A Combinatorial Algorithm for the Maximum Lifetime Data Gathering with Aggregation Problem in Sensor Networks

Konstantinos Kalpakis*,[a], Shilang Tang[a]

[a] *Computer Science & Electrical Engineering Department, University of Maryland Baltimore County. 1000 Hilltop Circle, Baltimore, MD 21250, USA*

Abstract

Performing tasks energy efficiently in a wireless sensor network (WSN) is a critical issue for the successful deployment and operation of such networks. Gathering data from all the sensors to a base station, especially with in–network aggregation, is an important problem that has received a lot of attention recently.

The Maximum Lifetime Data Gathering with Aggregation (MLDA) problem deals with maximizing the system lifetime T so that we can perform T rounds of data gathering with in–network aggregation, given the initial available energy of the sensors. A solution of value T to the MLDA problem consists of a collection of aggregation trees together with the number of rounds each such tree should be used in order to achieve lifetime T.

We describe a combinatorial iterative algorithm for finding an optimal continuous solution to the MLDA problem that consists of up to $n-1$ aggregation trees and achieves lifetime T_o, which depends on the network topology and initial energy available at the sensors. We obtain an α–approximate optimal integral solution by simply rounding down the optimal continuous solution, where $\alpha = (T_o - n + 1)/T_o$. Since in practice $T_o \gg n$, $\alpha \approx 1$. We get asymptotically optimal integral solutions to the MLDA problem whenever the optimal continuous solution is $\omega(n)$. Furthermore, we demonstrate the efficiency and effectiveness of the proposed algorithm via extensive experimental results.

Key words: wireless sensor networks, lifetime maximization, combinatorial algorithms, linear programming

1. Introduction

Recent technological advances have led to the development of wireless networks of micro–sensors [17, 25, 31]. Such networks are expected to consist of numerous inexpensive micro–sensors, readily deployable in various physical environments to collect useful information (e.g. seismic, acoustic, medical and surveillance data) in a robust and autonomous manner. However, there are several obstacles that

*Corresponding author.

Email addresses: kalpakis@csee.umbc.edu (Konstantinos Kalpakis), stang2@csee.umbc.edu (Shilang Tang)

need to be overcome before this vision becomes a reality – see Akyildiz et al [1] for a comprehensive survey of issues arising in wireless sensor networks. These obstacles are due to the limited energy, computing capabilities, and communication resources available to the sensors. Often, replenishing the energy of the sensors by replacing their batteries is cost prohibitive or even infeasible. Conserving the sensor energy, so as to maximize the system's lifetime, has been one of the key challenges.

In this paper we consider the problem of Maximum Lifetime Data Gathering with in-network Aggregation (MLDA) in wireless sensor networks. This problem deals with maximizing the system lifetime T so that we can perform T rounds of data gathering with in–network aggregation, given the initial available energy of the sensors. A solution of value T to the MLDA problem consists of a collection of aggregation trees together with the number of rounds each such tree should be used in order to achieve lifetime T. The notion of a data aggregation tree is introduced in [18, 9] to specify how data are gathered and aggregated from the sensors to the base station, which is a directed tree in which every sensor has a directed path towards the base station.

Data gathering with in-network data aggregation is a useful paradigm for increasing the system's lifetime [15, 23, 24]. In–network aggregation allows sensors to aggregate multiple input data packets into one output data packet; often, in each round, a sensor aggregates all the data packets it receives with its own data packet. In-network aggregation is most useful in computing for each round various easily computable and composable (i.e. suitable for in–network aggregation) statistical descriptors of the sensor measurements, such as the minimum, maximum, average, variance, approximate histograms, uniform fixed-size samples, measurements of high frequency, and various sketches [10, 13, 30, 34].

Although the MLDA problem is NP–hard, heuristics in [18] show that (1) tight approximate solutions to the network design problem can be obtained in reasonable time for small to medium–size networks, and (2) a collection of aggregation trees, together with the number of rounds each such tree is to be used in order to achieve lifetime T, can be determined in polynomial–time.

Based on the formulation in [9], we present a simple and efficient combinatorial iterative algorithm for the MLDA problem, while implicitly considering all the up to n^{n-2} aggregation trees for a WSN with n nodes. Our algorithm finds optimal solutions of small number of aggregation trees, which is desired since the sensor nodes are computation and communication resources limited. Our algorithm is based on the Revised Simplex method with a column generation scheme. In summary, our original contributions are as follows:

- provide an efficient combinatorial iterative algorithm (called RSM–MLDA) for finding an optimal continuous solution to the MLDA problem. The solution consists of at most $n-1$ aggregation trees for a WSN with n nodes.
- find α–approximate integral solutions to the MLDA problem for a WSN with n nodes and optimal continuous lifetime T_o, where $\alpha = (T_o - n + 1)/T_o$. Such solutions are computed from rounding down solutions obtained by the RSM–MLDA algorithm. Since in practice $T_o \gg n$, $\alpha \approx 1$.

- provide an upper bound on the system lifetime, which is easily computable during each iteration of the RSM–MLDA algorithm. This upper bound enables us to estimate a lower bound on the approximate ratio of the solution at each iteration of RSM–MLDA, and hence terminate early if that ratio exceeds a desired threshold.

- provide sensitivity analysis of the solution for bounding the change of data gathering energy budget of each sensor. The energy variability bound can be used by sensors to set up or adjust their energy spending profile.

Moreover, we conduct an extensive experimental evaluation of the proposed approach illuminating its practicability.

The rest of the paper is organized as follows. In section 2 we give the network model and the notations used in this paper. We describe our combinatorial algorithm for the MLDA problem in section 3. Estimating an upper bound on the optimal solution of the MLDA problem is given in section 3.3. We analyze the sensitivity of the optimal solutions to bound the changes of data gathering energy budget of the sensors in section 4. Experimental results are provided in section 5. We discuss related work in section 6, and conclude in section 7.

2. Preliminaries

Consider a wireless sensor network (WSN) with n nodes. One node, denoted b, is designated as the base station, with the remaining nodes being sensors. Sensors are assumed to have limited non–replenishable energy while the base station has no energy limitations. Time is discrete, and at each time period, we are interested in gathering the data from all the sensors to make them available at the base station. During data gathering, in–network aggregation is assumed, i.e. any number of incoming data packets at a node can be aggregated into a single outgoing data packet. [1] Without loss of generality (w.l.o.g.), we assume that data packets have fixed size. The system lifetime T is the earliest time at which one or more sensors deplete their available energy. [2]

Given a graph G, we denote its vertex and edge sets with $V[G]$ and $E[G]$ respectively. For brevity, we often write $v \in G$ instead of $v \in V[G]$ for a vertex v, and $ij \in G$ instead of $ij \in E[G]$ for an edge ij. A subset of vertices $S \subseteq V[G]$ induces the subgraph $G[S] = (S, E[S])$ of G where $E[S] = E[G] \cap S \times S$. A subset $E' \subseteq E$ of edges induces the subgraph $G[E'] = (V, E')$ of G.

The topology of the wireless sensor network is modeled by a directed graph (digraph) G, with $V[G] = \{1, 2, \ldots, n\}$ and $E[G] \subseteq V \times V$. There exists an edge $ij \in E[G]$ whenever i can successfully send a packet to j. Let d_{ij} be the distance

[1] Incoming and outgoing packets carry concise fixed–size summaries of the sensor data.

[2] Or more precisely, the system lifetime T is the earliest time at which one or more sensors deplete their energy budgeted for data gathering communications. In this work, we assume each sensor has an energy spending profile and has made such a budget. It is a more realistic assumption than that data gathering communications can use all the energy of the sensors, and has no impact on the applicability of our algorithm.

between i and j. Let τ_{ij} be the energy consumed by node i in order to transmit a single packet to node j, and let r_j be the energy needed to receive such a packet at node j. Often τ_{ij} is monotonically non–decreasing with the Euclidean distance between nodes i and j, i.e. the same or more energy is required to transmit a packet at longer distances. Let ϵ_i be the energy available at node i. We assume, w.l.o.g., that $\epsilon_b = \infty$, $r_b = 0$, $\tau_{bi} = 0$, $\forall i \in V$. For simplicity we consider sensor networks with a single base station. Sensor networks with multiple base stations can be easily handled as follows. Introduce a new node, b', to serve as the new single base station, and then append to G, for each current base station i, the new edge ib' with $\tau_{ib'} = 0$.

Given a digraph G, a *branching* $\mathcal{T}$ rooted at node $b \in V[G]$ is a subgraph of G such that each node has a directed path to b in $\mathcal{T}$, and its undirected version is a tree spanning all the nodes in $V[G]$. In the context of WSNs, we refer to such a branching $\mathcal{T}$ as an *aggregation tree* since it indicates how to gather data from all the sensors to the base station with in–network aggregation.

We denote with $\mathfrak{I} = \langle G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \rangle$ an instance of the MLDA problem for a a wireless sensor network G with base station b, node transmit and receive energy costs $\boldsymbol{\tau}$ and $\mathbf{r}$, and node energy budget $\boldsymbol{\epsilon}$. A solution to $\mathfrak{I}$ with lifetime T consists of a collection of aggregation trees $\mathcal{T}$ together with the number of rounds each tree is to be used to achieve the maximum lifetime T (which is obviously equal to the total number of rounds).

