Local Approximation Algorithms for Scheduling Problems in Sensor Networks

Patrik Floréen, Petteri Kaski, Topi Musto, and Jukka Suomela

Helsinki Institute for Information Technology HIIT
Department of Computer Science, University of Helsinki
P.O. Box 68, FI-00014 University of Helsinki, Finland
{firstname.lastname}@cs.helsinki.fi

Abstract. We study fractional scheduling problems in sensor networks, in particular, sleep scheduling (generalisation of fractional domatic partition) and activity scheduling (generalisation of fractional graph colouring). The problems are hard to solve in general even in a centralised setting; however, we show that there are practically relevant families of graphs where these problems admit a *local* distributed approximation algorithm; in a local algorithm each node utilises information from its constant-size neighbourhood only. Our algorithm does not need the spatial coordinates of the nodes; it suffices that a subset of nodes is designated as *markers* during network deployment. Our algorithm can be applied in any marked graph satisfying certain bounds on the marker density; if the bounds are met, guaranteed near-optimal solutions can be found in constant time, space and communication per node. We also show that auxiliary information is necessary—no local algorithm can achieve a satisfactory approximation guarantee on unmarked graphs.

1 Introduction

The scalability of distributed algorithms presents a basic hurdle to the envisaged large-scale implementations of sensor networking, in particular due to the bounded resources of the individual network nodes. Simply put, if we want to operate arbitrarily large sensor networks, we cannot apply network control algorithms where the communication or computation per node increases with increasing network size. Indeed, if each individual network node is powered by a battery with bounded capacity, there is always a threshold size for the network beyond which the energy consumption for network control exceeds the battery capacity of a node.

1.1 Local Algorithms

In this work we study *local algorithms* [1], where each node must operate solely based on information that was available at system startup within a constant-size neighbourhood of the node. A local algorithm provides an extreme form of scalability: assuming constant-size input per node, the communication, space and time complexity of a local algorithm is constant per node. Thus, a local algorithm scales to an arbitrarily large (or even infinite) resource-constrained network. We detail the model of computation in Sect. 3.

A local algorithm is clearly the ideal choice for sensor networks, but even from a theoretical perspective it is not immediate whether such algorithms can exist for practical computational problems arising in network control. This work shows that various NP-hard *scheduling problems* admit *deterministic, local approximation algorithms* provided that the network meets certain assumptions on its structure.

1.2 Scheduling Problems

We study two basic scheduling problems pertinent to sensor networks: *sleep scheduling*, a fractional packing problem, and *(conflict-free) activity scheduling*, a fractional covering problem. Both problems can be formulated as a linear program (LP), but the number of variables in the LP can be exponential in the size of the network; both problems are NP-hard to solve even in a centralised setting.

To ease the exposition, we present the scheduling problems first in a centralised setting; the requirements for a proper distributed solution are detailed together with the local computational model in Sect. 3. We require a few preliminaries to present the definitions. All graphs are undirected. We model the network topology by a *communication graph* $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$, where each node $v \in V_\mathcal{G}$ corresponds to a sensor device and each edge $\{u, v\} \in E_\mathcal{G}$ indicates that u and v can directly communicate with each other. We denote by $d_\mathcal{G}(u, v)$ the shortest-path distance (hop count) between nodes $u, v \in V_\mathcal{G}$ in $\mathcal{G}$ and extend the notation to subsets $U \subseteq V_\mathcal{G}$ by $d_\mathcal{G}(U, v) = \min_{u \in U} d_\mathcal{G}(u, v)$. For $v \in V_\mathcal{G}$ and $r \geq 0$, we define the closed ball of radius r centred at v in $\mathcal{G}$ by $B_\mathcal{G}(v, r) = \{u \in V_\mathcal{G} : d_\mathcal{G}(u, v) \leq r\}$.

Problem 1 (Sleep Scheduling)**.** The input to the problem consists of (i) the communication graph $\mathcal{G}$; (ii) a subgraph $\mathcal{R}$ of $\mathcal{G}$ called the *redundancy graph*; and (iii) a battery capacity $b(v) \geq 0$ for each node $v \in V_\mathcal{R}$. Each edge $\{u, v\} \in E_\mathcal{R}$ indicates that the nodes u and v are pairwise redundant; each node may sleep only if at least one of its neighbours in $\mathcal{R}$ is awake. The valid sets of awake nodes are precisely the dominating sets of $\mathcal{R}$. For a dominating set D, we define $D(v) = 1$ if $v \in D$ and $D(v) = 0$ if $v \notin D$. Denoting by $x(D)$ the length of the time period associated with the dominating set D, the task in the problem is to maximise the total length $\sum_D x(D)$ subject to $\sum_D D(v)x(D) \leq b(v)$ and $x(D) \geq 0$, where v ranges over $V_\mathcal{R}$ and D ranges over all the dominating sets of $\mathcal{R}$. To simplify subsequent analysis, we assume that the values $b(v)$ are chosen from a fixed, finite set of nonnegative rational numbers (say, the capacities of the standard batteries in stock); in particular, a constant number of bits per node suffice to encode the input, as it is enough to identify which battery is installed in the device instead of encoding an arbitrary battery capacity.

The sleep scheduling problem is a generalisation of fractional domatic partition. It captures the problem of maximising the lifetime of a battery-powered sensor network by letting each node sleep occasionally, subject to coverage constraints under a pairwise redundancy model [2–6].

Problem 2 (ACTIVITY SCHEDULING)**.** The input to the problem consists of (i) the communication graph $\mathcal{G}$; (ii) a subgraph $\mathcal{C}$ of $\mathcal{G}$ called the *conflict graph*; and (iii) an activity requirement $a(v) \geq 0$ for each node $v \in V_{\mathcal{C}}$. Each edge $\{u, v\} \in E_{\mathcal{C}}$ indicates that the nodes u and v are mutually conflicting; at most one of the two nodes may be active at any given time. The valid sets of active nodes are precisely the independent sets of $\mathcal{C}$. For an independent set I, we define $I(v) = 1$ if $v \in I$ and $I(v) = 0$ if $v \notin I$. Denoting by $x(I)$ the length of the time period associated with the independent set I, the task in the problem is to minimise the total length $\sum_I x(I)$ subject to $\sum_I I(v)x(I) \geq a(v)$ and $x(I) \geq 0$, where v ranges over $V_{\mathcal{C}}$ and I ranges over all the independent sets in $\mathcal{C}$. Again, we assume that the values $a(v)$ are chosen from a fixed, finite set of nonnegative rational numbers.

