Resource Optimization Algorithm for Sparse Time-Driven Sensor Networks

María Luisa Santamaría[1], Sebastià Galmés[1] and Ramon Puigjaner[1]

[1] Dept. of Mathematics and Computer Science, Universitat de les Illes Balears
Cra. de Valldemossa, km. 7.5, 07122 Palma, Spain
{maria-luisa.santamaria, sebastia.galmes, putxi}@uib.es

Abstract. Time-driven sensor networks are devoted to the continuous reporting of data to the user. Typically, their topology is that of a data-gathering tree rooted at the sink, whose vertexes correspond to nodes located at sampling locations that have been selected according to user or application requirements. Thus, generally these locations are not close to each other and the resulting node deployment is rather sparse. In a previous paper, we developed a heuristic algorithm based on simulated annealing capable of finding an optimal or suboptimal data-gathering tree in terms of lifetime expectancy. However, despite the enhanced lifetime, the overall link distance is not optimized, fact that increases the need for additional resources (relay nodes). Therefore, in this paper we propose the Link Distance Reduction algorithm, with the goal of reducing link distances as long as network lifetime is preserved. The benefits of this new algorithm are evaluated in detail.

Keywords: Time-driven sensor networks, TDMA, spanning tree, minimum spanning tree, network planning.

1 Introduction

In *time-driven* or *continuous monitoring* sensor networks, communication is triggered by nodes, which regularly deliver sensed data to the user [1]. Thus, the traffic generated by these networks usually takes the form of a continuous and predictable flow, and therefore the use of a TDMA-based protocol becomes especially appropriate [2]. Furthermore, it is common that time-driven sensor networks are deployed in a structured manner [1][2], either by selecting strategic locations or by adopting some regular sampling pattern, and that data are routed to the sink through multi-hop predetermined paths [1]. These paths form a *reverse multicast* or *convergecast* structure called the *data-gathering tree* [2].

Because the strategic locations can well be far to each other or because the monitored variable can exhibit low spatial variability (as it is usually the case), it is not surprising that the resulting application-driven deployment is rather sparse. Sparse manually-deployed sensor networks have received less attention in literature (in contrast to dense and randomly-deployed networks), in spite of putting forward some interesting challenges. In essence, these come from the fact that large inter-node

distances can be impractical or unattainable for sensor nodes. Inevitably, given the limited communication range of nodes, any solution demands the introduction of additional resources, basically in the form of relay nodes (nodes with their sensing capability deactivated). This could be done by randomly scattering them over the whole sensor field, but this would yield to prohibitively large amounts of nodes precisely in sparse areas. Thus, we suggest that the problem can be better addressed from a network planning perspective, which tries to control the number of additional resources as long as the functionality and lifetime of the network are preserved.

As part of this strategy, in [3] we developed a heuristic method based on *simulated annealing* (SA) that finds an optimal (or suboptimal) data-gathering tree in terms of lifetime expectancy. However, in general the overall link distance of this tree can be substantially improved (decreased), fact that would contribute to reduce the amount of relay nodes required to make the network operational. Thus, in this paper we propose the *Link Distance Reduction* (LDR) algorithm, which starts from the data-gathering tree delivered by the SA-based algorithm, and then progressively reduces its overall link distance as long as the lifetime obtained from SA is not decreased.

Accordingly, the rest of the paper is organized as follows. In Section 2, we describe our network planning approach to the problem of topology construction for sparse deployments, along with related work. In Section 3, we motivate the need for an algorithm that reduces the sum of link distances of the SA graph. In Section 4, we describe the new algorithm (LDR) in detail. Then, in Section 5, we evaluate its performance via simulation. This also includes the issue of computational complexity. Finally, in Section 6, we draw the main conclusions and suggestions for further research.

2 Network Planning Approach and Related Work

The problem of constructing a data-gathering tree can be faced up from a network planning perspective or as part of a protocol-oriented design. In the latter case, a real-time distributed algorithm is designed, in order to be embedded into an adaptive routing protocol capable of supporting frequent topology updates. This strategy has been typically used in dense, randomly-deployed networks. In contrast, in the former case, the goal is to find an optimized static routing tree, and thus the algorithm to be designed is not subject to real-time requirements (although computation time is still of concern) and can be executed centrally. This approach becomes especially appropriate for sparsely-deployed sensor networks or for sensor networks where maintaining and updating a routing table at each node is computationally too expensive.