Given an instance $\mathfrak{I}$ of an optimization problem, let $\textsc{opt}(\mathfrak{I})$ and $\textsc{sol}(\mathfrak{I})$ be an optimal and a feasible solution to $\mathfrak{I}$, respectively. For brevity, $\textsc{opt}(\mathfrak{I})$ and $\textsc{sol}(\mathfrak{I})$ will also indicate the value of the corresponding solution. The relative error of a solution $\textsc{sol}(\mathfrak{I})$ is $|\textsc{opt}(\mathfrak{I}) - \textsc{sol}(\mathfrak{I})| / \textsc{opt}(\mathfrak{I})$ and its approximation ratio is $\textsc{sol}(\mathfrak{I})/\textsc{opt}(\mathfrak{I})$. We refer to continuous (integral) solutions to instances of optimization problems, whenever fractional values for their unknowns are allowed (not allowed).

We denote vectors and matrices with lower and upper case bold letters, e.g. $\mathbf{x}_{n\times 1}$ and $\mathbf{A}_{n\times m}$ is an n–dimensional vector and an $n \times m$ matrix respectively. For brevity we omit the vector and matrix dimensions when they are clear from the context. Given an $n \times m$ matrix $\mathbf{A}_{n\times m}$ and two index sequences $I \subseteq \{1, 2, \ldots, n\}$ and $J \subseteq \{1, 2, \ldots, m\}$, we denote by $\mathbf{A}_{I,J}$ the sub–matrix $\mathbf{X}_{|I|\times|J|} = (x_{ij})$ of $\mathbf{A}$ where $x_{ij} = a_{I_i,J_j}$, and by $\mathbf{A}_J$ the sub–matrix $\mathbf{X}_{n\times|J|} = (x_{ij})$ of $\mathbf{A}$, where $x_{ij} = a_{iJ_j}$. For brevity, we indicate the ith column of $\mathbf{A}$ with $\mathbf{A}_i$. We denote the transpose of a matrix $\mathbf{A}$ with $\mathbf{A}^T$. The support of a vector $\mathbf{x}$ is defined as $\mathbb{I}(\mathbf{x}) = \{i \ : \ x_i \neq 0\}$. Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$, we say that $\mathbf{x}$ *dominates* $\mathbf{y}$ and write $\mathbf{x} \geq \mathbf{y}$, if $x_i \geq y_i$ for $i = 1, 2, \ldots, m$. We say that $\mathbf{x}$ is *lexicographically larger (smaller)* than $\mathbf{y}$ if the smallest index non–zero component of $\mathbf{x} - \mathbf{y}$ is positive (negative).

3. A Combinatorial Algorithm for the MLDA Problem

In this section we describe our combinatorial algorithm for the MLDA problem using the same LP formulation used in [9], while considering all of the up to n^{n-2} aggregation trees of a WSN with n nodes. Our algorithm is based on the

Revised Simplex method with the lexico–min rule. [3]

Consider an instance $\mathfrak{I} = \langle G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \rangle$ of the MLDA problem. For brevity, let $V = V[G]$ and $E = E[G]$ be the set of vertices and edges of G respectively. Let $\mathcal{T}$ be the set of all aggregation trees (branchings rooted at b) for the instance $\mathfrak{I}$. A feasible solution to $\mathfrak{I}$ can be described by a collection of aggregation trees (or branchings) $\mathcal{T}_i \in \mathcal{T}$ together with the number of rounds x_i each such tree is used. There is a natural bijection between the branchings of G which are rooted at b and the up to $|V|^{|V|-2}$ labeled spanning trees of G: direct all edges of a labeled spanning tree towards b to get the corresponding branching. Hence, the size of $\mathcal{T}$ is at most $|V|^{|V|-2}$ (since G may not be the complete graph). Let $\mu_j(\mathcal{T}_i)$ be the energy consumed by node j when performing data gathering with in–network aggregation using the branching $\mathcal{T}_i \in \mathcal{T}$.

A continuous (integral) solution to the MLDA problem instance $\mathfrak{I}$ can be found by solving the following linear (mixed integer) program

$$\text{LP:} \left\{ \begin{array}{ll} \max \sum_{\mathcal{T}_i \in \mathcal{T}} x_i & \text{such that} \\ s_j + \sum_{\mathcal{T}_i \in \mathcal{T}} x_i \cdot \mu_j(\mathcal{T}_i) = \epsilon_j, & \forall j \in V \\ x_i \geq 0, & \forall \mathcal{T}_i \in \mathcal{T} \\ s_j \geq 0, & \forall j \in V \end{array} \right\} \tag{1}$$

Extend $\mathbf{x}$ with n components, such that x_{m+i} is identified with the ith slack variable s_i, where $m = |\mathcal{T}|$. Define the $n \times (m+n)$ matrix $\mathbf{A}$ so that $\mathbf{A}_i = \mu(\mathcal{T}_i)$, $i = 1, 2, \ldots, m$, and $\mathbf{A}_{m+i}$ is the ith column of the $n \times n$ identity matrix $\mathbf{I}_n$, $i = 1, 2, \ldots, n$. Let $\mathbf{c} \in \mathbb{R}^{m+n}$ have its first m components equal to -1, and with all the rest equal to 0. Let $\mathbf{b} \in \mathbb{R}^n$ be equal to the vector of initial energies $\boldsymbol{\epsilon}$ of the sensors budgeted for data gathering communications. The LP above is now in the standard form

$$\text{LP:} \left\{ \begin{array}{ll} \min \mathbf{c}^T \cdot \mathbf{x} & \text{such that} \\ \mathbf{A} \cdot \mathbf{x} = \mathbf{b}, & \\ \mathbf{x} \geq \mathbf{0}, & \end{array} \right\} \tag{2}$$

The value $\mathbf{c}^T \cdot \mathbf{x}$ of each feasible solution $\mathbf{x}$ is the achieved system lifetime. An initial feasible basis for LP is $\mathcal{B} = \langle m+1, m+2, \ldots, m+n \rangle$, with corresponding bfs $\mathbf{x}_\mathcal{B} = \mathbf{b} = \boldsymbol{\epsilon}$. In our algorithm for solving the LP above, we explicitly store up to n columns of $\mathbf{A}$ at any point in time. All other columns of $\mathbf{A}$ will be generated on demand as needed.

Consider a feasible basis $\mathcal{B}$ of the the LP above. The shadow prices vector for $\mathcal{B}$ is $\boldsymbol{\pi}^T = \mathbf{c}_\mathcal{B} \cdot \mathbf{A}_\mathcal{B}^{-1}$. We define the *unit energy price* for a node j to be equal to $\phi_j = -\pi_j$, and the energy cost of a branching $\mathcal{T}_k \in \mathcal{T}$ to be equal to $\sum_{j \in V} \phi_j \mu_j(\mathcal{T}_k)$. Observe that the energy cost of $\mathcal{T}_k$ is equal to

$$\begin{aligned} \sum_{j \in V} \phi_j \cdot \mu_j(\mathcal{T}_k) &= \sum_{j \in V} \phi_j \left(\sum_{ji \in \mathcal{T}_k} \tau_{ji} + \sum_{ij \in \mathcal{T}_k} r_j \right) \\ &= \sum_{ij \in \mathcal{T}_k} (\phi_i \tau_{ij} + \phi_j r_j) \end{aligned}$$

[3] We provide an overview of some concepts in linear programming used by this section in Appendix A. The reader is referred to a text such as [20, 4] for further details.

$$= \sum_{ij \in \mathcal{T}_k} w_{ij} = w(\mathcal{T}_k), \tag{3}$$

where we define the weight of an edge $ij \in E$ to be equal to $w_{ij} = \phi_i \tau_{ij} + \phi_j r_j$.

Consider now a non–basic column $\mathbf{A}_k$ corresponding to a branching $\mathcal{T}_k \in \mathcal{T}$. The relative cost of $\mathbf{A}_k$ is

$$\begin{aligned} \bar{c}_k &= c_k - \boldsymbol{\pi}^T \cdot \mu(\mathcal{T}_k) \\ &= -1 - \boldsymbol{\pi}^T \cdot \mu(\mathcal{T}_k) = w(\mathcal{T}_k) - 1. \end{aligned} \tag{4}$$

Column $\mathbf{A}_k$ may enter the current basis $\mathcal{B}$ if its relative cost is negative. Hence, the branching $\mathcal{T}_k$ is a candidate to enter $\mathcal{B}$ if its energy cost $w(\mathcal{T}_k)$ is less than 1. [4] Therefore, it is sufficient to find a branching $\mathcal{T}_k \in \mathcal{T}$ with minimum energy cost, and if its cost is < 1, then $\mathcal{T}_k$ enters $\mathcal{B}$, otherwise, the algorithm has found an optimal basis and it terminates. Upon finding such a branching $\mathcal{T}_k$, the variable x_k enters the basis $\mathcal{B}$, and a basic column $\mathbf{A}_l$ is chosen to leave $\mathcal{B}$ using the lexico–min rule. The basic column $\mathbf{A}_l$ corresponds to the branching $\mathcal{T}_{\mathcal{B}(l)}$, if $\mathcal{B}(l) \leq m$, or to the slack variable $s_{\mathcal{B}(l)-m}$ otherwise. [5]

The complete description of our iterative RSM–MLDA algorithm for finding a continuous optimal solution to the MLDA problem is given in Algorithm 1. Our algorithm finds minimum weight branchings in a digraph using an algorithm given in section 3.2 below. The correctness of our algorithm follows from the discussion above and the properties of the Revised Simplex method with the lexico–min rule. The memory requirements of our algorithm is $O(V^2)$. The worst–case running time of each iteration of our algorithm is bounded as follows: $O(VE+V^2)$ for finding a minimum weight branching, $O(V^3)$ to invert the matrix $\mathbf{D}$, and $O(V^2)$ total time for all other operations, for a grand total of $O(V^3)$ worst–case running time per iteration. The running time can be reduced to $O(VE + V^2)$ by utilizing the matrix–inversion lemma to update $\mathbf{D}^{-1}$ after each pivoting step in $O(V^2)$ time instead of recomputing $\mathbf{D}^{-1}$ each time from scratch.

