1

A Distributed Algorithm for Maximum Lifetime Routing in Sensor Networks with Mobile Sink

Marios A. Gatzianas and Leonidas G. Georgiadis, *Senior Member, IEEE*

Abstract

We consider a noise-limited wireless sensor network that consists of battery-operated nodes which can route information to a mobile sink in a multi-hop fashion. The problem of maximizing the network's lifetime, defined as the period of time during which the network can route a feasible flow to each sink location subject to power/energy constraints, is cast into a linear program, reduced into a simpler equivalent form and solved via dual decomposition. The unknowns are the sink sojourn times and the routing flow vector for each sink location. The presence of a mobile sink presents new challenges but the problem structure can still be exploited to find the optimal solution. A distributed algorithm based on the subgradient method and using the sink as leader is proposed and its performance is evaluated through simulation for random networks. The algorithm's requirements in memory are also provided.

Index Terms

Sensor networks, network lifetime, duality, subgradient method, primal recovery.

I. Introduction

Wireless sensor networks (WSNs) typically consist of a large number of nodes used for gathering information from a geographical area and sending it in a multi-hop fashion to designated sink nodes for further processing. They are often employed in environmental monitoring applications, which requires their topology to be either fixed or slowly varying in a controllable manner, and their operational lifetime

The authors are with the Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece, e-mails: {mgkatzia,leonid}@auth.gr.

Part of this work was presented at the 2006 European Wireless conference, April 2-5, Athens, Greece.

is of the order of weeks or months. In addition to the usual ad-hoc nature of their formation, they possess the unique characteristics of even more stringent energy/power requirements, due to their long-term operation, as well as high correlation between the data generated in proximal nodes. Hence, issues such as source coding, resource allocation (power, bandwidth etc.) and routing policies become very important. This paper deals with the last two issues by jointly optimizing resource allocation and routing in order to maximize a suitably defined notion of network lifetime for a sensor network where the sink is mobile. Furthermore, due to the ad-hoc nature of these networks, we specifically direct our attention towards algorithms that are amenable to a distributed implementation.

Although the individual facets of the energy-efficient routing problem, i.e. optimal routing and resource allocation, have been extensively studied in isolation (see [1], [2] and the references in [3]), recently the focus has shifted towards cross-layer optimization. Reference [3], with which this work shares many traits (though our model does not fit in its framework), provides a general methodology for optimizing various performance metrics through the formulation of convex problems with the flows and allocated resources as unknowns. Similarly, [4] derives a schedule and power allocation policy that, for given minimum rate requirements, minimizes the *average* total power expended by the network. Finally, in the limiting case where the number of nodes grows denumerably to infinity (i.e. the nodes are so closely placed that macroscopic quantities can be defined as aggregate averages) the problem of minimizing the average total power can be cast into a partial differential equation (pde) similar to the ones encountered in electrostatics [5], [6].

All of the aforementioned literature considers the total expended power (usually in its long-term average form) as the sole metric of power efficiency which, as observed in [7], can be misleading in the case of sensor networks, since the network lifetime is not directly related to the total expended power. Hence, a min-max formulation is more appropriate than the standard min-sum previously used. This is performed in [8], where an interference-free sensor network in the low-SNR regime with a single stationary sink is considered and the lifetime problem is solved through subgradient projection. Reference [9] considers a general interference-limited network and formulates the combined scheduling/routing/resource allocation as a non-convex problem which is then approximated by a convex one in the high-SNR regime. Again, a single stationary sink is considered.

The case of a mobile sink has received less attention than the stationary sink, although it has been demonstrated in [10]– [12] that a mobile sink can potentially increase the network's lifetime by causing lower saturation on the nodes close to the sink due to its changing positions. Such a sink may be a small vehicle, possibly unmanned, equipped with wireless transceiver. The vehicle may stop at specified

locations (i.e. terminals), where it can dock and collect available network data without obstructing other vehicles (e.g. consider a WSN employed over cultivated farmland for the purpose of measuring air humidity. Since an arbitrary sink movement could damage the crops, it is more likely that the sink moves among predetermined locations). Recently, [13] examined multiple mobile sinks on sensor networks and developed approximation algorithms for special cases of NP-complete problems. However, the proposed algorithm is not easily adapted to a distributed environment. The main contribution of this paper is the development of an efficient distributed (and parallel) algorithm for a single mobile sink sensor network, offering an alternative to the centralized solution of [12]. As such, considerable emphasis will be placed on issues related to distributed implementation.

The rest of the paper is organized as follows. Section II contains the model description and statement of the problem. The dual formulation, along with supporting analysis, is presented in Section III with Section IV describing in detail the distributed version of the proposed method. Numerical results and heuristics are presented in Section V while Section VI concludes the paper and offers directions for future work.

II. System model and statement of the problem

Consider a set $\mathcal{N}$ of battery-operated wireless sensors, which are randomly deployed over a given area and remain stationary once placed. In the following, we model the system at the network flow level, i.e. packet scheduling is not taken into account. Each sensor (a.k.a. node) $i \in \mathcal{N}$ produces information, at a fixed deterministic rate $Q_i \geq 0$, which must eventually be routed in a multi-hop fashion (i.e. using other stationary nodes), to a distinct sink node s, moving between different locations $l \in \mathcal{L}$. Such routing is possible only if there exists a directed path from each node to at least one sink location l, which is hereafter taken for granted. Node a communicates directly with node b, i.e. a directed link (a, b) exists, if the received power at b due to a's transmission is above a certain threshold, when a transmits at maximum power. To make the distributed implementation (to be presented later) more tractable, all links are assumed to be bi-directional, i.e. if a can communicate with b but not vice versa, none of the edges (a, b), (b, a) is considered to exist. This restriction is imposed for convenience purposes and is not required *per se* for the validity of the ensuing analysis.

The system is considered to operate in the low-SNR regime such that transmissions incur a power expenditure directly proportional to the amount of transmitted information rate. This assumption is justified in the context of, say, ultra-wide band (UWB) networks (see [14] for a detailed description of the underlying physical model) which have lately attracted significant attention for sensing applications.

We further assume, following a convention established in previous works, that receptions incur no cost (a similar model has been proposed in [8] for a stationary sink). Adding a reception cost to the destination node of each transmission does not change the main principles of our analysis or the structure of the algorithm to be presented[1] and is neglected in order to simplify the discussion.

The sink's movement over different locations effectively creates a new digraph $\mathcal{G}^l\left(\mathcal{N}\cup\{s\},\mathcal{E}^l\right)$ for each sink location l (where the sink is allowed to stay for an arbitrary amount of time) and, hopefully, distributes the routing and associated power costs more evenly compared to a stationary sink. All nodes $i\in\mathcal{N}$ except for the sink are equipped with a non-renewable amount of energy $E_i>0$, which is gradually depleted as the nodes participate in routing, and operate under a peak transmission power constraint $P_i>0$. Once a node's energy is drained, the node can no longer transmit. The lifetime of the network has been traditionally defined as the period of time until the first node runs out of energy. We start from a definition that captures the notion that a network is alive as long as it can transfer all generated traffic to the sink while satisfying the energy/power and flow conservation constraints.

