The complexity of data aggregation in static and dynamic wireless sensor networks

Quentin Bramas, Sébastien Tixeuil

The Complexity of Data Aggregation in Static and Dynamic Wireless Sensor Networks[☆]

Quentin Bramas[a,*], Sébastien Tixeuil[a,b]

[a] *Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 4 place Jussieu 75005 Paris*
[b] *Institut Universitaire de France*

Abstract

The key feature of wireless sensor networks is to aggregate data collected by individual sensors in an energy efficient manner. We consider two techniques to save energy. The first one is to avoid collisions due to simultaneous transmissions among neighboring nodes. Second, when a node receives data from multiple neighbors, it aggregates these with its own data. Then, one transmission is sufficient to transmit all consolidated data to another neighbor. If the overall delay has to be kept as low as possible, scheduling sensors to avoid collisions while aggregating data becomes challenging.

The contribution of this paper is threefold. First, we give tight bounds for the complexity of data aggregation in static networks. In more details, we show that the problem remains NP-complete when the graph is of degree at most three. As it is trivial to solve the problem in static graphs of degree at most two, our result implies that the problem is intrinsically difficult for any practical setting. Second, we investigate the complexity of the same problem in a dynamic network, that is, a network whose topology can evolve through time. In the case of dynamic networks, we show that the problem is NP-complete even in the case where the graph is of degree at most two (and it is trivial to solve the problem when the graph is of degree at most one). Third, we give the first lower and upper bounds for the minimum data aggregation time in a dynamic graph.

We observe that even in a well-connected evolving graphs, the optimal solution cannot be found by a distributed algorithm or by a centralized algorithm that does not know the future. Thus we finally give the first approximation algorithm (centralized that knows the future) whose approximation factor is $T(n-1)$ if there exists a bound T such that there is a journey (a path in a dynamic graph) for all pairs of nodes in every time interval $[t, t+T]$.

Keywords: Data aggregation, dynamic graphs, complexity

[☆]This work was performed within the Labex SMART supported by French state funds managed by the ANR within the Investissements d'Avenir programme under reference ANR-11-IDEX-0004-02.

[*]Corresponding author

Email addresses: quentin.bramas@lip6.fr (Quentin Bramas), sebastien.tixeuil@lip6.fr (Sébastien Tixeuil)

1. Introduction

The growing number of sensor nodes with sensing, computing and communication capabilities, was made possible by recent technological advances. This growth was encouraged by a variety of applications and contributes to the widespread interest in practical and theoretical aspects of wireless sensor networks. Sensor nodes should be inexpensive, small and sustainable in order to be easily deployed in a dangerous area, inside a human body or in vehicles, generally for monitoring applications. They generate data that have to be retrieved by an end-user or a base station. However, the environment and the lack of networking infrastructure does not permit direct transmission to the end user, but only transmissions between sensors that are close to one another. This raises various challenges, such as energy (sensors are battery powered) and delay efficiency (information is relevant for a short period of time only).

In a wireless sensor network (WSN), sensor nodes can communicate through a wireless ad hoc network. Then, there exists a communication link between two nodes if the Euclidean distance between them is smaller than their communication range. Since we assume all sensors to be identical, they have the same communication range and the communication graph can be modeled as a unit-disk graph[1] [1]. Sensor nodes typically generate data from their environment, such as temperature, number of vehicles on a road, or number of passenger in a bus. The end-user, called *sink node*, wants to extract this information. To do so, a node can send its data directly to the sink node if it is located within its communication range, or, if it is far from the sink, can use intermediate nodes to relay the data to the sink node.

In this paper we investigate the problem of retrieving data from a WSN whose data transmissions are constrained by two rules: avoiding collisions, and allowing data aggregation. In more details, the time is discretized and, at each time slot, a node is able to send its data to all of its neighbors (*i.e.*, all nodes within its communication range). Now, if two or more nodes send their data in the same time slot, their common neighbors do not receive any data, due to interference. Whenever a node successfully receives data, it aggregates the data with its own and stores the result as its new data. This process ensures energy-efficiency of the protocol. Indeed, n transmissions are sufficient to retrieve the data from n sensors to the sink node (compared to possibly $\Omega(n^2)$ without the capability to aggregate data). The problem of aggregating data from all nodes in the network in a minimum amount of time slots (delay-efficiency), assuming that a node sends its data at most once (energy-efficiency) is known as the *minimum data aggregation time* (MDAT) problem [2]. A solution to this problem consists of a transmission schedule, meeting the communication constraints, with minimum duration.

In this paper we also introduce the dynamic version of the MDAT problem, where individual sensors are now mobile entities. This could model cars evolving in a smart city, medical devices in a body area network, or mobile devices monitoring an area. A WSN whose topology evolves with time is modeled as a *dynamic unit-disk graph*, *i.e.*, a sequence of static unit-disk graphs. In this setting, the communication constraints hold at each time slot, and a solution of the MDAT problem consists of a transmission schedule with minimum duration.

[1] We suppose here that the area is a two dimensional plane, but our results naturally extend to greater dimensions

When sensor nodes have fixed positions, the maximum distance (in hops) from the sink node to any other node is a lower bound for the minimum data aggregation time [2]. Indeed, if no collision occurs, the data from the farthest node can be sent through a shortest path. Each avoided collision increases the duration of the schedule by one time slot. However, if we suppose the nodes are moving, avoiding collisions can intuitively have a much greater impact. Indeed, if a collision occurs and we delay the transmission of a node by one time slot, the node may not be able to transmit again (maybe the node remains isolated thereafter). In other terms, the existence of a journey (a path in a dynamic graph) from every node to the sink node is not sufficient to guarantee the existence of a collision-free schedule.

Related Work. The data aggregation problem we consider here was first studied by Anamalai *et. al.* [3]. The authors assume that a fixed number of channels is available for a transmission, and a collision occurs at a receiver whenever two of its neighbors transmit on the same channel at the same time. The authors propose an algorithm that constructs a collision free convergecast tree that can also be used for broadcasting. Then, Chen *et. al.* [2] present a well-defined model for the study of the MDAT problem in wireless sensor networks. The problem is equivalent to the convergecast problem defined by Anamalai *et al.* with a unique channel. In the same paper, Chen *et al.* prove that the problem is NP-complete, even in graphs of degree at most four (more precisely they restricted the problem to networks whose topology is a sub-graph of the grid, which cannot be considered directly as a wireless sensor network). They also gave a $(\Delta - 1)$-approximation algorithm (where Δ is the maximum node degree of the graph).