Despite its importance, the network planning approach has received less attention than the approach centered on adaptive routing. Yet, some works deserve a description. For instance, the work presented in [4] considers the problem of designing a data-gathering tree with the goal of reducing the total amount of data transmitted. Hence, in contrast to our premises (see Section 3), its main focus is on a packet aggregation scheme. Analogously, the emphasis of [5] is also on packet aggregation. A common feature to these works, as stated by [6], is that their ultimate goal is to minimize the total energy consumption, thus ignoring the fact that some

nodes may deplete their energy faster than others and cause loss of coverage. Thus, the algorithm proposed in [6] focuses on maximizing the lifetime of the network, defined as the time until first node death (optimization with strict coverage requirements). However, this work adopts several assumptions that significantly simplify the analysis of the problem. One of them is that nodes cannot adjust their transmission power, which is not always the case. This assumption eliminates the dependence on distance and thus it reduces the complexity of the proposed algorithm. Another simplifying assumption is the adoption of a packet aggregation scheme that reduces the number of packets to be forwarded by a node to the number of direct descendents it has. Compared to this work, our particular approach adopts the same definition of lifetime, but it uses a more complete energy consumption model that does not rely on the abovementioned assumptions.

Another simplification present in [6] is the assumption that the initial network is fully connected. Since this is not necessarily true, especially in sparse deployments, the network planning perspective has also considered the possibility to introduce additional resources in the network, in the form of relay nodes. In this context, several works assume that nodes have a maximum transmission range, and then formulate the problem of introducing the fewest number of relay nodes so that a certain degree of network connectivity is achieved. Examples of these works are [7] and [8], which consider flat topologies, and [9] and [10], which assume a two-tiered architecture where relay nodes always belong to the backbone network. Essentially, the theoretical formulation of these topological problems is that of a *Steiner tree with minimum number of Steiner points* (ST-MSP), a problem that is known to be NP-hard. However, as stated above, the focus of these works is on network connectivity rather than on lifetime (at most, lifetime is partially addressed by considering transmission ranges and the so-called *node degrees*). By contrast, our network planning approach addresses both network lifetime and minimization of the number of Steiner points (resource optimization) by means of the following two stages:

- *Elementary tree construction*. The construction of an elementary data-gathering tree consists of determining a spanning tree that connects all regular or duty nodes (nodes at locations of interest) to the sink, with a view to maximizing the lifetime of the sensor network. In this process, we assume that these nodes have full power control capability with the only limit of their own energy resources (battery). In other words, the likely existence of a maximum transmission range is ignored at this stage. Under this assumption, the work presented in [3] showed that the problem of finding a data-gathering tree with optimal lifetime, by including in the analysis the workload of every node to a maximum extent, is NP-hard. Then, a heuristic method based on simulated annealing was proposed, which it was shown to yield, in general, a spanning tree with near-optimal lifetime in linear time.
- *Placement of additional nodes*. In the case of sparse deployments, the data-gathering tree that results from the previous stage usually contains link distances that exceed some maximum transmission range. In this case, our strategy consists of regularly inserting the necessary number of relay nodes into the uplink of every regular node. This is the technique of *steinerized edges*, which was also considered in [7], [8], [10] and [11]. In the latter, the number of inserted nodes was related to the corresponding lifetime enhancement.

3 Problem Motivation

The work presented in this paper is halfway between the two stages of the topology construction process described in the previous section. The reason is that although the SA-based algorithm generates a spanning tree with optimal or suboptimal lifetime, in general the sum of the resulting link distances can be substantially decreased without penalizing such lifetime. This, in turn, may contribute to reduce the amount of additional resources required in the second stage. In this section, we illustrate these ideas by means of a simple test scenario; however, for the sake of completeness, we first set up a lifetime model for a time-driven sensor network with N nodes.

3.1 Lifetime Model

A lifetime model requires an energy consumption model. The energy consumption model used in the present paper was first developed in [3] (on the basis of a radio model described in [12]), under the following premises:

- Only the energy wasted by the node transceivers is considered.
- No data aggregation, since the spatial correlation among samples (readings) in sparse scenarios tends to be low. In addition, by ignoring data aggregation a worst-case lifetime analysis is performed.
- Some low duty-cycle MAC protocol, such as TDMA, governs the access of nodes to the wireless medium.
- Time is divided into rounds or reporting periods, whose duration is specified at the application or user levels and is typically much larger than the duration of packets and TDMA frames. In addition, this allows for neglecting the delays incurred by TDMA, whatever slot assignment scheme is used and despite the absence of a packet aggregation scheme.