Specifically, we say that the network is *survivable* up to time T if for each sink location $l\in\mathcal{L}$ a) there exists a sequence of time intervals $\tau_k^l,\ k=1,\ldots,K_l$ (representing the time intervals during which the sink resides at location l) such that $T=\sum_{l\in\mathcal{L}}\sum_{k=1}^{K_l}\tau_k^l$ and b) there exist *feasible* flow/power allocations $f_{ij}^{k,l}$, $p_{ij}^{k,l}$ for each time interval τ_k^l. Here $f_{ij}^{k,l}$ and $p_{ij}^{k,l}$ denote the traffic rate and transmission power, respectively, on link (i,j) during the interval τ_k^l. Denoting with $\mathcal{S}_i^l=\left\{n\in\mathcal{N}:(i,n)\in\mathcal{E}^l\right\}$ the set of outgoing neighbors of node i for sink location l, the term *feasible* means that the following conditions are true[2]

$$\sum_{l\in\mathcal{L}}\sum_{k=1}^{K_l}\sum_{j\in\mathcal{S}_i^l}p_{ij}^{k,l}\tau_k^l\le E_i,\quad\forall\, i\in\mathcal{N}, \tag{1}$$

$$\sum_{j\in\mathcal{S}_i^l}p_{ij}^{k,l}\le P_i,\quad\forall\, i\in\mathcal{N},\ \forall\, k\in\{1,\ldots,K_l\},\ \forall\, l\in\mathcal{L}, \tag{2}$$

$$f_{ij}^{k,l}\le h\left(p_{ij}^{k,l}\right),\quad\forall\, k\in\{1,\ldots,K_l\},\ \forall\,(i,j)\in\mathcal{E}^l,\ \forall\, l\in\mathcal{L}, \tag{3}$$

$$\sum_{j\in\mathcal{S}_i^l}f_{ij}^{k,l}=Q_i+\sum_{j:i\in\mathcal{S}_j^l}f_{ji}^{k,l},\quad\forall\, i\in\mathcal{N}\cup\{s\},\ \forall\, k\in\{1,\ldots,K_l\},\ \forall\, l\in\mathcal{L}. \tag{4}$$

Equation (1) ensures that the total expended energy by node i during the network's operation will not exceed the initial energy reserve E_i, while (2) places a peak transmission power constraint P_i (say, due

[1] as a foresight, only the edge costs d_{ij}^l of the minimum cost flow problem developed in (17) of Section III-B are modified.

[2] the specification of the vector $\left(\tau_k^l,f_{ij}^{k,l},p_{ij}^{k,l}\right)$ for all i,j,k,l can be regarded as a policy specification. Hence, a policy achieves survivability up to time T if $T=\sum_{l\in\mathcal{L}}\sum_{k=1}^{K_l}\tau_k^l$ and $\left(f_{ij}^{k,l},p_{ij}^{k,l}\right)$ is a feasible allocation.

to FCC regulations) to node i for all time intervals and sink locations. Equation (3) provides a capacity-related upper bound $h\left(p_{ij}^{k,l}\right)$, where h is a non-decreasing concave function, on the achievable link rate $f_{ij}^{k,l}$ on link (i,j) using power $p_{ij}^{k,l}$. Finally, (4) ensures long-term network stability under a fluid model, which is most suitable for the macroscopic (flow) level at which we examine the problem. For consistency reasons, we also define $Q_s^l \triangleq Q_s = -\sum_{i\in\mathcal{N}} Q_i, \;\; \forall\, l \in \mathcal{L}$. Notice that flow conservation is imposed for each individual sink location.

In this paper we are interested in maximizing the survivability time of the network, i.e. the problem

$$\begin{aligned} &\text{maximize} \quad T = \sum_{l\in\mathcal{L}} \sum_{k=1}^{K_l} \tau_k^l, \\ &\text{s.t.} \quad \text{constraints of (1)–(4).} \end{aligned} \tag{5}$$

The maximum value of T in (5) is defined as the network "lifetime" (equivalently, the network "dies" at time T). This definition justifies the imposition of (4) as a problem constraint since, under the optimal solution, the network lifetime will usually be sufficiently large for buffer overflow to occur in any location of non-zero sojourn time (even though the latter may be a small portion of the total lifetime) where (4) is violated. This is especially true considering the hardware capabilities of typical sensors.

Using a standard convex combination argument (see Appendix I), it is easy to show that we can restrict our attention to time-invariant flows for each sink location (i.e. a single flow allocation for each location suffices), thereby removing the k summation from all previous relations. The following observations are also useful. First, it is clear that (3) holds with strict equality at optimality, since if there was an allocation such that $f_{ij}^{k,l} < h\left(p_{ij}^{k,l}\right)$ for a specific interval τ_k^l and link (i,j) we could reduce $p_{ij}^{k,l}$ until equality is achieved with no reduction in network lifetime. For an interference-free environment, the low-SNR regime assumption allows us to use the following approximation to the Shannon rate formula $f_{ij}^l = h\left(p_{ij}^l\right) = r\ln\left(1 + a_{ij}^l\, p_{ij}^l/N_0\right) \approx r a_{ij}^l\, p_{ij}^l/N_0 = p_{ij}^l/e_{ij}^l$, where r, a_{ij}^l are link specific quantities that depend on path quality, employed transmission scheme etc. Hence, e_{ij}^l represents the power required to sustain a rate of 1 bps from node i to node j when the sink is at location l. Clearly, if both i,j are static nodes, the l superscript is redundant since the same channel exists for all sink locations (i.e. $e_{ij}^l = e_{ij}\;\forall\, l$). It is only when j becomes equal to the sink index s that the l superscript becomes meaningful and in fact necessary to avoid ambiguity. The above results in the original problem being reduced to the equivalent form

$$\text{maximize} \quad \sum_{l\in\mathcal{L}} t^l, \tag{6}$$

$$\text{s.t} \quad \sum_{l\in\mathcal{L}}\sum_{j\in\mathcal{S}_i^l} f_{ij}^l e_{ij}^l t^l \le E_i, \quad \forall\, i\in\mathcal{N}, \tag{7}$$

$$\sum_{j\in\mathcal{S}_i^l} f_{ij}^l e_{ij}^l \le P_i, \quad \forall\, i\in\mathcal{N},\ \forall\, l\in\mathcal{L}, \tag{8}$$

$$\sum_{j\in\mathcal{S}_i^l} f_{ij}^l = Q_i + \sum_{j:i\in\mathcal{S}_j^l} f_{ji}^l, \quad \forall\, i\in\mathcal{N}\cup\{s\},\ \forall\, l\in\mathcal{L}. \tag{9}$$

Introducing the auxiliary variable $b_{ij}^l = f_{ij}^l\, t^l$ and multiplying (8), (9) by t^l converts the above problem into a linear program with respect to t^l, b_{ij}^l. Since the original problem has been reduced to a simpler form, we summarize the simplified notation in Table I for the reader's convenience. Although in principle any LP solution technique can be applied (as was the case in [12] where a centralized simplex algorithm was used), we will focus instead on a distributed algorithm derived via duality.

The power constraint of (2), (8) models a broadcast transmission, where each node operates in a frame fashion and allocates individual slots in its frame to each of its neighbors. In that sense, P_i is understood as the maximum power expenditure per frame. Another meaningful constraint could be applied on a slot basis and it would be expressed by dropping the j summation from (2), (8), so that an additional constraint of the form $f_{ij}^l e_{ij}^l \le \tilde{P}_{ij}$ would appear for each edge and sink location. As will be seen in Section III, this additional constraint is trivial to incorporate in our model, since the problem remains convex and only the upper bounds of the flow in (13) (to appear soon) are affected, while the dual minimum cost flow problem remains unaffected.

Finally, it is interesting to consider the relationship between the network's lifetime and the first time at least one node dies under the optimal policy of time-invariant flows for each sink location. For an immobile sink, the previous definition of lifetime implies that, when the problem has a feasible solution, the death of the first node occurs at the time the network dies under the optimal time-invariant policy[3]. However, this is not the case for a mobile sink, as the trivial counterexample of Fig. 1 demonstrates. In Fig. 1, the sink is allowed to move between two locations (marked as squares) while information is injected into the black node with rate $Q = 1$ (the white nodes act as relays only). All nodes have a power constraint of $P = 1$ and energy constraint of $E = 1$ for the white nodes and $E = 2$ for the black node. The link labels represent the edge costs e_{ij}^l of (7)–(8). Clearly, an optimal policy consists of the source always transmitting with rate 1 to each relay node, which then forwards this rate to the sink. Regardless

[3] this does not exclude the existence of optimal time-varying policies where the network survives the first node's death. However, Appendix I guarantees that these policies can be replaced by time-invariant policies, of optimal lifetime, where the network dies exactly when the first node dies.

of the order in which the locations are visited, each relay can transmit for at most one unit of time, leading to a network lifetime of 2. Furthermore, there exists no time-invariant policy which can force both relay nodes to die at time 2. It is clear though (applying the argument of Appendix I verbatim) that if, for a mobile sink, the problem has a feasible solution, then there exists an optimal policy such that for *each* location of non-zero sojourn, a node dies exactly at the end of the respective sojourn. At that point, the sink must move to the next location.

