Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 31 (2008) 160–173
computer communications
www.elsevier.com/locate/comcom

Efficient construction of network topology to conserve energy in wireless ad hoc networks ☆

Sajjad Zarifzadeh *, Amir Nayyeri, Nasser Yazdani

Router Laboratory, School of Electrical and Computer Engineering, University of Tehran, North Karegar Avenue, Tehran, Iran

Received 24 February 2007; received in revised form 15 October 2007; accepted 29 October 2007
Available online 7 November 2007

Abstract

Wireless ad hoc networks are usually composed of battery constraint devices, which make energy conservation a vital concern of their design. Reducing energy consumption has been addressed through different aspects till now. Topology Control (TC) is a well-known approach which tries to assign transmission ranges of nodes to optimize their energy utilization while keeping some network properties like connectivity. However, in current TC schemes, the transmission range of each node is mostly accounted as the exclusive estimator for its energy consumption, while ignoring the amount of data it sends or relays. In this paper, we redefine the problem of Topology Control regarding both transmission range and traffic load parameters. After proving the NP-hardness of the new problem, we mathematically formulate it as a mixed integer linear programming problem to find optimal solutions. Then, we introduce polynomial-time heuristic algorithms to practically solve the problem. During construction of network topology, we deliberately take into account the impact of the employed routing method on load of individual nodes. Finally, we show the advantages of our proposals through simulations.

Keywords: Wireless ad hoc networks; Energy conservation; Topology control; Routing; Traffic load

1. Introduction

Wireless ad hoc networks are composed of several wireless devices that form a network without any special infrastructure. Energy conservation is perhaps the most important issue in such networks since battery charging is usually difficult. This fact becomes more vital in some special-purposed wireless networks such as sensor networks or networks deployed in military or critical environments.

Thus far, different techniques have been suggested to address energy conservation problem, ranging from efficient hardware design [1], to appropriate placing of communicating codes in the network [2]. One of the most well-known approaches to this problem, which is called Topology Control (TC), is based on constructing an efficient topology for the network such that the energy consumption becomes optimum while some essential properties like connectivity are preserved in the induced network graph. Minimizing the transmission ranges of ad hoc nodes such that the resulting topology remains connected is one of the main TC approaches [3]. In brief, the main intuition behind this approach is that the amount of communication's energy that each node consumes is highly related to its transmission range.

However, we believe that there is a shortcoming in the conventional definition of the TC problem that negatively affects all existing proposals. Factually, in the TC problem, the goal of optimization is solely based on minimizing the transmission ranges of wireless nodes. However, transmission range together with the load on a device will determine its actual energy consumption rate. For instance, a node with a very large range that forwards only a small fraction

☆ An earlier and shorter version of this paper appeared in IEEE SECON, September 2006.

* Corresponding author. Tel.: +98 21 61114352; fax: +98 21 88778690.
E-mail addresses: szarifzadeh@ece.ut.ac.ir (S. Zarifzadeh), a.nayyeri@ece.ut.ac.ir (A. Nayyeri), yazdani@ut.ac.ir (N. Yazdani).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.10.040

of network's traffic may consume much less energy than another node with a smaller transmission range that forwards much more packets per time.

In this paper, we try to consider the above deficiency. More precisely, we formulate a new problem, called *MinMax Load Sensitive Topology Control* (*MLSTC*), for multihop wireless ad hoc networks. We first provide a general description for this problem, and then discuss its tractability under two main constraints. Next, we mathematically formulate MLSTC as a Mixed Integer Linear Programming (MILP) problem to obtain optimal solutions. Particularly, we consider our problem separately under the presence of the shortest-path routing and multipath routing methods. We also introduce heuristics to effectively approximate the problem in polynomial time. At last, through experimental results, we show the superiority of the MLSTC approach over former TC schemes.

The remainder of this paper is organized as follows: In the next section, we survey the previous works on topology control in wireless ad hoc networks. In Section 3, we present the motivation and also the exact description of the MinMax Load Sensitive Topology Control (MLSTC) problem. Section 4 is devoted to the explanation of MILP approach, and Section 5 introduces the proposed heuristic methods. We demonstrate the simulation results in Section 6 and finally conclude the paper in Section 7.

2. Related work

Being one of the main sources of energy consumption, different techniques have been proposed to reduce the required energy of communications among wireless nodes. One of the most challenging problems is selecting transmission ranges of nodes so that the energy utilization becomes optimal while some important properties of the network, for example connectivity, are conserved. In [3], the authors provide a well formulation of this problem together with some efficient algorithms. In their graph-based model, topology control problem is represented by a triple $\langle M, P, O\rangle$ where M is the model of the graph, i.e., directed or undirected, P represents the network properties that are important for us to conserve, like connectivity, bi-connectivity, and strong connectivity, and O denotes the objective that should be optimized, like the maximum power consumption or total power consumption. Categorizing different versions of topology control problem according to this model, they presented valuable results including polynomial algorithms for some cases and proofs of NP-hardness for others.

In addition, many other works have been proposed as heuristic and optimal solutions for various forms of topology control problem [4–6]. As one of the initial papers, [4] has suggested two centralized algorithms along with two heuristic distributed methods. Later, Wattenhofer et al. [5] proposed an elegant distributed algorithm which works locally. The idea is to adjust the transmission range of every node so that it has at least one neighbor on each α angle. Surprisingly, they proved that global connectivity will be conserved if α is not bigger than $2\pi/3$. Another worth noting work is [6], where the authors have tried to find an equal and small range value for all nodes to obtain a strongly connected graph structure. Different from other proposals, they applied probabilistic methods rather than deterministic algorithms to attain such network property.

Other practical approaches are COMPOW [7] and CLUSTERPOW [8], implemented in the network layer. Both rely on the idea that if each node uses the smallest common power required to maintain connectivity, then the capacity of the entire network with respect to carrying traffic is maximized, the battery life is extended, and the MAC-level contention is mitigated. The major drawback of these approaches is their significant message overhead, since each node has to run multiple daemons, each of which has to exchange link-state information with their counterparts at other nodes.

In sensor networks, the problem of energy conservation can be stated in a different form that is how to build a broadcast/multicast tree that conserves energy well. This problem has been studied in [9], where the authors tried to adjust nodes' power, such that the total energy cost of a broadcast/multicast tree becomes optimized. Heuristics were proposed to address the issue, namely Broadcast Incremental Power (BIP), Multicast Incremental Power (MIP), Minimum Spanning Tree (MST), and Shortest Path Tree (SPT) algorithms. The proposed algorithms were evaluated through simulations. Later, Wan et al. [10] presented a quantitative analysis to evaluate the performance of these heuristics.

Working upon the topology constructed during topology control process, the method of routing can significantly influence the forwarding load and the energy utilization of nodes in the network. Numerous works have been suggested for energy efficient routing in wireless ad hoc and sensor networks; such as [11,12]. Five important metrics for energy efficient routing have been studied in [12], like minimizing energy consumed per packet, minimizing variance in node power levels, minimizing cost per packet, and so on.

One fundamental shortcoming which is appeared in all of these TC proposals is that they try to minimize energy utilization of nodes only through reducing their transmission ranges without taking into consideration the amount of data they send or forward. However, as we will show later, energy utilization in each wireless node is significantly affected by the volume of traffic it forwards. As an example, one common negative phenomenon in the process of data forwarding in multihop wireless networks is the appearance of highly-loaded and early-depleted areas in special geometric places like centric parts of the network [13,14]. This deficiency is caused since usually most of packets should pass over this small area to reach their destinations. In brief, the main contribution of this paper is to address this deficiency by constructing an efficient topology for the network.

3. System model

Energy consumption in wireless ad hoc networks usually roots in either communication or processing efforts of the system. However, in many cases like sensor networks or highly-loaded networks, the amounts of energy that nodes use for message passing is orders of magnitude higher than what they consume for processing. Thus, in many proposals in this field, optimizing the communication's energy consumption is assumed as the ultimate goal.

3.1. Basics of wireless communication

Communication's energy consumption itself is caused by either transmitting or receiving packets. Here, we have used the popular modeling also used by [15]. In this model, the required energy for transmitting a bit-stream at rate a over the Euclidian distance of d is assessed by:

$$E_t(a,d) = a(\alpha_1 + \alpha_2 d^n), \tag{1}$$

in which α_1 is the distance-independent term (i.e., the energy utilized in the transmitter circuit) and α_2 captures the distance-dependent one. Moreover, n is a real value which is usually $2 \leqslant n \leqslant 4$ for the free-space and short-to-medium range communications. Likewise, the amount of energy used to receive a bit-stream again at rate a can be calculated as:

$$E_r(a) = a\beta. \tag{2}$$

Therefore, when a node forwards data by rate a over the distance d, it consumes $E(a, d)$ units of energy per time where $E(a, d)$ is computed according to the following formula:

$$E(a,d) = a(\gamma + \alpha d^n), \tag{3}$$

in which $\gamma = \alpha_1 + \beta$ and $\alpha = \alpha_2$.

