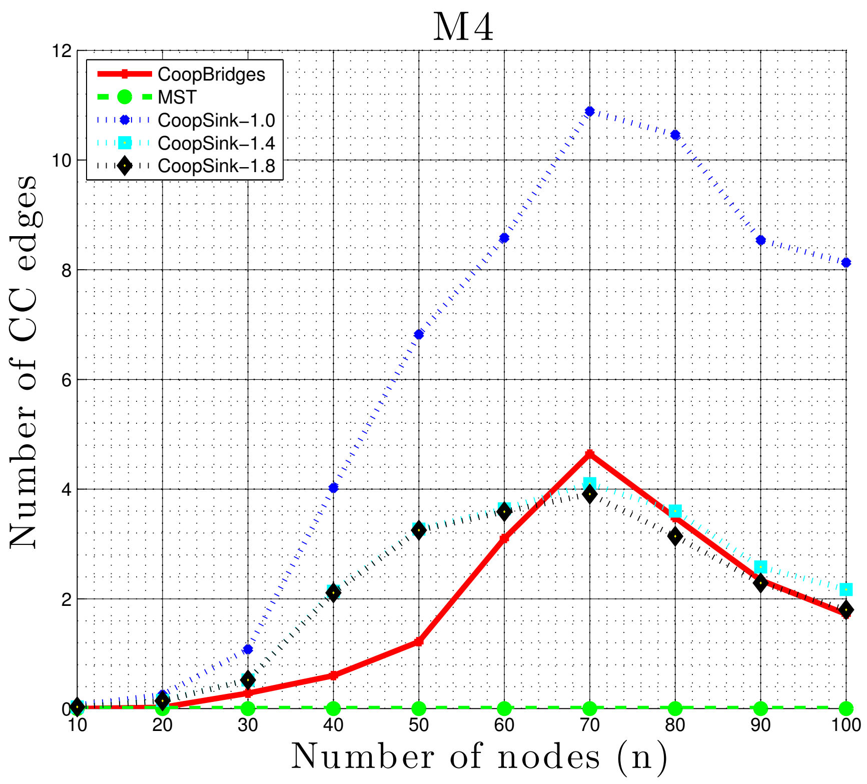

Fig. 7. Simulation results for M4.

happens because the connectivity to the sink in CoopBridges improves with $n \geq 50$ nodes, explaining this increase in the number of CC edges;

- From 60 nodes, the CoopBridges remains with basically the same number of CC edges than CoopSink-1.4 and CoopSink-1.8. However, the CoopBridges still has less connectivity to the sink. This shows that CoopSink is able explore CC links more efficiently than CoopBridges when the connectivity to the sink is considered.

6 Conclusion

In this work, it was presented a new technique, named CoopSink, that can be used to perform topology control in cooperative ad hoc networks. More specifically, CoopSink is a technique developed for ad hoc networks where there is the necessity to create efficient routes to a sink node. With this purpose, CC is used to increase connectivity and reduce energy consumption while maintaining efficient routes to the sink. When compared to other similar proposals, CoopSink was able to improve network connectivity to the sink up to 6.8 times while routes are up to 2.3 times better when the ESF to the sink is considered. Future works aim to analyse the impact of the required node synchronisation that is necessary for CC communication.

References

[1] Agarwal, M., L. Gao, J. H. Cho and J. Wu, *Energy efficient broadcast in wireless ad hoc networks with hitch-hiking*, Mobile Networks and Applications **10** (2005), pp. 897–910.

[2] Basagni, S., M. Conti, S. Giordano and I. Stojmenovic, “Mobile ad hoc networking,” Wiley.com, 2004.

[3] Blough, D. M., M. Leoncini, G. Resta and P. Santi, *On the symmetric range assignment problem in wireless ad hoc networks*, in: *Proceedings of the IFIP 17th World Computer Congress-TC1 Stream/2nd IFIP International Conference on Theoretical Computer Science: Foundations of Information Technology in the Era of Networking and Mobile Computing*, 2002, pp. 71–82.

[4] Boukerche, A., H. Oliveira, E. F. Nakamura and A. A. F. Loureiro, *A novel location-free greedy forward algorithm for wireless sensor networks*, in: *Communications, 2008. ICC'08. IEEE International Conference on*, IEEE, 2008, pp. 2096–2101.

[5] Cardei, M., J. Wu and S. Yang, *Topology control in ad hoc wireless networks using cooperative communication*, Mobile Computing, IEEE Transactions on **5** (2006), pp. 711–724.

[6] Chen, W.-T. and N.-F. Huang, *The strongly connecting problem on multihop packet radio networks*, Communications, IEEE Transactions on **37** (1989), pp. 293–295.

[7] Clementi, A. E., P. Penna and R. Silvestri, *On the power assignment problem in radio networks*, Mobile Networks and Applications **9** (2004), pp. 125–140.

[8] de Morais Cordeiro, C., H. Gossain and D. P. Agrawal, *Multicast over wireless mobile ad hoc networks: present and future directions*, Network, IEEE **17** (2003), pp. 52–59.

[9] Ekbatanifard, G. and R. Monsefi, *Mamac: A multi-channel asynchronous mac protocol for wireless sensor networks*, in: *Broadband and Wireless Computing, Communication and Applications (BWCCA), 2011 International Conference on*, IEEE, 2011, pp. 91–98.