III. Dual formulation

In [8] a distributed algorithm was presented for the solution of the sensor lifetime problem when the sink location is fixed. As explained next, this algorithm cannot be applied to the problem at hand. Since [8] considers a stationary sink, we must temporarily remove the l summation to be on common ground. The LP in [8] is derived from (6)–(9) (without the l summation) by performing the substitution $q = 1/t$, thus converting "maximize t" into "minimize q" and (7) into

$$\sum_{j \in \mathcal{S}_i} f_{ij}\, e_{ij} \leq q E_i \quad \forall\, i \in \mathcal{N}. \tag{10}$$

Equations (8), (9) don't contain t and are therefore unaffected. The problem is now reduced into LP form with respect to q, f_{ij}. In order for the same artifice to be applicable to our mobile sink case, we should set $q = 1/\sum_{l \in \mathcal{L}} t^l$ and transform (7) into a form containing q only. However, the latter is impossible since the l summation now includes a product of three terms, which cannot be expressed as a function of q. Hence, a different approach is required. We present a method in the following after discussing some preliminary results.

A. Solution bounds and feasibility

To justify later analysis, we provide finite upper bounds on both f_{ij}^l and t^l, the lower bounds being obviously zero. Total flow conservation requires $f_{ij}^l \leq \sum_{i \in \mathcal{N}} Q_i,\ \forall\, l \in \mathcal{L},\ (i,j) \in \mathcal{E}^l$. Additionally, if the alternative slot-based power constraint is applied, it also holds $f_{ij}^l \leq \tilde{P}_{ij}/e_{ij}^l$. In both cases, we can define an upper bound u_{ij}^l such that $f_{ij}^l \leq u_{ij}^l\ \forall\, l \in \mathcal{L},\ \ (i,j) \in \mathcal{E}^l$. For the sojourn time, we have

$$\begin{aligned} E_i &\geq \sum_{l \in \mathcal{L}} \sum_{j \in \mathcal{S}_i^l} e_{ij}^l f_{ij}^l\, t^l \geq \sum_{l \in \mathcal{L}} \sum_{j \in \mathcal{S}_i^l} t^l \min_{j \in \mathcal{S}_i^l} \left(e_{ij}^l\right) f_{ij}^l \geq \sum_{l \in \mathcal{L}} t^l \min_{j \in \mathcal{S}_i^l} \left(e_{ij}^l\right) \sum_{j \in \mathcal{S}_i^l} f_{ij}^l \geq \\ &\overset{(9)}{\geq} \sum_{l \in \mathcal{L}} t^l \min_{j \in \mathcal{S}_i^l} \left(e_{ij}^l\right) Q_i \geq t^l \min_{j \in \mathcal{S}_i^l} \left(e_{ij}^l\right) Q_i, \quad \forall\, i \in \mathcal{N},\ \forall\, l \in \mathcal{L}. \end{aligned} \tag{11}$$

Since the last bound applies to all nodes, we extract the following global upper bound for each sink location

$$t^l \leq \min_{i\in\mathcal{N}} \underbrace{\left(\frac{E_i}{Q_i \min_{j\in\mathcal{S}_i^l} e_{ij}^l}\right)}_{\hat{T}_i^l} \triangleq \hat{T}^l, \quad \forall\, l \in \mathcal{L}, \tag{12}$$

the important property being that these bounds are independent of the flow. Their utility will become apparent later.

The constraint set, when non-empty, is compact so that a solution always exists, although it may not be unique (a common nuisance in linear programming). When only energy constraints are present (i.e. (8) is removed) the set is always non-empty and the problem is feasible. This is justified as follows. If there exists one sink location for which a directed path exists from each node to the sink, then Theorem 6.12 of [15] asserts that a valid flow can be constructed for this location. Accordingly, selecting a sufficient small sojourn time allows us to satisfy (7) for all nodes, leading to a feasible solution. On the other hand, it is easy to construct trivial networks where the addition of (8) results in infeasible problems. Since there is no way of *a priori* knowing whether the problem is feasible, the proposed algorithm must systematically detect (preferably as soon as possible) infeasible problems and halt properly. This will be addressed in a later section.

B. Minimum cost flow via duality

Contrary to previous work that treats (7)–(9) as constraints for the dual formulation and introduces Lagrange multipliers for each of them, we only regard (7), (8) as constraints. We therefore define for each sink location l the set

$$\mathcal{X}^l = \left\{ \left(\boldsymbol{f}^l, t^l\right) : \sum_{j:i\in\mathcal{S}_j^l} f_{ji}^l + Q_i = \sum_{j\in\mathcal{S}_i^l} f_{ij}^l,\ 0 \leq f_{ij}^l \leq u_{ij}^l,\ 0 \leq t^l \leq \hat{T}^l \right\}, \tag{13}$$

which is essentially the set of all flow conserving vectors when the sink is at location l. The primal problem then becomes

$$\begin{aligned} \underset{\cap_l \mathcal{X}^l}{\text{minimize}} \quad & \sum_{l\in\mathcal{L}} t^l, \\ \text{s.t.} \quad & \text{constraints of (7)–(8).} \end{aligned} \tag{14}$$

Removing the flow conservation from the constraint set imposes additional structure to the dual problem, which makes it more amenable to a distributed solution. Furthermore, it forces all intermediate numerical

solutions obtained during the subgradient projection iterations to be flow-conserving. Hence, exploiting the fact that both sides of (8) can be multiplied by t^l, the Lagrangian is written as

$$\begin{aligned} L(\boldsymbol{f},\boldsymbol{t},\boldsymbol{\nu},\boldsymbol{\mu}) &= -\sum_{l\in\mathcal{L}} t^l + \sum_{i\in\mathcal{N}} \nu_i \left[\sum_{l\in\mathcal{L}}\sum_{j\in\mathcal{S}_i^l} f_{ij}^l e_{ij}^l t^l - E_i\right] + \sum_{l\in\mathcal{L}}\sum_{i\in\mathcal{N}} \mu_i^l \left[\sum_{j\in\mathcal{S}_i^l} f_{ij}^l e_{ij}^l t^l - P_i t^l\right] = \\ &= -\sum_{i\in\mathcal{N}} \nu_i E_i + \sum_{l\in\mathcal{L}} t^l \left\{-1 - \sum_{i\in\mathcal{N}} \mu_i^l P_i + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{S}_i^l} \left(\nu_i + \mu_i^l\right) f_{ij}^l e_{ij}^l\right\}, \end{aligned} \tag{15}$$

which results in a dual-function of

$$q\left(\boldsymbol{\nu},\boldsymbol{\mu}\right) = \inf_{\cap_l \mathcal{X}^l} L(\boldsymbol{f},\boldsymbol{t},\boldsymbol{\nu},\boldsymbol{\mu}) = -\sum_{i\in\mathcal{N}} \nu_i E_i + \sum_{l\in\mathcal{L}} \inf_{\mathcal{X}^l} \left\{ t^l \left[-1 - \sum_{i\in\mathcal{N}} \mu_i^l P_i + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{S}_i^l} \left(\nu_i + \mu_i^l\right) f_{ij}^l e_{ij}^l\right]\right\}, \tag{16}$$

where we exploited the separability of the Lagrangian with respect to l to pass the min through the summation (in the case where only slot-based power constraints are applied and (8) is not imposed, it trivially follows that $\mu_i^l \equiv 0 \quad \forall\, i,l$ in all subsequent expressions).