3.2. Motivation

The above formula apparently shows that the amount of consumed energy in each wireless node not only depends on its transmission range (TR) but also on the volume of traffic it relays. Hence, nodes that only forward a limited number of packets can have longer ranges and, at the same time, consume even less amount of energy than other highly-loaded nodes. Consequently, for the sake of minimizing the energy consumption, it seems rational to select longer TR values for lightly-loaded nodes or equivalently smaller ranges for those that experience high traffic loads.

In this paper, we consider both load and range parameters to minimize the maximum energy consumption among wireless nodes. Our solution to this problem is primarily based on the observation that in wireless ad hoc networks, the devices near some special places, like centric parts of the network, usually bear a high traffic load [13]. Therefore, it makes sense to use variable transmission range idea to assign lower ranges to such close nodes, and higher ones to those placed far from the center. Unfortunately, solving this decision problem is not straightforward. The major difficulty is that changing the TR values of nodes may greatly alter their traffic loads, and as a result, a highly-loaded node may become idle as the ranges change and vice versa. This means that the ranges of wireless nodes should be reassigned again and again and evidently, it might bring a loop in the process of topology management by which the TR values never converge to the optimal ones.

A simple clarifying example is shown in Fig. 1. As the left figure depicts, if all five nodes A, B, C, D, and E choose a relatively equal and small value for their transmission range, then node A which is located at the center of the network will be obliged to receive and forward a huge volume of traffic generated by other nodes. Unfortunately, this overloading of node A quickens its energy expiration and so breaks the network connectivity rapidly. On the other hand, when there is a link between node B and C and also between node D and E (as the right figure shows), these nodes will be able to send their packets directly to each other without involving node A. In this case, although node A will bear a lower traffic load but due to the selection of longer ranges by B, C, D and E, their energy will drain much earlier than before. In fact, as this example demonstrates, the TC process makes a trade-off between load and transmission range of nodes. Namely, as much as ranges of wireless devices increase, although the traffic load will be reduced and also balanced over the network, but the increase in their energy consumption (arising from the selection of higher ranges) may significantly influence the virtues of such load balancing approach. The goal of this work is indeed to prolong the network lifetime through an efficient combination of both range minimization and load balancing approaches. In this context, the lifetime of the network is the time elapses from the network start-up time till the first node of the network stops working.

Notice that while managing the topology in this manner (i.e., by adjusting the ranges of nodes) can result in a longer lifetime for the network, it may increase the total energy consumption of nodes. Yet, most of practical works try to increase the lifetime of the network rather than its total energy consumption. For instance, when a sensor network stops working, the amount of energy remained in the still alive sensors is not of importance.

3.3. Problem statement

The discussion so far must convince the reader that in the process of constructing network topology not only the transmission range but also the amount of traffic load on individual nodes should be considered. In this section, we declare a new TC problem chiefly, because the conventional TC approaches hide the load parameter that is crucial for energy conservation.

Definition 1. A *wireless ad hoc network* is represented as $W = (N, L)$, where N is a set of nodes and L is a set of coordinates on the plane denoting the locations of the nodes.

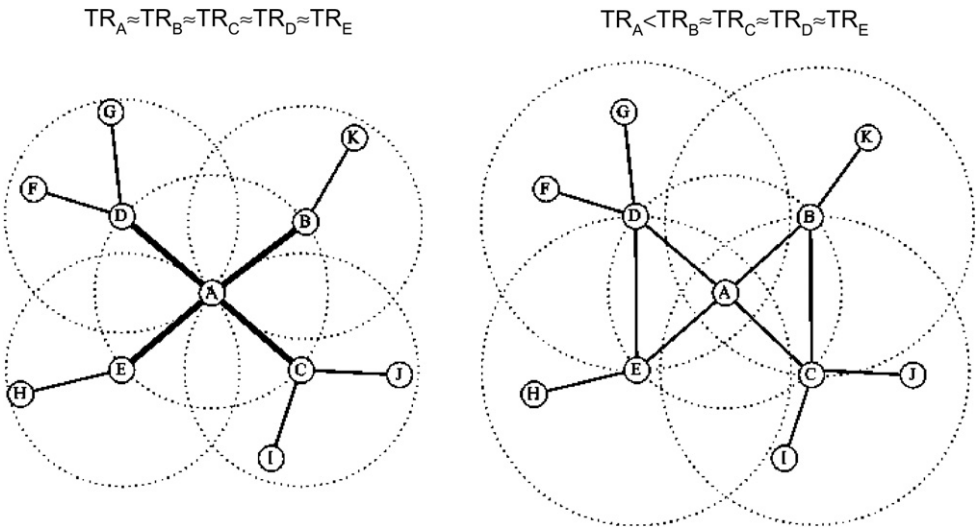

Fig. 1. A simple motivating example. The bold lines indicate the highly-loaded links (TR_A is the TR value of node A).

Let $d_{u,v}$ be the geographical distance between node u and v. For the sake of simplicity, we assume that the energy consumption of each wireless node can be computed according to (3) with $\gamma = 0$ and $\alpha = 1$ (this is because $\alpha d^n \gg \gamma$). Hence, the amount of energy that node u consumes to relay traffic by rate a to node v will be:

$$E_u(a, v) = a \cdot d_{u,v}^n. \quad (4)$$

Let $r_{\max}$ be the maximum transmission range of all nodes in the network. We denote the network topology constructed under the common range $r_{\max}$ as an undirected simple graph $G_{\max} = (V, E)$, where V is the set of nodes in the network and $E = \{(u, v) | d_{u,v} \leqslant r_{\max}, u, v \in V\}$ is the link set.

Definition 2. A *range assignment function* for a wireless network $W = (N, L)$ is represented as $r : N \to R^+$, where $r(u)$ denotes the transmission range of node u.

Definition 3. Given a wireless ad hoc network $W = (N, L)$, and a range function $r : N \to R^+$, the *induced graph* is represented by $G_r = (V, E)$, where V is a set of vertices corresponding to the nodes in N, and E is a set of undirected edges that $(u, v) \in E$, if and only if, $r(u) \geqslant d_{u,v}$ and $r(v) \geqslant d_{v,u}$.

Generally, the TC problem can be thought as the problem of optimizing a set of cost metrics under a given set of constraints. Examples of cost metrics include total energy consumption and maximum energy consumption. Examples of constraints include connectivity, degree boundedness and tree-ness. In this paper, we consider a single cost metric, namely the maximum energy consumption, and two constraints connectivity and tree-ness. The objective of minimizing the maximum energy consumption rather than the total over all nodes is because battery life is a local resource so collective minimization has a little practical value [4]. Also, when the constraint is connectivity (or tree-ness), we are looking only for those topologies that form a spanning graph (or tree) among all nodes of the network. It is remarkable that making topology in the form of a spanning tree can be very helpful, because: (1) It is the minimum-sized graph structure that provides connectivity between every pair of nodes; thus, the degree of nodes are usually small and also computation and maintenance of the resulted topology is simple, and (2) It provides loop-freedom for distributed routing algorithms, since there is no loop in a tree. For convenience, we use CGraph and CTree terms to refer to the connectivity and tree-ness constraints, respectively.

Definition 4. Problem *MinMax Range Assignment* (*MRA*). Given a wireless network $W = (N, L)$, find a range assignment function $r : N \to R^+$, such that the induced graph preserves the property $P \in \{CGraph, CTree\}$, and $\max_{\forall u \in N}\{r(u)\}$ becomes minimized.

The main intuition behind MRA is that through minimizing the transmission ranges of nodes, the amount of consumed energy is more likely to be reduced, according to (4). As you can see, MRA ignores the impact of traffic load on energy consumption of nodes. Based on the graph property P, two separate problems derive from the MRA problem; namely MRA-CTree, if $P =$ CTree and MRA-CGraph, if $P =$ CGraph. In [4], the authors introduced an optimal polynomial-time algorithm, named BICONN-AUGMENT, to solve the MRA-CGraph problem. But, [3] theoretically proves that MRA-CTree is NP-hard.