3.1. Getting α–approximate Integral Solutions

Consider an instance $\mathfrak{I} = \langle G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \rangle$ of the MLDA problem and a continuous optimal basic solution $\mathbf{x}$ with lifetime T, computed by the RSM–MLDA algorithm. Since the support of $\mathbf{x}$ has size at most $|V[G]|$ and the slack variable that corresponds to b is always basic, the number of distinct aggregation trees in the continuous optimal bfs is $\leq |V[G]|-1$. By rounding down the components of $\mathbf{x}$, we get an integral solution to $\mathfrak{I}$ with lifetime $\geq T-|V[G]|+1$, i.e. we get an integral solution to MLDA which is optimal within a factor of $\alpha = (T-|V[G]|+1)/T$. For example, if $T = 1000$ and $|V[G]| = 50$, we get an integral solution which is optimal within a factor of 95%. Whenever $T = \Theta(\boldsymbol{\epsilon}) = \omega(|V[G]|)$, we find asymptotically optimal integral solutions to the MLDA problem.

[4] The energy cost of the branching that corresponds to a basic column is equal to 1.

[5] The slack variable x_{m+b} never exits the basis, since $\epsilon_b > 0$ and $\mu_b(\mathcal{T}_i) = 0$ for all $\mathcal{T}_i \in \mathcal{T}$.

Algorithm 1 The RSM–MLDA algorithm for finding an optimal continuous solution to the MLDA problem.

Input: MLDA instance $\mathfrak{I} = \langle\, G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \,\rangle$

Output: maximum lifetime T, together with at most $|V| - 1$ branchings rooted at b and their #rounds

// initialize

$\mathbf{D} \longleftarrow$ identity matrix of order $|V|$

let $\mathbf{d}_i$ denote the ith row of $\mathbf{D}$

for each $1 \leq i \leq |V|$ **do**

  $branching[i] \longleftarrow$ **null**

  $rounds[i] \longleftarrow 0$

  $c[i] \longleftarrow 0$

// iterate

for at most $|V|^{|V-2|}$ iterations **do**

  $\boldsymbol{\phi} \longleftarrow -\mathbf{c} \cdot \mathbf{D}^{-1}$ and $\mathbf{x} \longleftarrow \mathbf{D}^{-1} \cdot \boldsymbol{\epsilon}$

  for each edge $ij \in E$ **do**

    $w_{ij} \longleftarrow \phi_i \tau_{ij} + \phi_j r_j$

  // find candidate branchings to enter and leave the current basis

  $\mathcal{T}_k \longleftarrow$ GetMWB $(G, b, \mathbf{w})$

  if $w(\mathcal{T}_k) \geq 1$ **then**

    break

  $\mathbf{u} \longleftarrow \mathbf{D}^{-1} \cdot \mu(\mathcal{T}_k)$

  if the support $\mathbb{I}(\mathbf{u})$ is empty **then**

    return *instance is unbounded*

  $l \longleftarrow \arg\,\text{lexico–min}\{[x_i, \mathbf{d}_i]/u_i \;:\; i \in \mathbb{I}(\mathbf{u})\}$

  $branching[l] \longleftarrow \mathcal{T}_k$

  $\mathbf{D}_l \longleftarrow \mu(\mathcal{T}_k)$

  $c[l] \longleftarrow -1$

  // update the rounds for each branching in the basis

  $rounds \longleftarrow \mathbf{D}^{-1}\boldsymbol{\epsilon}$

  for $1 \leq i \leq |V|$ and $branching[i] =$ **null do**

    $rounds[i] \longleftarrow 0$

// return optimal solution

$T \longleftarrow$ sum of all $rounds[i]$

return $\langle\, T, rounds, branching \,\rangle$

3.2. Computing Minimum Weight Branchings

We turn our attention to finding a minimum weight branching for a digraph $G = (V, E)$ rooted at a node $r \in V$ and with edge weights $w : E \longrightarrow \mathbb{R}$. Tarjan [36] provides an more efficient algorithm for the minimum weight branching for a digraph G whose running time is $O(\min\{E \log V, V^2\})$. We provide a simple, but slower, algorithm whose detailed description is given in Algorithm 2. The algorithm is an instance of the local–ratio algorithm design paradigm [3]. Correctness of the algorithm can be shown by induction on the number of iterations of the outer–loop, and using the following observation. For each iteration and each strongly connected component (SCC) G_i found during that iteration, any minimum weight branching must use an edge leaving each such G_i that does not contain the root. Since each iteration of the outer–loop takes $O(V + E)$ time, and the number of iterations is at most $|V|$, the worst–case running time of GetMWB is $O(V^2 + VE) = O(V^3)$. Its total memory requirements is $O(V^2)$.

Algorithm 2 The GetMWB algorithm for finding a minimum weight branching rooted at r.

Input: $\langle\, G = (V, E), r, \mathbf{w} \,\rangle$
Output: minimum weight branching $\mathcal{T}$ rooted at r

while at most $|V|$ iterations **do**
  let $E_o = \{\ e \in E\ :\ w_e = 0\}$
  let $G_o = (V, E_o)$
  let $\mathcal{T}$ be the transpose of a BFS tree of G_o^T rooted at r
  if $\mathcal{T}$ spans all the nodes in V **then**
    return $\mathcal{T}$
  for each SCC $G_i = (V_i, E_i)$ of G_o **do**
    $\partial \longleftarrow E \cap V_i \times \overline{V_i}$
    $\theta \longleftarrow \min\{w_e\ :\ e \in \partial\}$
    for each $e \in \partial$ **do**
      $w_e \longleftarrow w_e - \theta$
return no branching exists

3.3. An Upper Bound on the Lifetime

We compute an upper bound on the lifetime during each iteration of the RSM–MLDA algorithm using duality theory of linear programming. Consider an instance $\langle\, G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \,\rangle$ of the MLDA problem, a convenient way to write a primal linear program equivalent to the one in (1) is

$$\left\{ \begin{array}{ll} \max \mathbf{1}^T \cdot \mathbf{x} & \text{such that} \\ \mathbf{A} \cdot \mathbf{x} \leq \boldsymbol{\epsilon}, & \\ \mathbf{x} \geq \mathbf{0}, & \mathbf{x} \in \mathbb{R}^{|\mathcal{T}|} \end{array} \right\} \tag{5}$$

where the matrix $\mathbf{A}$ has as columns the vectors $\mu(\mathcal{T}_k)$, for all branching $\mathcal{T}_k \in \mathcal{T}$, and $\mathcal{T}$ is the set of all branchings of G rooted at b. The dual of this linear program is

$$\left\{ \begin{array}{ll} \min \boldsymbol{\epsilon}^T \cdot \mathbf{y} & \text{such that} \\ \mathbf{A}^T \cdot \mathbf{y} \geq \mathbf{1}, & \\ \mathbf{y} \geq \mathbf{0}, & \mathbf{y} \in \mathbb{R}^{|V|} \end{array} \right\} \tag{6}$$

Observe that both primal and dual linear programs above are feasible, which implies that strong duality holds. Consider a pair of feasible solutions $\mathbf{x}$ and $\mathbf{y}$ to the linear programs in (5) and (6). Strong duality implies that

$$\mathbf{1}^T \cdot \mathbf{x} \leq \boldsymbol{\epsilon}^T \cdot \mathbf{y} \tag{7}$$

with equality when both $\mathbf{x}$ and $\mathbf{y}$ are respectively optimal.

Algorithm 3 The GetUB procedure for computing an upper bound on the system lifetime.

Input: MLDA instance $\langle G, b, \boldsymbol{\tau}, \mathbf{r}, \boldsymbol{\epsilon} \rangle$ and energy prices $\boldsymbol{\phi}$
Output: upper bound on the system lifetime.

```
1: z ⟵ max{−φ_{V−b}, 0}
2: for each edge ij ∈ E do
3:     w_ij ⟵ z_i τ_ij + z_j r_j
4: T_k ⟵ GetMWB (G, b, w)
5: y ⟵ z/w(T_k)
6: θ ⟵ ε^T · y
7: return θ
```

In Algorithm 3, we provide the detailed description of the GetUB algorithm for computing an upper bound on the lifetime of the given MLDA instance and node energy prices $\boldsymbol{\phi} \in \mathbb{R}^{|V|}$. Recall that the shadow prices for the constraints of the linear program in (5) correspond to the negated energy prices of the nodes. Let $\mathbf{z} = \max\{-\boldsymbol{\phi}, \mathbf{0}\}$. Consider the edge weight function $w : E \longrightarrow \mathbb{R}$ that corresponds to $\mathbf{z}$, i.e. $w_{ij} = z_i\tau_{ij} + z_j r_j$ for each edge $ij \in E$. Let $\mathcal{T}_k \in \mathcal{T}$ be a minimum weight branching for the edge weights $\mathbf{w}$. Observe that each row of $\mathbf{A}^T \cdot \mathbf{z}$ is equal to the energy cost of the corresponding branching, with energy prices given by $\mathbf{z}$, which in turn is equal to the weight of the same branchings under the edge weights $\mathbf{w}$. Therefore,

$$\mathbf{A}^T \cdot \mathbf{z} \geq w(\mathcal{T}_k) \cdot \mathbf{1}. \tag{8}$$

Consequently, the vector $\mathbf{y} = \max\{-\boldsymbol{\phi}, \mathbf{0}\}/w(\mathcal{T}_k)$ is a feasible solution to the dual linear program in (6). Moreover, $\boldsymbol{\epsilon}^T \cdot \mathbf{y}$ is an upper bound on the system lifetime for the given MLDA instance and energy prices $\boldsymbol{\phi}$.