The activity scheduling problem is a generalisation of fractional graph colouring. It captures the problem of minimising the total duration of radio transmissions subject to pairwise interference constraints [7].

1.3 Assumptions on Network Structure

Unfortunately, both scheduling problems just presented are hard to solve exactly or approximately [8–10], even in a centralised setting. To arrive at problem instances that can be solved approximately in a distributed manner, one must impose constraints on the structure of the communication graph $\mathcal{G}$. Furthermore, to obtain a local approximation algorithm, there is a need to break symmetry between the nodes to obtain any satisfactory approximation guarantee, as we will make apparent in Lemma 1.

An embedding of $\mathcal{G}$ in a low-dimensional ambient space could be used as a remedy for both aforementioned difficulties. Indeed, graphs with geometric constraints (for example, unit-disk graphs) in many cases admit efficient approximation algorithms at least in the centralised case, and the spatial coordinates of the nodes break symmetry. However, equipping the nodes with self-positioning capabilities (such as GPS) may not be feasible in large-scale installations, and neither is it practical to inform each node about its physical location during network deployment.

Rather than rely on a geometric embedding, in this work we investigate a minimalistic solution to break symmetry—one *marker* bit of information per node. Furthermore, we use purely combinatorial constraints on the marked $\mathcal{G}$ to arrive at a locally tractable setting. We characterise the admissible distributions of the nodes with the marker bit set (the *markers*) in $\mathcal{G}$ by nonnegative integer parameters ℓ_1, ℓ_μ, and μ, where $\ell_1 < \ell_\mu$.

Definition 1. *A* $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph *is a pair* $(\mathcal{G}, M)$*, where* $\mathcal{G}$ *is a graph and* $M \subseteq V_\mathcal{G}$ *is a set of* markers *such that, for all* $v \in V_\mathcal{G}$*, (i) the degree of* v *in* $\mathcal{G}$ *is at most* Δ*; (ii)* $d_\mathcal{G}(M, v) \leq \ell_1$*; and (iii)* $|M \cap B_\mathcal{G}(v, \ell_\mu)| \leq \mu$*.*

In other words, every node has at most Δ neighbours, there is at least one marker within ℓ_1 hops from any node, and there are at most μ markers within ℓ_μ hops from any node. Examples of marked graphs appear in Sect. 5.

1.4 Contributions

As the main technical contribution, we prove the following theorems in Sect. 4. In both theorems, the marking constraint applies to the communication graph $\mathcal{G}$ only.

Theorem 1. *There is a local* $(1+\epsilon)$*-approximation algorithm for sleep scheduling in* $(\Delta, \ell_1, \ell_\mu, \mu)$*-marked graphs for any* $\epsilon > 4\Delta/\lfloor(\ell_\mu - \ell_1)/\mu\rfloor$*.*

Theorem 2. *There is a local* $1/(1-\epsilon)$*-approximation algorithm for activity scheduling in* $(\Delta, \ell_1, \ell_\mu, \mu)$*-marked graphs for any* $\epsilon > 4/\lfloor(\ell_\mu - \ell_1)/\mu\rfloor$*.*

To contrast these positive results, we also demonstrate that the algorithms in Theorems 1 and 2 make near-optimal use of the marking information. In particular, we present a family of marked graphs where our algorithm for sleep scheduling (respectively, activity scheduling) achieves the approximation ratio $1 + 9\epsilon$ (respectively, $1/(1 - 9\epsilon)$) while no local approximation algorithm can achieve the approximation ratio $1 + \epsilon$ (respectively, $1/(1 - \epsilon)$).

2 Earlier Work

2.1 Local Algorithms

Previous work on local algorithms mainly focuses on combinatorial problems such as independent set and graph colouring. Linial [11] shows that any distributed algorithm requires $\Omega(\log^* n)$ communication rounds to find a maximal independent set or a 3-colouring of a ring with n nodes, implying in particular that no local algorithm exists for these tasks. Naor and Stockmeyer [1] present positive results for *locally checkable labelling* problems; for example, it is possible to 2-colour the nodes of a graph using a local algorithm so that each node has at least one neighbour with a different colour, provided that all nodes have odd degree.

Closer to the present work is the work of Kuhn et al. [12], who present local approximation algorithms for fractional covering and packing problems. However, in their work the size of the LP is polynomial in the size of the network, while the size of the LPs that arise from sleep scheduling and activity scheduling can be exponential.

2.2 Shifting Strategy

The present work can be seen as an extension of a classical design paradigm for geometric approximation algorithms—the *shifting strategy* [13]—into a local, coordinate-free, and nongeometric setting. In a typical application of the shifting strategy [13–17], one uses a grid to partition the (low-dimensional) geometric space into small cells. Each cell defines a subproblem; for example, the subgraph induced by the nodes which are located within or near the cell. Each subproblem is solved optimally, and the solutions are combined to form a feasible global solution. A number of possible locations for the grid are evaluated and the best one is chosen as the solution.

Unfortunately, there are two basic obstacles hindering the application of the shifting strategy in large-scale distributed systems. First, it has been argued that the shifting strategy is "inherently central" [18]; in particular, the final step involves determining which of the candidate solutions is the best one. Second, a straightforward application of the shifting strategy requires that we know how the input is embedded in an ambient space.

Our previous work [3] partially overcomes the aforementioned obstacles in a specific problem: sleep scheduling. To avoid the need for centralised control, we note that the scheduling problem is of fractional nature: one can take two valid schedules and interleave them in order to obtain another valid schedule. To avoid a global coordinate system, we place markers in the underlying communication graph; the constraints for the locations of the markers are geometric, but the algorithm does not use the locations. The present work generalises this previous work in the following aspects: (i) The algorithm is extended to fractional covering problems in addition to fractional packing problems. (ii) No geometric constraints are required; in particular, $\mathcal{G}$ need not have an embedding in a low-dimensional space. (iii) There is no lower bound for the distance between a pair of markers.