Let $\lambda : N \times N \to R^+$ be the rate function that indicates the rate of traffic between every pair of nodes, i.e., $\lambda(s, d)$ denotes the traffic rate from node s to d. As discussed before, we take both range and load of wireless nodes into account for the purpose of energy conservation. Specifically, we consider the following constrained optimization problem:

Definition 5. Problem *MinMax Load Sensitive Topology Control* (*MLSTC*). Given a wireless network $W = (N, L)$ and a rate function $\lambda : N \times N \to R^+$, find a sub-graph of

G_{max} that preserves the property $P \in \{CGraph, CTree\}$ and also minimizes $E_{max} = \max_{\forall u \in N}\{L_u \times r^n(u)\}$ where L_u is the load of node u.

Overall, the problem definition is "how to construct a graph of type P so that E_{max} becomes optimized". Note that MLSTC mainly tries to find the desired graph of topology rather than determining the optimal range assignment function. Clearly, the range of each node $u \in N$ can be obtained indirectly from the resulting topology by the following relation:

$$r(u) = \max_{\forall v \in N(u)}\{d_{u,v}\}, \tag{5}$$

where $N(u)$ is the set of all neighbor nodes of u. In other words, MLSTC differs from MRA in the following two senses: (1) unlike MRA, MLSTC considers load of nodes when computing their energy utilization; and (2) in MRA, the range of every node exactly determines its neighbors while in MLSTC, this statement may not be true always. For example, suppose that $G(V, E)$ is a solution to MLSTC. Then, we may encounter a situation in G where $r(u) \geqslant d_{u,v}$, but $(u, v) \notin E$. Similar to MRA, we use MLSTC-CGraph when we consider the MLSTC problem under the CGraph constraint and accordingly, MLSTC-CTree when the constraint is CTree. Unfortunately, we have found that both variants of MLSTC problem are hard to be solved in polynomial time.

Theorem 1. *The problem MLSTC-CTree is NP-hard.*

Proof. The proof is presented in the Appendix A. □

Following the proof in the Appendix A, it is not a difficult task to show that MLSTC remains NP-hard under the CGraph constraint as well. In the next section, we formulate the MLSTC problem as an MILP problem. Later, we suggest heuristic methods to approximate it. Without loss of generality, we hereafter assume that the rate of traffic between every two nodes is the same rate λ, i.e., $\lambda(s, d) = \lambda, (\forall s, d \in N)$. Note that due to difficulty of determining the exact amount of traffic demands between node pairs, most of works in this field assume a uniform distribution for traffic demands of nodes [13,14].

It should be mentioned here that the choice of the routing algorithm clearly affects the load of individual nodes which in consequence yields different rate of energy consumption for every node. Hence, it is necessary to take the routing method into our consideration during constructing network topology. Due to dominance of hop-count shortest-path (SP) routing in current networks, we concentrate here on finding solutions to MLSTC under the SP routing algorithm. In addition, we investigate the problem under a restricted type of multipath routing in which only node-disjoint paths are considered to route packets of the same source-destination pair in order to avoid possible collisions between paths. In this paper, we consider only undirected graphs, that is, all links in the graph are bi-directional. The capability of forming a topology that consists of only bi-directional links is important for link level acknowledgement and critical for proper operation of the RTS/CTS mechanism [20]. Bi-directional links also ensure existence of reverse paths for transferring ACKs. In addition, we assume that each node can transmit signals to its neighbors in a conflict free fashion. Thus, like many other topology control proposals, we do not consider signal interference in this paper. There are many MAC layer and code assignment protocols [16,17] that have been proposed to avoid or reduce signal interference in radio transmissions.

4. MILP approach to the MLSTC problem

In this section, we introduce our mixed integer linear programming (MILP) formulation for both variants of the MLSTC problem (i.e., MLSTC-CTree and MLSTC-CGraph). First, we introduce the formulation for the case where the shortest path (SP) routing algorithm is applied on both variants. Later, a similar MILP approach under multipath (MP) routing will be formulized. To simplify the formulation, we use the following variables:

- $l_{i,j}$: Boolean variable, $l_{i,j} = 1$ if there is a link from node i to node j; otherwise $l_{i,j} = 0$.
- $l_{i,j}^{s,d}$: Boolean variable, $l_{i,j}^{s,d} = 1$ if the route from s to d goes through the link (i, j); otherwise $l_{i,j}^{s,d} = 0$.
- $f_{i,j}^{s,d}$: The amount of traffic from node s to node d that goes through the link (i, j).
- E_{max}: Maximum energy consumption among all nodes.
- L_i: The transmitting load on node i.

4.1. Single path routing

Now, we present the MILP formulation of MLSTC-CTree:

Given: A wireless ad hoc network $W = (N, L)$, a traffic demand λ between each pair of nodes and a maximum transmission range r_{max}. T denotes the total number of nodes placed in the network, i.e., $T = |N|$.

Optimization goal: Minimize the maximum energy consumption of nodes:

$$Minimize \quad E_{max} \tag{6}$$

Constraints:

– Bi-directional link constraint:

$$\forall i, j \in N: \qquad l_{i,j} = l_{j,i} \tag{7}$$

– Route constraints:

$$\forall s, d, i \in N: \sum_{\forall j \in N} l_{i,j}^{s,d} - \sum_{\forall j \in N} l_{j,i}^{s,d} = \begin{cases} +1 & s = i \\ -1 & d = i \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$\forall s, d, i, j \in N: l_{i,j}^{s,d} \leqslant l_{i,j} \tag{9}$$

– Load constraint:

$$\forall i \in N : L_i = \sum_{\forall s,d \in N} \sum_{\forall j \in N} l_{i,j}^{s,d} \times \lambda = (T-1) \times \lambda + \sum_{\forall s,d \in N} \sum_{\forall j \in N} l_{j,i}^{s,d} \times \lambda \quad (10)$$

– Maximum transmission range constraint:

$$\forall i \in N : r_{\max} \geqslant \max_{\forall j \in N}\{d_{i,j} \times l_{i,j}\} \quad (11)$$

– Energy constraint:

$$\forall i \in N : E_{\max} \geqslant L_i \times \max_{\forall j \in N}\{d_{i,j}^{n} \times l_{i,j}\} \quad (12)$$

– SP routing constraints:

$$\forall s,d,t \in N : \sum_{\forall i,j \in N} l_{i,j}^{s,d} \leqslant \sum_{\forall i,j \in N} l_{i,j}^{s,t} + l_{t,d} + T \times (1 - l_{t,d}) \quad (13)$$

$$\forall s,d \in N : l_{s,d}^{s,d} = l_{s,d} \quad (14)$$

– Tree constraint:

$$\sum_{\forall i \in N} \sum_{\forall j \in N} l_{i,j} = 2 \cdot (T-1) \quad (15)$$

Remarks.

- Constraint (7) ensures that each edge is bi-directional.
- Constraints (8) and (9) ensure the validity of the route for each source-destination pair. The first constraint means that a link should exist before belonging to any particular path. As stated in the second constraint, for a node j along the path from s to d, the number of input links of the path at node j equals the outputs. Since unicast routing is used, if $l_{i,j}^{s,d} = 1$, then the whole traffic from node s to node d must go through link (i, j).
- Constraint (10) determines the traffic load that each node experiences. It also ensures that the total outgoing traffic from a node is equal to the total arriving traffic at the node plus the traffic originated by the node itself.
- Constraint (11) states that the range of every node must be no longer than the maximum range $r_{\max}$.
- Constraint (12) determines the maximum energy consumption among all nodes.
- Constraints (13) and (14) ensure the validity of the route under the shortest path routing for each source-destination pair. In other words, constraint (13) asserts that if there is an arbitrary node t such that $(t,d) \in E$, then the length of the path from s to d $\left(\text{i.e.}\sum_{\forall i,j \in N} l_{i,j}^{s,d}\right)$ should not be larger than the length of the path from s to t plus 1 $\left(\text{i.e.}\sum_{\forall i,j \in N} l_{i,j}^{s,t} + l_{t,d}\right)$. If there is not any link between t and d (i.e. $l_{t,d} = 0$), the term $T \times (1 - l_{t,d})$ in (13) causes the inequality of (13) to be always satisfied, since the length of any simple path from s to d is certainly less than the total number of nodes.
- Constraint (15) ensures the tree-ness property of the network graph.
- Note that the constraints (6)–(12) are applied on both versions of MLSTC. However, constraints (13) and (14) are applied only on MLSTC-CGraph (because there is totally one route between every pair of nodes in a tree) and constraint (15) is used just in the MLSTC-CTree problem.

4.2. Multipath routing

In many situation like fault tolerant networks, the traffic between a source-destination pair may take multiple paths due to congestion or failures in the network. In this subsection, we consider our problem in the case where the traffic demand of a node-pair can be split among multiple node-disjoint paths.