Then, in order to characterize the traffic workload of each node $i = 1 \ldots N$, two magnitudes were introduced:

- $g(i)$: number of packets to be generated by node i per communication round (*generating parameter*). This specification takes into account heterogeneous scenarios where the sampling process needs to be more intensive in some locations than in others.
- $\sigma(i)$: number of packets to be forwarded by node i during every communication round (*forwarding parameter*).

Note that these magnitudes become time-independent when dealing with static data-gathering trees, as it is the case of the present paper. Note also that if $CH(i)$ denotes the set of child nodes of node i, the following equality holds:

$$\sigma(i) = \sum_{j \in CH(i)} g(j) + \sum_{j \in CH(i)} \sigma(j) \,, \forall i \,. \tag{1}$$

Now, let $E_G(i)$ ($E_F(i)$) be the energy consumed by node i to generate and transmit (receive and forward) a packet. Then, the total energy consumed by this node during a communication round, namely $E(i)$, can be expressed as follows:

$$E(i) = g(i) \cdot E_G(i) + \sigma(i) \cdot E_F(i) \,,\, \forall i \,. \tag{2}$$

Furthermore, if E_R is the energy consumed by any node to receive a packet and $E_T(i)$ is the energy consumed by node i to transmit a packet at distance $d(i)$, we can set up the following equations:

$$E_G(i) = E_T(i) \,,\, \forall i \,. \tag{3a}$$

$$E_F(i) = E_R + E_T(i) \,,\, \forall i \,. \tag{3b}$$

The energy consumed to receive a packet is the energy dissipated by the node transceiver circuitry to perform such operation. Because all nodes are assumed to be provided by the same vendor, this energy can be considered constant throughout the network. On the other hand, the energy wasted to transmit a packet includes a distance-dependent component in addition to the energy dissipation, and thus it generally varies from node to node (in spite of being provided by the same vendor). These statements are made explicit via the so-called radio model [12]:

$$E_R = E_{elec} \cdot m \,. \tag{4a}$$

$$E_T(i) = E_{elec} \cdot m + E_w \cdot m \cdot d^f(i) \,,\, \forall i \,. \tag{4b}$$

Here, E_{elec} stands for the energy dissipated by the transceiver circuitry to transmit or receive a single bit, $E_w \cdot d^f(i)$ is the energy radiated to the wireless medium to transmit a single bit over a link of distance $d(i)$, f is the path-loss exponent and m is the packet size in bits. In turn, the distance-dependent component depends on the distance itself:

$$d \le d_0 \Rightarrow E_w = E_{fs} \,,\, f = 2 \,. \tag{5a}$$

$$d > d_0 \Rightarrow E_w = E_{mp} \,,\, f > 2 \,. \tag{5b}$$

Equation (5a) models free space propagation, which is valid below the reference distance (d_0) (although beyond the so-called far-field region of the transmitting antenna [12]), whereas equation (5b) reflects multi-path fading effects. In this latter case, a typical value for the path-loss exponent is 4. By combining equations (2), (3a)-(3b) and (4a)-(4b), the total energy wasted by a node in every communication round can be rewritten in the following way:

$$E(i) = (g(i) + 2\sigma(i)) \cdot E_{elec} \cdot m + (g(i) + \sigma(i)) \cdot E_w \cdot m \cdot d^f(i) \,, \forall i \,. \quad (6)$$

Note that in this analysis the energy wasted to wake up a node has been neglected. This is an acceptable and usual approximation, which in addition makes the analysis independent of the particular slot assignment scheme in the case of TDMA. It can also be noticed that expression (6) reveals that the energy consumed by a node depends on three major factors: workload, represented by the traffic parameters $g(i)$ and $\sigma(i)$, target distance and packet size.