After the work of Chen *et. al.* [2], a variety of papers proposed centralized and distributed approximation algorithms using geometric aspect of the MDAT problem to improve the data aggregation delay. Yu et. al. [4] give a distributed algorithm with an upper bound at $24D + 6\Delta + 16$ (where D is the diameter, and Δ the maximum degree of the graph). Xu et. al., in [5] and Ren et. al. in [6] propose centralized algorithms with upper bounds at $16R + \Delta - 14$ and $16R + \Delta - 11$, respectively (where R is the radius of the graph). The best known bound is due to Nguyen *et al.* in [7], as they give a centralized algorithm that takes at most $12R + \Delta - 11$ time slots to aggregate all data.

Related problems such as in-network aggregation [8] focus on an orthogonal perspective. They assume that collisions are handled by the MAC layer, and aim to find routes that minimize the delay. So, those works actually differ significantly from the MDAT problem.

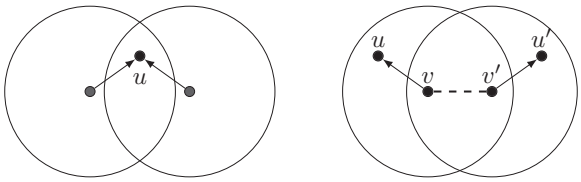
On the other hand, dynamic graphs have received a lot of interest recently and efforts have been done in order to standardize the underlying model [9, 10, 11]. Various problems have been studied in a distributed setting, such as designing foremost, fastest, and shortest broadcast algorithms [12, 13]. For each problem, sufficient and necessary conditions on the (dynamic) graph are given. The opposite problem of data dissemination (or flooding) has also received a lot of attention in random dynamic networks. Clementi *et al.* [14] gave an almost tight bound for any random dynamic graphs including edge-markovian evolging graphs and geometric graphs (where nodes follow a random walk or a random waypoint). A conclusion of their work is that increasing the dynamicity of the graph (for instance by increasing the speed of the nodes) implies faster data dissemination, even if the network is at each instant very sparse (due to a short communication range). Most related to our concern are two previous attempts that consider

collecting data in dynamic networks [15, 16], however they use a much more powerful communication model where no collision occurs. In more details, continuous aggregation [15] assumes that data have to be aggregated, and that the result of the aggregation is then disseminated to all participating nodes. The main metric is then the delay before aggregated data is delivered to all nodes, as no particular node plays the role of a sink. The alternative problem considered by Cornejo *et al.* [16] aims to minimize the number of nodes that owns data (or partially aggregated data) assuming *unicast* communications. In this work [16], the time is finite and no particular node plays the role of sink node as any node can retrieve the data from any other node. To summarize, no previous work considers the data aggregation problem in dynamic networks *allowing* collisions.

Our Contribution. The contribution of this paper is threefold. First, in order to compare the complexity of the data aggregation in static and dynamic WSN, we give a tight bound for the complexity of the MDAT problem in static WSN. In more details, we show that, in a *static* WSN, the problem remains NP-complete when the graph is a partial grid of degree at most three (a particular case of WSN topology). As it is trivial to solve the problem in static graphs of degree at most two, our result implies that the problem is intrinsically difficult for any practical setting. This result closes the complexity gap in the static case.

Second, we introduce an extension of the MDAT problem in dynamic WSNs, and we prove that the dynamic MDAT is NP-complete in a *dynamic* partial grid of degree at most two (and it is trivial to solve the problem if the graph is of degree at most one). As our result does not use the geometric properties of the graph, it remains valid for arbitrary dynamic graphs whose degree is greater than one. We also show that allowing simultaneous transmissions to the same node is not intrinsically helpful as it only delays the complexity wall: we show that the problem remains NP-complete if a node can correctly receive up to $K > 1$ simultaneous packets from different neighbors, if the maximum node degree of the graph is $K + 2$ in the static case (and $K + 1$ in the dynamic case).

Third, we give the first lower and upper bounds for the dynamic MDAT problem. More precisely, we define the notion of the foremost convergecast tree to the sink node as a convergecast tree to the sink node, with the earliest arrival time. Then, the minimum time to aggregate all data in a dynamic network is greater than the duration of a foremost convergecast tree (this result is valid in *any* graph, and for any degree Δ, there exists a dynamic graph such that the bound is attained) and is smaller than the duration of $n - 1$ independent foremost convergecast trees (this later bound is valid for any graph, but actually obtained for dynamic graphs of degree $n - 1$). If we restrain the class of dynamic graphs to those of degree smaller than $n - 1$, we prove that the upper bound is greater or equal to the duration of l independent foremost convergecast trees (with $l = (\Delta - 1) \log_{\Delta} (n (\Delta - 1) + 1) - \Delta + 2$), which prevents previous approximation algorithms for the static case to be extended to the dynamic case. Finally, we observe that, even in periodic graphs, optimal solutions cannot be computed by an algorithm that is unaware of the future of the graph or by a distributed algorithm (even if each node knows its own future). This motivates our simple approximate algorithm presented in section 6 to be centralized with full knowledge (yet, it does not assume that the graph is a dynamic WSN in the sense that it can perform on arbitrary graphs). The approximation factor is $T(n - 1)$ if there exists a bound T such that there is a journey between every



(a) node u does not receive data (b) node u receives data from v and u' from v'

Figure 1: Communication constraints

two nodes in every time interval $[t, t+T]$.

2. Model and Preliminaries

Wireless sensor networks (WSNs) containing n nodes with transmission range normalized to 1, are modeled by unit disk graphs [1] *i.e.*, intersection graphs of n equal-sized circles. Each vertex corresponds to a circle of radius 1/2, and an edge exists between two vertices when the corresponding circles intersect (tangent circles are assumed to intersect).

We model a *dynamic WSN* as a *discrete-time-varying* graph [10]. According to this model, we consider a discrete lifetime $\mathcal{T} = \mathbb{N}$ with a constant *latency* function ρ that equals one for every edge at any time (messages can travel at most one hop at a given time). Under those assumptions, a dynamic graph is seen as an *evolving* graph *i.e.*, a sequence of *snapshots*, where each snapshot is a static graph that represents the evolving graph at a given time $t \in \mathbb{N}$. The maximum node degree of a dynamic graph, denoted Δ, is the maximum node degree among all its snapshots. We recall that in dynamic graphs, an edge is a pair $((u, v), t)$, where u and v are two nodes that are connected at time t.