Given: All the parameters in the formulation of former case, as well as k as the maximum number of node-disjoint paths used between a node-pair (s, d).

Optimization goal: Minimize the maximum energy consumption of nodes:

$$Minimize E_{\max} \quad (16)$$

Constraints:

– Bi-directional link constraint:

$$\forall i,j \in N : l_{i,j} = l_{j,i} \quad (17)$$

– Route constraints:

$$\forall s,d,i \in N : \quad \sum_{\forall j \in N} f_{i,j}^{s,d} - \sum_{\forall j \in N} f_{j,i}^{s,d} = \begin{cases} +\lambda & s = i \\ -\lambda & d = i \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

$$\forall s,d,i,j \in N : \quad f_{i,j}^{s,d} \leqslant l_{i,j} \times f_{i,j}^{s,d} \quad (19)$$

$$\forall s,d,i,j \in N : \quad l_{i,j}^{s,d} = \begin{cases} 0 & f_{i,j}^{s,d} = 0 \\ 1 & f_{i,j}^{s,d} > 0 \end{cases} \quad (20)$$

– Load constraint:

$$\forall i \in N : \quad L_i = \sum_{\forall s,d \in N} \sum_{\forall j \in N} f_{i,j}^{s,d} \times \lambda = (T-1) \times \lambda + \sum_{\forall s,d \in N} \sum_{\forall j \in N} f_{j,i}^{s,d} \times \lambda \quad (21)$$

– Energy constraint:

$$\forall i \in N : \quad E_{\max} \geqslant L_i \times \max_{\forall j \in N}\{d_{i,j}^{n} \times l_{i,j}\} \quad (22)$$

– MP routing constraint:

$$\forall s,d,i \in N : \quad \sum_{\forall j \in N} l_{i,j}^{s,d} = \begin{cases} \leqslant k & i = s \\ \leqslant 1 & i \neq s \end{cases} \quad (23)$$

Remarks.

- The objective and the most of constraints are the same as the case with the SP routing algorithm.
- Constraint (18) specifies the flow conservation along all the routes used between the node-pair (s, d).

- Constraint (23) ensures the node-disjointness of the routes between every source-destination pair. It also guarantees that at most k routes are used to convey the traffic between s and d.

Now, the problems of MLSTC (under both SP and MP routing algorithms) have been formulated as mixed integer linear programming problems. There are several tools that can be employed to compute the solutions to these problems. After calculating $l_{i,j}$ $(\forall i,j \in N)$, the transmission range of each node can be determined by its distance to the farthest neighbour node, as stated in (5). We use MATLAB 7 (with TOMLAB package) to solve our MILP problems for experiment purpose. The results are presented in Section 6.

5. Heuristic approaches to the MLSTC problem

Although in the previous section, we presented the MILP formulation of MLSTC to find the optimal solutions, it is impossible to find such optimal solutions within polynomial time due to the NP-hardness of MLSTC, unless $P = \text{NP}$. Hence, we need approximation algorithms to practically solve this problem for large scale networks. In this section, we introduce three separate heuristics for MLSTC-CTree and MLSTC-CGraph problems. In Section 6, we perform simulations to compare their results with the optimal ones obtained by solving the corresponding MILP problems.

5.1. Optimized minimum spanning tree (OMST) for MLSTC-CTree

In this section, we describe the *Optimized Minimum Spanning Tree* (*OMST*) method as a heuristic solution to the MLSTC-CTree problem. Before explaining our solution, we should first define an elementary issue.

Definition 6. Assume that $(x,y) \in E$. The *input load* of a node x from a node y, denoted by $s(x,y)$, is defined as the number of nodes whose paths to x goes through link (x,y).

Apparently, if x and y are not neighbors then both $s(x,y)$ and $s(y,x)$ will be equal to zero. Note that, according to the property that there exists only one route between every pair of nodes in a tree, the input load is more straightforward in tree structure. Immediately from the above definition and also the definition of tree, we can conclude the following relations:

$$\forall x \in N: \quad \sum_{\forall y \in N} s(x,y) = T - 1, \tag{24}$$

$$\forall (x,y) \in \textit{Tree}: \quad s(x,y) = T - s(y,x), \tag{25}$$

where T is the total number of nodes in the network. Note that (24) remains still true in general graph structures. Having the input loads of a node x (located on a tree) from all its neighbors, the total traffic load on x can be easily computed according to the following lemma.

Lemma 1. *The total load on a node x (located on a tree) can be computed by*:

$$L_x = \left(\left(T - \frac{1}{2} \right) \cdot (T-1) - \sum_{\forall y \in N} s^2(x,y) \right) \cdot 2\lambda \tag{26}$$

Proof. Consider L_x^r as the amount of data relayed by node x and L_x^t as the traffic load that node x produces itself. Clearly, the total load of node x, i.e., L_x, is:

$$L_x = L_x^r + L_x^t$$

L_x^r can be simply computed from the following formula:

$$\begin{aligned} L_x^r &= \left(\sum_{\forall y \in N(x)} 2 \cdot s(x,y) \cdot s(y,x) \cdot \lambda \right) \\ &= \left(\sum_{\forall y \in N(x)} s(x,y) \cdot ((T-1) - s(x,y)) \right) \cdot 2\lambda \\ &= \left((T-1) \sum_{\forall y \in N(x)} s(x,y) - \sum_{\forall y \in N(x)} s^2(x,y) \right) \cdot 2\lambda \\ &= \left((T-1)(T-1) - \sum_{\forall y \in N(x)} s^2(x,y) \right) \cdot 2\lambda \end{aligned}$$

On the other hand, L_x^t equals the total traffic that all nodes (except x itself) receive from x, i.e.,:

$$L_x^t = \left(\sum_{\forall y \in N(x)} s(x,y) \right) \cdot \lambda = (T-1) \cdot \lambda$$

Conclusively:
$L_x = L_x^r + L_x^t = ((T - \frac{1}{2}) \cdot (T-1) - \sum_{\forall y \in N} s^2(x,y)) \cdot 2\lambda$. □

Corollary 1. *The maximum energy consumption over the entire network is*:

$$E^* = \max_{\forall x \in N} \left\{ r(x) \cdot \left(\left| T - \frac{1}{2} \right| |T-1| - \sum_{\forall y \in N} s^2(x,y) \right) \cdot 2\lambda \right\} \tag{27}$$

Basically, our OMST algorithm is an iterative variant of MST algorithm, which is optimized for energy conservation. It consists of two major steps. In the initial step, OMST constructs a minimum spanning tree over the graph $G_{\max}$ of $W = (N, L)$ using the well-known Prim's algorithm [18]. During the second step, OMST tries to optimize the constructed tree by incremental changes in the initial MST. First, the node that utilizes the maximum energy is found using (27). For example, let node z be such a node in the topology example of Fig. 2. As depicted in the figure, every pair of adjacent neighbors of z, like v and u, composes a semi-triangle shape with node z as its center vertex. In our terminology, we call this shape *adj-triangle* which is *centered at* z and refer to it by Λzvu.

Intuitively, to decrease the energy utilization of z, the adj-triangle Λzvu can be transformed to one of Λuzv and

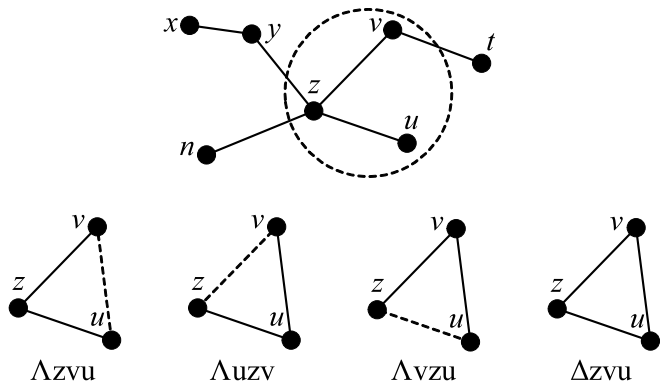

Fig. 2. Different types of adj-triangle transformation.

Λvzu, or remain unchanged if both transformations increase the maximum energy consumption of the network. Since such transformations, as we will show later, will not affect the energy utilization of nodes other than z, u, and v, it suffices to check whether the maximum required energy among these three nodes is less than what was consumed by z before or not. Among all adj-triangles centered at z, OMST chooses and then transforms the one that reduces the maximum required energy more than others. OMST repeats step 2 (i.e., finding the node with maximum energy and transforming one of its adj-triangles) again and again until the maximum energy can not be reduced any more. Of course, after each transformation, the input load of all vertices of the modified adj-triangle should be updated properly. Fortunately, this update can be done in O(1) using the lemma below.