The energy consumption model given by (6) can now be embedded into a lifetime model. By assuming that lifetime is defined as the time until first node death, we can express it in rounds as follows (B is the energy initially available at all nodes, that is, the battery):

$$L = \frac{B}{\max\{E(i) \,, i = 1 \ldots N\}} = \frac{B}{E(i)\big|_{\max}} \,. \quad (7)$$

3.2 Illustrative Example

We consider the scenario detailed in Fig. 1, where 8 regular nodes are supposed to send 1 packet to the sink per round of communication, that is, $g(i) = 1 \,, i = 1 \ldots N$ (for simplicity, with no loss of generality). For the radio model, we take the data from a realistic radio module previously introduced in [13]: E_{elec} = 50 nJ/bit, E_w = 10 pJ/bit/m^2 and f = 2 if the transmission distance is below 75 m (the so-called reference distance), E_w = 0.0013 pJ/bit/m^4 and f = 4 if the transmission distance is above 75 m, and B = 15 kJ. For the packet size, we assume a slightly large value of 125B, although the choice for this parameter is not relevant to our analysis. Under all these assumptions, Fig. 1 also shows the graph generated by the SA algorithm developed in [3]. This graph yields a lifetime of 722168 rounds, determined by node 8 – see expression (7). Besides, this graph reveals one of the main features of the SA algorithm: it focuses on maximizing the lifetime of the network as defined by (7), but not on enhancing the lifetimes or transmission distances of particular nodes unless this could have a positive impact on the final result. For instance, it is very probable that node 4 could have been connected to node 2 instead of node 5 without perturbing the lifetime of the network.

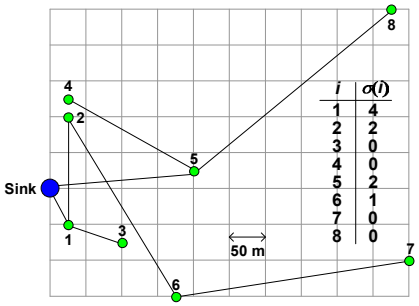


Fig. 1. Node deployment and graph generated by the SA algorithm. Forwarding parameters are indicated.

In this context, a reference algorithm is the Minimum Spanning Tree, which finds the spanning tree that minimizes the overall link distance (if link cost is defined as the Euclidean distance). In particular, when applied to the node deployment of Fig. 1, the result is the graph shown in Fig. 2 in solid lines. The average link distance has decreased from 207.970 m (with SA) to 148.302 m, which is the minimum achievable value. This represents a reduction of approximately 30%. However, the lifetime of the network has also decreased to 514451 rounds (this time determined by node 7), which is also about 30% smaller. The reason is that, by focusing exclusively on the total link distance, the MST algorithm ignores the workload supported by nodes and thus it does not optimize the lifetime of the network.

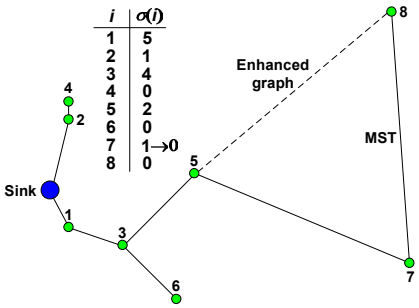


Fig. 2. MST and enhanced graphs. Forwarding parameters are indicated, including the change of $\sigma(7)$ from one graph to the other.

Obviously, a question that arises is whether an intermediate solution is feasible. In this respect, the graph shown in Fig. 2 – with the dashed line instead of line 8-7, constitutes a positive answer. This graph, which results from applying the combined SA+LDR algorithm (LDR is described in the next section), exhibits the same lifetime as the SA graph (722168 rounds, again determined by node 8), but its average link distance is now 148.855 m, very close to the minimum achieved by the MST algorithm. In fact, it can be noticed that the two graphs in Fig. 2 are very similar, although their small dissimilarity has an important effect on network lifetime.

4 Link Distance Reduction Algorithm

Starting from the spanning tree generated by the SA algorithm, the proposed Link Distance Reduction algorithm is a heuristic iterative method that is intended to transform this tree into another one with less average link distance, as long as lifetime is not degraded. In essence, the strategy is to check the connections of all nodes to their parents, trying to find closer parents while preserving the tree structure (no cycles are formed) and maintaining or even increasing the lifetime of the network (recall that, in general, the SA algorithm yields a near-optimal solution).