The RSM–MLDA algorithm, computes at each iteration a basic feasible solution $\mathbf{x}$ for the primal linear program in (5), together with a vector of node energy prices $\boldsymbol{\phi}$. By calling the GetUB routine with parameters $\mathfrak{I}$ and $\boldsymbol{\phi}$, we obtain an upper bound on the optimal system lifetime. At each iteration of the RSM–MLDA algorithm, we use the smallest upper bounds computed so far as our best upper bound on the system lifetime. Therefore, we can continuously assess the distance of the current feasible solution $\mathbf{x}$ from optimality, and we can terminate the RSM–MLDA algorithm early if that distance is below some desired threshold.

4. Sensitivity to the Variability of Initial Data Gathering Energy Budget

We consider the impact of changes on the sensor's data gathering energy budget $\boldsymbol{\epsilon}$ to the optimal solution for MLDA instances found by our RSM–MLDA

algorithm. In real deployment, it is not uncommon for the available data gathering energy of sensors to become different to the initial value $\boldsymbol{\epsilon}$ during the operation of the network. Such changes may be due to a number of factors, e.g. unexpected computations, battery leakages, extra energy for transmissions due to channel errors, etc.

Let $\mathcal{B}$ and $\mathbf{x}$ be the basis and the corresponding bfs of the optimal solution by RSM–MLDA algorithm. Suppose that the energy budget of a single sensor i changes from ϵ_i to $\epsilon_i - \delta_i$, while the energy budget of all other sensors remains the same. Since the current solution $\mathbf{x}$ is still feasible when $\delta_i \leq 0$, we limit our attention to the cases where $\delta_i \geq 0$, i.e. a sensor has less energy budget for data gathering than initially assumed.

We compute upper bounds on δ_i to achieve certain goals. Let $\mathbf{e}^{(i)}$ be the ith unit vector in $\mathbb{R}^n$ and $\boldsymbol{\nu}^{(i)} = \mathbf{A}_{\mathcal{B}}^{-1} \cdot \mathbf{e}^{(i)}$. Intuitively, to compensate for the loss of a unit of energy at sensor i, the number of rounds of the jth aggregation tree needs to be reduced by $\nu_j^{(i)}$. Observe that when the energy budget changes from $\boldsymbol{\epsilon}$ to $\boldsymbol{\epsilon} - \delta_i \cdot \mathbf{e}^{(i)}$, the current bfs changes from $\mathbf{x} = \mathbf{A}^{-1} \cdot \boldsymbol{\epsilon}$ to

$$\mathbf{A}^{-1} \cdot (\boldsymbol{\epsilon} - \delta_i \cdot \mathbf{e}^{(i)}) = \mathbf{x} - \delta_i \boldsymbol{\nu}^{(i)}. \tag{9}$$

Therefore, in order for the current basis $\mathcal{B}$ to still be feasible, we need $\mathbf{x} - \delta_i \boldsymbol{\nu}^{(i)} \geq \mathbf{0}$, which implies that it is both necessary and sufficient to have

$$\delta_i \leq \min_{\nu_j^{(i)} > 0} \{\frac{x_j}{\nu_j^{(i)}}\} = \delta_i^{(\max)}. \tag{10}$$

Intuitively, $\delta_i^{(\max)}$ is the maximum energy loss at sensor i that can be compensated by reducing the number of rounds of each aggregation tree with $\nu_j^{(i)} > 0$ without affecting the energy balance of the other sensors. [6]

We may desire to achieve additional goals besides feasibility of the current basis $\mathcal{B}$. We consider two such additional goals below. First, we may want to bound the change in the number of rounds that any single aggregation tree is used under the modified energy budget by some parameter $|\triangle x|$. To this end, it is sufficient to ensure that $|\delta_i \nu_j^{(i)}| \leq |\triangle x|$ for all $1 \leq j \leq n$. Hence, to maintain feasibility of the current basis and bound the change on the number of rounds of any aggregation tree, it is both necessary and sufficient to have

$$\delta_i \leq \min_{\boldsymbol{\nu}_j^{(i)} \neq 0} \{\frac{|\triangle x|}{|\boldsymbol{\nu}_j^{(i)}|}\}. \tag{11}$$

Second, we may want to ensure that when the energy budget changes from $\boldsymbol{\epsilon}$ to $\boldsymbol{\epsilon} - \delta_i \cdot \mathbf{e}^{(i)}$, the optimal lifetime changes from T to $T - \triangle T \geq (1 - \alpha)T$, for some parameter α. Since $\triangle T = \mathbf{c}^T \cdot (\delta_i \boldsymbol{\nu}^{(i)})$ and $T = \mathbf{c}^T \cdot \mathbf{A}_{\mathcal{B}}^{-1} \cdot \boldsymbol{\epsilon}$, we need

$$\mathbf{c}^T \cdot (\delta_i \boldsymbol{\nu}^{(i)}) \leq \alpha \cdot \mathbf{c}^T \cdot \mathbf{A}_{\mathcal{B}}^{-1} \cdot \boldsymbol{\epsilon}. \tag{12}$$

[6] If $\nu_j^{(i)} < 0$ then the number of rounds of the jth tree is increased.

Therefore, to ensure both the feasibility and the the upper bound on the change of the optimal lifetime it is both necessary and sufficient to have

$$\delta_i \leq \min \left\{ \alpha \frac{\mathbf{c}^T \cdot \mathbf{A}_{\mathcal{B}}^{-1} \cdot \boldsymbol{\epsilon}}{\mathbf{c}^T \cdot \boldsymbol{\nu}^{(i)}}, \delta_i^{(\max)} \right\}. \tag{13}$$

The above analysis assumes that the energy budget changes for a single sensor only, while the energy budget of all other sensors remains the same. We can extend this analysis to compute energy budget bounds for individual sensors to the case where any subset of $k \leq n$ sensors experiences a change in their energy budget. To this end, we simply divide the bounds on the change of the data gathering energy budget given by in Eqs. (10), (11), or (13) by k.

The bounds δ_i on the change of the data gathering energy budget of a sensor i provided by Eqs. (10), (11), or (13) can be utilized reactively and/or proactively. In reactive mode, each sensor i is required to notify the base station when its data gathering energy budget is reduced by more than δ_i. Upon receiving such notification, the base station may need to update the number of rounds that each aggregation tree is used, or re–run the RSM–MLDA algorithm to find a new optimal solution. In proactive mode, each sensor i would dynamically adjust its energy spending profile to ensure that its data gathering energy budget stays is at least $\epsilon_i - \delta_i$.

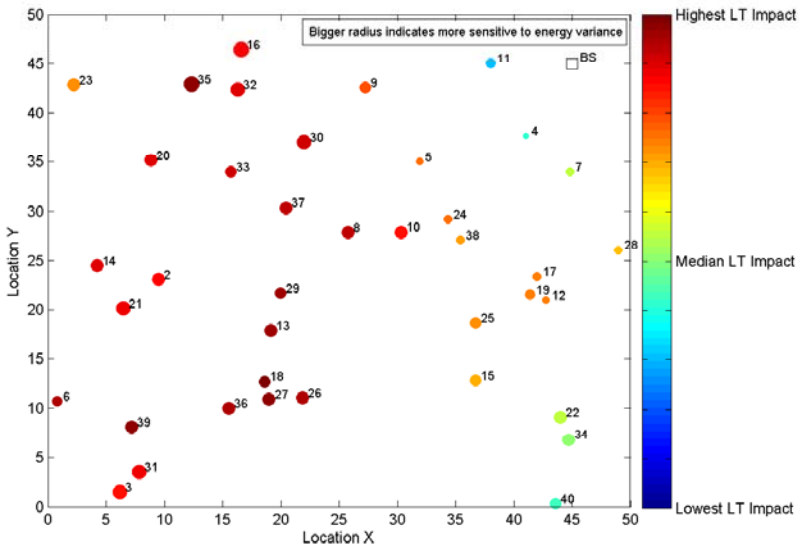

Figure 1: Energy sensitivity map of a network instance of 40 sensors deployed in a $50m \times 50m$ field.

To visualize the impact on the network lifetime and the intolerance to the changes of data gathering energy budget for each sensor in the current feasible solution, we plot a sensitivity map. The intolerance to energy budget change for sensor i is the inverse of its bound $\delta_i^{(\max)}$ given in Eq. (10). In this map, each sensor is marked by a colored circle at its location in the field. The radius of a sensor's circle is proportional to its intolerance to energy changes. The color of the circle indicates the impact (change) of a unit energy of the sensor on the optimal

lifetime, from red for the highest impact to blue for the lowest impact. Fig. 1 shows an example sensitivity map for a network instance of 40 nodes with each sensor having $1J$ as its initial data gathering energy budget. Such a sensitivity map helps us to quickly and visually identify sensitive sensors, for instance sensors 13 and 35 which have red–colored circles with rather large radius, and to develop network management strategies accordingly.