Lemma 2. *Let Λzvu be an adj-triangle which is transformed to Λuzv during the construction of OMST. The input loads of z, v and u are updated as follows*:

$$s_{new}(z,v) = s_{new}(v,z) = 0, \tag{28}$$

$$s_{new}(z,u) = s_{old}(z,u) + s_{old}(z,v), \tag{29}$$

$$s_{new}(u,z) = s_{old}(u,z) - s_{old}(z,v), \tag{30}$$

$$s_{new}(u,v) = s_{old}(z,v), \tag{31}$$

$$s_{new}(v,u) = s_{old}(z,u), \tag{32}$$

$$\text{other cases} : s_{new}(x,y) = s_{old}(x,y), \tag{33}$$

where s_{old} is the value of input load before transforming Λzvu and s_{new} refers to the new value.

Proof. Considering the acyclic property of OMST, all relations are obtained straightforwardly. □

Now, we state some properties of the network topology derived from OMST. Since throughout the construction of OMST, the resulted graph remains spanning tree with bi-directional links, OMST guarantees to hold connectivity. Remind that preserving connectivity is the primary objective of topology control algorithms. Moreover, because only adj-triangles are transformed in OMST, it can be shown that the resulting topology is planar (i.e., no two edges crossing each other in the graph). This enables several localized routing algorithms, such as [19], to be performed on top of this structure to provide packet delivery without routing table.

In many TC schemes, it is desirable that the node degree[1] in the resulting topology is bounded by a small value. This feature is favorable because a small node degree reduces the MAC-level contention and interference [20]. It has been observed that any MST of a simple undirected graph in the plane has a maximum degree of 6 [20]. However, our investigation suggests that the maximum degree in OMST can theoretically reach a high value (For example, it can reach 12 as shown in Fig. 3). To avoid such a high node degree, our algorithm can follow a heuristic guideline in adj-triangle transformations such as not exceeding an upper bound d_{max} on the degree of every node. More precisely, we apply a transformation on the tree, if such a change does not cause the degree of any node in the network to become more than d_{max}.

It is worth mentioning that because the maximum energy consumption decreases step by step during each adj-triangle transformation, our algorithm ensures to converge to the optimal OMST. Besides, we can prove that OMST has $O(|N|^4)$ complexity in the worst case. The main intuition is that the maximum energy takes its value from a finite discrete set of values, which is determined by the range and load of nodes. More precisely, we have $O(|N|)$ choices of transmission range for each node and also the load on a node can vary from $|N| \times \lambda$ to at most $|N|^2 \times \lambda$. Since at most $|N|$ nodes can be chosen as the node of maximum energy during transformations, we can conclude that the worst-case complexity of the OMST algorithm is $O(|N| \times |N| \times |N|^2) = O(|N|^4)$. Note that because every adj-triangle takes O(1) to be transformed, it has no effect on this upper bound. Later, by presenting experimental results, we show that the actual running time of OMST is much smaller than the worst case and is fortunately comparable to the running time of the original MST algorithm.

5.2. Refined OMST for MLSTC-CGraph under shortest path routing

With some slight modifications, OMST can be extended to be used as an efficient heuristic solution to the MLSTC-CGraph problem (when SP algorithm is used to route packets). We call this new algorithm *Refined OMST* (*ROMST*). Initially, like OMST, the node consuming the maximum energy must be found. Then, adj-triangle transformations should take place. In addition to two possible choices of transformations introduced in the previous section, ROMST can use a new one in which the edge (u, v) is added into Λzvu without any elimination of edges (z, v) and (z, u). We call an adj-triangle whose all three sides exist

[1] The degree of a node is the number of nodes connected directly to that node.

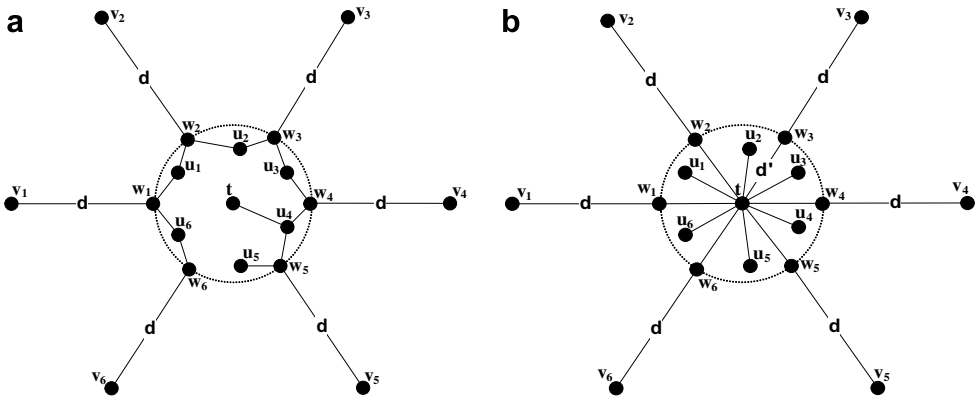

Fig. 3. An example that shows the node degree in OMST can reach 12. The number on each link is the length of that link. Note that $d \gg d'$. (a) MST of the network. (b) OMST of the network.

in the graph *complete adj-triangle* and use Δzvu notation to denote it. Unfortunately, the introduction of complete adj-triangles makes the computation of input loads very complicated, because in each transformation, we may need to re-calculate data paths between every pair of nodes to compute input loads. To avoid such a large order of complexity, we put the following constraints on adj-triangle transformations in ROMST:

C1. An edge belonging to a complete adj-triangle is never removed.

C2. An adj-triangle can be transformed into a complete adj-triangle, if none of its sides belong to a complete adj-triangle.

Notice that by preserving the above constraints during transforming Λzuv, only the portion of routes that goes over $\langle u, z, v\rangle$ will be replaced with $\langle u, v\rangle$. The main reason is that if an arbitrary node t $(t \neq z)$ is cut-vertex[2] before transforming Λzuv to Δzuv, then it will be still cut-vertex after transformation. In this manner, only paths which traverse through $\langle u, z, v\rangle$ will change and instead go over the shortest sub-path $\langle u, v\rangle$. Fortunately, this causes that the input loads of all nodes (other than z, u, and v) remain the same as before. This fact is formally confirmed below.

Lemma 3. *Given an adj-triangle Λzuv, if a node x $(x \neq z)$ is cut-vertex before transforming Λzuv to Δzuv, it remains a cut-vertex after transformation.*

Proof. The proposition is trivial for the case when $x \notin \{u, v\}$. Now, assume $x \in \{u, v\}$ and for simplicity, suppose that $x = u$. The other case can be similarly proved. Since u is a cut-vertex before transformation, its degree should be at least 2 and hence, its degree increases by one after transformation, due to addition of (u, v) into the graph. As the degree of u is greater than 2, there exists at least one adjacent vertex to x, other than u, and v, named x'. Contradictory, assume that x is not a cut-vertex in the new graph after transformation. Then, there exists a path P from x' to v, that does not contain x. Thus, eliminating x would not partition even the old graph, and as a result, x was not a cut-vertex before transformation, which contradicts our assumption. Hence, the lemma is proved. □

The main issue that should be addressed now is how to update the input loads of nodes involved in an adj-triangle transformation. Clearly, in the case that Λzvu is transformed to either Λvzu or Λuzv, the input loads are simply computed according to Lemma 2. However, transforming to Δzvu raises a different form of input load update, which can be calculated in $O(|N|)$ time through the following lemma.

Lemma 4. *Let Λzvu be an adj-triangle which changes to Δzvu under the constraints C1 and C2. The input loads of z, v and u are updated as follows*:

$$s_{\text{new}}(z,u) = s_{\text{old}}(z,u) \quad \textit{and} \quad s_{\text{new}}(z,v) = s_{\text{old}}(z,v), \tag{34}$$

$$s_{\text{new}}(u,z) = s_{\text{old}}(z,v) \quad \textit{and} \quad s_{\text{new}}(v,z) = s_{\text{old}}(z,u), \tag{35}$$

$$s_{\text{new}}(u,v) = \sum_{\forall x \in \{N(v)-z\}} s_{\text{old}}(v,x) + 1, \tag{36}$$

$$s_{\text{new}}(v,u) = \sum_{\forall x \in \{N(u)-z\}} s_{\text{old}}(u,x) + 1, \tag{37}$$

$$\text{other cases}: s_{\text{new}}(x,y) = s_{\text{old}}(x,y). \tag{38}$$

Proof. Since connecting u and v together does not make any node to reach z through a shorter path, then the input loads of node z from both nodes u and v remain unaffected, as stated by (34). However, by adding the link (u, v) into the topology, all nodes that access node z through the link (v, z) will change their routes to reach u by the link (v, u), instead of the longer path $\langle v, z, u\rangle$. This apparently yields the equations declared in (35). The relations stated in (36) and (37) derive from Lemma 3, as u, and v are cut-vertices. At last, the relation in (38) is concluded from respecting the constraints C1 and C2, as discussed above. □

Because ROMST is not acyclic, the total amount of traffic load on a node cannot be computed through Lemma 1. Instead, Lemma 5 allows us to compute the load of every node in ROMST.