Before proceeding to the specification of the algorithm, let us introduce some useful definitions and naming conventions:

- Let $t = (V, E)$ denote a particular spanning tree, where V is the set of $N+1$ vertices representing the N (regular) nodes and the sink, and E is a collection of two-element subsets of V that represent connections between pairs of nodes. Let T denote the set of all trees spanning all vertices in V.
- Given that t is a spanning tree, every edge in E can be expressed as a pair $(i, p(i)), i = 1 \ldots N$, where $p(i)$ denotes the parent of node i (that is, the node to which node i sends packets on the way to the sink). Thus, $p(i) = 0$ if node i is directly connected to the sink (it is assumed that node 0 is the sink).
- As stated above, the algorithm explores potential parents for each node. Thus, we introduce $p^*(i)$ to denote the parent that is tried for node i at a particular stage of the execution process.
- In our formal specification of the algorithm, a state corresponds to a particular spanning tree. Accordingly, three variables are managed. The first one is `Initial State`, which denotes the spanning tree obtained from the SA algorithm. The LDR algorithm starts from this state. Then, `Current State` and `New State` denote the spanning trees at the beginning and end of each iteration, respectively. Therefore, `New State` is different from `Current State` as long as at least one parent of any node is changed during the corresponding iteration.
- Let $d(i, j)$ be the Euclidean distance between any two vertices $i, j = 0 \ldots N$. Accordingly, let **A** be an NxN matrix such that $\mathbf{A}(u, v) = i$ and $\mathbf{A}(u, v') = j$ with $v < v'$ means that $d(u, i) \le d(u, j)$, with $u, v, v' = 1 \ldots N$, $i, j = 0 \ldots N$ and $i, j \neq u$. This matrix depends exclusively on the deployment of the network and thus it needs to be calculated only once. Also, let **B** be a row vector with N components such that $\mathbf{B}(v) = i$ and $\mathbf{B}(v') = j$ with $v < v'$ means that $d(i, p(i)) \ge d(j, p(j))$, with $i, j, v, v' = 1 \ldots N$. Essentially, for a given spanning tree, **B** is the list of nodes in decreasing order of their uplink distances. Thus, it depends on the particular spanning tree. In the algorithm, **B** is re-calculated at the beginning of each iteration.
- Finally, `No Cycle` and `Lifetime Not Degraded` are two self-explanatory logical variables that represent the two conditions that must be fulfilled in order to accept a potential parent. The former implies determining that the connection of a node to a potential parent does not form any cycle. Under this condition, the latter

implies the re-evaluation of network lifetime according to (6) and (7) (note that the two conditions should be checked in this order).

Based on these definitions, the pseudo-code for the algorithm can be written as follows:

```
Link Distance Reduction Algorithm
1  Set A;
2  New State ← Initial State; {Initial State = Spanning tree
   obtained from SA}
3  repeat
4     Current State ← New State;
5     Set B;
6     for i = 1 to N do
7        j ← 1;
8        p*(B(i)) ← A(B(i), 1);
9        while p*(B(i)) <> p(B(i)) do
10          if (No Cycle == True) and (Lifetime Not Degraded
            == True) then
11             p(B(i)) ← p*(B(i))
12          else
13             j ← j+1;
14             p*(B(i)) ← A(B(i), j);
15          end if
16       end while
17    end for
18 until New State == Current State
```

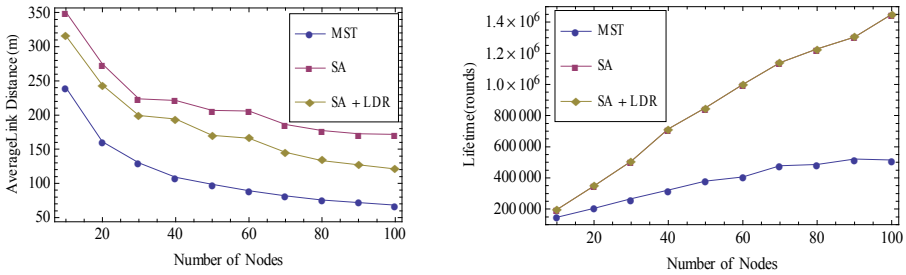
In this algorithm, lines 3-18 correspond to a single iteration. During an iteration, the connections of all nodes to their parents are checked in decreasing order of their link distances (lines 6-17), and, for each node, potential parents are examined in increasing order of their distance to such node, that is, starting with the closest one (lines 9-16). Thus, for any given node, in the worst case this process yields the same parent node as the previous iteration. Also note that, as stated before, the algorithm executes a new iteration as long as at least one change is registered in the previous one. In effect, any change requires revisiting the situation of all nodes in a subsequent iteration; for instance, if node i was checked before node j in iteration k, and the connection of node j was changed during such iteration, maybe a potential parent for node i that was rejected in iteration k can now be accepted in iteration k+1.