Since the two product terms inside the infimum are bounded and independent of each other, the infimum can be further decomposed as follows: for a given $(\boldsymbol{\nu},\boldsymbol{\mu})$ pair, we compute

$$\tilde{f}^l = \arg\inf_{\mathcal{X}^l} \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{S}_i^l} \underbrace{\left(\nu_i + \mu_i^l\right) e_{ij}^l}_{d_{ij}^l} f_{ij}^l, \tag{17}$$

which corresponds to a minimum cost flow problem [16], since the set $\mathcal{X}^l$ contains all flow-conserving vectors[4]. Denoting as $\tilde{F}^l$ the corresponding minimum cost, the minimizer of the separable Lagrangian becomes

$$\tilde{t}^l = \begin{cases} 0 & \text{if } \tilde{F}^l \geq 1 + \sum_{i\in\mathcal{N}} \mu_i^l P_i \\ \hat{T}^l & \text{otherwise} \end{cases}, \tag{18}$$

so that the dual function can be written as

$$q\left(\boldsymbol{\nu},\boldsymbol{\mu}\right) = -\sum_{i\in\mathcal{N}} \nu_i E_i + \sum_{l\in\mathcal{L}} \hat{T}^l\, \mathbb{I}\left[\tilde{F}^l < 1 + \sum_{i\in\mathcal{N}} \mu_i^l P_i\right], \tag{19}$$

where $\mathbb{I}[\ ]$ denotes the indicator function of the enclosed logic statement. Hence, minimization of the Lagrangian in (15) is equivalent to the solution of $|\mathcal{L}|$ minimum cost flow problems, which can be performed in a distributed manner [16].

[4] for the case where link (i,j) incurs a power reception cost on the receiver j equal to $a_j f_{ij}$, a straightforward repetition of the previous procedure results in $d_{ij}^l = \left(\nu_i + \mu_i^l\right) e_{ij}^l + a_j \left(\nu_j + \mu_j^l\right)$.

The dual problem now becomes

$$\begin{aligned} &\max\ q(\boldsymbol{\nu},\boldsymbol{\mu}), \\ &\text{s.t.}\quad \boldsymbol{\nu},\,\boldsymbol{\mu}\in\mathcal{F}, \end{aligned} \tag{20}$$

where $\mathcal{F}=\{(\boldsymbol{\nu},\boldsymbol{\mu}):\boldsymbol{\nu},\,\boldsymbol{\mu}\geq 0,\ q\left(\boldsymbol{\nu},\boldsymbol{\mu}\right)>-\infty\}$ is the set of Lagrange multipliers leading to a bounded (from below) Lagrangian. It is clear from (19) that the last predicate in $\mathcal{F}$ is redundant so that $\mathcal{F}=\{(\boldsymbol{\nu},\boldsymbol{\mu}):\ \boldsymbol{\nu},\,\boldsymbol{\mu}\geq 0\}$. The dual problem is solved via standard subgradient projection [17], i.e. an update procedure of the form

$$\begin{aligned} \boldsymbol{\nu}^{k+1} &= P_{\mathcal{F}}\left[\boldsymbol{\nu}^{k}+s^{k}\left(\sum_{l\in\mathcal{L}}\sum_{j\in\mathcal{S}_i^l}\tilde{f}_{ij}^{l}\,\tilde{t}^{l}e_{ij}^{l}-E_i\right)\right]=\left[\boldsymbol{\nu}^{k}+s^{k}\left(\sum_{l\in\mathcal{L}}\sum_{j\in\mathcal{S}_i^l}\tilde{f}_{ij}^{l}\,\tilde{t}^{l}e_{ij}^{l}-E_i\right)\right]^{+}, \\ \boldsymbol{\mu}^{k+1} &= P_{\mathcal{F}}\left[\boldsymbol{\mu}^{k}+s^{k}\left(\sum_{j\in\mathcal{S}_i^l}\tilde{f}_{ij}^{l}e_{ij}^{l}-P_i\right)\tilde{t}^{l}\right]=\left[\boldsymbol{\mu}^{k}+s^{k}\left(\sum_{j\in\mathcal{S}_i^l}\tilde{f}_{ij}^{l}e_{ij}^{l}-P_i\right)\tilde{t}^{l}\right]^{+}, \end{aligned} \tag{21}$$

where the $P_{\mathcal{F}}$ operator denotes a projection (in the metric space context) of the enclosed argument onto space $\mathcal{F}$, which is then reduced to the simple $[\ ]^{+}=\max([\ ],0)$ operator, s^k is a suitably chosen scalar step-size for subgradient iteration k and the quantities between parentheses in (21) represent the subgradients for the energy/power constraints, respectively.

On a more technical note, the subgradient method is conditionally convergent upon a suitable choice for s^k. According to Proposition 8.2.6 of [17], choosing $s^k=1/k$ ensures convergence to the dual optimal $(\boldsymbol{\nu}^*,\boldsymbol{\mu}^*)$ solution as $k\to\infty$. This choice for s^k also facilitates the primal recovery scheme to be explained later. Additionally, since the problem is effectively an LP, through the substitution $b_{ij}^{l}=f_{ij}^{l}t^{l}$, the Slater conditions hold [18] and there is no duality gap so that a primal solution can be obtained through the minimization of $L\left(\boldsymbol{f},\boldsymbol{t},\boldsymbol{\nu}^*,\boldsymbol{\mu}^*\right)$.

C. Some fine points

This section provides information on certain important details of the proposed method as well as additional issues alluded to in the previous analysis. It is clear that the heart of the problem is the efficient solution of a minimum (linear) cost flow problem for each Lagrange multiplier pair. This problem has been extensively studied and various techniques have been proposed. We choose the synchronous scaled ϵ-relaxation (SER) [16] algorithm due to its ease of distributed implementation and use it exclusively in this paper.

The SER algorithm, as most flow algorithms, implicitly assumes integer edge costs, which translates in our case to d_{ij}^{l} being integer in (17). This cannot be satisfied *a priori* since d_{ij}^{l} encapsulate the Lagrange

multipliers ν_i, μ_i^l, which are most likely floating point numbers. Thankfully, this can be circumvented by exploiting the scale invariance of a linear problem with respect to its (linear) objective. Specifically, all edge costs d_{ij}^l are up-scaled by an appropriate power of 10, so that the minimum modified edge cost becomes of the order of 1000, and all fractional parts are then truncated. Since all costs are larger than 1000, the truncation effect is expected to be minimal. Notice that this does not affect the optimal flow allocation since the constraint set remains unaffected (the scaling is essentially performed on the $\nu_i + \mu_i^l$ part of the cost). The SER algorithm is of pseudo-polynomial time complexity so that the scaling affects the execution time; however the time dependence on the maximum cost exists only in the form of a log function, which imposes a minimal computational burden. Once the solution is obtained, the costs are down-scaled to their original values so that (18) can be correctly computed.

Furthermore, the Lagrange minimizer in (18) is essentially a binary decision process, i.e. the sink stays either for the maximum allowable time $\tilde{T}^l$ or leaves immediately. Since the original bounds may be too loose, we seek to refine them during the algorithm's execution. This can be performed through the weak duality theorem [17], which implies

$$q(\boldsymbol{\nu}^k, \boldsymbol{\mu}^k) \leq q^* \leq -\sum_{l\in\mathcal{L}} t^{*\,l} \leq -\sum_{l\in\mathcal{L}} t^l \Rightarrow t^l \leq \sum_{l\in\mathcal{L}} t^l \leq -q(\boldsymbol{\nu}^k, \boldsymbol{\mu}^k), \tag{22}$$

where $q^*, t^{*\,l}$ are the exact dual/primal optimal values, respectively. Therefore, for every iteration where it holds $-q(\boldsymbol{\nu}^k, \boldsymbol{\mu}^k) < \hat{T}^l$, $\hat{T}^l$ can be set to the left hand side of the last inequality (we hereafter use the notation $\hat{T}^{l,k}$ to refer to the upper sojourn bound for location l at the end of iteration k). Experience has shown this procedure to lead to increased execution speed. The above property can also be used to derive an infeasibility detection criterion according to the following fact (see Appendix II for the proof)

Corollary 1 *If during the execution of the subgradient projection algorithm there exists an iteration k^* such that $\hat{T}^{l,k^*} = 0$ for all $l \in \mathcal{M} \subseteq \mathcal{L}$, then for all $l \in \mathcal{M}$ there exist no flows f_{ij}^l satisfying the power constraints of (8). If, additionally, it holds $\mathcal{M} = \mathcal{L}$, then the original problem is infeasible.*

Regarding the primal recovery procedure, we adopt the methodology in [19], from which we have the following

Proposition 1 *Assume that for a step size $s^k = 1/k$, the subgradient projection method converges to a pair $(\boldsymbol{\nu}^*, \boldsymbol{\mu}^*)$. For each iteration k, denote $\mu_j^k = 1/k,\ \forall\, j = 1 \ldots k$ and $\tilde{\boldsymbol{x}}^k = (\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{t}})^k$ the minimizer of*

the Lagrangian for the k iteration. Construct the sequence $\{\boldsymbol{x}^k\}$ as follows

$$\begin{aligned} \boldsymbol{x}^0 &= \tilde{\boldsymbol{x}}^0, \\ \boldsymbol{x}^{k+1} &= \frac{k}{k+1}\,\boldsymbol{x}^k + \frac{1}{k+1}\,\tilde{\boldsymbol{x}}^k. \end{aligned} \tag{23}$$

Then, the sequence $\boldsymbol{x}^k$ converges to a vector $\bar{\boldsymbol{x}}$ which is primal optimal.