[2] A vertex whose deletion along with its incident edges results in a graph with more components than the original graph.

Lemma 5. *Consider L_x^{r1} as the amount of data relayed by node x over links of adj-triangles centered at x, L_x^{r2} as the amount of data forwarded through links of complete adj-triangles (centered at x), and finally L_x^t as the traffic load that node x produces itself. Also, let $S_x(\Delta xyt) = s(x,y) + s(x,t)$. The total load on a node x in ROMST can be computed by*:

$$L_x = L_x^{r1} + L_x^{r2} + L_x^t, \tag{39}$$

where

$$L_x^{r1} = \left(\sum_{\forall \Delta xyz} S_x(\Delta xyz) \cdot ((T-1) - S_x(\Delta xyz)) \right) \cdot 2\lambda, \tag{40}$$

$$L_x^{r2} = \left(\sum_{\forall \Lambda xyz} s(x,y) \cdot ((T-1) - s(x,y)) \right) \cdot 2\lambda, \tag{41}$$

$$L_x^t = (T-1) \cdot \lambda. \tag{42}$$

Proof. The proof of this lemma is very similar to that of Lemma 1. For conciseness, we omit the proof here. □

5.3. Extending to multipath (MP) routing

Both heuristic methods introduced so far are based on the assumption that the shortest path (SP) algorithm is used to route packets. Hence, they may not be originally suitable to work under multipath (MP) routing approach. For example, a good network topology for the multipath case should be enough dense to provide several node disjoint paths between node pairs. However, ROMST is so sparse that we rarely can find multiple paths between every two nodes. It is noticeable that devising an efficient method that properly works under the multipath case is much more difficult than the former single path approach. The reason is that the concurrent usage of multiple paths in a topology (which should be preferably dense enough) extremely complicates the computation of load parameter. Thus, understanding the exact amount of traffic load that each node experiences is somehow impossible within tolerable time. Here, we only try to modify our ROMST structure to operate as an effective topology under the MP routing.

ROMST can be helpfully applied to the case of MP routing with one minor modification. More specifically, after construction of ROMST, we consider all nodes which are directly visible by a node u as the neighbour nodes of u, i.e.,:

$$N(u) = \{v | \forall v \in N, d_{u,v} \leqslant r(u)\}. \tag{43}$$

The main intuition behind such an approach is to increase the degree of nodes in the resulting graph (i.e., to make the topology denser), while keeping the transmission range of nodes as optimized as possible.

6. Performance evaluation

In the preceding sections, we proposed some optimal solutions, based on MILP formulation, along with three heuristic methods to solve both variants of MLSTC problem (i.e., MLSTC-CTree and MLSTC-CGraph). In this section, we analyze the performance of the proposed algorithms through computer simulations. Since there is no existing approximation algorithm for MRA-CTree, we compare OMST with the simple Minimum Spanning Tree (MST) method which works based on Prim's algorithm. We also evaluate our ROMST method in comparison with the BICONN-AUGMENT algorithm [4] (which is the optimal solution to MRA-CGraph). For simplicity, we use BICONN to refer to this algorithm.

6.1. Simulation environment

Our experiments were carried out using a customized implementation with an experimental setup similar to the one described in [3]. The simulations are conducted by varying the node density δ from 0.5 to 6.25 node/sq unit in a 4×4 two-dimensional area, while nodes are uniformly distributed in the area. In each experiment, after generating a placement of the nodes, we run the algorithms on the network consisting of those nodes. For each algorithm, we measure both the max energy consumption, as well as the max node degree in the resulting topology. We assume that the energy consumption in each node can be computed according to (4) with $n = 2$. Also, we fix the traffic rate λ to 1. For the sake of computing multiple node-disjoint paths in Multipath ROMST (to calculate the load and energy of nodes), we first choose the shortest path between source and destination nodes and then eliminate the links of this path and find the next shortest path, and so on, until no path remains between the pair. Finally, we break the traffic demand λ evenly between all computed paths. To obtain the maximum degree of nodes in OMST, we set d_{max} to ∞. Note that the results of the max energy consumption (the max node degree) are the average (the maximum) over 10 trials.

6.2. Experimental results

As the first simulation result, we compare the results of the max energy consumption acquired by our single-path heuristics OMST and ROMST, with the optimal ones obtained by the MILP approach. For convenience, we refer to the optimal solutions to MLSTC-CTree and MLSTC-CGraph by Opt-Tree and Opt-Graph, respectively. Also, we use Opt-Multipath to denote the optimal solution to MLSTC-CGraph under the MP routing method. Since the execution time of the MILP approaches becomes exponentially high in large topologies, we run this experiment only for small values of node density.

As we can see from Fig. 4, the maximum amount of energy consumed by OMST and ROMST is about 10–25% larger than the optimal ones used by Opt-Tree and Opt-Graph, respectively. Note that to find the optimal solutions, we must solve the MILP problems and unfortunately, this takes an inexplicable execution time even in small sized networks. Another point to mention is that as

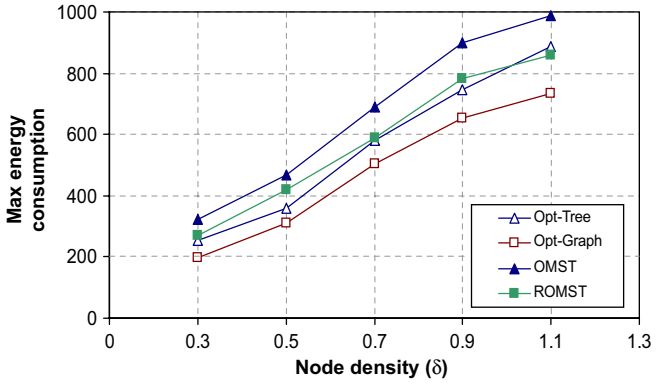

Fig. 4. Maximum energy consumption vs. node density under SP routing.

much as δ (or node density) increases, the max energy consumption increases too. With a higher value of node density in a fixed area, although the transmission ranges of nodes may become smaller, but due to the increase in traffic demand of nodes (which is equal to $(|N|-1)\times\lambda$ for each node), more energy should be consumed. Similarly, Fig. 5 shows the results of the max energy consumption obtained by two multipath approaches, Opt-Multipath and Multipath ROMST. As the figure illustrates, the optimal solution always yields a better result than the heuristic Multipath ROMST algorithm (i.e., 20–40% improvement, on average).

Figs. 6 and 7 compare the results of various algorithms in regard to the max energy consumption and the max node degree, respectively. First, with respect to maximum energy, our heuristic algorithms OMST and ROMST show much better results than the other two single-path methods (i.e., BICONN and MST). For example, ROMST outperforms BICONN by 40–50%. Since ROMST can exploit more adj-triangles for transformation, it results in a lower max energy utilization (about 10–20%), as compared with OMST. As we can see from the figure, the BICONN algorithm in many cases shows the worst results. This is mainly due to the fact that BICONN tries to minimize energy utilization of nodes only through reducing their ranges, while

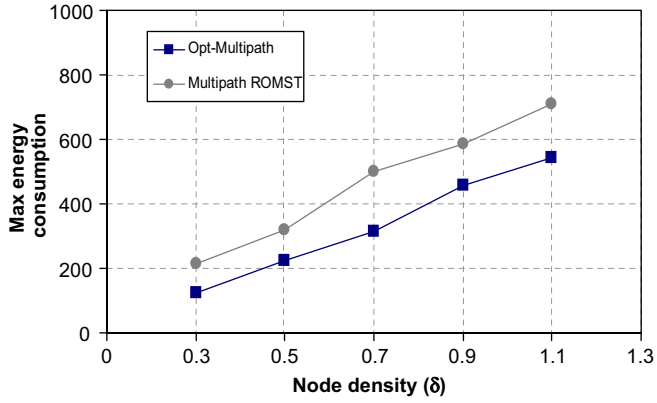

Fig. 5. Maximum energy consumption vs. node density under MP routing.