5 Simulation Results

In order to validate the proposed algorithm and evaluate its computational complexity, we performed several simulation experiments using the *Mathematica* software. The scenario for these experiments was a 1000m x 1000m field, with coordinate $x \in [0,1000]$ and coordinate $y \in [-500,500]$. We placed the sink at $(0,0)$ and for the radio model we used the values of subsection 3.2. Each simulation experiment corresponded to a given number of nodes, which was varied between 10

and 100 in steps of 10. The number of runs in all simulation experiments was adjusted so as to produce 95% confidence intervals below 10%. In turn, each run consisted of generating a random deployment with the corresponding number of nodes, and then executing, on the one hand, the MST algorithm, and, on the other hand, the SA followed by the LDR algorithms. The result of each run consisted of the graphs generated by the MST, SA and SA+LDR algorithms, as well as several performance metrics evaluated on these graphs. Specifically, the most significant performance metrics were the following ones:

- *Average link distance*. For any data-gathering tree, this is the sum of link distances divided by the number of nodes (there are as many links as nodes). In order to reduce the amount of additional resources that could be required, the goal is to reduce the average link distance as much as possible.
- *Lifetime*. This is the network lifetime as given by (7).
- *Number of relay points*. This refers to the total number of relay points (nodes) that should be marked on the graph of the network in order to fulfill some maximum transmission range requirement.



(a) (b)

Fig. 3. Average link distance (a) and lifetime (b) versus number of nodes. The SA and SA+LDR curves in (b) are in full agreement.

Fig. 3 shows the results obtained for the first two metrics. In particular, Fig. 3(a) shows the downward trend of the average link distance (in meters) with the number of nodes, irrespective of the algorithm. This behavior was completely predictable, because the greater the amount of nodes deployed in a fixed-size region, the closer they are to each other. Moreover, this figure shows that the lowest average link distance corresponds to the MST algorithm, which is obvious since the only goal of MST is to minimize the sum of link distances. On the other hand, since the SA algorithm only takes link distances into account as long as they contribute to increase the lifetime of the network, it generates an average link distance substantially greater than that of MST. Between these two extreme cases, the combined SA+LDR algorithm represents the best tradeoff, as it tries to reduce the overall link distance as long as the lifetime obtained from SA is not degraded. Specifically, it can be noticed that the gap between the SA and SA+LDR curves is significant, fact that validates the role of the LDR algorithm. Finally, another observation is that the MST curve is smoother than the others. This is due to the inherent randomness of the simulated annealing technique on which the SA and SA+LDR algorithms are based, which

generally leads to a larger solution space than with the MST algorithm (in this case the solution space is typically a single graph or at most a very small set of graphs).

The downward trend of curves in Fig. 3(a) is in line with the upward trend of the lifetime curves shown in Fig. 3(b). This is due to the fact that transmission distance plays a dominant role in energy consumption, and its progressive decrease with the number of nodes has more impact than the corresponding increase in traffic load (represented by the forwarding parameters) – see expression (6). As it can also be noticed, the curves corresponding to SA and SA+LDR are completely superposed, which is not surprising as the LDR algorithm does not degrade the lifetime of the SA graph, and usually does not improve it either. Finally, another observation about Fig. 3(b) is the growing value of the gap between the two curves. This is explained by the fact that only the SA and SA+LDR algorithms focus on network lifetime.

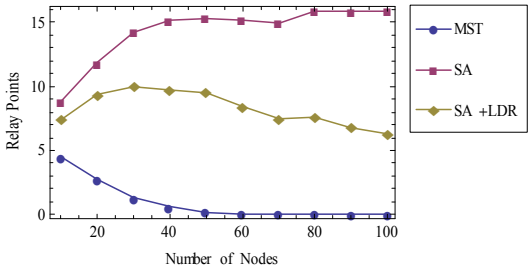


Fig. 4. Number of relay points versus number of nodes.

Fig. 4 plots the evolution of the number of relay points as the number of nodes increases, for a maximum transmission range of 250m. In this case, the three algorithms experience different trends. These can be explained by viewing the average link distance and the number of nodes as separate factors: on the one hand, reducing the average link distance (obviously as a consequence of increasing the number of nodes) contributes to reducing the number of relay points required for network connectivity; on the other hand, since there can be one or more relay points per (regular) node, there is some positive correlation between these two magnitudes. In the case of MST, which focuses exclusively on reducing the overall link distance, the former factor dominates. However, the opposite happens with SA, since the whole set of link distances play a secondary role in this algorithm. Again, the SA+LDR curve is in an intermediate position, and thus it exhibits a local maximum at a relatively low number of nodes. In particular, the gap between the SA and SA+LDR curves as well as the decreasing trend of the latter beyond its local maximum validate the significance of LDR as a resource optimization algorithm.