3 Preliminaries

3.1 Model of Computation

We assume a communication graph $\mathcal{G}$ where each node has degree bounded by a constant Δ. Each node in $\mathcal{G}$ executes the same distributed deterministic algorithm.

An algorithm is *local* if there exist a constant L ("the local horizon") such that for every problem instance, each node $v \in V_{\mathcal{G}}$ makes its decisions based on information in the nodes $B_{\mathcal{G}}(v, L)$ only. In the sleep scheduling problem, this information consists of the identifiers of the nodes $B_{\mathcal{G}}(v, L)$, the subset of markers $M \cap B_{\mathcal{G}}(v, L)$, the subgraph of the communication graph $\mathcal{G}$ induced by $B_{\mathcal{G}}(v, L)$, the subgraph of the redundancy graph $\mathcal{R}$ induced by $B_{\mathcal{G}}(v, L) \cap V_{\mathcal{R}}$, and the battery capacity $b(u)$ for each node $u \in B_{\mathcal{G}}(v, L) \cap V_{\mathcal{R}}$. The definition is analogous for activity scheduling.

We assume that the node identifiers form an ordered set. An algorithm cannot access the absolute value of an identifier, but only the ordering of the identifiers. In particular, the identifiers need only be unique in $B_{\mathcal{G}}(v, L)$ for each $v \in V_{\mathcal{G}}$. Therefore our computational model is slightly weaker in comparison with the model used by Linial [11]. (To motivate this weakening, see Naor and Stockmeyer [1, Theorem 3.3].)

With these definitions, the number of bits communicated, stored and processed by any node during the execution of a local algorithm is bounded by a constant. Thus also the time complexity is constant per node.

In the scheduling problems, a node does not report any output; instead, a node executes the schedule it has locally computed by controlling its sleeping (respectively, activity). To enable execution of the schedule, it is assumed that (i) each node has access to a clock and (ii) the clocks are (locally) synchronised.

A local $(1+\epsilon)$-approximation algorithm for sleep scheduling guarantees that the nodes that are awake form a dominating set of the redundancy graph at any point in time during the first $q/(1+\epsilon)$ time units, where q is the length of an optimal solution. A local $(1+\epsilon)$-approximation algorithm for activity scheduling guarantees that the nodes that are active form an independent set of the conflict graph at any point in time and each node completes its activity within $(1+\epsilon)q$ time units, where q is the length of an optimal solution.

3.2 Limitations

The chosen local model of computation is very restrictive. For example, Linial [11] shows that (with respect to a strictly stronger model of computation) no local algorithm can properly 3-colour rings. Thus, it is not surprising that scheduling problems in rings are not approximable by local algorithms.

Lemma 1. *No local algorithm on an unmarked graph has an approximation ratio better than* 3 *for the sleep scheduling problem or any finite approximation ratio for the activity scheduling problem.*

Proof. Consider an arbitrary local algorithm with local horizon $L \in \mathbb{N}$. Let the communication graph $\mathcal{G}$ be a ring of $6L$ nodes, that is, $V_{\mathcal{G}} = \{0, 1, \ldots, 6L-1\}$ and $E_{\mathcal{G}} = \{\{0,1\}, \{1,2\}, \ldots, \{6L-2, 6L-1\}, \{6L-1, 0\}\}$. The node identifiers are ordered by $0 < 1 < \ldots < 6L-1$. For sleep scheduling, let $\mathcal{R} = \mathcal{G}$ and $b(v) = 1$ for each $v \in V_{\mathcal{R}}$; for activity scheduling, let $\mathcal{C} = \mathcal{G}$ and $a(v) = 1$ for each $v \in V_{\mathcal{C}}$. Now the local neighbourhood $B_{\mathcal{G}}(v, L)$ has the same structure for each node in $U = \{L, L+1, \ldots, 5L-1\}$. At any point in time, all these nodes have to make the same decision.

In the case of sleep scheduling we can obtain a schedule of length 3 by choosing the congruence classes modulo 3 as the dominating sets and by assigning 1 time unit to each. However, if each node in U makes the same decision in the local algorithm, then all of them have to be awake at any point in time; otherwise, e.g., the node $L+1$ would not be dominated. Thus if the local algorithm produces a feasible sleep schedule, the length of the schedule is at most 1, implying that the local algorithm cannot guarantee an approximation ratio better than 3.

In the case of activity scheduling we can obtain a schedule of length 2 by choosing the congruence classes modulo 2 as the independent sets and by assigning 1 time unit to each. However, if each node in U makes the same decision in the local algorithm, then none of them can be active at any point in time; otherwise a conflicting pair of nodes $\{L, L+1\}$ would be active simultaneously. Thus, the nodes in U can never complete their activities, implying that the local algorithm cannot guarantee any finite approximation ratio. $\square$

Therefore one has to incorporate auxiliary information to the communication graph to obtain satisfactory approximation guarantees for scheduling.

4 Local Approximability of Scheduling

We assume that the marked communication graph $(\mathcal{G}, M)$ is a $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph with $k = \lfloor(\ell_\mu - \ell_1)/\mu\rfloor > 0$. Intuitively, a large k is desirable for a good approximation and a small ℓ_μ is desirable in limiting the computational effort.

4.1 Finding Cells

Each node $v \in V_\mathcal{G}$ applies the following algorithm:

```
FIND-CELLS
1  $d \leftarrow d_\mathcal{G}(M, v)$
2  **for** $i \leftarrow 0$ **to** $k\mu - 1$
3        **do** $m(v,i) \leftarrow \min(M \cap B_\mathcal{G}(v, d+i))$
```

First, the node finds the distance d to its nearest marker; note that $d \le \ell_1$. Then, for each *configuration* $i = 0, 1, \ldots, k\mu - 1$, the node finds the smallest marker within the distance $d+i$; here we use the total order on the identifiers.