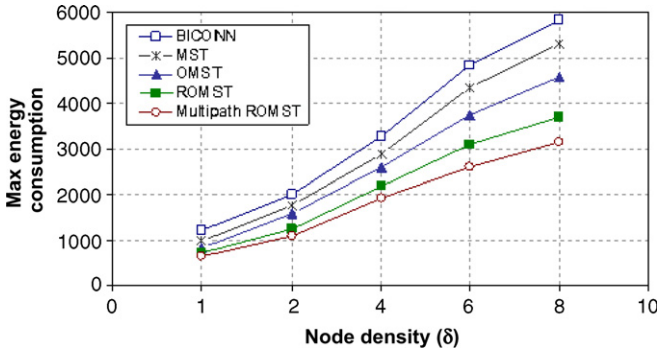

Fig. 6. Maximum energy consumption vs. node density.

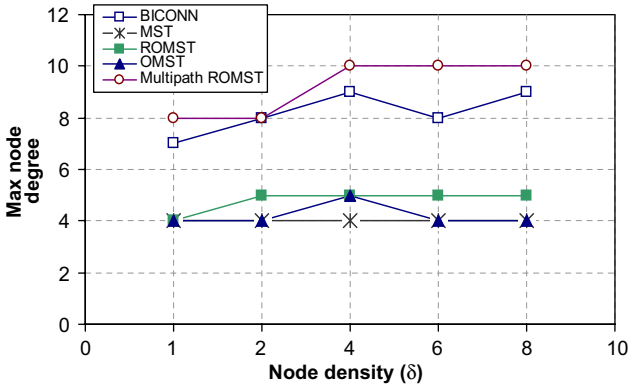

Fig. 7. Maximum node degree vs. node density.

ignoring the amount of traffic they transmit or forward. Moreover, due to concurrent usage of multiple paths, Multipath ROMST better distributes the traffic load over the network while also keeping the transmission range values small enough and thus, it gives much better results than all single-path methods (e.g., it outperforms ROMST by 10–15%, on average). On the other hand, as shown in Fig. 7, the MST algorithm exhibits the best performance with respect to the max node degree. However, our methods, specifically OMST, work very close to MST. Again, the BICONN algorithm produces the worst results among the single-path methods. Since ROMST does not preserve the tree-ness property, it usually incurs larger node degrees than OMST. Also, when using either of the algorithms, the max node degree does not differ much as the density changes. In addition, as we expect, the max node degree in Multipath ROMST is much higher than its single-path counterpart. This is mainly because Multipath ROMST considers all nodes within the transmission range of a node as its neighbours.

Finally, Table 1 shows the running time of various algorithms in topologies with different number of nodes. All of our programs run on one same workstation with a 2.4 GHZ Pentium IV processor. As the table clearly demonstrates, to obtain the optimal OPT-Tree solutions by the MILP approach may take an order of magnitude longer time than to get sub-optimal solutions by using

Table 1
Approximate execution times (ms) of different algorithms

Topology	MST	OMST	ROMST	Opt-Tree
$\|N\| = 10$	94	99	108	5000
$\|N\| = 15$	101	125	138	781000
$\|N\| = 20$	113	142	157	$>10^7$

some heuristic algorithms. These results become even worse for finding the Opt-Graph and Opt-Multipath solutions. Although in practice it is not efficient to use our integer programming model, it provides us a lower bound to evaluate our heuristic algorithms. Interestingly, the running times of our heuristics are comparable to the MST method. The main reason is that because of the simple computation of energy and load in our algorithms (which are stated in Lemmas 1–5), each adj-triangle takes a little time to be transformed in OMST and ROMST. Hence, constructing the initial MST takes the major portion of running time in both of these algorithms. Note that due to a slight difference between ROMST and Multipath ROMST, both methods have a very similar execution time.

7. Conclusions

In this paper, we proposed a new approach to the topology control (TC) problem for the purpose of energy conservation. In spite of classical topology management schemes that consider transmission range as the only factor which affects energy utilization, we also emphasized on the total load that each wireless node experiences. Consequently, a new problem, called MLSTC, was formally defined for topology control in wireless ad hoc networks. After discussing the tractability of the new problem, we proposed an approach based on integer programming to obtain the optimal solutions. Then, some heuristic algorithms were designed to find near-optimal solutions in polynomial time. Finally, the superiority of our algorithms over the existing TC methods was demonstrated through simulations.

Although the proposed algorithms can run in a distributed manner, each node needs the positional information of the whole network, or at least its cluster, to construct the topology. As a future work, we recommend design of algorithms that can work solely based on local information. From another perspective, working on schemes that consider interference as an additional optimization parameter beside energy can be truly useful.

Appendix A. NP-hardness Proof of MLSTC-CTree

For the purpose of showing that MLSTC-CTree is NP-hard, it can be replaced by the following simplified problem P1.

P1- The simplified MLSTC-CTree problem:

Instance: Given an energy constraint C, a network graph $G(V, E)$, and a destination node $t \in V$, where all nodes in V have the same traffic rate λ toward t.

Question: Is there a spanning tree T in G, rooted at t, that satisfies the following constraint: $\forall v \in V$, then $E(v) = L_v \cdot r^n(v) \leqslant C$?

As we can see, the problem P1 simplifies the original MLSTC-CTree problem in the senses that (1) the traffic rates are unified to a constant rate λ; and (2) instead of considering all nodes in V, only one destination node $t \in V$ is considered in P1. In other words, the rate function in P1 is defined by:

$$\lambda(x,y) = \begin{cases} \lambda & \text{if } (x \neq t) \text{ and } (y = t) \\ 0 & \text{otherwise} \end{cases}$$

Our objective here is to show that even the simplified problem P1 is too hard to solve in polynomial time. Before presenting the formal NP-hardness proof of P1, we first introduce one existing NP-hard problem, known as the *Non-uniform Load Balancing* problem [21], from which we construct our reduction to P1. For the sake of simplicity, we refer to the Non-uniform Load Balancing problem as P0.

P0- The Non-uniform Load Balancing problem:

Instance: A set of jobs $J = j_1, j_2, \ldots, j_k$, and a set of machines $M = m_1, m_2, \ldots, m_n$; for each job there is a set $S_i \subset M$ on which j_i can be run; each job has a requirement r_i which is equal to either 1 or 2.

Question: Is there an assignment from J to M such that each job $j \in J$ is assigned to a machine $m \in M$ so that the sum of the job requirements assigned to each machine is at most 2?

Problem P0 is NP-hard [21]. Now, we prove that the problem P1 is also NP-hard by constructing a reduction from P0 to P1. The reduction algorithm begins with an instance of P0. We construct a graph $G(V, E)$ to encode the instance of P0, as illustrated in Fig. 8.

Without loss of generality, we suppose that, among k jobs in J, there are $k1$ jobs with requirements of 1 and $k2$ jobs with requirement of 2. For simplicity of notation, we denote by $J1$ the set of jobs with requirements of 1 and by $J2$ the set of jobs with requirements of 2, respectively. It is apparent that $J = J_1 \bigcup J_2$ and $k = k_1 + k_2$, where $k_1 = |J_1|$ and $k_2 = |J_2|$. For each job j_i in J_1, we add one single node $s_{1,i}$ into V. For each job j_p in J_2, we first add two nodes $s'_{2,p}$ and $s_{2,p}$ into V; secondly, a link $(s'_{2,p}, s_{2,p})$ between these two nodes is added into E. Also, for each machine m_u in M, we place a node t_u into V. Finally, a termination node t is inserted into V and for each node t_u, a link (t_u, t) is added into E, which connects t_u to t. Note that as shown in Fig. 8, we place all nodes (except node t) inside a circle with radius of 0.5 unit such that the distance between every pair of nodes becomes less than 1. Moreover, every nodes t_u is laid on a circle centered at t with radius of one unit in a way that the distance of each node t_u from t becomes exactly 1. Apparently, such placement of nodes is always feasible and realistic. So far, the construction of V is completed.