This leads to a scheme that allows "on-the-fly" update of the primal feasible candidate solution with no backstorage required.

Application of the subgradient method requires an initial guess $\left(\boldsymbol{\nu}^0, \boldsymbol{\mu}^0\right)$ for the Lagrange multipliers, as well as two scalar parameters $maxiter$ and $winrec$ denoting, respectively, a maximum number of subgradient iterations and the number of iterations during which the primal recovery scheme is applied (i.e. the subgradient method is run for at most $maxiter$ iterations, barring earlier convergence, and the primal recovery scheme is switched "on" during the last $winrec < maxiter$ iterations). It is well known that choosing an initial guess sufficiently close to the dual solution $(\boldsymbol{\nu}^*, \boldsymbol{\mu}^*)$ is crucial in obtaining fast convergence. Unfortunately, $\left(\boldsymbol{\nu}^0, \boldsymbol{\mu}^0\right)$ is usually chosen at random so that the parameter $maxiter$ is used to guard against a very slow (numerical) convergence due to a poor initial guess. The term "slow" refers to the fact that the recovered primal solution at the end of $maxiter$ iterations may be of poor "quality", i.e. the energy and power constraints may be violated by a significant amount. In practice, the network designer may construct quality metrics such as the maximum normalized energy and power violations, $\max_{i\in\mathcal{N}}|\mathring{e}_i - E_i|/E_i$ and $\max_{i\in\mathcal{N}}|\mathring{p}_i - P_i|/P_i$, respectively (where $\mathring{e}_i, \mathring{p}_i$ is the energy/power expended by node i under the recovered solution. These two definitions are in accordance with [8]) and deem a recovered solution as acceptable only when these metrics satisfy given thresholds (say below 10%). Obviously, a way of modifying $\left(\boldsymbol{\nu}^0, \boldsymbol{\mu}^0\right)$ must be provided for the contingency where a solution is considered unacceptable. To this end, three heuristics were proposed in [20] and compared via simulations. It was concluded that one of them significantly outperforms the others and hence results will be presented for this heuristic, only, in a later section. The reader is referred to [20] for further details on the rationale behind each heuristic.

IV. Distributed implementation

The previous analysis already suggests the distributed implementation of the above solution. Specifically, we assume that each node locally stores its Lagrange multipliers ν_i and μ_i^l (for a total of $|\mathcal{L}| + 1$ values), properly initialized. Furthermore, for each sink location, every node can determine its outgoing

neighbors and estimate the associated link costs. For each location, a spanning tree[5], rooted at the sink, is constructed in a distributed manner and used for message exchange between the nodes and the sink. Its utility stems from the fact that summations and min/max operations over nodes (which, as will be seen, are the only operations needed by the sink in order to perform its task) can be easily performed in a distributed manner [21] through the spanning tree with only $O(|\mathcal{N}|)$ messages (rather than having each node transmit its individual variable to the sink with the latter carrying out the entire computation). Also, the sink can broadcast control instructions to all nodes through the spanning tree using $O(|\mathcal{N}|)$ messages.

The algorithm now proceeds as follows: the sink starts at the first location (say $l = 0$) and performs some standard initializations (among which is the determination of $\hat{T}^l$). It then instructs the network to solve, in the distributed manner of [16], the minimum cost flow problem of (17) using the current Lagrange multipliers. Once the minimum cost flow solution has been attained, the total minimum cost objective $\tilde{F}^l$ is computed in a distributed manner and is transmitted to the sink along with the quantities $\sum_{i\in\mathcal{N}} \mu_i^l P_i$, $-\sum_{i\in\mathcal{N}} \nu_i E_i$, which are also computed in a distributed way. These values travel towards the sink through the spanning tree and once they reach the sink, the quantity $\tilde{t}^l$ in (18) is evaluated and transmitted back to the entire network through the spanning tree. Each node then updates its current power Lagrange multiplier μ_i^l according to (21), since the outgoing flows are locally stored. Additionally, each node adds the quantity $\sum_{j\in\mathcal{S}_i^l} \tilde{f}_{ij}^l \tilde{t}^l e_{ij}^l$ to a locally stored variable called $sumR$ and performs the primal recovery scheme of (23), if applicable.

At this point, the sink moves to the next location ($l = 1$) and performs exactly the same steps as in the previous paragraph, with the minor difference that the quantity $-\sum_{i\in\mathcal{N}} \nu_i E_i$ need not be computed again for the duration of the current round-trip (i.e. until the sink returns to the original location). A spanning tree rooted at the new sink location must also be constructed and used for the exchange of control messages. Again, once the new minimum cost flow problem is solved, each node updates its μ_i^l and increments its $sumR$ variable. This continues as the sink visits all locations in sequence. The sink will eventually return to its initial location $l = 0$, where it will detect that a round-trip has been completed (we assume that the sink is equipped with such a capability). At this point, and before starting a second round-trip, the sink transmits a special message through the spanning tree instructing the nodes to update their ν_i according to (21). If desired, the sink can also transmit back the value of $q(\boldsymbol{\nu}, \boldsymbol{\mu})$, which has been computed during the previous round-trip, so that the nodes can refine their $\hat{T}^l$ bounds from (22), if

[5] since all links are assumed bi-directional, the same property applies to the tree as well.

applicable. Once this is performed, the sink starts a second round-trip applying the previous procedure verbatim. The algorithm continues in that manner until it is terminated by a convergence criterion being satisfied. Once the complete solution has been recovered, the sink moves to each location and stays there for the specified optimal sojourn time. In this sense, the round-trips performed until convergence is achieved can be considered as part of the network's initialization period (i.e. the network is *truly* operational only after the algorithm has converged to a solution).

A pseudo-code representation of the above algorithm is given in Fig. 2. It is clear that the core of the algorithm is the distributed solution of the minimum cost flow problem. The number and type of messages that must be exchanged for this operation is described in detail in [20], and omitted from here due to space restrictions. However, it should be mentioned that a second variant of the above algorithm exists, in which the sink performs a single round-trip, allows the nodes to discover their neighbors for each location, and then returns to the initial location and stays there for the remainder of the algorithm. Obviously, "virtual" sink movement must be emulated by the sink generating appropriate messages. This variant requires the construction of a single spanning tree only, which is performed when the sink returns to the initial location after the round-trip. It offers the advantage of decreased initialization time (convergence is achieved in less "actual time"[6]) but implicitly assumes that the network topology is time-invariant. If this is not the case, the sink must move to each location and monitor the network topology anew.

It follows from the previous discussion that each node (including the sink, unless otherwise stated) must locally store the following information.