Definition 1. *A* dynamic wireless sensor network G *is a dynamic graph* $(V, (E_t)_{t\in\mathbb{N}})$ *where* V *is the set of vertices and* $(E_t)_{t\in\mathbb{N}}$ *a sequence that represents the edges of the graph over the time, such that for each* t, (V, E_t) *is a unit-disk graph.*

Data Aggregation Schedule. The time is discrete and at each time round, called *time slot*, communications are constrained by the following rule. Sensor nodes can send or receive data, but cannot do both at the same time. Moreover, if two nodes send their data simultaneously, all their common neighbors do not receive anything, due to interference (see figure 1). A node can aggregate a received data with its own data, according to a given aggregation function (simple examples of aggregation functions include maximum and addition). The aggregation is supposed to be atomic, and the resulting data can be sent like the original data *i.e.*, in one time slot.

Let $G = (V, (E_t)_{t\in\mathbb{N}})$, $A \subseteq V$, and $B \subseteq A$. We say that *data is aggregated from* A *to* B *at time* t, denoted by $(G, A, t) \to (G, B, t+1)$, if nodes in $A \setminus B$ transmit their data simultaneously and all the data is received by at least one node in B. Formally if:

$$\forall u \in A \setminus B,\ \exists v \in B,\ \forall u' \in A \setminus B \setminus \{u\} :$$
$$(u, v) \in E_t \wedge (u', v) \notin E_t$$

A *dynamic data aggregation schedule* to s of duration l is a decreasing sequence of sets $V = R_0 \supseteq R_1 \supseteq \ldots \supseteq R_l = \{s\}$ satisfying the following condition:

$$\forall i \text{ s.t. } 0 \leq i < l, \quad (G, R_i, i) \rightarrow (G, R_{i+1}, i+1)$$

Dynamic Minimum Data Aggregation Time Problem. An instance of the dynamic MDAT problem is a pair (G, s), where $G = (V, (E_t)_{t\in\mathbb{N}})$ models a dynamic WSN, and $s \in V$ denotes the sink node. A solution of an instance (G, s) is a dynamic data aggregation schedule to s with minimum duration. The minimum duration is denoted by $MDAT_{Opt}(G, s)$.

Remark 1. *The dynamic minimum data aggregation time problem may have no solution, even in a dynamic WSN G connected over time (i.e., such that for all $u, v \in V$, there exists a journey from u to v). Indeed, consider a set of edges defined as follow: $E_0 = V \times V$ and $\forall i \neq 0$, $E_i = \emptyset$. Then, the graph is connected, but only one node can send its data to the sink node at time* 0*, and the other nodes are never able to send their data. A simple sufficient (but not necessary) assumption that ensures the existence of a solution is that the graph is recurrent connected (see our algorithm GDAS in the sequel).*

3. NP-Hardness

3.1. Static grid graphs of degree at most three

A grid graph is a unit disk graph where all disks have centers with integer coordinates and radius 1/2 *i.e.*, an induced sub-graph of the grid. However, a sub-graph of the grid (not necessarily induced, called partial grid) is not necessarily a grid graph.

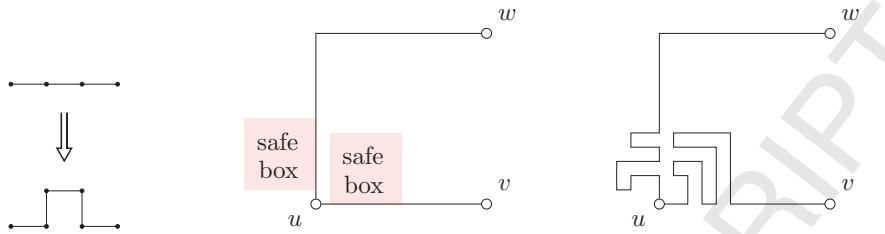
Chen *et. al.* prove in [2] that finding the minimum data aggregation time is $NP-$hard, even when the network is restricted to partial grid (with maximum degree $\Delta = 4$). On the other side, if the maximum degree of a static graph is $\Delta = 2$, the graph is either a line or a cycle and the minimum data aggregation time is easy to compute. Let ε be the eccentricity of the sink node and n the number of nodes. If n is odd and $\varepsilon = (n-1)/2$ (the graph is either a cycle or a line with the sink node in the middle) then the MDAT is $\varepsilon + 1$. Otherwise the MDAT is ε.

In this section we close the complexity gap of the MDAT problem in static networks by proving that the MDAT is $NP-$hard, even when restricted to grid graphs with maximum degree $\Delta = 3$.

We use a construction that is similar to that of Chen *et al.* [2] with some improvements about the topology (grid graph instead of partial grid) and about the maximum node degree (3 instead of 4). We first state a lemma slightly different from their lemma 2 [2], and follow with our first theorem.

Lemma 1. *Let H be a planar graph with $n > 6$ nodes and maximum degree $\Delta \leq 4$, there exist an orthogonal planar embedding of H such that each edge has the same length. This embedding can be computed in time polynomial in n.*

Proof. From [17], we know that there exists an orthogonal embedding of H in a grid g of size n where each edge has at most 2 bends (so each edge has length smaller than $3n$). Let g_i be the grid g where the unit has been divided by $4i$, of size $4in$. We consider the corresponding embedding of H in g_i (so that the coordinates of vertices has been multiplied by $4i$). In g_i, the maximum length of an edge is $12in$.



(a) replacement of an edge of length 3 by an edge of length 5

(b) positions of the safe boxes

(c) both edges have the same length

Figure 2: Construction of the orthogonal embedding

We can increase the length of an edge by 2 by replacing a piece of edge between two points with integer coordinates by a new piece of edge, like in figure 2a. This is possible if there are no other edges around.

By choosing $i \geq 48n$, there is enough space in a subgrid $(2i-2) \times (2i-1)$, called *safe box*, to contain an edge of length $3 \times 4in$. So that, if $i \geq 48n$, since all the length are even, we can lengthen all the edges to make them have a length $12in$. Indeed, we can lengthen a given edge in the first $(2i-2) \times (2i-1)$ subgrid of an end point, in the anticlockwise direction (see figure 2). □

Before stating our first theorem, we recall the definition of the restricted planar 3-SAT [18]. Let φ be a 3−SAT formula composed by a set C of m clauses $c_1, \ldots, c_m$ over a set V of n variables $v_1, \ldots, v_n$. We define the corresponding formula graph $G_\varphi = (V \cup C, E_1 \cup E_2)$, where $E_1 = \{(x_i, c_j) : x_i \in c_j \text{ or } \bar{x}_i \in c_j\}$ and $E_2 = \{(x_i, x_{i+1}) : 1 \leq i \leq n-1\} \cup \{(x_n, x_1)\}$. φ is said to be *planar* if the formula graph G_φ is planar. φ is said to be *restricted* if *(i)* each variable appears in at most three clauses, *(ii)* both negated and unnegated forms of each variable appears at least once, and *(iii)* clauses drawn on the same side of the cycle E_2 must share the same literal if they share the same variable (*i.e.*, at a variable vertex in G_φ, incident edges from one side correspond to the same literal). It is known that restricted planar 3-SAT is NP-complete [18].