We define the *cell* of the marker m in configuration i by $C(m,i) = \{v \in V_\mathcal{G} : m(v,i) = m\}$. We say that a node $v \in V_\mathcal{G}$ is a *boundary node* in configuration i if v has a neighbour u in $\mathcal{G}$ such that $m(v,i) \ne m(u,i)$. The following lemma captures a key property of the configurations (cf. Floréen et al. [3, Lemma 4]).

Lemma 2. *For any $v \in V_\mathcal{G}$, there are at most 4μ configurations i such that v is a boundary node in i.*

To prove Lemma 2, we start with two technical lemmata.

Lemma 3. *For any node $v \in V_\mathcal{G}$, there are at most μ different values of $m(v,i)$.*

Proof. On line 3 in FIND-CELLS, it holds that $d + i < \ell_1 + k\mu \le \ell_\mu$, which implies $m(v,i) \in M \cap B_\mathcal{G}(v, \ell_\mu)$ for each configuration i. By definition, $|M \cap B_\mathcal{G}(v, \ell_\mu)| \le \mu$. $\square$

Lemma 4. *For any node $v \in V_\mathcal{G}$, there are at most $\mu - 1$ configurations i such that $m(v,i) \neq m(v,i+1)$.*

Proof. Consider an arbitrary $v \in V_\mathcal{G}$. By Lemma 3, it suffices to show that each distinct value of $m(v,i)$ corresponds to a single interval of configurations i; once $m(v,i)$ changes its value from m_1 to $m_2 \neq m_1$, it never changes back to m_1.

Assume that $m(v,i_1) = m(v,i_2) = m$ for arbitrary m and $i_1 \leq i_2$. Then m is a member of the ball $B_\mathcal{G}(v, d_\mathcal{G}(M,v) + i_1)$, and m is the smallest marker in the larger ball $B_\mathcal{G}(v, d_\mathcal{G}(M,v) + i_2)$. Thus, for any $i_1 \leq i \leq i_2$, it holds that m is the smallest smallest marker in $B_\mathcal{G}(v, d_\mathcal{G}(M,v) + i)$, implying $m(v,i) = m$ for all $i_1 \leq i \leq i_2$. □

Proof of Lemma 2. Consider an arbitrary $v \in V_\mathcal{G}$. By Lemma 4, we can divide the list of configurations $(0, 1, \ldots, k\mu - 1)$ into at most μ intervals, such that $m(v,i)$ is constant within each interval. We now prove that v can be a boundary node at most 4 times on each interval.

This clearly holds for intervals of length at most 4. Next, consider an interval from i_1 to i_2 with $i_2 \geq i_1 + 4$ such that $m(v,i) = m$ for each configuration i with $i_1 \leq i \leq i_2$.

Let $u \in V_\mathcal{G}$ be any neighbour of v. Because $d_\mathcal{G}(u,v) = 1$, it holds that $|d_\mathcal{G}(M,v) - d_\mathcal{G}(M,u)| \leq 1$. By construction, $m = m(v,i_1)$ is a marker in $B_\mathcal{G}(v, d_\mathcal{G}(M,v) + i_1) \subseteq B_\mathcal{G}(u, d_\mathcal{G}(M,v) + i_1 + 1) \subseteq B_\mathcal{G}(u, d_\mathcal{G}(M,u) + i_1 + 2)$, and $m = m(v,i_2)$ is the smallest marker in $B_\mathcal{G}(v, d_\mathcal{G}(M,v) + i_2) \supseteq B_\mathcal{G}(u, d_\mathcal{G}(M,v) + i_2 - 1) \supseteq B_\mathcal{G}(u, d_\mathcal{G}(M,u) + i_2 - 2)$.

Therefore m is a marker in $B_\mathcal{G}(u, d_\mathcal{G}(M,u) + i_1 + 2)$ and furthermore m is the smallest marker in $B_\mathcal{G}(u, d_\mathcal{G}(M,u) + i_2 - 2) \supseteq B_\mathcal{G}(u, d_\mathcal{G}(M,u) + i_1 + 2)$. We obtain $m(u,i) = m = m(v,i)$ for $i_1 + 2 \leq i \leq i_2 - 2$.

As this holds for any neighbour u, the node v cannot be a boundary node in the configurations $i_1 + 2 \leq i \leq i_2 - 2$. There are at most 4 configurations in the ends of the interval such that v may be a boundary node. □

The algorithm Find-Cells is local. In the following sections, we use the cells and Lemma 2 to obtain local algorithms for the scheduling problems.

4.2 Sleep Scheduling

Let $\bar{C}(m,i) = \{v \in V_\mathcal{G} : d_\mathcal{G}(C(m,i), v) \leq 1\}$. For each marker m and configuration i, solve the LP

$$\begin{array}{lll} \text{maximise} & \sum_K x_{m,i}(K) & \\ \text{subject to} & \sum_K K(v) x_{m,i}(K) \leq b(v) & \text{for all } v, \\ & x_{m,i}(K) \geq 0 & \text{for all } K, \end{array} \tag{1}$$

where v ranges over all nodes in $\bar{C}(m,i) \cap V_\mathcal{R}$, and K ranges over all subsets $K \subseteq \bar{C}(m,i) \cap V_\mathcal{R}$ such that K dominates $C(m,i) \cap V_\mathcal{R}$ in $\mathcal{R}$. Note that the boundary nodes may participate in domination, but they need not be dominated.

The LP has constant size and depends on the local information only. Let $q_{m,i} = \sum_K x_{m,i}(K)$ be the total length of the solution.

Based on the computed solutions, each node controls its sleeping as follows. We use the synchronised clocks to proceed in cycles of length δ time units for some δ. Each cycle is further divided into $k\mu$ steps of length $\delta/(k\mu)$. We label the steps within each cycle by $0, 1, \dots, k\mu - 1$. The behaviour of each node at step i is controlled as follows by the local solutions $x_{m,i}$ associated with the configuration i.