1) a node tag, possibly hardware-coded, serving as a unique identifier. The sink identifier is globally recognized by all nodes, i.e. a node can always distinguish between communication with the sink and with another static node (i.e. any node other than the sink) .
2) the maximum energy and instantaneous power constraints E_i, P_i, the Lagrange multipliers ν_i, μ_i^l and the exogenous rate of information Q_i (static nodes only).
3) a group of variables representing the flow surplus g_i and flow conservation cost p_i. These are used by the minimum cost flow algorithm [16].
4) the outgoing and incoming edges in suitable doubly-linked lists. The acquisition of this information is described in [20]. The incoming edges also require some memory to be reserved for the primal recovery scheme.
5) an array of maximum length $(\text{outdeg}(i) + 2)$ necessary for storing the children and parent of the

[6] "actual time", measured in seconds, is used here in contrast to "algorithmic time", measured in subgradient iterations.

node for the spanning tree.

6) a number of bookkeeping variables that is independent of network size, i.e. $O(1)$.

A simple inspection of the previous list reveals that the memory required by the algorithm is $O\left(\text{outdeg}(i)+|\mathcal{L}|+|\mathcal{L}|\cdot\text{outdeg}(i)\right)=O\left(|\mathcal{L}|\cdot\text{outdeg}(i)\right)$ per node, where the last term comes from the primal recovery scheme. This compares very favorably to the memory requirements of a centralized solution method such as simplex. Specifically, a full tableau [18] simplex requires $O\left(|\mathcal{N}|\cdot|\mathcal{L}|\cdot(|\mathcal{L}|+|\mathcal{E}|)\right)$ memory while a revised simplex requires $O\left(|\mathcal{L}|^2\cdot|\mathcal{N}|^2\right)$. The proposed distributed algorithm clearly has superior memory scalability than either simplex method. Furthermore, the sink needs no special knowledge of the network, other than the ability to communicate with its direct neighbors (essentially, the sink's neighbors carry the necessary topology information to the sink through the spanning tree) and requires minimal CPU resources since the sum/max/min operations are performed by the nodes themselves in a distributed manner. We consider these properties as significant assets of the proposed algorithm.

V. Numerical results

A large number of simulations was performed in order to evaluate the proposed algorithm. Specifically three sets of network topologies with 20, 40 and 80 nodes, respectively, were constructed as explained in [12], each set containing 100 random network instances. The sink was allowed to move over 4 locations, which were the same for all networks. In all cases, each node had an exogenous rate of $Q_i=1$. The flow cost of edge (i,j) was assumed proportional to d_{ij}^2, with d_{ij} the physical distance between the two nodes. Two scenarios were studied: in the first one, only a power constraint of $E_i=1000$ was applied while in the second one a power constraint of $P_i=100$ was also imposed (the values were chosen for demonstration purposes only and don't correspond to any actual specs. Additionally, we use dimensionless quantities since the paper's focus is on the algorithm's convergence properties and not on actual values). The case of energy constraints only may arise when the exogenous rates are so low that the power constraints are inactive for all nodes. Two different $(maxiter, winrec)$ pairs were used, $(400,200)$ and $(200,100)$, respectively, to study the algorithm's sensitivity to these parameters. All these combinations resulted in a huge data set, so that only the most representative results will be given. The following results were obtained using the H3 heuristic of [20] since it provided the best performance relative to the other heuristics.

A typical evolution of the lifetime solution as well as the constraint violation is shown in Fig. 3 for the pair $(400,200)$ and an energy-constrained 20-node network. The normalized maximum lifetime, measured by the left axis in Fig. 3 using the circle markers, is defined as the ratio of the lifetime predicted

by the distributed algorithm over the lifetime computed by a standard LP solver while the normalized maximum energy violation, previously defined in Section III, is measured by the right axis in Fig. 3. A casual inspection of Fig. 3 reveals that convergence is practically achieved even after 300 iterations, assuming a 10% violation threshold in order to consider a solution acceptable. Hence, no additional runs are required. However, this is not the case for the second scenario where, using the pair $(200, 100)$ for a 40-node network, 7 individual runs were required for a total of 1380 iterations. This is depicted in Fig. 4, which shows the energy/power violations for the last 100 iterations of the last run, during which the primal recovery scheme was applied, using the left axis with the circle marker and the right axis with the solid line, respectively. The convergence of the normalized computed lifetime is also shown in the subplot of Fig. 4. Since the solution's evolution will generally vary among different random graphs, further insight may only be gained by cumulative results which capture the "average" performance of the algorithm. Specifically, we define the following quantities as performance metrics for the proposed distributed algorithm combined with the H3 heuristic.

- number of network instances (out of the original 100 instances) for which the algorithm produces an acceptable solution, i.e. a solution that minimally violates the energy/power constraints (according to criteria imposed by the network designer). Denote this number as I.
- the average (with respect to the I instances) relative error between the optimal lifetime predicted by the proposed algorithm and the optimal lifetime obtained through a centralized solution (say, simplex method).
- the average (with respect to the I instances) number of iterations required to achieve an acceptable solution.

Under the previous definitions, for the 20-node networks with energy constraints only and using the pair $(200, 100)$ the algorithm produced 98 acceptable solutions with an average relative error of 3.04% and 379 iterations on average. This performance can be further improved by using the pair $(400, 200)$, which results in 98 acceptable solutions with an average error of 1.93%. This is, obviously, achieved at the cost of more iterations, which now rise to an average of 514. The corresponding results for the 40-node networks with energy constraints only are a) 89 instances with 5.37% error and 647 iterations for the $(200, 100)$ pair and b) 100 instances with 4.2% error and 800 iterations for the $(400, 200)$ pair. Imposing power constraints naturally leads to an increase in iterations (since there exist more dual variables). Specifically, for the 20-node networks, the pair $(200, 100)$ produced 89 acceptable solutions with a 3.42% average error and 819 iterations, while the pair $(400, 200)$ provided 92 solutions with

2.9% error and 1519 iterations. For each pair, five infeasible problems were also detected. For the 40-node networks, the $(200, 100)$ pair provided 86 solutions with 2.67% error and 1141 iterations while the $(400, 200)$ pair produced 93 solutions with 2.77% error and 1681 iterations. Finally, for the 80-node networks under power constraints, the $(200, 100)$ pair produced only 71 acceptable solutions with 3.99% average error and 1498 iterations. Using the $(400, 200)$ resulted in a huge improvement: 96 acceptable solutions with 5.11% error and 1875 iterations. Finally, in order to examine the effect of the number of sink locations on the algorithm's performance, the power-constrained simulations were repeated using 8 and 2 sink locations, respectively. The results are summarized in Table II, which reveals that the effect of the number of sink locations on the total number of iterations is more significant as the network size increases. Even in these cases, however, the increase appears to be sublinear and exhibits a diminishing trend. This is another indication of the algorithm's robustness.

Although the algorithm's execution time (as measured by the number of iterations above) may seem too large in absolute terms, we note that it should be viewed relative to the expected lifetime of the WSN. WSNs are often designed to operate unattended for weeks or months; hence, the allocation of some initial time interval (in the order of seconds or minutes, which is the algorithm's typical execution time) in order to determine the policy that maximizes the network lifetime may be worthwhile.

Another practical issue is the algorithm's adaptability to changing network conditions, since our method implicitly assumes that the problem parameters e_{ij}^l, Q_i do not change during the algorithm's execution. Since the nodes remain stationary, the most common source of variation is a change in Q_i. Obviously, our algorithm should be applied anew to the modified problem; however we claim that if the variations are slight,[7] the new problem can be solved relatively fast using the last known Lagrange multipliers of the last problem as initial guesses for the new one. The number of iterations cannot become too small, as there is always the need for the primal recovery, but it can still be smaller than solving the problem anew.

To examine this claim, the following experiment was performed. Consider a power-constrained network whose nodes have a time-varying Q_i in the interval $[7 \quad 13]$ with a mean value of 10. Initially, all nodes have $Q_i = 10$ and $E_i = P_i = 1000$. However, during the network's lifetime four "disturbances" occur at random epochs. In each disturbance epoch, a random node changes its Q_i randomly in the allowed interval. Hence, in addition to the original problem, four new LP problems need to be solved (note that apart from Q_i, the residual energy E_i of each node also changes between the disturbances since the

[7] meaning that each Q_i, viewed as a random variable, has a small standard deviation

network is already operational). The two options are either solving the problem anew without exploiting any past information (approach A) or using the last known Lagrange multipliers as initial guess for the new "disturbed" problem (approach B). The results of each approach are summarized in Table III, which contains the lifetime obtained from both approaches (with a centralized solution lifetime used as benchmark), as well as the maximum constraint violation and the number of required iterations. The results were obtained for a typical 20 and 80-node network.