Theorem 1. *The MDAT problem restricted to grid graphs of degree at most three is NP-complete.*

Proof. The proof is by reduction from restricted planar 3−SAT. Let φ be an instance of the restricted planar $3-SAT$ on n variables and m clauses. From the planar formula graph G_φ, we construct a planar graph G with maximum degree $\Delta = 3$. The idea behind the construction is that, in order to have a fast data aggregation, the schedule must "choose" between two sides (*i.e.*, the data are aggregated along one of two possible paths) representing the true or false instantiation of a variable. The aggregation is fast if all clauses are connected to the correct side of at least one variable. First we construct the sub-graph X_i that represents the variable x_i (see figure Fig. 3.1).

X_i is composed of a cycle of nodes e_i, l_i, r_i, s_i, o_i, $\bar{s}_i$, $\bar{r}_i$, $\bar{l}_i$. Then, we connect to l_i (resp. $\bar{l}_i$) a line L_i (resp. $\bar{L}_i$) of length $5i-3$. Thus l_i cannot sends its data before

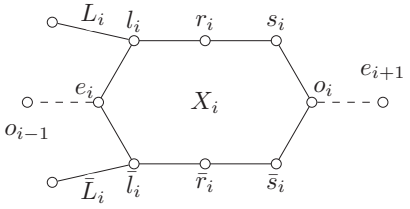


Fig. 3.1: sub-graphs X_i, L_i and $\bar{L}_i$

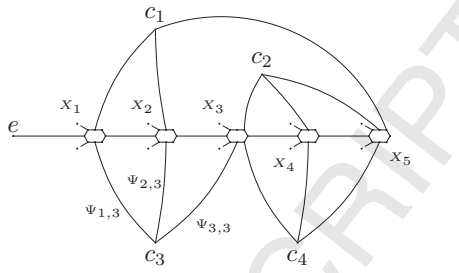


Fig. 3.2: example for $\varphi = (x_1 \vee x_2 \vee \bar{x}_5) \wedge (x_3 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_3 \vee x_4 \vee x_5)$

aggregating data from L_i *i.e.*, before $5i-3$ timeslots. For $1 \leq i < n$ we connect o_i to e_{i+1} and we connect to e_1 a new node e_0.

Each clause c_j is represented by a single node and for each variable x_i (resp. negation $\bar{x}_i$) in clause c_j, we connect c_j to r_i or s_i (resp. $\bar{r}_i$ or $\bar{s}_i$) by a line $\Psi_{i,j}$ of length $(5i-2)$ to r_i (resp. $\bar{r}_i$) or $(5i-1)$ to s_i (resp. $\bar{s}_i$). Let $G = \bigcup_{1 \leq i \leq n}^{1 \leq j \leq m} \left(X_i \cup L_i \cup \bar{L}_i \cup \Psi_{i,j}\right)$ (see figure Fig. 3.1 and Fig. 3.2).

In order to use the previous lemma we need to be able to change the distance between nodes. So we define G_T obtained from G by replacing every edge by a line of length T *i.e.*, by adding $T-1$ nodes between two connected nodes, and by adding a pending node e connected to e_0.

In the next three Claims we show that there exists a T such that G_T is a grid graph (of degree at most 3) and that the minimum time to aggregate data from G_T to o_n is $5nT+1$ if and only if φ is satisfiable. Then, the theorem follows from the NP-completeness of restricted planar 3−SAT [18].

Claim 1. *There exists a T such that G_T is a grid graph.*

From Lemma 1 we deduce that G has an orthogonal embedding such that every edge has the same length $l \geq 1$ in a grid of size s. We divide the unit by 4 so that the embedding is in a grid of size $4s$ and every edge has length $4l$ (every vertex has their coordinates multiplied by 4). Then, we replace, in its embedding, each node by a disk of radius 1/2, and every edge by a chain of $4l-1$ disk of radius 1/2, centered at integer coordinates along the edge. Finally, we add a disk of radius 1/2, centered at integer coordinates, at distance 1 from e_0 and at distance greater than 1 from other disks. The corresponding unit disk graph (that is also a grid graph) is exactly G_{4l}.

Claim 2. *For all $T \geq 1$, if φ is satisfiable, then the minimum time to aggregate data from G_T to o_n is $5nT+1$.*

First of all, we remark that the distance between e and o_n is $5nT+1$, so that $5nT+1$ is a lower bound for the minimum data aggregation time.

Now suppose that φ is satisfiable and let I be an interpretation of the truth-functional propositional calculus satisfying φ. Let $i \in [1..n]$ and suppose $I(x_i) = true$. Suppose that e_i has aggregated at time $(5i-4)T+1$ data from previous nodes $X_k \cup (\Psi_{k,j} - \{c_j\})$, $k < i$ and from clauses c_j containing for every $j < i$, x_j if $I(x_j) = true$, and $\bar{x}_j$ otherwise. As we said before, l_i (resp. $\bar{l}_i$) can aggregate all data from L_i (resp. $\bar{L}_i$) at time $(5i-3)T$.

At the negative side of X_i, $\bar{l}_i$ can also aggregate all data of the $T-1$ nodes between

e_i and $\bar{l}_i$ before time $(5i-3)T$. If $\bar{r}_i$ is connected to a clause c_j, it can receive all data from nodes between $\bar{r}_i$ and c_j (c_j excluded) at time $(5i-2)T-1$. Then $\bar{r}_i$ can receive all data from $\bar{l}_i$ at time $(5i-2)T$. If $\bar{s}_i$ is connected to c_j, it can receive all data from nodes between $\bar{s}_i$ and c_j (c_j excluded) at time $(5i-1)T-1$. Then $\bar{s}_i$ can aggregate all data from $\bar{r}_i$ at time $(5i-1)T$. Thus o_i can aggregate all data from $\bar{s}_i$ at time $5iT$.