For each job j_i in J, since it has a machine set $S_i \subset M$ on which it can run, then for each machine $m_u \in S_i$, we add one link $(s_{1,i}, t_u)$ into E if j_i is in J_1, and if j_i is in J_2, we

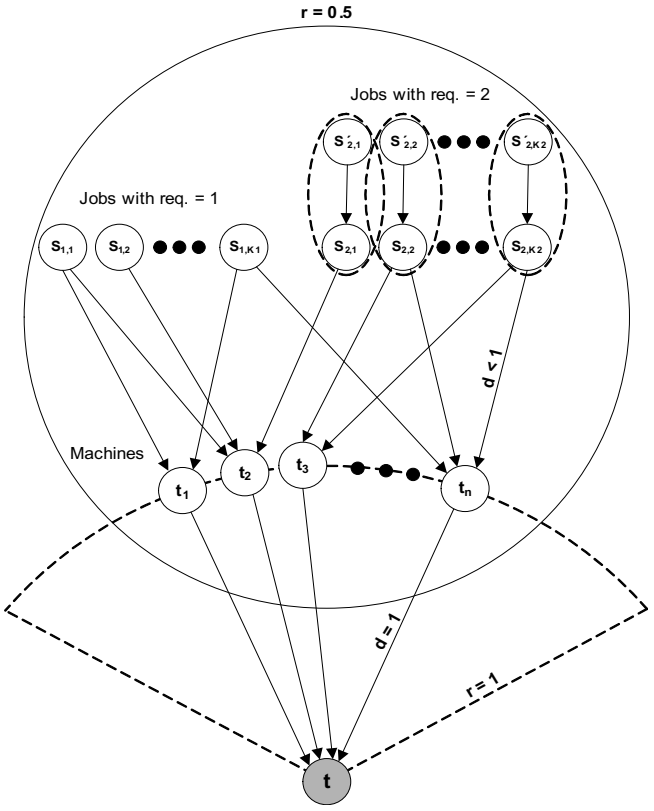

Fig. 8. The NP-hardness reduction.

add a link $(s_{2,i}, t_u)$ into E instead. Through these links, we encode the condition that job j_i can be assigned to a machine in S_i (namely node t_u is reachable from $s_{1,i}$ or $s_{2,i}$) if and only if $m_u \in S_i$. Now, the link set E and consequently the construction of graph $G(V, E)$ is completed.

It is clear that such construction of G can be made within polynomial time. Now, the problem P0 is transformed into an instance of problem P1 where the constructed graph G is the graph in P1, $C = 3, \lambda = 1$ and the question is to look for a spanning tree T for G that satisfies the energy constraint. In order to show that this transformation of the instance of P0 into the above instance of P1 is a reduction, we must verify that (1) a feasible assignment of jobs to machines in problem P0 always constructs a feasible spanning tree T in G; and (2) a solution to above instance of P1 always implies a feasible assignment to P0, accordingly.

First, if there is a feasible assignment of jobs to machines in problem P0, then it is not difficult to make a spanning tree T for G, accordingly. Specifically, for each job $j_i \in J_1$, if it is assigned to machine m_u, then we take the corresponding link $(s_{1,i}, t_u)$; for any job $j_i \in J_2$, if it is assigned to machine m_u, then we take both links $(s_{2,i}, t_u)$ and $(s'_{2,i}, s_{2,i})$. Moreover, all links (t_u, t) are taken. As such, we can easily show that the resulting sub-graph G' is a spanning tree for G. Remind that for every u, x $(1 \leqslant u \leqslant n, x = 0, 1)$, the distances from $s'_{2,i}$ to $s_{2,i}$, and also from $s_{x,i}$ to t_u are smaller than one unit. Considering $\lambda = 1$, since each machine has workload of at most 2 and each job can be assigned to at most 1 machine, it can be easily verified that $L_v \leqslant 3$ for any $v \in V$. More precisely, we have $L_{s_{1,i}} = 1, L_{s'_{2,i}} = 1, L_{s_{2,i}} = 2$, and $L_{t_i} \leqslant 3$. On the other hand, since the maximum distance of two nodes in V is one unit, i.e., $r(v) \leqslant 1 \quad (\forall v \in V)$, we can conclude that $E(v) = L_v \cdot r^n(v) \leqslant L_v \leqslant 3$, meaning that the energy constraint is simply satisfied. Therefore, T is a solution to problem P1 on G.

Conversely, suppose that there is a solution T satisfying P1, namely T is a spanning tree for G rooted at t such that the energy constraint of each node is preserved. Since we have assumed $C = 3$ and $\lambda = 1$ and also by observing that $r(t_u) = 1$ for each node t_u, at most 2 nodes can be contained in the sub-tree rooted at t_u. Node t_u itself consumes 1 unit of energy from the link (t_u, t). Therefore, at most two other paths can pass through this link. Since for each node $s'_{2,i}$, there is no direct link between it and t_u or t (there is only one link connecting it to $s_{2,i}$), its path to t must pass through the corresponding $s_{2,i}$. In addition, this path has to pass the same path from $s_{2,i}$ to t (otherwise, T will no longer be tree). So, the nodes $s_{2,i}$ and $s'_{2,i}$ are connected to each other in T. Hence, each sub-tree rooted at t_u can only have the following four options: contains 0 node, or any 1 node $s_{1,i}$, or any 2 nodes $s_{1,i}$ and $s_{1,j}$, or any one pair of bound nodes $s_{2,i}$ and $s'_{2,i}$. Following the spanning tree T, we can easily find an assignment of jobs accordingly: for any link $(s_{x,i}, t_u)$ in T, we simply assign the corresponding job j_i to the machine m_u. As we discussed above, the energy constraint $C = 3$ guarantees that the workload at each machine does not exceed 2. Moreover, since all nodes in V are covered by T, all jobs are assigned. Therefore, the corresponding assignment is a solution to the instance of P0.

References

[1] M. Hempstead, N. Tripathi, P. Mauro, G.Y. Wei, D. Brooks, An ultra low power system architecture for sensor network applications, Proc. ISCA (2005).
[2] H. Liu, T. Roeder, K. Walsh, R. Barr, E.G. Sirer, Design and implementation of a single system image operating system for ad hoc networks, Proc. ACM Mobisys (2005).
[3] E.L. Lloyd, R. Liu, M.V. Marathe, R. Ramanathan, S.S. Ravi, Algorithmic aspects of topology control problems for ad hoc networks, Proc. ACM MobiHoc (2002).
[4] R. Ramanathan, R. Rosales-Hain, Topology control of multihop wireless networks using transmit power adjustment, Proc. IEEE INFOCOM (2000).
[5] R. Wattenhofer, L. Li, P. Bahl, Y. Wang, Distributed topology control for power efficient operation in multihop wireless ad hoc networks, Proc. IEEE INFOCOM (2001).
[6] P. Santi, D.M. Blough, F. Vainstein, A probabilistic analysis for the range assignment problem in ad hoc networks, Proc. ACM MobiHoc (2001).
[7] S. Narayanaswamy, V. Kawadia, R.S. Sreenivas, P.R. Kumar, Power control in ad-hoc networks: theory, architecture, algorithm and implementation of the COMPOW protocol, in Proceedings of the European Wireless 2002, Next Generation Wireless Networks: Technologies, Protocols, Services and Applications, Florence, Italy, February, 2002.
[8] V. Kawadia, P.R. Kumar, Power control and clustering in ad hoc networks, Proc. IEEE INFOCOM (2003).

[9] J.E. Wieselthier, G.D. Nguyen, A. Ephremides, On the construction of energy-efficient broadcast and multicast trees in wireless networks, Proc. IEEE INFOCOM (2000).
[10] P.J. Wan, G. Calinescu, X.Y. Li, O. Frieder, Minimum-energy broadcast routing in static ad hoc wireless networks, Proc. IEEE INFOCOM (2001).
[11] V. Rodoplu, T.H. Meng, Minimum energy mobile wireless networks, IEEE J. Select. Areas Commun. 17 (8) (1999) 1333–1344.
[12] S. Singh, M. Woo, C.S. Raghavendra, Power-aware routing in mobile ad hoc networks, Proc. ACM Mobicom (1998).
[13] P.P. Pham, Sylvie Perreau, Performance analysis of reactive shortest path and multi-path routing mechanism with load balance, Proc. IEEE INFOCOM (2003).
[14] Y. Ganjali, A. Keshavarzian, Load balancing in ad hoc networks: single-path routing vs. multi-path routing, Proc. IEEE INFOCOM (2004).
[15] J. Pan, Y.T. Hou, L. Cai, Y. Shi, S.X. Shen, Topology control for wireless sensor networks, Proc. ACM Mobicom (2003).
[16] A. Muqattash, M. Krunz, CDMA-based MAC protocol for wireless ad hoc networks, Proc. ACM MobiHoc (2003).
[17] L. Hu, Distributed code assignments for CDMA packet radio networks, IEEE/ACM Trans. Netw. (1993) 668–677.
[18] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, Massachusetts, 1990.
[19] F. Kuhn, R. Wattenhofer, A. Zolinger, Worst-case optimal and average-case efficient geometric ad-hoc routing, Proc. ACM MobiHoc (2003).
[20] N. Li, J.C. Hou, L. Sha, Design and analysis of an MST-Based topology control algorithm, Proc. IEEE INFOCOM (2003).
[21] J. Kleinberg, Y. Rabani, and E. Tardos, Fairness in routing and load balancing, in: Proceedings of 40th Symposium on Foundations of Computer Science, 1999.