Clearly, approach B can obtain very good solutions with acceptable constraint violations and, most importantly, using at most 400 or 160 iterations for the 80 and 20 node network, respectively. Solving the problem from scratch (approach A) allows us to obtain superior accuracy compared to approach B in some cases (i.e. for the 20-node network) but at the cost of far more iterations. Although the decision of which approach to use lies with the network designer, we believe the above example illustrates the algorithm's ability to adapt to small network perturbations.

VI. Conclusions and future work

This paper presented a distributed algorithm for computing the maximum lifetime of a sensor network which routes data to a mobile sink. By imposing flow conservation to **all** sink locations and removing some constraints from the dual problem, the number of Lagrange multipliers was reduced and the dual problem acquired special structure that allowed for an efficient solution using the standard subgradient method and the minimum cost flow ϵ-relaxation algorithm. Details about the memory requirements and types of exchanged messages were also provided. The efficiency of the proposed algorithm was studied by numerical simulation.

The case of a mobile sink has not received as much attention as the single immobile sink. Hence, most open problems regarding the latter case, such as various convexifications of the general non-convex interference-based model or incorporation of stochastic components into the model, carry over to the former. We consider the stochastic aspects of the problem (including fading effects) to be of more interest, especially since previous work, such as [22], has been restricted to a semi-deterministic setting and we plan to investigate these in the future.

Appendix I
Sufficiency of time-invariant flows

Let the network be survivable up to time T. By definition, there exist values T^l such that $\sum_{l\in\mathcal{L}} T^l = T$ as well as a sequence of valid $f_{ij}^{k,l}$, $p_{ij}^{k,l}$ values, meaning that they (1)–(4) are satisfied. We construct the

convex combinations

$$\begin{aligned} \hat{f}_{ij}^{l} &\triangleq \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} f_{ij}^{k,l}, \\ \hat{p}_{ij}^{l} &\triangleq \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} p_{ij}^{k,l}, \end{aligned} \tag{24}$$

for each sink location $l \in \mathcal{L}$, where $T^l = \sum_{k=1}^{K_l} \tau_k^l$. We only need to show that this flow is feasible for the special case of (1)–(4) with $K_l = 1,\ \forall\, l \in \mathcal{L}$. We verify each constraint separately

$$\sum_{l\in\mathcal{L}} \sum_{j\in\mathcal{S}_i^l} \hat{p}_{ij}^{l} T^l \overset{(24)}{=} \sum_{l\in\mathcal{L}} \sum_{j\in\mathcal{S}_i^l} \sum_{k=1}^{K_l} \frac{\tau^{k,l}}{T^l} p_{ij}^{k,l} T^l \overset{(1)}{\leq} E_i, \tag{25}$$

$$\sum_{j\in\mathcal{S}_i^l} \hat{p}_{ij}^{l} \overset{(24)}{=} \sum_{j\in\mathcal{S}_i^l} \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} p_{ij}^{k,l} \overset{(2)}{\leq} \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} P_i \leq P_i, \tag{26}$$

$$\hat{f}_{ij}^{l} = \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} f_{ij}^{k,l} \overset{(3)}{\leq} \sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} h\left(p_{ij}^{k,l}\right) \overset{\text{conc}}{\leq} h\left(\sum_{k=1}^{K_l} \frac{\tau_k^l}{T^l} p_{ij}^{k,l}\right) \triangleq h\left(\hat{p}_{ij}^{l}\right), \tag{27}$$

where the last inequality is due to the concavity of h. Finally, the flow conservation for $\hat{f}_{ij}^{l}$ is obtained by multiplying both sides of (4) with τ_k^l/T^l, summing over k for each location l and interchanging the summations. This results in

$$\sum_{j\in\mathcal{S}_i^l} \hat{f}_{ij}^{l} = Q_i + \sum_{i\in\mathcal{S}_j^l} \hat{f}_{ji}^{l}, \quad \forall\, i \in \mathcal{N} \cup \{s\},\ \forall\, l \in \mathcal{L}. \tag{28}$$

Hence, under the convex combination of (24), the network is still survivable up to time T. This concludes the proof.

APPENDIX II
PROOF OF COROLLARY 1

Proof is by contradiction. Specifically, assume that there exists a flow vector f_{ij}^{l} satisfying power and flow conservation constraints for some $l \in \mathcal{M}$. Then, for this l we could pick a sufficiently small, but still positive t^l, such that the energy constraint is satisfied for all $i \in \mathcal{N}$ and arrive at a feasible solution with positive sojourn. However, this is a contradiction since the assumption $\hat{T}^{l,k^*} = 0$ implies, according to (22), that the sojourn time at location l is zero for any feasible solution. If $\mathcal{M} = \mathcal{L}$, then, from the above, no power-satisfying feasible solutions exist for any $l \in \mathcal{L}$ and the problem is therefore infeasible.

REFERENCES

[1] A. Michail and A. Ephremides, "Energy-efficient routing for connection-oriented traffic in wireless ad-hoc networks," *Mobile Networks and Application*, no. 8, pp. 517–533, 2003.

[2] V. Rodoplu and T. Meng, "Minimum energy mobile wireless networks," vol. 3. IEEE ICC, June 1998, pp. 1633–1639.

[3] L. Xiao, M. Johansson, and S. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Trans. Commun.*, vol. 52, no. 7, pp. 1136–1144, July 2004.

[4] R. Cruz and A. Santhaman, "Optimal routing, link scheduling and power control in multi-hop wireless networks." INFOCOM, March 2003, pp. 702–711.

[5] M. Kalantari and M. Shayman, "Energy efficient routing in wireless sensor networks." CISS, March 2004.

[6] S. Toumpis and L. Tassiulas, "Optimal deployment of large wireless sensor networks," submitted to IEEE Trans. on Inform. Theory.

[7] J. Chang and L. Tassiulas, "Energy conserving routing in wireless ad-hoc networks." IEEE INFOCOM, 2000, pp. 22–31.

[8] R. Madan and S. Lall, "Distributed algorithms for maximum lifetime routing in wireless sensor networks," *IEEE Trans. Wireless Commun.*, to appear.

[9] R. Madan, S. Cui, S. Lall, and A. Goldsmith, "Cross-layer design for lifetime maximization in interference-limited wireless sensor networks," vol. 3. INFOCOM, March 2005, pp. 1964–1975.

[10] S. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," in *Proceedings of IEEE GLOBECOM*, vol. 1, 2003, pp. 377–381.

[11] Z. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor networks lifetime," in *Proceedings of the 38th HICSS*, 2005.

[12] I. Papadimitriou and L. Georgiadis, "Maximum lifetime routing to mobile sink in wireless sensor networks." SoftCOM, September 2005.

[13] J. Luo, "Mobility in wireless networks: friend or foe, network design and control in the age of mobile computing," Ph.D. dissertation, EPFL, 2006.

[14] B. Radunovic and J. L. Boudec, "Optimal power control, scheduling and routing in UWB networks," *IEEE J. Select. Areas Commun.*, vol. 22, no. 7, pp. 1252–1270, 2004.

[15] R. Ahuja, T. Magnanti, and J. Orlin, *Network flows: theory, algorithms, and applications*. Prentice Hall, 1993.

[16] D. Bertsekas and J. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice Hall, 1989.

[17] D. Bertsekas, A. Nedić, and A. Ozdaglar, *Convex analysis and optimization*. Athena Scientific, 2003.

[18] D. Bertsimas and J. Tsitsiklis, *Introduction to linear optimization*. Athena Scientific, 1997.

[19] H. Sherali and G. Choi, "Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs," *Operations Research Letters*, vol. 19, pp. 105–113, 1996.