On the other side of the cycle, l_i has to wait time $(5i-3)T+1$ to receive data from e_i. Again, if r_i is connected to c_j, it can receive all data from nodes between r_i and c_j (c_j included) at time $(5i-2)T$. Then r_i can receive all data from l_i at time $(5i-2)T+1$. If s_i is connected to c_j, it can receive all data from nodes between s_i and c_j (c_j included) at time $(5i-1)T$. Then s_i can aggregate all data from r_i at time $(5i-1)T+1$. Finally, o_i can aggregate all data from s_i at time $5iT+1$.

If $i=n$, then it's over, otherwise o_i can start transmitting to the next block X_{i+1} and e_{i+1} can aggregate data from o_i at time $(5(i+1)-4)T+1$, the data includes data from clauses containing x_i. If $I(x_i)=false$, the schedule on the positive and negative side of the cycle are exchanged in order to aggregate data from clauses containing $\bar{x}_i$ instead of x_i. Recursively, since all clauses are connected to a variable x_i with $I(x_i)=true$ or to $\bar{x}_i$ with $I(x_i)=false$, o_n aggregates all data at time $5nT+1$.

Claim 3. *For all $T \geq 1$, if the minimum time to aggregate data from G_T to o_n is $5nT+1$, then φ is satisfiable.*

Suppose that all data from G_T are aggregated to o_n at time $5nT+1$. Since e is at distance $5nT+1$, its data is sent directly through a shortest path *i.e.*, there is a shortest path P from e to o_n such that, when a node in P receives at time t a data from e (an aggregation that contains e's data) it sends the aggregation to the next node in the path at time $t+1$. There are 2^n shortest paths from e to o_n, indeed, at each block X_i, the path can use the positive or the negative side, and implicitly choose to interpret x_i as true or false. Let $i \in [1..n]$, and suppose P uses the positive side of X_i. As we saw before, data from L_i and $\Psi_{i,j}$ for clauses c_j that contain x_i can be aggregated to l_i, r_i and s_i before the data from e. The first observation is that o_i receives data from e, L_i and $\Psi_{i,j}$ for clauses c_j containing x_i at time $5iT+1$ and has to send the aggregation just after (it cannot receive data after time $5iT+1$).

On the other side, $\bar{l}_i$ can start sending at time $(5i-3)T$ (because it must receive data from $\bar{L}_i$ first), and its data must be aggregate through a path to o_n of length smaller than or equals to $5nT+1-(5i-3)T$. Because of its length, this path must pass through o_i, by the negative side of X_i. Thus data from $\bar{l}_i$ must be aggregate through a path to o_i of length smaller than or equals to $3T+1$ (the maximum length of the path minus the minimum distance from o_i to o_n). Adding this to the first observation, we know that data from $\bar{l}_i$ must be received by o_i at time $5iT$ (and cannot be received before, because $\bar{l}_i$ and o_i are at distance $3T$).

Thus, data from $\bar{l}_i$ cannot be delayed (except by o_i) *i.e.*, if a node receives a data from $\bar{l}_i$ at time t, it must send the aggregation to the next node at time $t+1$. Thus $\bar{r}_i$ and $\bar{s}_i$ can receive data from other nodes only before time $(5i-2)T$ and $(5i-1)T$, and so this data cannot include data from a connected clause c_j (only from $\Psi_{i,j}-\{c_j\}$). Since all paths from a clause c_j to o_n passing trough X_i contain a node r_i, $\bar{r}_i$, s_i or $\bar{s}_i$, a connected clause can send its data through X_i only if it contains x_i.

The same thing happens if P uses the negative side of X_i: a connected clause can send its data through X_i only if it contains $\bar{x}_i$.

For a shortest path P from e to o_n used to aggregate e's data, we say a variable x_i

is true if P uses the positive side of X_i, and $\bar{x}_i$ is true otherwise. We have shown that if the data of a clause c_j is received by o_n on time (before time $5nT+1$), then c_j contains a true literal.

Finally, if data from all clauses is received on time, then every clauses contains at least one true literal, and the formula is satisfiable. □

An interesting extension can be to investigate the case when a constant number K of simultaneous transmissions are feasible. Let $K \in N^*$, we define the MDAT$_K$ problem as the MDAT problem, with the additional assumption that nodes can simultaneously receive up to K messages from K different neighbors. Note that simultaneously receiving $K+1$ or more messages still results in a collision. We now revisit the previous result with $K > 1$. We show that allowing one more simultaneous transmission results in increasing by one the maximum node degree of the graph for the problem to be NP-complete. This may seems obvious at first sight, but the fact that the problem concerns unit-disk graphs makes the proof slightly technical. Indeed, every time we want to create a collision, to force the algorithm to make a choice, we have to create a node of degree $K+2$, and in a unit-disk graph, if K is large enough, such a node must have two neighbors that are connected. So that we cannot connect an arbitrary amount of lines to a node while keeping the whole network as a unit-disk graph. The idea of the proof is as follow. For each node where we need a collision (see proof of theorem 1), we add a single special line of nodes. The resulting graph has now a maximum node degree of 4 and still have an embedding in the grid. Then, after replacing each edge in the embedding by nodes to create a unit-disk graph, we replace each special line by a unit-disk graph that creates the desired collision.

Theorem 2. *The MDAT$_K$ problem restricted to graphs of degree at most $K+2$ is NP-complete.*

Proof. Let φ be an instance of the restricted planar $3-SAT$ with n variables and m clauses. From the planar formula graph G_φ, we construct a graph G as in the proof of Theorem 1, with 5 additional lines C_1, C_2, C_3, C_4, C_5 connected to nodes $r_i, l_i, \bar{r}_i, \bar{l}_i$, and o_i. We connect C_1 (resp. C_2) of length $5i-2$ to r_i (resp. $\bar{r}_i$), C_3 (resp. C_4) of length $5i-1$ to l_i (resp. $\bar{l}_i$), C_5 of length $5i+1$ to o_i. As in the proof of Theorem 1 we define G_T obtained from G by replacing every edge by a line of length T *i.e.*, by adding $T-1$ nodes between two connected nodes, and by adding a pending node e connected to e_0. G has a maximum node degree of 4, so it has an orthogonal embedding such that every edge has the same length $l \geq 1$ in a grid of size s (see Lemma 1). We divide the unit distance by 4 so that two edges are at distance at least 4 from each other. Then, we divide the unit distance by $2(K-1)$.