5. Experimental Evaluation

To evaluate the performance of our RSM–MLDA algorithm, we consider sensor networks in which sensors are uniformly distributed in a $50m \times 50m$ sensing field while the base station is fixed at location $(45, 45)$. We generate 20 random networks for each network size n, where n takes values between 10–100 in multiples of 10. Our algorithm requires as input the transmit and receive energies τ_{ij} and r_j for each sensor node i and j, respectively. To this end, in our experiments, we assume packets of size 1000 bits, and we use the simple first order radio model introduced by the seminal paper of Heinzelman et al [11].

The first order radio model [11] has four parameters: the energy/bit dissipated at the transmitter circuitry (E_{Tx}), the receiver circuitry (E_{Rx}), and the power amplifier circuitry (E_{amp}), and a path loss exponent α. The path loss exponent α depends on the propagation environment and typically varies between 2 (for free–space communications) and 4 (for areas with obstacles etc) [19]. [7] In the first order radio model the energy required to transmit a k–bit message at distance d is $(E_{Tx} + E_{amp} \cdot d^{\alpha}) \cdot k$ and the energy to receive a k–bit message is $E_{Rx} \cdot k$. The default values for these parameters, as given in Heinzelman et al [11], are $E_{Tx} = 50nJ/bit$, $E_{Rx} = 50nJ/bit$, $E_{amp} = 100pJ/bit/m^2$, and $\alpha = 2$. These default parameter values provide energy consumption that is similar to the actual energy consumption of various low–power radios [29]. [8] We refer to the first order radio model with the default parameter values as the *default (first order) radio model*. Hereafter, unless we state otherwise, we assume the default radio model, and hence we have $r_i = 5\ mJ$ and $\tau_{ij} = (5 + 0.1 \cdot d_{i,j}^2)\ mJ$ for each sensor node i and j.

Each sensor has a data gathering energy budget of $1J$. All of our experiments were done in Matlab running on a standard desktop PC.

First, we evaluate the practicality of our algorithm. We see in Fig. 2 that the RSM–MLDA algorithm takes a considerable number of iterations to reach the optimal continuous lifetime when the network size n becomes big, while in Fig. 3 we see that it gets within a large percentage of the optimal lifetime with few iterations

[7] The first order radio model considers only path loss and ignores more complex wireless channel effects such as fading, absorption, multi–path loss, etc.

[8] For Bluetooth radios with EDR operating at 2.7V, with peak current $< 65mA$, and 3Mbps (eg. BlueCore4 [6], BR-C46AR [5]) we have $E_{Tx} = E_{Rx} = 2.7V \times 65mA \div 3Mbps \approx 56nJ/bit$. For low–power Bluetooth radios (eg. BlueCore7 [6]) we have $E_{Tx} = E_{Rx} \approx 20mW \div 2Mbps \approx 10nJ/bit$. For the nFR2401 radio [26] we have $E_{Rx} = 3V \times 19mA \div 1Mbps \approx 56nJ/bit$, $E_{Tx} = 3V \times 13(8.8)mA \div 1Mbps \approx 38(26)nJ/bit$ at 0 (-20) dBm. For the CC2420 radio [38] we have $E_{Rx} \approx 162nJ/bit$, $E_{Tx} \approx 143(70)nJ/bit$ at 0 (-25) dBm.

for all the network sizes n considered in our experiments; e.g. for networks of size $n = 100$, it takes 2300 and 4200 iterations on average to reach 90% and 95% of the optimal lifetime, respectively. In practice, for medium–to–large WSNs, it may be acceptable to terminate the RSM–MLDA as soon as it reaches a user defined threshold for the approximation ratio, e.g. 95%. In such scenarios, we believe the RSM–MLDA would be very useful and effective.

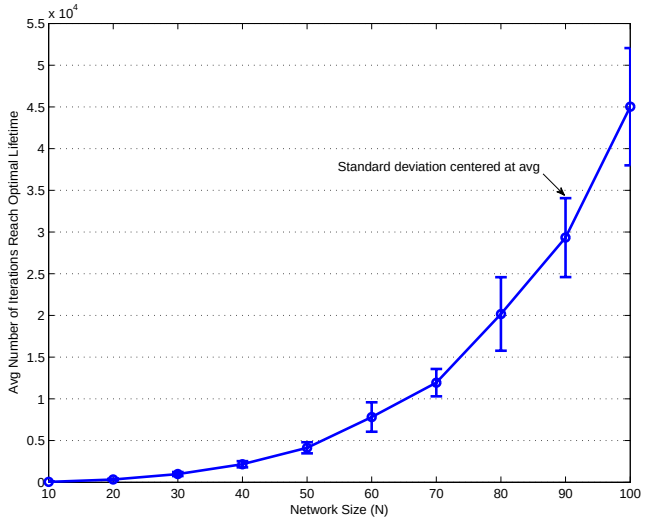

Figure 2: Average number of iterations for the RSM–MLDA algorithm to reach the optimal continuous lifetime.

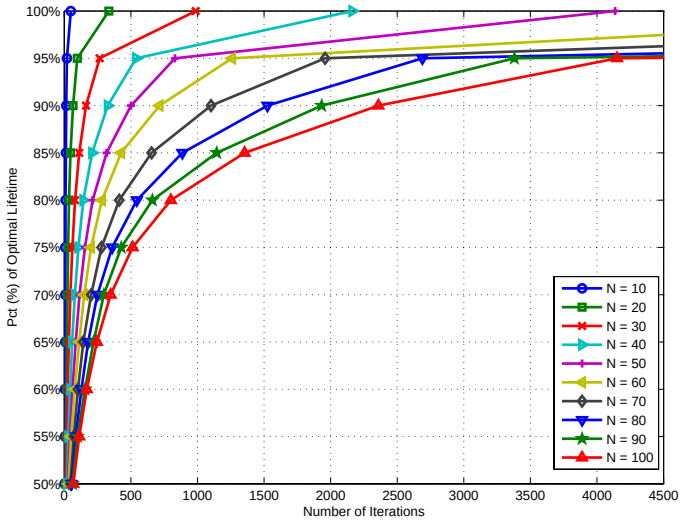

Figure 3: Average approximation ratio vs. number of iterations of the RSM–MLDA algorithm for different network sizes n.

We provided GetUB, an algorithm for computing a running upper bound of the system lifetime during the execution of the RSM–MLDA algorithm. Such an upper bound can be used to terminate the algorithm early, as soon as the estimated approximation ratio exceeds a user defined threshold. We evaluate the performance of the RSM–MLDA algorithm with early termination as well as the quality of the estimated upper bound on the lifetime, for two threshold values:

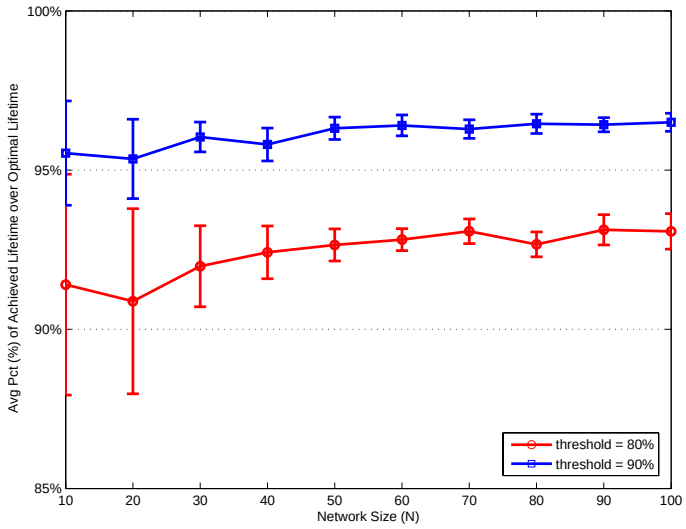

Figure 4: Average approximation ratio achieved by the RSM–MLDA algorithm with early termination.

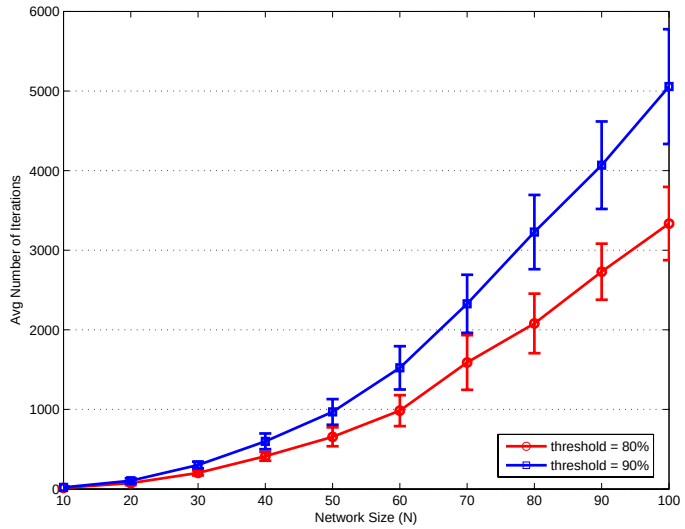

Figure 5: Average number for iterations for the RSM–MLDA algorithm with early termination.