[20] M. Gatzianas and L. Georgiadis, "Application of a distributed minimum cost flow algorithm to the maximum network lifetime problem." [Online]. Available: http://users.auth.gr/∼mgkatzia/maxlife_dmcf.ps

[21] N. Lynch, *Distributed algorithms*. Morgan Kaufmann, 1996.

[22] J. Li and S. Dey, "Lifetime optimization for wireless sensor networks with outage probability constraints." 12^{th} European Wireless, April 2006, available in electronic form only.

TABLE I

LIST OF NOTATIONS

Notation	Definition
$\mathcal{N}$	set of static sensor nodes
$\mathcal{L}$	set of locations where sink may stay
$\mathcal{E}^l$	set of edges (i.e. links) between the static nodes and the sink when the latter is at location $l \in \mathcal{L}$
t^l	sink sojourn time at location $l \in \mathcal{L}$, also an unknown in the formulated linear program
$\hat{T}^l$	upper bound on sink sojourn time for location $l \in \mathcal{L}$.
Q_i	exogenous traffic rate for static node $i \in \mathcal{N}$
$\mathcal{S}_i^l$	set of outgoing neighbors of node i when the sink is at location $l \in \mathcal{L}$
e_{ij}^l	amount of power needed to transmit 1 bps from node i to node j when the sink is at location $l \in \mathcal{L}$
f_{ij}^l	time-invariant rate of flow (in bps) from node i to node j when the sink is at location $l \in \mathcal{L}$
p_{ij}^l	amount of power needed to transmit with rate f_{ij}^l from node i to node j when sink is at location $l \in \mathcal{L}$
b_{ij}^l	$f_{ij}^l t^l$: an unknown in the formulated linear program

TABLE II

NUMBER OF ITERATIONS FOR VARIOUS SETS OF SINK LOCATIONS

	20 nodes		40 nodes		80 nodes	
sinks	200/100	400/200	200/100	400/200	200/100	400/200
2	859	1539	851	1644	1004	1567
4	819	1519	1141	1681	1498	1875
8	879	1538	1178	1751	1646	1942

TABLE III

EXAMPLE OF ALGORITHM'S ADAPTABILITY TO PARAMETER CHANGES

		approach A			approach B		
	20 nodes						
problem	centralized sol. lifetime	lifetime	violation (%)	iterations	lifetime	violation (%)	iterations
original	3.3557	3.66161	9.11	800	N/A	N/A	N/A
disturb	2.91661	2.91358	1.74	600	2.9306	6.19	80
disturb	0.865215	0.874121	6.44	800	0.689	7.73	160
disturb	0.382747	0.384058	0.345	800	0.40288	5.26	80
disturb	0.261876	0.275827	5.33	800	0.221995	0.0	80
	80 nodes						
original	1.89755	1.91398	2.01	1200	N/A	N/A	N/A
disturb	1.69949	1.70545	4.81	1200	1.69068	9.48	200
disturb	1.5178	1.58869	5.73	1200	1.53319	2.51	200
disturb	1.36093	1.4077	4.25	1200	1.31622	0.00	400
disturb	1.22384	1.27272	8.59	1200	1.22355	4.47	200

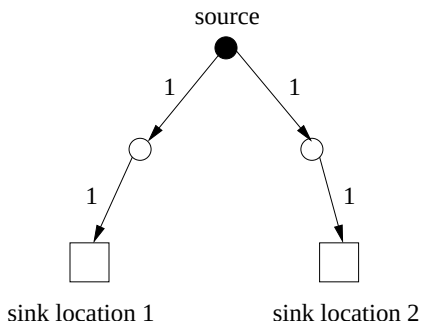

Fig. 1. Trivial network demonstrating the relationship between network lifetime and time of first node's death.

Algorithm 1: solve_maxlife.

Objective: solve problem defined in (6)–(9).

Input: the digraphs $\mathcal{G}^l$ for all l, vectors $\boldsymbol{E}, \boldsymbol{P}, \boldsymbol{Q}$, an initial guess vector $(\boldsymbol{\nu}, \boldsymbol{\mu})^0$, a starting iteration index k_0 and a maximum number of iterations $maxi$.

Output: the optimal flows $f_{ij}^{*l} \ \ \forall\, (l,i,j) \in \mathcal{L} \times \mathcal{E}^l$ and sojourn times $t^{*l} \ \ \forall\, l \in \mathcal{L}$.

initialization phase ;
$k \leftarrow k_0$;
select subgradient step-size $s^k = 1/k$;
while *number of iterations* $< maxi$ **do**
  for $l \leftarrow 0$ ***to*** $|\mathcal{L}| - 1$ **do**
    if $l = 0$ **then**
      perform round-trip initialization ;
    endif
    solve minimum cost flow problem for digraph $\mathcal{G}^l$ with respect to $(\boldsymbol{\nu}, \boldsymbol{\mu})^k$;
    determine $\tilde{t}^l$ according to (18) ;
    update b_{ij}^l, t^l for all $(i,j) \in \mathcal{E}^l$ according to (23) if applicable ;
    update μ_i^l from k to $k+1$ for all nodes if power constraints are enforced ;
  endfor
  update $\boldsymbol{\nu}$ from k to $k+1$ for all nodes ;
  refine all lifetime upper bounds $\hat{T}^{l,k}$ according to (22) if applicable ;
  test termination criterion for $(\boldsymbol{\nu}, \boldsymbol{\mu})^k$ and $(\boldsymbol{\nu}, \boldsymbol{\mu})^{k+1}$;
  if *termination criterion is true* **then**
    return;
  else
    $k \leftarrow k+1$;
  endif
endw

Fig. 2: Pseudocode for the maximum lifetime algorithm.

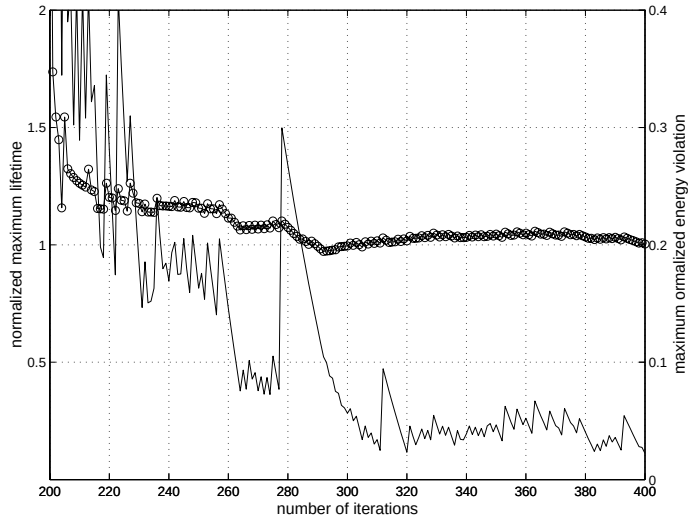

Fig. 3. Lifetime and violation evolution for a sample network under energy constraints.

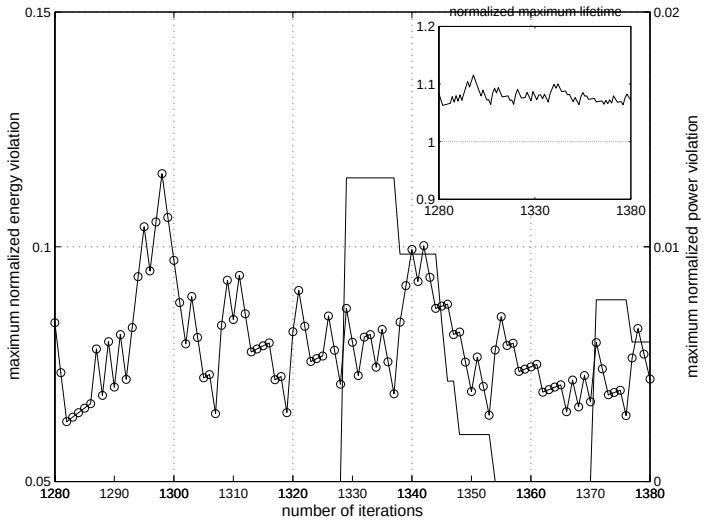

Fig. 4. Lifetime and violation evolution for a sample network under energy/power constraints.