First, consider a non-boundary node $v \in V_{\mathcal{R}}$. The node constructs a schedule based on the solution $x_{m,i}$ where $m = m(v,i)$. All nodes in $\bar{C}(m,i) \cap V_{\mathcal{R}}$ consider the sets K with a nonzero $x_{m,i}(K)$ in the same order $K_1, K_2, \dots$ (say, the lexicographic order). Let $t_j = \delta x_{m,i}(K_j)/(k\mu q_{m,i})$. First, if $v \in K_1$, the node is awake for t_1 time units; otherwise it is asleep for t_1 time units. Then, if $v \in K_2$, the node is awake for t_2 time units, and so on. This way we have scaled down the entire schedule x_m into one time step of length $\delta/(k\mu)$.

Second, consider a boundary node $v \in V_{\mathcal{R}}$. As above, we construct a schedule based on $x_{m(v,i),i}$. Additionally we construct a schedule based on $x_{m(u,i),i}$ for every u such that $m(u,i) \neq m(v,i)$ and $\{u,v\} \in E_{\mathcal{G}}$. We take the union of these schedules: at any point in time, the node v is awake if it is awake according to at least one of the schedules.

In each configuration i, each node is a member of $C(m,i)$ for some m, and the local solution $x_{m,i}$ guarantees that $C(m,i) \cap V_{\mathcal{R}}$ is dominated at every point in time. Thus, this procedure is correct in the sense that $V_{\mathcal{R}}$ is dominated at every point in time, as long as no node runs out of battery.

Proof of Theorem 1. Let x be an optimal sleep schedule in the centralised setting; let $q = \sum_D x(D)$. The solution x can be used to construct a feasible solution to each local LP (1): for each dominating set D, add $x(D)$ units to $x_{m,i}(K)$ where $K = D \cap \bar{C}(m,i)$; as D dominates $V_{\mathcal{R}}$, it follows that K dominates $C(m,i) \cap V_{\mathcal{R}}$. Thus, $q_{m,i} \geq q$, as $x_{m,i}$ is an optimal solution to (1).

Consider an arbitrary node $v \in V_{\mathcal{R}}$. During each step i when v is not a boundary node, it is awake for at most $\delta b(v)/(k\mu q_{m,i}) \leq \delta b(v)/(k\mu q)$ time units. When v is a boundary node, this is increased by at most a factor $\Delta + 1$ because there are at most Δ neighbours and thus at most Δ different neighbouring cells. By Lemma 2, the node v is a boundary node in at most 4μ configurations out of $k\mu$. Thus, v is awake for at most $(k\mu + 4\Delta\mu)\delta b(v)/(k\mu q) = (1 + 4\Delta/k)\delta b(v)/q$ time units during an entire cycle of length δ. During $\lfloor q/(\delta(1 + 4\Delta/k)) \rfloor$ cycles, v is awake for at most $b(v)$ time units. Thus, the battery of v lasts at least $\lfloor q/(\delta(1 + 4\Delta/k)) \rfloor \delta \geq q/(1 + 4\Delta/k) - \delta$ time units. By choosing a small δ, we can obtain an approximation ratio $1 + \epsilon$ for any $\epsilon > 4\Delta/k$.

To choose a small enough δ, we need some information on q. If $q > 0$ then each node has to have at least one neighbour u with $b(u) > 0$; by letting all nodes be awake as long as their batteries last, we obtain a trivial constant lower bound for q from the smallest nonzero element of the finite set of possible $b(v)$; we use this bound to choose δ. The obtained δ (as well as any other value) is trivially valid also in the case $q = 0$. $\square$

4.3 Activity Scheduling

Let $C^\circ(m,i)$ be the set of nodes $v \in C(m,i)$ such that v is not a boundary node in configuration i. For each marker m and configuration i, solve the LP

$$\begin{array}{lll} \text{minimise} & \sum_K x_{m,i}(K) & \\ \text{subject to} & \sum_K K(v) x_{m,i}(K) \geq a(v) & \text{for all } v, \\ & x_{m,i}(K) \geq 0 & \text{for all } K, \end{array} \tag{2}$$

where v ranges over all nodes in $C^\circ(m,i) \cap V_\mathcal{C}$ and K ranges over all subsets $K \subseteq C^\circ(m,i) \cap V_\mathcal{C}$ such that K is an independent set in $V_\mathcal{C}$. Note that the boundary nodes are not considered. The LP has constant size and depends on the local information only. Let $q_{m,i} = \sum_K x_{m,i}(K)$ be the total length of the solution.

As in Sect. 4.2, we proceed in cycles of length δ and steps of length $\delta/(k\mu)$. Also the translation of local solutions into schedules is the same for nonboundary nodes. However, the boundary nodes are inactive.

Proof of Theorem 2. Let x be an optimal activity schedule in the centralised setting; let $q = \sum_D x(D)$. The solution x can be used to construct a feasible solution to each local LP (2): for each independent set I, add $x(I)$ units to $x_{m,i}(K)$ where $K = I \cap C^\circ(m,i)$; as I is an independent set in $\mathcal{C}$, so is K. Thus, $q_{m,i} \leq q$, as $x_{m,i}$ is an optimal solution to (2).

Consider an arbitrary node $v \in V_\mathcal{C}$. During each step i when v is not a boundary node, it is active for at least $\delta a(v)/(k\mu q_{m,i}) \geq \delta a(v)/(k\mu q)$ time units. By Lemma 2, the node v is a boundary node in at most 4μ configurations out of $k\mu$. Thus, v is active for at least $(1-4/k)\delta a(v)/q$ time units during an entire cycle of length δ. During $\lceil q/(\delta(1-4/k))\rceil$ cycles, v is active for at least $a(v)$ time units. Thus, the node can complete its activities in $\lceil q/(\delta(1-4/k))\rceil\delta \leq q/(1-4/k)+\delta$ time units. By choosing a small δ, we can obtain an approximation ratio $1/(1-\epsilon)$ for any $\epsilon > 4/k$. Again an appropriate δ can be chosen by bounding q using the information on the possible values of $a(v)$. □

4.4 A Lower Bound for Local Approximability

We proceed to show that the value ϵ in the approximation guarantees of Theorems 1 and 2 cannot be improved by a constant factor larger than 9.