In addition to guaranteeing network connectivity, the introduction of relay nodes enhances the lifetime of the network beyond the values shown in Fig. 3(b) (according to the energy consumption model considered so far). Certainly, this is applicable to both the SA+LDR and MST algorithms, but the difference is that the starting point is generally better for the former than for the latter. Thus, the enhanced lifetime is expected to be greater with SA+LDR than with MST, at least in a statistical sense.

In addition to the above metrics, we introduced two complementary measures:

- *Average energy consumption*. This is the average energy consumed by a node per round of communication. It is the result of dividing the total energy consumed by the network during a round of communication by the number of nodes. Although the definition of lifetime used in this paper relegates this measure to a secondary role, it could achieve more importance from alternative viewpoints. For instance, if electrical power supply for nodes was feasible (as it is the case of some scenarios), lifetime would not be crucial anymore, but preserving energy consumption would still be a relevant issue (for environmental protection).
- *Number of cross-points*. This is the number of crossing or intersection points among the edges of the graph that represents the sensor network. It can be viewed as a rough estimate of the entropy of such graph. Although this measure may also appear to be secondary, it could become more meaningful in some scenarios. For instance, in case directive antennas were used for inter-node communication, the number of cross-points would have an impact on the amount of interferences.

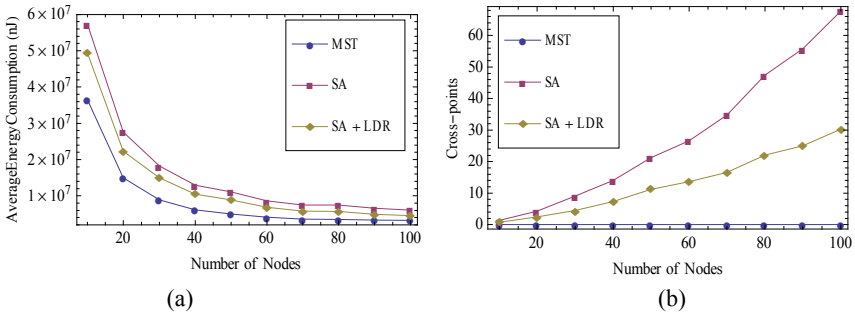


Fig. 5. Average energy consumption (a) and number of cross-points (b) versus number of nodes.

Fig. 5 plots the evolution of these additional metrics as the number of nodes increases, for the three algorithms considered so far. In particular, the results shown in Fig. 5(a) (expressed in nJ) are completely correlated with those in Fig. 3(a). Again, this is due to the fact that transmission distance plays a dominant role in energy consumption. So, we can conclude that MST outperforms SA and SA+LDR when the emphasis is put on the overall energy consumption, in contrast to what is shown in Fig. 3(b), which focuses on network lifetime. Regarding Fig. 5(b), we should recall that the SA algorithm focuses on maximizing the lifetime of the network as defined by (7), but not on enhancing the connections of the nodes that do not determine this lifetime. This, in addition to the inherent randomness of this algorithm, makes the resulting graph rather chaotic, with a large amount of cross-points that obviously increases with the number of nodes. From this point of view, the MST algorithm exhibits an opposite behavior: it produces a graph with no cross-points, as shown in Fig. 5(b). Between these two extremes, the SA+LDR curve also shows an upward trend, but smoother than in the SA case.

Finally, we also evaluated the computational complexity of the proposed algorithm. Although from a network planning perspective the topology design

algorithms are not subject to strong real time requirements, the issue of computational complexity is still of concern, since these algorithms may have to be executed from time to time for network reconfiguration. In this sense, the results obtained for LDR are very satisfactory, since the trend of the regression curve shown in Fig. 6 is practically linear. Specifically, this curve represents the evolution of the average number of iterations (see Section 4) as the number of nodes increases. In addition to its linear trend, this average takes on relatively moderate values, as it varies from approximately 30 for 10 nodes to around 1800 for 100 nodes.

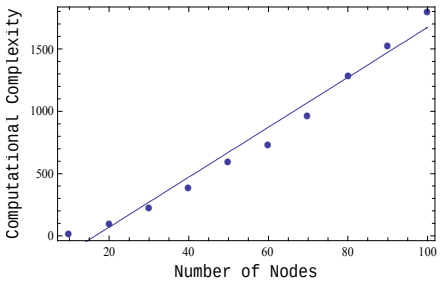


Fig. 6. Computational complexity of LDR as a function of the number of nodes.