[10] Grant, M., S. Boyd and Y. Ye, *Cvx: Matlab software for disciplined convex programming* (2008).

[11] Hou, T.-C. and V. Li, *Transmission range control in multihop packet radio networks*, Communications, IEEE Transactions on **34** (1986), pp. 38–44.

[12] Ibrahim, A. S., A. K. Sadek, W. Su and K. R. Liu, *Cooperative communications with relay-selection: when to cooperate and whom to cooperate with?*, Wireless Communications, IEEE Transactions on **7** (2008), pp. 2814–2827.

[13] Jakllari, G., S. V. Krishnamurthy, M. Faloutsos, P. V. Krishnamurthy and O. Ercetin, *A framework for distributed spatio-temporal communications in mobile ad hoc networks*, in: *Proc. IEEE Infocom*, 2006, pp. 1–13.

[14] Javali, N., “Topology Control for Wireless Ad-hoc Networks,” ProQuest, 2008.

[15] Jia, X., D. Li and D. Du, *QoS topology control in ad hoc wireless networks*, in: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1 **2**, IEEE, 2004, pp. 1264–1272.

[16] Kruskal, J. B., *On the shortest spanning subtree of a graph and the traveling salesman problem*, Proceedings of the American Mathematical society **7** (1956), pp. 48–50.

[17] Kurth, M., A. Zubow and J.-P. Redlich, *Cooperative opportunistic routing using transmit diversity in wireless mesh networks*, in: *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, IEEE, 2008, pp. 1310–1318.

[18] Laneman, J. N., *Cooperative communications in mobile ad hoc networks*, IEEE Signal Processing Magazine **1053** (2006).

[19] Laneman, J. N., D. N. Tse and G. W. Wornell, *Cooperative diversity in wireless networks: Efficient protocols and outage behavior*, Information Theory, IEEE Transactions on **50** (2004), pp. 3062–3080.

[20] Li, L., J. Y. Halpern, P. Bahl, Y.-M. Wang and R. Wattenhofer, *Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks*, in: *Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, ACM, 2001, pp. 264–273.

[21] Li, N., J. C. Hou and L. Sha, *Design and analysis of an mst-based topology control algorithm*, in: *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, 1 **3**, IEEE, 2003, pp. 1702–1712.

[22] Li, X.-Y., *Topology control in wireless ad hoc networks*, Mobile Ad Hoc Networking (2003), pp. 175–204.

[23] Li, X.-Y., P.-J. Wan and Y. Wang, *Power efficient and sparse spanner for wireless ad hoc networks*, in: *Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on*, IEEE, 2001, pp. 564–567.

[24] Lima, M. M., H. A. de Oliveira, E. F. Nakamura, A. A. Loureiro and B. H.-M. Gerais-Brasil, *Roteamento e agregaç ao de dados baseado no rssi em redes de sensores sem fio*, in: *Simposio Brasileiro de Redes de Computadores, 2013. SBRC*, SBC, 2013.

[25] Mohapatra, P. and S. V. Krishnamurthy, "Ad Hoc Networks - Technologies and Protocols," Springer Sciente + Business Media, Inc. chapters 1, 3, 6, 2005.

[26] Nosratinia, A., T. E. Hunter and A. Hedayat, *Cooperative communication in wireless networks*, Communications Magazine, IEEE **42** (2004), pp. 74–80.

[27] Ramanathan, R. and R. Rosales-Hain, *Topology control of multihop wireless networks using transmit power adjustment*, in: *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 1 **2**, IEEE, 2000, pp. 404–413.

[28] Sadek, A. K., W. Su and K. R. Liu, *Multinode cooperative communications in wireless networks*, Signal Processing, IEEE Transactions on **55** (2007), pp. 341–355.

[29] Shi, Y., S. Sharma, Y. T. Hou and S. Kompella, *Optimal relay assignment for cooperative communications*, in: *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, ACM, 2008, pp. 3–12.

[30] Tang, L., Y. Sun, O. Gurewitz and D. B. Johnson, *Pw-mac: An energy-efficient predictive-wakeup mac protocol for wireless sensor networks*, in: *INFOCOM, 2011 Proceedings IEEE*, IEEE, 2011, pp. 1305–1313.

[31] Wang, L., B. Liu, D. Goeckel, D. Towsley and C. Westphal, *Connectivity in cooperative wireless ad hoc networks*, in: *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, ACM, 2008, pp. 121–130.

[32] Wu, J., M. Cardei, F. Dai and S. Yang, *Extended dominating set and its applications in ad hoc networks using cooperative communication*, Parallel and Distributed Systems, IEEE Transactions on **17** (2006), pp. 851–864.

[33] Yu, J., H. Roh, W. Lee, S. Pack and D.-Z. Du, *Cooperative bridges: topology control in cooperative wireless ad hoc networks*, in: *INFOCOM, 2010 Proceedings IEEE*, IEEE, 2010, pp. 1–9.

[34] Zhang, J., G. Zhou, C. Huang, S. Son and J. Stankovic, *Tmmac: An energy efficient multi-channel mac protocol for ad hoc networks*, in: *Communications, 2007. ICC'07. IEEE International Conference on*, IEEE, 2007, pp. 3554–3561.

[35] Zhu, Y., M. Huang, S. Chen and Y. Wang, *Energy-efficient topology control in cooperative ad hoc networks*, Parallel and Distributed Systems, IEEE Transactions on **23** (2012), pp. 1480–1491.