Select integers $N \geq 100$ and $\mu \geq 10$. Consider an arbitrary local algorithm with local horizon $L \in \mathbb{N}$. Construct the communication graph $\mathcal{G}$ by forming a ring of $n = (6N+1)6L$ nodes, that is, $V_\mathcal{G} = \{0,1,\ldots,n-1\}$, $E_\mathcal{G} = \{\{0,1\}, \{1,2\},\ldots,\{n-2,n-1\},\{n-1,0\}\}$. The identifiers are ordered by $0 < 1 < \ldots < n-1$. Place the markers at the nodes v where $v \equiv 0 \pmod{6N+1}$. The construction is a $(2, 3N, \lceil\mu(3N+1/2)\rceil - 1, \mu)$-marked graph.

For sleep scheduling, let $\mathcal{R} = \mathcal{G}$ and $b(v) = 1$ for each $v \in V_\mathcal{R}$; for activity scheduling, let $\mathcal{C} = \mathcal{G}$ and $a(v) = 1$ for each $v \in V_\mathcal{C}$. Consider the nodes $U = \{L, L+1, \ldots, n-L-1\}$. For each $j \in \{0,1,\ldots,6N\}$, the local neighbourhoods of the nodes $v \in U$ with $v \equiv j \pmod{6N+1}$ are identical; thus, each of these nodes must make the same decision at any point in time.

In the case of sleep scheduling, there exists a schedule of length 3. However, the local algorithm cannot achieve an optimal solution. Consider a chain of $6N+1$ nodes in U. If only $2N$ nodes are awake at a given point in time, then only $6N$ nodes are awake in a chain of $18N+3$ nodes, as each subchain of length $6N+1$ behaves identically. However, $6N$ nodes cannot dominate a chain of $18N+1$ nodes; thus, there is at least one node in the chain which cannot be dominated. Therefore at least $2N+1$ nodes have to be awake, and the total lifetime of the nodes in a subchain of $6N+1$ nodes is thus at most $(6N+1)/(2N+1) = 3/(1+\epsilon)$ for $\epsilon = 2/(6N+1) > 0.33/N$. Our local algorithm achieves the approximation guarantee $1+\epsilon$ for any $\epsilon > 8/k$ where $k \geq 3N - 1/\mu - 3N/\mu - 1/2 \geq 2.694N$. That is, we can achieve $\epsilon = 9 \times 0.33/N$.

In the case of activity scheduling, there exists a schedule of length 2. In the local algorithm, for each chain of $6N+1$ nodes there can be at most $3N$ nodes active simultaneously, implying that the length of the schedule obtained by the arbitrary local algorithm is at least $(6N+1)/(3N) = 2/(1-\epsilon)$ for $\epsilon = 1/(6N+1) > 0.165/N$. Our local algorithm achieves $\epsilon = 9 \times 0.165/N$.

In conclusion, we have presented an infinite family of parameters $(\Delta, \ell_1, \ell_\mu, \mu)$ such that the ϵ in our approximation guarantees for both sleep scheduling and activity scheduling is within factor 9 of the best possible that any deterministic local approximation algorithm can achieve.

In this lower bound, we focused on the case $\ell_1 \approx \ell_\mu/\mu \to \infty$. The following lemma shows that the case $\ell_1 \ll \ell_\mu/\mu$ is trivial to local algorithms.

Lemma 5. *If $\ell_\mu \geq (\mu+1)(\ell_1+1/2)$, then the size of each connected component of a $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph is bounded by a constant.*

Proof. Let $(\mathcal{G}, M)$ be a $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph with $\ell_\mu \geq (\mu+1)(\ell_1+1/2)$. To reach a contradiction, assume that there exist nodes $v_0, v_\mu \in V_\mathcal{G}$ such that $d_\mathcal{G}(v_0, v_\mu) = \mu(2\ell_1+1)$. Then there exist nodes $v_1, v_2, \ldots, v_{\mu-1} \in V_\mathcal{G}$ such that $d_\mathcal{G}(v_i, v_{i+1}) = 2\ell_1+1$ and a node $u \in V_\mathcal{G}$ such that $d_\mathcal{G}(v_i, u) \leq \lceil \mu(\ell_1+1/2) \rceil$. For $i = 0, 1, \ldots, \mu$, let $m_i \in M$ be a marker having the minimum distance to v_i in $\mathcal{G}$; the nodes $m_0, m_1, \ldots, m_\mu$ are distinct. Furthermore, it holds that $d_\mathcal{G}(m_i, u) \leq d_\mathcal{G}(m_i, v_i) + d_\mathcal{G}(v_i, u) \leq \ell_1 + \lceil \mu(\ell_1+1/2) \rceil \leq \ell_1 + \mu(\ell_1+1/2) + 1/2 = (\mu+1)(\ell_1+1/2) \leq \ell_\mu$. This implies that we have $\mu+1$ markers in $B_\mathcal{G}(u, \ell_\mu)$, which is a contradiction with the assumption that $(\mathcal{G}, M)$ be a $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph. Thus, the diameter of each connected component of $\mathcal{G}$ is less than $\mu(2\ell_1+1)$, and each connected component consists of at most $1 + \Delta^{\mu(2\ell_1+1)}$ nodes. □

5 Deploying a Marked Network

Any local algorithm for scheduling requires some auxiliary information, marking or otherwise, to break symmetry (Lemma 1). Thus, to apply a local algorithm, one must incorporate this information into the network when the network is deployed. In particular, a *practically feasible* way to deploy the network is required. We conclude this paper by developing a series of examples that illustrate how one might go about and deploy a marked network in a physical area so that Definition 1 is met.

An intuitive picture to keep in mind in what follows is a graduate student walking about an area where a network is to be deployed with two (heavy) bags of sensor devices. One bag contains devices with the marker bit set, and the other bag devices with the bit reset.

We start with a purely combinatorial setup and proceed in steps towards more realistic scenarios.