80% and 90%, see Figs. 4 and 5. We note that the approximation ratio achieved at early termination can be higher than the chosen threshold — as shown in Fig. 4, on average the approximation ratio achieved is 92% and 95% for threshold 80% and 90%, respectively. Fig. 5 shows the average number of iterations for the estimated approximation ratio to exceed the chosen threshold — RSM–MLDA exceeds that threshold within a small number of iterations.

Next, we compare our RSM–MLDA algorithm with an iterative algorithm proposed by Stanford and Tongngam [35], which we call the GK algorithm, as it is based on the Garg–Konemann approach [12]. We run both algorithms on all the generated networks of various sizes n. For the GK algorithm, we use $\epsilon = 0.1$. We allow each algorithm to iterate at most $5n$ times.

Fig. 6 shows the approximation ratio achieved by the GK and RSM–MLDA algorithms – each point in the figure corresponds to a network instance. The

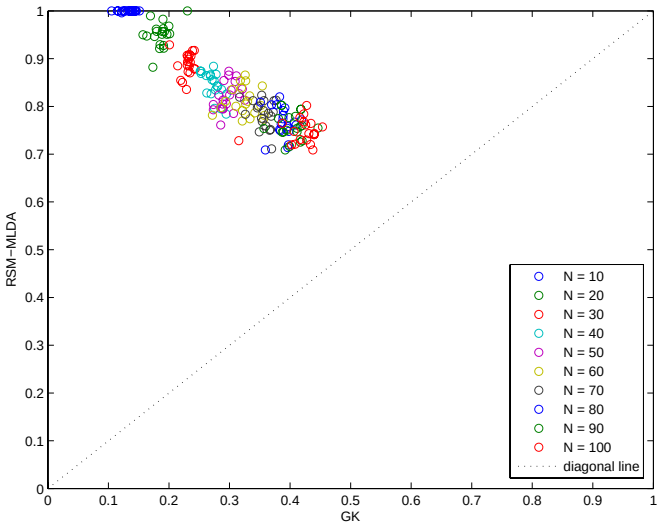

Figure 6: Approximation ratio achieved by the GK and RSM–MLDA algorithms for each network instance.

approximation ratio is calculated as the lifetime achieved at termination over the optimal lifetime for that network. We see that RSM–MLDA outperforms GK in every network instance. Moreover, the minimum number of aggregation trees generated by GK at termination is 3, 69, 146, 200, 250, 300, 350, 400, 450, and 500 for networks of size $n = 10, 20, \ldots, 90, 100$, respectively. The number of aggregation trees used by the GK algorithm seems to be close to the number of iterations it performs. Given the limited computational resources of the sensors, and the large number of iterations for GK to converge for small ϵ, distributing the relevant information about a large number of aggregation trees to the sensors is challenging and limits the usability of the GK algorithm. For example, as we see in Fig. 6, since GK achieved an average approximation ratio of 30% for networks of 50 nodes by 250 iterations, one needs to use 250 trees to get 30% of the optimal lifetime using the GK algorithm vs. at most 50 trees to get over 75% of the optimal lifetime using our RSM–MLDA algorithm.

5.1. On the impact of the radio model and size of the sensing field.

Next, we consider two additional factors, the radio model parameter values and the size of the sensing field, that affect the transmit and receive energies, and how they may impact the RSM–MLDA algorithm. Note that RSM–MLDA always converges to an optimal solution irrespective of the exact values of the transmit and receive energies.

We conduct experiments with the first order radio model with parameter values that approximately match the energy consumption profile of the commonly used Chipcon CC2420 radio [38]. We estimate E_{Tx} and E_{amp} using the transmit current consumption at -25dBm and 0dBm to correspond to transmission distances of $0m$ and $100m$ (its outdoors transmission range) respectively. In particular, $E_{Tx} = 2.1V \times 8.5mA \div 250Kbps = 69.7nJ/bit$, $E_{Rx} = 2.1V \times 19.7mA \div 250Kbps = 161.6nJ/bit$, and $E_{amp} = 2.1V \times (17.4mA - 8.5mA) \div 250Kbps \div (100m)^2 = 7.3pJ/bit/m^2$. We refer to this model as the *CC2420–based (first order) radio*

model. The main differences between the CC2420–based and the default radio models are that (a) in the CC2420–based radio model the packet receive energy is larger than the packet transmit energy for distances less than $\approx 112.2m$, and (b) the CC2420–based radio model has larger packet transmit energy than the default radio model for distances less than $\approx 14.57m$.

We also consider sensing fields of various sizes. To this end, we scale our default 50×50 sensing field by a factor of $\lambda = 2$, 4, and 10, so that a sensor located at (x, y) in the default field is located at $(\lambda x, \lambda y)$ in the scaled field. Increasing the scaling factor increases the transmit energy without affecting the receive energy of any sensor.

For each instance of the problem we compute the following five metrics. The number of iterations to converge within a factor of 99.9% of the optimal lifetime T_{opt} (which is set to the upper bound on the optimal lifetime computed by our GETUB algorithm), and the difference $\text{gap}_{99.9\%}$ between the lifetime $T_{99.9\%}$ achieved and T_{opt}. For each aggregation tree $\mathcal{T}$, we compute its height $h(\mathcal{T})$, the in–degree $\Delta_{\text{base}}(\mathcal{T})$ of the base station, and the maximum in–degree $\Delta_{\text{sensors}}(\mathcal{T})$ of the sensors. For each problem instance, we compute the average $\overline{h}, \overline{\Delta}_{\text{base}}, \overline{\Delta}_{\text{sensors}}$ of these three metrics weighted by the number of rounds each aggregation tree is used, for example

$$\overline{\Delta}_{\text{sensors}} = \frac{\sum_{\mathcal{T}} rounds[\mathcal{T}] \cdot \Delta_{\text{sensors}}(\mathcal{T})}{\sum_{\mathcal{T}} rounds[\mathcal{T}]}. \tag{14}$$

We report the average of these metrics across all the problem instances for the two radio models and the four scaling factors in Table 1.

Even though, the average number of iterations is still increasing exponentially with the number of sensor nodes, as the receive energy becomes significantly larger than the transmit energy or vice versa, fewer iterations are needed to converge to a near optimal solution. Furthermore, despite the early termination, on average, RSM–MLDA finds solutions with lifetimes that are within at most 9 rounds of the optimal lifetime. The behavior of the RSM–MLDA algorithm is consistent and robust under different sensing field sizes and radio model parameters. Moreover, as expected, the radio model parameters and sensing field size can have a significant impact on the (optimal) system lifetime.

We also observe that as the transmit energy becomes significantly larger than the receive energy, the aggregation trees use edges of smaller length (leading to an increase of tree height), the maximum in-degree of the sensor nodes slowly increases, while the in–degree of the base station decreases (as fewer sensors tend to transmit directly to the base station).

We note that the average in–degree of the sensors in the aggregation trees the RSM–MLDA algorithm finds is ≤ 4.03 for networks with up to $n \leq 30$ nodes. Therefore, even though any number of input packets is allowed to be aggregated into a single output packet, only a small number of input packets need to be aggregated into a single output packet in the aggregation trees our algorithm computes. Having sensors with small in–degree is beneficial for constructing small latency collision–free schedules of all the packet transmissions prescribed by each aggregation tree [14, 27, 37].

Table 1: Impact of different sensing field sizes and radio model parameter values on RSM–MLDA and its computed solutions, with an early termination threshold of 99.9%. Number of iterations ($\#\text{iters}_{99.9\%}$), sensor and base station in–degrees ($\overline{\Delta}_{\text{sensors}}$ and $\overline{\Delta}_{\text{base}}$), aggregation tree height ($\overline{h}$), lifetime achieved ($T_{99.9\%}$), and difference from optimal lifetime ($\text{gap}_{99.9\%} = T_{\text{opt}} - T_{99.9\%}$), averaged over 20 network instances, each with n nodes placed in the sensing field scaled by a factor of λ.

n	λ	$\#\text{iters}_{99.9\%}$	$\overline{\Delta}_{\text{sensors}}$	$\overline{\Delta}_{\text{base}}$	$\overline{h}$	$T_{99.9\%}$	$\text{gap}_{99.9\%}$
CC2420–based radio model							
10	1	25.35	0.36	8.61	1.35	11443.60	1.76
10	2	30.55	0.73	8.00	1.71	8181.81	1.86
10	4	37.20	1.31	4.94	3.77	4693.89	1.70
10	10	43.45	2.15	1.99	5.64	2431.77	0.24
20	1	156.20	0.42	18.23	1.36	11498.88	7.04
20	2	237.90	0.86	16.93	1.66	8180.60	6.00
20	4	233.85	1.67	9.20	5.83	4856.13	3.77
20	10	366.40	2.84	2.17	10.47	3215.05	0.45
30	1	534.20	0.63	27.77	1.50	11525.10	8.74
30	2	801.95	1.11	25.75	1.72	8282.97	6.93
30	4	678.10	1.88	14.66	6.43	4966.30	4.29
30	10	1396.40	2.79	2.60	14.47	3551.45	1.70
Default radio model							
10	1	44.75	1.71	2.39	5.89	8597.31	1.28
10	2	48.15	2.42	1.83	5.22	4830.74	5.47
10	4	33.50	2.48	1.77	5.12	1691.03	0.08
10	10	31.40	2.29	1.79	5.33	304.31	0.32
20	1	297.20	2.27	3.67	9.78	9526.24	4.72
20	2	339.00	3.40	2.18	9.11	6676.06	2.32
20	4	207.65	3.19	2.08	9.16	2795.26	0.48
20	10	175.15	2.89	1.97	9.41	538.49	0.33
30	1	874.15	2.30	5.46	12.35	9844.28	4.83
30	2	1374.90	3.96	2.47	12.14	7433.17	2.06
30	4	771.55	4.03	2.20	10.50	3195.92	0.59
30	10	529.95	3.06	1.92	12.37	621.56	0.36

6. Related Work

In wireless sensor networks, the notion of in-network fusion or aggregation was first introduced by Intanagonwiwat et al [16] to opportunistically eliminate duplicates in the context of directed diffusion. Intanagonwiwat et al [15] extend this work by constructing a routing tree where paths are shared as much as possible to increase the possibilities of duplicate elimination. The potential benefits of in-network duplicate elimination have been studied from a theoretical perspective in [21]. Data gathering with in-network data aggregation is a useful paradigm for increasing the system's lifetime, since it leads to significant energy savings [15, 23, 24].