Let $\varepsilon > 0$ and $K' = K-1$. For each line $C \in \{C_1, C_2, C_3, C_4, C_5\}$ of length d in the embedding (d is a multiple of $8K'$), we split the line in small parts of length $2K'$ at the two extremities, and length $4K'$ otherwise (see Figure 3). Each small part is either a straight line or two orthogonal straight lines. The goal is to replace each part by a unit-disk graph $\hat{C}$ such that, when aggregating all the data from $\hat{C}$, K' nodes then want to transmit simultaneously at time d their data to the node that connects $\hat{C}$ to the remaining of the graph.

Each straight line of length $4K'$ is replaced by $K' \times 4K'$ disks of radius $1/2$ spread across a grid of size $K' \times 4K'$, that is constrained in a rectangular area of width $4K'$,

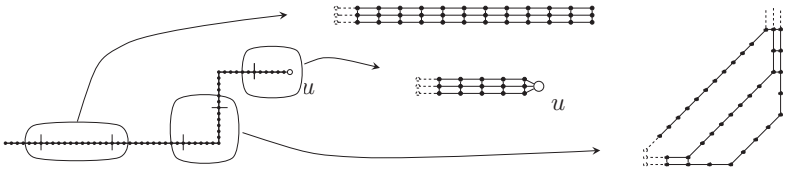


Figure 3: The replacement of a line connected to node $u \in \{r_i, \bar{r_i}, l_i, \bar{l_i}, o_i\}$ by graphs containing $K' = K - 1$ lines of length $2K' - 1$ for the part connected to u, and $4K'$ for the parts not at an extremity.

height ε, and centered at the middle of the initial line. This implies that the intersection graph of those disks (see Figure 3) is composed of $4K'$ cliques of size K' that are connected by K' lines of length $4K'$. This graph has the property that the distance between two nodes located at distinct extremities is either $4K'$ if the nodes are in the same line, or $4K' + 1$ otherwise. We repeat the same process for the parts at the extremities, using only $K' \times (2K' - 1)$ disks for the part connected to the remaining of the graph, and $K' \times 2K'$ disks for the part at the other extremity.

Similarly, each part composed by two lines at a right angle is replaced by $K' \times 4K'$ disks of radius $1/2$ in such a way that the intersection graph contains K' lines of length $4K'$ (connected with some additional edges), and has the same property as the previous intersection graph (see Figure 3). This implies that the intersection graph produced by all the disks of all the small parts contains K' lines of length d such that the distance between two nodes at distinct extremities is d if they are in the same line, or $d + 1$ otherwise.

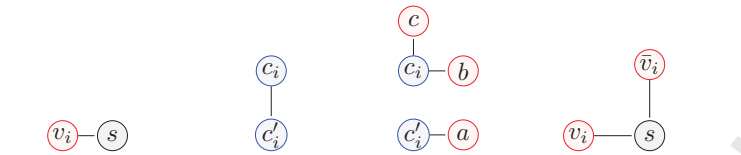
All the disks are slightly translated toward the node $u \in \{r_i, \bar{r_i}, l_i, \bar{l_i}, o_i\}$ that connects C to the remaining of the graph, so that the K' disks at the extremity of the line are at distance at most 1 from u. Also, ε is chosen sufficiently small so that disks that replace Line C do not intersect with other disks, located in the remaining of the graph.

Finally, as in the previous proof, we replace in the remaining of the embedding each node by a disk of radius $1/2$, and every edge by a chain of $8K'l - 1$ disks of radius $1/2$, centered at integer coordinates along the edge. Finally, we add a disk of radius $1/2$, centered at integer coordinates, at distance 1 from e_0. The obtained graph is $G_{8K'l}$, where each line C_i is replaced by a subgraph $\hat{C}_i$ such that, to aggregate all data from $\hat{C}_i$ without delaying the data aggregation in the whole graph, the node u that connects $\hat{C}_i$ to the remaining of the graph has to aggregate the $K - 1$ neighbors in $\hat{C}_i$ simultaneously at time d, which equals the length of C_i. Then, only one other neighbor of u can transmit its data to u at time d. This simulates the constaint that no two neighbors of u could transmit at time d without interference in the previous setting. So, we can apply the same proof as the case $K = 1$ on the transformed graph. □

One can observe that the problem is easy to solve in static graphs of degree at most $K + 1$. Indeed, when aggregating data along the shortest path tree, no collision occurs, except maybe at the sink node. Any schedule, for the transmissions of the neighbors of the sink, that avoid collisions is optimal.

3.2. Dynamic graphs of degree at most two

In a dynamic network we prove that, even when the maximum degree is two, the dynamic MDAT problem is NP-hard. This result is optimal since the problem is easy to



(a) configuration during T_1 (b) configuration during T_2 (c) configuration during T_3

Figure 4: Node Configurations (clauses are blue and literals are red).

solve in a graph of degree at most one (where no collision occurs).

Theorem 3. *The dynamic MDAT problem is NP-complete even in a dynamic wireless sensor network of degree at most two.*

Proof. The proof is by reduction from 3-SAT. Given any 3-SAT instance φ of n variables $v_1, \ldots, v_n$ and m clauses $c_1, \ldots, c_m$, we construct the dynamic grid graph $G_\varphi(V, E)$ as follow:

Nodes are composed of one sink node, literals, clauses, and copy of clauses:

$$V = \{s\} \bigcup_{1 \leq i \leq n} \{v_i\} \cup \{\bar{v}_i\} \bigcup_{1 \leq i \leq m} \{c_i\} \cup \{c'_i\}$$

Let $t_f = 3n + 2m$. We decompose the time interval $[1, t_f]$ in three periods T_1, T_2 and T_3 (see figure 4):

- During $T_1 = [1, 2n]$, we have for all $i \in [1, n]$:

$$E_{2i-1} = \{(v_i, s)\}, \quad E_{2i} = \{(\bar{v}_i, s)\}$$

- During $T_2 = [2n + 1, 2n + 2m]$ we have for all $j \in [1, m]$:

$$\text{with } \{a, b, c\} = c_j, \quad E_{2n+2j-1} = \{(c_j, c'_j)\}, \; E_{2n+2j} = \{(c'_j, a), (c_j, b), (c_j, c)\}$$

- During $T_3 = [2n + 2m + 1, 2n + 2m + n]$ we have for all $i \in [1, n]$:

$$E_{2n+2m+i} = \{(v_i, s), (\bar{v}_i, s)\}$$

The remaining of the dynamic graph (after time t_f) can contain for instance composed only empty snapshots, or be such that the graph is periodic. This does not change the proof, but can be used to analyze the best approximation ratio for this problem (see remark 2).