6 Conclusions and Further Research

In this paper, we have developed the Link Distance Reduction algorithm, a heuristic method that minimizes the overall link distance of a time-driven sensor network graph while preserving or even improving its lifetime. This algorithm uses hill-climbing without backtracking, and thus it could stop at a local optimum. However, in contrast, this limitation has favored its simplicity and computational efficiency, and at the same time it has not precluded the algorithm from yielding significant reductions in average link distance and number of relay points.

The work presented in this paper can be extended to other energy-consumption models, as well as to scenarios where the deployment density is kept constant or the maximum transmission range of nodes is varied. A more advanced research would consist of developing a modified version of the original SA algorithm that included the average link distance minimization, in order to enable the achievement of the global optimum (although at the expense of increasing the computational complexity). Another issue would be the distributed implementation of the proposed algorithms. Also, the issues of connectivity, lifetime enhancement and fault-tolerance could be jointly treated by replacing single nodes with clusters of cooperative transmitting nodes (see [14] for a survey on MIMO/MISO techniques). Finally, our research results are progressively being integrated in *NetLife*, a software tool for planning time-driven sensor networks.

Acknowledgments. This work has been supported in part by the Spanish Ministry of Science and Technology under contract TIN2009-11711.

References

1. Akkaya, K., Younis, M.: A Survey on Routing Protocols for Wireless Sensor Networks. In: Ad Hoc Networks, vol. 3, pp. 325-349. Elsevier (2005)
2. Krishnamachari, B.: Networking Wireless Sensors. Cambridge University Press (2005)
3. Santamaría, M. L., Galmés, S., Puigjaner, R.: Simulated Annealing Approach to Optimizing the Lifetime of Sparse Time-Driven Sensor Networks. In: 2009 IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 193-202. IEEE, Inc. (2009)
4. Goel, A., Estrin, D.: Simultaneous Optimization for Concave Costs: Single Sink Aggregation or Single Source Buy-at-Bulk. In: ACM Symposium on Discrete Algorithms (2003)
5. Enachescu, M., Goel, A., Govindam, R., Motwani, R.: Scale Free Aggregation in Sensor Networks. In: First International Workshop on Algorithmic Aspects of Wireless Sensor Networks (2004)
6. Wu, Y., Fahmy, S., Shroff, N. B.: On the Construction of a Maximum-Lifetime Data Gathering Tree in Sensor Networks: NP-Completeness and Approximation Algorithm. In: 27th Conference on Computer Communications (INFOCOM) (2008)
7. Cheng, X., Du, D., Wang, L., Xu, B.: Relay Sensor Placement in Wireless Sensor Networks. In: Wireless Networks, vol. 14, no. 3, pp. 347-355. Springer (2008)
8. Chen, D., Du, D., Hu, X., Lin, G., Wang, L., Xue, G.: Approximations for Steiner Trees with Minimum Number of Steiner Points. In: Journal of Global Optimization, vol. 18, no. 1, pp. 17-33. Springer (2000)
9. Tang, J., Hao, B., Sen, A.: Relay Node Placement in Large Scale Wireless Sensor Networks. In: Computer Communications, vol. 29, no. 4, pp. 490-501. Elsevier (2006)
10. Kashyap, A., Khuller, S., Shayman, M.: Relay Placement for High Order Connectivity in Wireless Sensor Networks. In: 25th Conference on Computer Communications (INFOCOM) (2006)
11. Galmés, S.: Lifetime Planning for TDMA-Based Proactive WSN with Structured Deployment. In: 2007 IFIP/ACM Latin American Networking Conference (LANC) (2007)
12. Rappaport, T. S.: Wireless Communications: Principles and Practice. Prentice-Hall (2002)
13. Heinzelman, W. B., Chandrakasan, A. P., Balakrishnan, H.: An Application-Specific Protocol Architecture for Wireless Microsensor Networks. In: IEEE Trans. on Wireless Communications, vol. 1, no. 4, pp. 660-670 (2002)
14. Ahmad, M. R., Dutkiewicz, E., Huang, X.: Performance Evaluation of MAC Protocols for Cooperative MIMO Transmissions in Sensor Networks. In: 5th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (2008)