Kalpakis et al [18] tackle the MLDA problem by reducing it to a directed network design problem: maximize T such that all sensor–base station directed cuts have capacity of at least T while the total energy consumed by sending/receiving messages along the edges incident to each sensor does not exceed each sensor's available energy. They first use linear programming to find tight approximate solution T to the network design problem, and then derive a collection of aggregation trees together with the number of rounds each tree is to be used in order to achieve lifetime T. We propose a new efficient combinatorial algorithm that finds near optimal solution directly.

Dasgupta et al [9] consider an equivalent formulation of the MLDA problem and provide approximate solutions by selecting aggregation trees from a pool of randomly constructed aggregation trees $\mathcal{T}$, and then using linear programming to determine the number of rounds each tree in the pool should be used in order to maximize the lifetime. The number of the aggregation trees included in the pool limits the approximation ratio achieved by them. Our algorithm considers, implicitly, all the up to n^{n-2} aggregation trees for a WSN with n nodes. Dasgupta et al [9] also consider a partial in–network aggregation where some fixed number K of input packets can be aggregated into a single output packet, and they find that as K decreases from ∞ to 2 the optimal system lifetime decreases by about 50%.

Stanford and Tongngam [35] give an $(1-\epsilon)^2$–approximation iterative algorithm, based on the Garg–Konemann approach [12] for solving packing linear programs, where the linear program is the same LP formulation of the MLDA problem considered here. Their approximation algorithm inherits the properties of Garg–Konemann approach. A markable limitation of the Garg–Konemann approach is the slow convergence for small ϵ due to the large number of iterations. A large number of iterations typically leads to a drastic slowdown of this algorithm. Moreover, it does not guarantee a small number of aggregation trees – a property that is critical in WSNs since the overhead of distributing information about these trees to the sensors must be limited. In addition, the integral solution obtained by rounding down the achieved continuous solution may have substantially reduced lifetime.

Xue at al [39] approach the MLDA problem as a maximum concurrent multicommodity flow problem in digraphs. Their tree–based $(1-\epsilon)^2$–approximation algorithm is also based on the Garg–Konemann approach [12] therefore it inherits

its performance characteristics as well. Furthermore, it turns out, as Stanford and Tongngam [35] point out, that the model used by Xue at al [39] does not allow the same manner of data aggregation we consider here.

There has been a considerable amount of work on the maximum lifetime data gathering problem without in–network aggregation. Chang and Tassiulas [8] propose a shortest path routing algorithm using link costs that reflect both the communication energy consumption rates and the residual energy levels at the two end nodes. Sankar and Liu [33] provide a distributed algorithm for maximizing the lifetime for data gathering in wireless sensor networks without in–network aggregation. They adapt a technique developed for distributed maximum flow computations by Awerbuch and Leighton [2]. Yu, Prasanna, and Krishnamachari [40] consider minimizing the total additive energy for data gathering without aggregation in WSNs subject to latency constraints. They present off–line and distributed on–line algorithms for scheduling packet transmissions on a data gathering tree by exploring energy-latency tradeoffs. Luo and Hubaux [22] combine mobility strategies with routing to increase the lifetime of WSNs in the the Chang–Tassiulas model [8]. They find that when the base station is static, the sensors close to it deplete their batteries early, leading to smaller lifetimes. They show that a mobile base station offers better load balancing, reducing substantially the load of the most loaded sensors. Pan et al [28] present methods for the placement of base stations in WSNs to achieve maximum lifetime for data gathering without in–network aggregation.

Finally, let us comment on the work by Garg and Konemann [12]. Garg and Konemann [12] describe an important approach to obtain simple iterative algorithms with provable approximation ratios for the maximum multicommodity flow, packing linear programs, maximum concurrent flow, and minimum cost multicommodity flow problems. These algorithms utilize a parameter ϵ, and at each iteration compute a variable of least relative cost whose value is incremented by a certain small amount. For the maximum multicommodity flow and the packing LP problems they achieve an $(1-\epsilon)^2$ approximation ratio with at most $km\lceil\log_{1+\epsilon} m/\epsilon\rceil$ iterations, where m is the number of edges and constraints and k is the number of commodities or 1, respectively. For the maximum concurrent flow and minimum cost multicommodity flow problems they achieve an $(1-\epsilon)^3$ approximation ratio with at most $3k \lg k\lceil\log_{1+\epsilon}(\frac{m}{1-\epsilon})/\epsilon$ iterations, where m is the number of edges and k is the number of commodities. Note that, in all four problems, the number of iterations needed to achieve an approximation ratio of $\alpha \leq 1$ is $O([(1-\sqrt[K]{\alpha})\log(2-\sqrt[K]{\alpha})]^{-1})$ for fixed problem size, with $K = 2$ or 3. Further, when applied to solve packing linear programs, it finds solutions whose support can be as high as the number of iterations. Nevertheless, the Garg–Konemann approach is attractive since it takes considerable time to solve such problems using traditional linear programming algorithms as the size of the problem increases, and this beautiful approach has been applied to various problems in WSNs, e.g. to the maximum lifetime routing problem [7] and to the problem of maximum data collection in store–and-extract networks [32].

7. Conclusions

In this paper we consider the problem of Maximum Lifetime Data Gathering with in–network Aggregation (MLDA) in wireless sensor networks. This is a NP–hard problem and the size of the solution, which consists of a collection of aggregation trees together with the number of rounds each such tree should be used, is desired to be small, given the limited computation and communication resources of sensor nodes. We describe a simple and efficient combinatorial iterative algorithm for finding an optimal continuous solution that has up to $n-1$ aggregation trees and achieves lifetime T_o, which depends on the network topology and initial energy available at the sensors. We obtain an α–approximate optimal integral solution by simply rounding down the optimal continuous solution, where $\alpha = (T_o - n + 1)/T_o$. Since in practice $T_o \gg n$, $\alpha \approx 1$. We get asymptotically optimal integral solutions to the MLDA problem whenever the optimal continuous solution is $\omega(n)$. We also present sensitivity analysis of the solution that can be used to bound the change of data gathering energy budget of each sensor, and demonstrate the practicability of the proposed algorithm via extensive experimental results.

As part of future work, we plan to extend our approach to obtain distributed and adaptive algorithms for the MLDA problem with provable performance guarantees.

Acknowledgments.

We thank the referees for their thoughtful comments and suggestions.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubermanian, and E. Cayirici. A survey of sensor networks. *IEEE Communications Magazine*, 40:102–114, 2002.

[2] B. Awerbuch and F. T. Leighton. Improved approximation algorithms for the multi–commodity flow problem and local competitive routing in dynamic networks. In *Proc. ACM Symposium on Theory of Computing*, pp. 487–496, 1994.

[3] R. Bar-Yehuda, K. Bendel, A. Freund, and D. Rawitz. Local ratio: A unified framework for approximation algorithms. *ACM Computing Surveys*, 36(4):422–463, 2004.

[4] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[5] BlueRadios, *BR-C46AR Datasheet*. http://www.blueradios.com

[6] CSR, *BlueCore4 and BlueCore7 Datasheets*. http://www.csr.com

[7] J. H. Chang and L. Tassiulas. Fast approximate algorithm for maximum lifetime routing in wireless ad-hoc networks. In *Networking 2000*, volume LNCS 1815, pages 702–713, 2000.

[8] J.-H. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(4):609–619, 2004.

[9] K. Dasgupta, K. Kalpakis, and P. Namjoshi. Improving the lifetime of sensor networks via intelligent selection of data aggregation trees. In *Proc. of the Communication Networks and Distributed Systems Modeling and Simulation Conference*, Orlando, Florida, January 2003.

[10] A. Deligiannakis, Y. Kotidis and N. Roussopoulos. Processing Approximate Aggregate Queries in Wireless Sensor Networks. *Information Systems*, 31 (8), pp. 770–792, 2006.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proc. of the 33rd Hawaii International Conference on System Sciences* 2000.

[12] N. Garg and J. Konemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *IEEE Symposium on Foundations of Computer Science*, pages 300–309, 1998.

[13] M. Garofalakis. Processing Massive Data Streams. Invited full-day seminar at the 2008 VLDB Database School, Cairo University, Cairo, Egypt, March 2008. http://www.softnet.tuc.gr/ minos/Talks/vldbSchool08.pdf

[14] S.C.H. Huang, P. Wan, C. T. Vu, Y. Li, and F. Yao. Nearly Constant Approximation for Data Aggregation Scheduling in Wireless Sensor Networks. In *Proc. IEEE INFOCOM*, pp. 366-372, 2007.