5.1 Grids

Consider an infinite 2-dimensional grid graph $\mathcal{G}$, where $V_\mathcal{G} = \mathbb{Z}^2$ and $E_\mathcal{G} = \{\{(x_1, x_2), (y_1, y_2)\} : |x_1 - y_1| + |x_2 - y_2| = 1\}$. Choose an integer $N > 1$. Deploy the markers at nodes $M_\mathcal{G} = \{(Ni, Nj) : i + j \text{ odd}\}$. The constructed $(\mathcal{G}, M)$ is a $(4, N, 2N - 1, 4)$-marked graph. Generalisation to higher dimensions is immediate.

For large N we obtain an approximation ratio $1 + \epsilon$ where $\epsilon \approx 64/N$ for sleep scheduling and $\epsilon \approx 16/N$ for activity scheduling. In this sense, our local approximation algorithm is a *local approximation scheme* for grid graphs: any approximation ratio above 1 can be achieved by deploying the markers in a sufficiently sparse manner. Furthermore, the rule for deploying the markers is arguably practically feasible from the perspective of a combinatorial entity traversing the grid.

5.2 Globally Grid-like Graphs

The communication topology in a real physical environment does not have a perfect grid structure. To arrive at a more versatile model, consider an infinite connected graph $\mathcal{H}$ where every node has degree at most $\Delta_\mathcal{H}$. We assume that the large-scale structure of $\mathcal{H}$ is similar to a 2-dimensional grid graph $\mathcal{G}$, but the small-scale structure of $\mathcal{H}$ can be arbitrary. In precise terms, we assume that the metric spaces $(V_\mathcal{G}, d_\mathcal{G})$ and $(V_\mathcal{H}, d_\mathcal{H})$ are *quasi-isometric*[1]; that is, we assume that there exist mappings $h \colon V_\mathcal{G} \to V_\mathcal{H}$, $g \colon V_\mathcal{H} \to V_\mathcal{G}$ and constants $C \geq 0$, $\lambda \geq 1$ such that $d_\mathcal{H}(h(x), h(y)) \leq \lambda d_\mathcal{G}(x, y) + C$, $d_\mathcal{G}(g(u), g(v)) \leq \lambda d_\mathcal{H}(u, v) + C$, $d_\mathcal{G}(g(h(x)), x) \leq C$, and $d_\mathcal{H}(h(g(v)), v) \leq C$ for all $x, y \in V_\mathcal{G}$ and $u, v \in V_\mathcal{H}$. Define the marking of $\mathcal{H}$ from a marking of $\mathcal{G}$ by $M_\mathcal{H} = h(M_\mathcal{G})$.

Lemma 6. *Any marked graph $(\mathcal{H}, M_\mathcal{H})$ that satisfies the above conditions is a $(\Delta_\mathcal{H}, \lfloor \lambda N + 2C \rfloor, \lfloor 2\lambda N - (2C + 1)/\lambda \rfloor, \lceil 2\lambda^2 \rceil^2)$-marked graph where N is the constant used to mark $\mathcal{G}$.*

[1] Ghys [19] attributes this definition of quasi-isometry to G. A. Margulis.

Proof. Let $v \in V_{\mathcal{H}}$. Let m be the marker closest to $g(v)$ in $\mathcal{G}$; $d_{\mathcal{G}}(m, g(v)) \leq N$. We have $h(m) \in M_{\mathcal{H}}$ and $d_{\mathcal{H}}(h(m), v) \leq d_{\mathcal{H}}(h(m), h(g(v))) + d_{\mathcal{H}}(h(g(v)), v) \leq \lfloor \lambda d_{\mathcal{G}}(m, g(v)) + C \rfloor + \lfloor C \rfloor \leq \lfloor \lambda N + C \rfloor + \lfloor C \rfloor \leq \lfloor \lambda N + 2C \rfloor$. We can choose $\ell_1 = \lfloor \lambda N + 2C \rfloor$.

Let $\ell_\mu = \lfloor 2\lambda N - (2C+1)/\lambda \rfloor$. Let $v \in V_{\mathcal{H}}$ and $m \in M_{\mathcal{G}}$ be such that $d_{\mathcal{H}}(v, h(m)) \leq \ell_\mu$. Then $d_{\mathcal{G}}(g(v), m) \leq d_{\mathcal{G}}(g(v), g(h(m))) + d_{\mathcal{G}}(g(h(m)), m) \leq \lambda \ell_\mu + 2C \leq 2\lambda^2 N - 1 \leq \lceil 2\lambda^2 \rceil N - 1$. For any $x \in V_{\mathcal{G}}$ and positive integer κ there are at most κ^2 markers in $B_{\mathcal{G}}(x, \kappa N - 1)$; thus there are at most $\lceil 2\lambda^2 \rceil^2$ markers in $B_{\mathcal{H}}(v, \ell_\mu)$. We can choose $\mu = \lceil 2\lambda^2 \rceil^2$. $\square$

Again we obtain an approximation scheme; any approximation ratio above 1 can be achieved by placing the markers in a sufficiently sparse manner in $\mathcal{G}$.

Intuitively, each element of $V_{\mathcal{G}}$ corresponds to a geometric area and $g(v) \in V_{\mathcal{G}}$ is the area where the device $v \in V_{\mathcal{H}}$ is located. The choice of $M_{\mathcal{H}}$ reflects the following rule for deploying the markers. First, some geometric areas $M_{\mathcal{G}}$ are selected based on the grid structure. Second, one marker is deployed in each of these geometric areas.

In the small scale, quasi-isometry allows arbitrary structure to occur; the small-scale structure of realistic communication graphs is irregular and unpredictable due to the complex nature of the physics of radio propagation. In the large scale, quasi-isometry requires that shortest-path distances in the communication graph reflect the distances in the ambient space; this is a reasonable assumption from a dense deployment of sensors in an area devoid of large-scale obstructions.

5.3 Serendipity of Locality

The defining property of a local algorithm is that the behaviour of each network node is uniquely determined by the radius-L neighbourhood of the node. In other words, all things being equal in the neighbourhood, the large-scale topology of the network has no effect in the operation of a network node. This is particularly useful from the perspective of network deployment—to fulfil the intended sensing objective, it suffices to deploy the sensor nodes in a manner that, from the perspective of mission-critical sensor nodes, *looks like* a benign topology, even if the actual topology is not.