During T_3, either a variable or its negation can send its data to the sink node s, but not both, so that the set of literals that send data can be seen as an interpretation of a truth-functional propositional calculus.

During T_1, variables that does not send their data during T_3 can send their data directly to the sink node.

During T_2, there is a link between a clause c_i and its copy c'_i so that either c_i or c'_i can send both data. Since all clauses can send their data only once to all the literals it contains, the data is successfully sent to the sink node if and only if at least one literals it contains sends its data in T_3 *i.e.*, is *true*.

Thus, if the interpretation chosen in T_3 satisfies the formula φ, then each clause contains a literal that transmits during T_3, and the minimum data aggregation time is exactly t_f. Otherwise, some clauses must send their data before $t = 1$, and the minimum data aggregation time is greater than t_f.

So that the 3-SAT instance φ is satisfiable if and only if the minimum data aggregation time ending before t_f is t_f. □

Remark 2. *Theorem 3 raises the question of the best approximation ratio achievable by an approximation algorithm. We observe that using time as a complexity measure is not relevant in the dynamic case. Contrary to the static case, where good approximation ratios have been found, the problem is not approximable in the dynamic case using the duration of the solution as a measure of complexity. Indeed, the dynamic graph constructed in the proof of Theorem 3, with empty snapshots when time is greater than t_f, only contains the optimal solution. So, if an approximation algorithm finds an approximate solution, it actually finds the optimal solution, which is not possible in polynomial-time (unless $P = NP$). Even when restricted to smaller classes of graphs, such as periodic dynamic graphs, the approximation ratio (with respect to duration) can be made arbitrary large by increasing the period. The approximation ratio can be bounded in periodic graphs, but only when the period is itself bounded (or in time-bounded recurrent connected graphs with fixed bound, as defined in Section 5). This remark justifies (i) our use of Foremost Convergecast Trees, defined in Section 4, as a complexity measure to establish upper and lower bounds, and (ii) the approximation ratio of our approximate algorithm, presented in Section 6, when restricted to time-bounded recurrent connected graphs.*

As in the static case, we show that when K simultaneous transmissions are allowed without collisions, the problem is similar. Here, the geometric constraints do not hold, giving a more straightforward proof.

Theorem 4. *The dynamic $MDAT_K$ problem restricted to evolving graphs of degree at most $(K + 1)$ is NP-complete.*

Proof. In the dynamic case, there is a simpler way to apply the same trick as in the static case. From a 3-SAT instance, we create the same evolving graph as in Theorem 3, but with $(K - 1) \times n$ additional nodes $(r^1_1, r^1_2, \ldots, r^n_{K-1})$, and where edges in the period T_3 are defined as follow: for all $i \in [1, n]$,

$$E_{2n+2m+i} = \{(v_i, s), (\bar{v}_i, s)\} \cup \bigcup_{j=1}^{K-1} \{(r^i_j, s)\}$$

So, in order for the data aggregation to terminates at time $t_f = 3n + 2m$, all nodes r^i_j have to transmit during T_3. This implies, like in the proof of Theorem 3, that either a variable or its negation can transmit during T_3. □

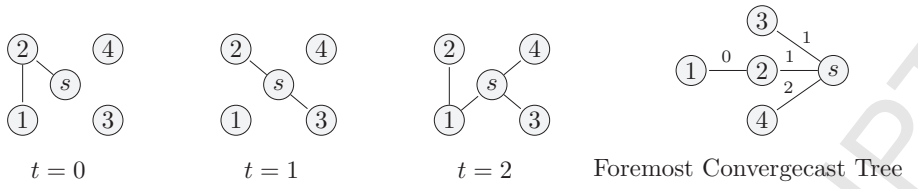


Figure 5: An example of foremost convergecast tree.

4. Upper and Lower Bounds

In this section, we introduce the notion of foremost convergecast trees. Then we propose the first upper and lower bounds for the dynamic MDAT problem, given in terms of foremost convergecast tree duration.

A journey from a node u to node a v is a sequence of edges $((e_1, t_1), \ldots, (e_r, t_r))$ such that $(e_1, e_2, \ldots, e_r)$ is a path from u to v in the static graph $(V, \cup_{i\in\mathbb{N}} E_i)$ and

$$\forall i \in [1..r-1],\ t_i < t_{i+1} \qquad \forall i \in [1..r],\ e_i \in E_{t_i}$$

For a journey J, we denote by $departure(J)$ the starting time t_1 and by $arrival(J)$ the arrival time $t_r + 1$ of the journey. The arrival time corresponds to the time of the existence of the last edge plus the latency to travel along the last edge. Then, $duration(J) = arrival(J) - departure(J)$ denotes the duration of the journey. We denote by $\mathcal{J}_{(u,v)}$ the set of journeys from u to v and by $\mathcal{J}_{(u,v)}^{[t_s,t_e]}$ the subset of journeys that start and end between t_s and t_e.

Definition 2. *Let $G(V, E)$ be a dynamic graph. A* convergecast tree *to node s is a pair (T, c), where $T(V, T_{edges})$ is a tree rooted at s, and c is a function $c : T_{edges} \to \mathbb{N}$ that satisfies: if u is a descendant of v in T and $(e_1, e_2, \ldots, e_r)$ is the path from u to v in T, then*

$$((e_1, c(e_1)), (e_2, c(e_2)) \ldots, (e_r, c(e_r)))$$

is a journey in G called the journey from u to s induced by T. The departure *(respectively, the* arrival*) of the convergecast tree is the departure of the first journey in T (respectively, the arrival of the last journey in T):*

$$departure(T, c) = \min_{e\in T_{edges}} c(e) \qquad and \qquad arrival(T, c) = \max_{e\in T_{edges}} c(e) + 1$$

Definition 3. *Let $G(V, E)$ be a dynamic graph. A* foremost convergecast tree *(abbreviated as FCT) to node s starting at time t_s is a convergecast tree (T, c) to s such that $departure(T, c) \geq t_s$ with minimum arrival time.*

$FCT(G, s, t_s)$ denotes the set of foremost convergecast trees of G to node s starting after time t_s. The common duration of foremost convergecast trees starting after t_s is denoted $FCTD(G, s, t_s)$.