[15] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *In Proceedings of International Conference on Distributed Computing Systems (ICDCS)*, Vienna, Austria, July 2002.

[16] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. of 6th ACM/IEEE Mobicom Conference*, 2000.

[17] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Mobile Networking for Smart Dust. In *Proc. of 5th ACM/IEEE Mobicom Conference*, 1999.

[18] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, 42(6):697–716, 2003.

[19] H. Karl and A. Willig. *Protocols and architectures for wireless sensor networks.* John Wiley and Sons, 2005.

[20] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms.* Springer Verlag, 1991.

[21] B. Krishanamachari, D. Estrin, and S. Wicker. The impact of data aggregation in wireless sensor networks. In *In International Workshop of Distributed Event Based Systems (DEBS)*, Vienna, Austria, July 2002.

[22] J. Luo and J.-P. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proc. of INFOCOM*, pages 1735–1746, 2005.

[23] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *Proc. of 5th Symposium on Operating Systems Design and Implementation*, pages 131–, 2002.

[24] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks. In *Proc. of 4th IEEE Workshop on Mobile Computing and Systems Applications*, 2002.

[25] R. Min, M. Bhardwaj, S. Cho, A. Sinha, E. Shih, A. Wang, and A. Chandrakasan. Low-power wireless sensor networks. In *VLSI Design*, 2001.

[26] Nordic Semiconductor, *nRF2401 Datasheet*. http://www.nordicsemi.com

[27] K. Oikonomou and I. Stavrakakis. An Adaptive Time-Spread Multiple-Access Policy for Wireless Sensor Networks. *EURASIP Journal on Wireless Communications and Networking*, Vol. 1, pp. 24–32, 2007.

[28] J. Pan, L. Cai, Y. T. Hou, Y. Shi, and S. X. Shen. Optimal base-station locations in two-tiered wireless sensor networks. *IEEE Transactions on Mobile Computing*, 4(5):458–473, 2005.

[29] J. Polastre, R. Szewczyk and D. Culler. Telos: Enabling Ultra-Low Power Wireless Research. In *Proc. of IPSN*, pp. 364–369, 2005.

[30] V. Puttagunta and K. Kalpakis. Accuracy vs. Lifetime: Linear Sketches for Aggregate Queries in Sensor Networks. *Algorithmica*, 49 (4), pp. 357–385, 2007.

[31] J. Rabaey, J. Ammer, J. da Silva Jr, and D. Patel. PicoRadio: Ad-hoc wireless networking of ubiquitous low-energy sensor/monitor nodes. In *Proc. of the IEEE Computer Society Annual Workshop on VLSI*, 2000.

[32] N. Sadagopan and B. Krishnamachari. Maximizing data extraction in energy-limited sensor networks. In *Proc. of INFOCOM*, 2004.

[33] A. Sankar and Z. Liu. Maximum lifetime routing in wireless ad-hoc networks. In *Proc. of INFOCOM*, 2004.

[34] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri. Medians and Beyond: New Aggregation Techniques for Sensor Networks. In *Proc. ACM SenSys*, 2004.

[35] J. Stanford and S. Tongngam. Approximation algorithm for maximum lifetime in wireless sensor networks with data aggregation. In *Proc. of the Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'06)*, pages 273–277, Washington, DC, USA, 2006.

[36] R. Tarjan. Finding optimum branchings. *Networks*, 7(1):25–35, 1977.

[37] Y.C. Tay, K. Jamieson and H. Balakrishnan, Collision-minimizing CSMA and its applications to wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6), pp. 1048–1057, 2004.

[38] Texas Instruments, *Chipcon CC2420 Datasheet*. http://www.ti.com

[39] Y. Xue, Y. Cui, and K. Nahrstedt. Maximizing lifetime for data aggregation in wireless sensor networks. *Mobile Networks and Applications*, 10(6):853–864, 2005.

[40] Y. Yu, V. K. Prasanna, and B. Krishnamachari. Energy minimization for real-time data gathering in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 5(11):3087–3096, 2006.

Appendix A. Linear Programming Primer

This appendix provides an overview of some concepts in linear programming used in this paper. Further details can be referred in a text such as [4, 20].

Consider a linear program in standard form

$$\left\{ \begin{array}{ll} \min \mathbf{c}^T \mathbf{x} & \text{such that} \\ \\ \mathbf{A} \cdot \mathbf{x} = \mathbf{b} & \\ \\ \mathbf{x} \geq \mathbf{0} & \end{array} \right\} \tag{15}$$

where $\mathbf{A} \in \mathbb{R}^{n \times m}$, $\mathbf{c}, \mathbf{x} \in \mathbb{R}^m$, $\mathbf{b} \in \mathbb{R}^n$, and $n \leq m$. The linear program above defines a convex polyhedron $\mathcal{P} = \{\mathbf{x} \ : \ \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$. For convenience, and without loss of generality, suppose that the constraint matrix $\mathbf{A}$ is of full–rank n and that $\mathbf{b} \geq \mathbf{0}$. The case where $\mathbf{A}$ has rank less than n leads to degeneracies requiring special handling, see [4]. We further assume, w.l.o.g., that the polyhedron $\mathcal{P}$ is bounded and non–empty, i.e. the linear program has a bounded optimal solution.

Let $\mathcal{B}$ be a sequence (ordered set) of n column indexes in $\{1, \ldots, m\}$. Let $\mathbf{A}_\mathcal{B}$ be the $n \times n$ sub–matrix of $\mathbf{A}$ whose ith column is $\mathbf{A}_{\mathcal{B}(i)}$. A sequence $\mathcal{B}$ is called a base if $\mathbf{A}_\mathcal{B}$ is of full-rank (invertible). It is called a feasible basis if $\mathbf{A}_\mathcal{B}^{-1}\mathbf{b} \geq \mathbf{0}$. Since $\mathbf{A}$ is of full–rank and the linear program is feasible, a feasible basis always exists. A variable x_i (column $\mathbf{A}_i$) with index in $\mathcal{B}$ is called a *basic variable (basic column)*, otherwise it is called a *non–basic variable (non–basic column)*.

Construct a feasible solution $\mathbf{x}$ corresponding to a feasible base $\mathcal{B}$ by taking $\mathbf{x}_\mathcal{B} = \mathbf{A}_\mathcal{B}^{-1}\mathbf{b}$ and $\mathbf{x}_{\overline{\mathcal{B}}} = \mathbf{0}$. Such a solution is called a *basic feasible solution* (bfs). There is a bijection between basic feasible solutions and vertices (extreme points) of the polytope defined by $\mathbf{A}$. Furthermore, an optimal solution always occurs at one of its vertices.

Associate with each constraint a *shadow price* (or *dual variable*). The shadow prices $\boldsymbol{\pi} \in \mathbb{R}^n$ corresponding to a base $\mathcal{B}$ is given by

$$\boldsymbol{\pi}^T = \mathbf{c}_\mathcal{B}\mathbf{A}_\mathcal{B}^{-1}. \tag{16}$$

The *relative cost* $\bar{c}_j$ of each non–basic column $\mathbf{A}_j$ is given by

$$\bar{c}_j = c_j - \boldsymbol{\pi}^T \mathbf{A}_j \tag{17}$$

The Simplex method, discovered by Dantzig, systematically explores the set of basic feasible solutions, starting from an initial bfs, until an optimal bfs is found. The process of moving from a bfs to an adjacent bfs is called *pivoting*. In pivoting, we exchange a basic column with a non–basic column, without increasing the cost of the best feasible solution so far.

We describe next a variant of the Simplex method, the Revised Simplex Method (RSM) with the lexico–min rule. An arbitrary non–basic column $\mathbf{A}_j$ enters the current base $\mathcal{B}$ if its relative cost $\bar{c}_j < 0$. If all non–basic columns have relative cost ≥ 0, then the current bfs is optimal and Simplex terminates. Otherwise, a basic column to exit the current base $\mathcal{B}$ needs to be selected. There are multiple approaches to do so. We describe the lexico–min approach for choosing the basic column to exit the current basis $\mathcal{B}$, since it guarantees termination in a finite number of pivoting steps [4]. Let $\mathbf{a}_i$ denote the ith row of the matrix $\mathbf{A}_\mathcal{B}$. Let l be the index of the lexicographically smallest row $[x_i, \mathbf{a}_i]/u_i$ with $u_i > 0$,

$$l = \arg \text{lexico–min} \left\{ \frac{[x_i, \mathbf{a}_i]}{u_i} \;\; : \;\; u_i > 0 \right\}, \tag{18}$$

where $\mathbf{u} = \mathbf{A}_\mathcal{B}^{-1} \mathbf{A}_j$ and $\mathbf{x} = \mathbf{A}_\mathcal{B}^{-1} \mathbf{b}$. Column $\mathbf{A}_j$ enters the base $\mathcal{B}$ replacing column $\mathbf{A}_{\mathcal{B}(l)}$, i.e. $\mathcal{B}(l) \longleftarrow j$. An index l always exists, since otherwise $u_i \leq 0$ for all i and the problem is unbounded.

Extensive computational experience since the discovery of the Simplex method demonstrated that in practice it is an efficient algorithm. The Revised Simplex method offers computational advantages for linear programs with sparse constraint matrices. Moreover, observe that RSM allows us to solve linear programs with exponentially many variables by performing few pivots in practice, provided that we can either find, in polynomial–time, a non–basic column with negative relative cost or show that no such column exists.