In more concrete terms, let us assume that we have some two-dimensional area $\mathcal{A}$ of arbitrary shape that we want to monitor by a sensor network. Let us also assume that we have a method of network deployment that would produce a $(\Delta, \ell_1, \ell_\mu, \mu)$-marked graph if applied to the infinite two-dimensional plane; say, the method produces a globally grid-like marked graph $(\mathcal{H}, M_{\mathcal{H}})$.

Now, to deploy a network to monitor $\mathcal{A}$, all one has to do is to apply the deployment method to $\mathcal{A}$ plus its constant-width surroundings. More precisely, we deploy so that for any node v that is placed within $\mathcal{A}$, its $(L+1)$-hop neighbourhood is indistinguishable from its neighbourhood in $(\mathcal{H}, M_{\mathcal{H}})$. By locality it follows immediately that any node within $\mathcal{A}$ (or with a neighbour in $\mathcal{A}$) operates exactly as it would operate in the case of $(\mathcal{H}, M_{\mathcal{H}})$. For example, if the nodes execute the sleep scheduling algorithm, then full coverage for every node within $\mathcal{A}$ is guaranteed, with a lifetime at least as good as in the case of an infinite graph. Other nodes may fail in an arbitrary manner, but this does not affect the operation within $\mathcal{A}$; for example, in the case of activity scheduling, these nodes cannot conflict with the nodes within $\mathcal{A}$.

Again it can be argued that this deployment scheme is straightforward to implement in practice. The overhead (in the number of extra nodes that need to be deployed outside $\mathcal{A}$) depends on the shape and size of $\mathcal{A}$, but if the shape of $\mathcal{A}$ is not too irregular, the relative overhead approaches zero as the surface area of $\mathcal{A}$ increases.

Both the deployment of the markers and the deployment of the extra nodes in the surroundings of $\mathcal{A}$ can be seen as examples of a basic tradeoff in computational effort: minor (and, from the perspective of the deployer, computationally straightforward) extra effort invested in deployment pays off by enabling local approximation of fundamental scheduling problems.

Acknowledgements. This research was supported in part by the Academy of Finland, Grants 116547 and 117499, and by Helsinki Graduate School in Computer Science and Engineering (Hecse).

References

1. Naor, M., Stockmeyer, L.: What can be computed locally? SIAM Journal on Computing **24**(6) (1995) 1259–1277
2. Cardei, M., MacCallum, D., Cheng, M.X., Min, M., Jia, X., Li, D., Du, D.Z.: Wireless sensor networks with energy efficient organization. Journal of Interconnection Networks **3**(3–4) (2002) 213–229
3. Floréen, P., Kaski, P., Suomela, J.: A distributed approximation scheme for sleep scheduling in sensor networks. In: Proc. 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON, San Diego, CA, June 2007), Piscataway, NJ, IEEE (2007) 152–161
4. Koushanfar, F., Taft, N., Potkonjak, M.: Sleeping coordination for comprehensive sensing using isotonic regression and domatic partitions. In: Proc. 25th Conference on Computer Communications (INFOCOM, Barcelona, April 2006), Piscataway, NJ, IEEE (2006)
5. Moscibroda, T., Wattenhofer, R.: Maximizing the lifetime of dominating sets. In: Proc. 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS, Denver, CO, April 2005), Los Alamitos, CA, IEEE Computer Society Press (2005) 242b
6. Pemmaraju, S.V., Pirwani, I.A.: Energy conservation via domatic partitions. In: Proc. 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc, Florence, May 2006), New York, NY, ACM Press (2006) 143–154
7. Jain, K., Padhye, J., Padmanabhan, V.N., Qiu, L.: Impact of interference on multi-hop wireless network performance. Wireless Networks **11**(4) (2005) 471–487
8. Feige, U., Halldórsson, M.M., Kortsarz, G., Srinivasan, A.: Approximating the domatic number. SIAM Journal on Computing **32**(1) (2002) 172–195

9. Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. Journal of the ACM **41**(5) (1994) 960–981
10. Suomela, J.: Locality helps sleep scheduling. In: Working Notes of the Workshop on World-Sensor-Web: Mobile Device-Centric Sensory Networks and Applications (WSW, Boulder, CO, October 2006) 41–44; available at `http://www.sensorplanet.org/wsw2006/`
11. Linial, N.: Locality in distributed graph algorithms. SIAM Journal on Computing **21**(1) (1992) 193–201
12. Kuhn, F., Moscibroda, T., Wattenhofer, R.: The price of being near-sighted. In: Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA, Miami, FL, January 2006), New York, NY, ACM Press (2006) 980–989
13. Hochbaum, D.S., Maass, W.: Approximation schemes for covering and packing problems in image processing and VLSI. Journal of the ACM **32**(1) (1985) 130–136
14. Erlebach, T., Jansen, K., Seidel, E.: Polynomial-time approximation schemes for geometric intersection graphs. SIAM Journal on Computing **34**(6) (2005) 1302–1323
15. Hunt III, H.B., Marathe, M.V., Radhakrishnan, V., Ravi, S.S., Rosenkrantz, D.J., Stearns, R.E.: NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. Journal of Algorithms **26**(2) (1998) 238–274
16. Jiang, T., Wang, L.: An approximation scheme for some Steiner tree problems in the plane. In: Proc. 5th International Symposium on Algorithms and Computation (ISAAC, Beijing, August 1994), Berlin, Springer-Verlag (1994) 414–422
17. Suomela, J.: Approximability of identifying codes and locating-dominating codes. Information Processing Letters **103**(1) (2007) 28–33
18. Kuhn, F., Nieberg, T., Moscibroda, T., Wattenhofer, R.: Local approximation schemes for ad hoc and sensor networks. In: Proc. Joint Workshop on Foundations of Mobile Computing (DIALM-POMC, Cologne, September 2005), New York, NY, ACM Press (2005) 97–103
19. Ghys, É.: Les groupes hyperboliques. Astérisque **189–190** (1990) 203–238 [Séminaire Bourbaki, Vol. 1989/90, Exp. No. 722]