In dynamic WSNs, a foremost convergecast tree plays the same role as a shortest path tree in static WSNs. Indeed it gives the same lower bound as in the static version of the problem. Figure 5 shows an example of the unique foremost convergecast tree to the sink node s starting at time 0 of a simple dynamic graph (for the sake of simplicity, the position of the nodes do not change with time).

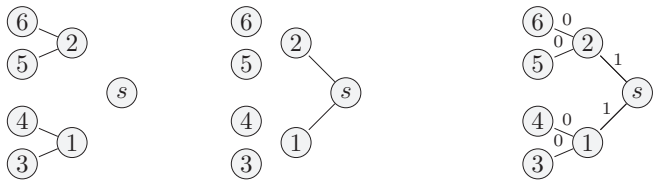


Figure 6: Creation of a perfect binary FCT.

Lemma 2. *Let G be a dynamic graph, we have:*

$$MDAT_{Opt}(G, s) \geq FCTD(G, s, 0)$$

Proof. Let G be a dynamic graph, s be a sink node, and $S = \{S_t\}_t$ be a data aggregation schedule to s of duration $l = MDAT_{Opt}(G, s)$.

Let x_0 be a node different from the sink. We know that there exists i is such that $x_0 \in S_i$ and $x_0 \notin S_{i+1}$. Since $(G, S_i, i) \to (G, S_{i+1}, i+1)$, there exists a node $x_1 \in S_{i+1}$ such that $(x_0, x_1) \in E_i$. We can apply the same argument to x_1 if it is different from the sink.

Recursively we have x_1, x_2, ..., $x_p = s$ and $t_1 < t_2 \ldots < t_p < l$ such that for all $1 \leq i \leq p$, $(x_{i-1}, x_i) \in E_{t_i}$. Thus $J = \{((x_{i-1}, x_i), t_i) \mid 1 \leq i \leq p\}$ is a journey from x_0 to s ending before l. We can do this with every nodes in $V - \{s\}$, which proves $FCTD(G, s, t_s) \leq l$. □

In a static WSN, the same shortest path tree can be used to avoid collisions. But in a dynamic WSN, a FCT $\mathcal{T}_1$ that exists at a given time may not exists thereafter. If we delay the transmission of a node, to avoid a collision, another FCT $\mathcal{T}_2$ is then used to retry a transmission. In order to be sure that $\mathcal{T}_2$ can be used by all delayed nodes, it has to start after the end of $\mathcal{T}_1$. In this case we say that $(\mathcal{T}_1, \mathcal{T}_2)$ is a 2-time-independent FCT.

Definition 4. *A l-time-independent FCT of G to s starting at time t_s is a sequence of l foremost convergecast trees of G to s $((T_1, c_1), \ldots, (T_l, c_l))$ such that:*

- *(T_1, c_1) is a FCT starting at t_s.*
- *for all $1 < i \leq l$, (T_i, c_i) is a FCT starting at $arrival(T_{i-1}, c_{i-1})$.*

Its duration is the sum duration of all FCT in the sequence and also equals to $tarrival(T_1, c_1) - t_s$. The set of l-time-independent FCT of G to s starting at t_s is denoted $FCT^l(G, s, t_s)$. The common duration of all l-time-independent LTJs in $FCT^l(G, s, t_s)$ is denoted $LTJD^l(G, s, t_s)$.

This definition is used to give lower and upper bounds for the minimum data aggregation time in a dynamic graph G with n nodes as follow:

$$FCTD^{n-1}(G, s, 0) \geq MDAT_{Opt}(G, s) \geq FCTD(G, s, 0)$$

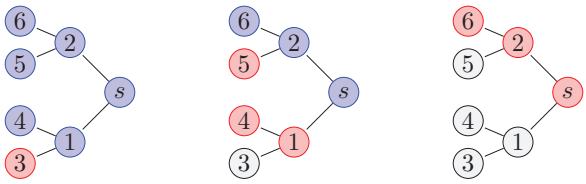


Figure 7: Optimal data aggregation schedule when FCTs are complete binary trees.

Where the right-hand inequality comes from lemma 2 and the left-hand inequality comes from the fact that a node can send its data during a FCT, so that $n-1$ foremost convergecast trees are sufficient to aggregate the data of every nodes.

The lower bound and the upper bound are tight in the sense that there exists a graph that reaches the lower bound (any graph of degree at most one) and a graph that reaches the upper bound (for instance a graph whose sink node is of degree $n-1$ at each time).

If we consider only graphs with a given maximum node degree Δ, the lower bound is still tight, but the upper bound is no longer tight. The following lemma gives a graph with a minimum data aggregation time that lowers the upper bound, for an arbitrary maximum node degree Δ. We conjecture that it also gives the worst data aggregation duration *i.e.*, that it also give an upper bound that remains tight for an arbitrary maximum node degree.

Lemma 3. *Let $n \in \mathbb{N}^*$ and $\Delta \leq n$, there exist a dynamic graph G with n nodes of degree at most Δ such that:*

$$FCTD^m(G,s,0) = MDAT_{Opt}(G,s) < +\infty$$

with $m = (\Delta-1)\log_\Delta(n(\Delta-1)+1) - \Delta + 2$

Proof. Let $\Delta \geq 2$. We consider the dynamic graph $G(V,E)$. For the sake of simplicity, we suppose that there exists $h \in \mathbb{N}$ such that $|V| = n = \frac{\Delta^{h+1}-1}{\Delta-1}$. One can construct G such that, there is a perfect $\Delta-$ary tree T (of height h) such that, for all $t \equiv 0 \mod h$, $FCT(G,s,t) = \{(T,c_t)\}$ and (T,c_t) is of duration h (and thus with collision appearing between every nodes having the same parent). See for instance figure 6 with $\Delta = 2$ and $h = 3$. The path associated to a journey from a node u to the sink is unique. So that a node has to wait to receive the data from all its children before transmitting.

Since no two children of s can transmit at the same time, s needs Δ FCTs to receive the data from all its children. Let s' be the first direct child of s that transmits, and $T(s')$ the sub-tree of T rooted at s'. $T(s')$ is a perfect $\Delta-$ary tree of height $h-1$. When s' transmits, its data contains the data of its children. Again, Δ FCTs are needed to aggregate all the data from all its children. One of this FCT can be used to aggregate the data from s' to s. For each layer of the tree, $\Delta-1$ consecutive FCTs are needed.

Recursively, we need at least $(\Delta-1)h+1$ time-independent $FCTs$ to aggregate all the data from G. One can show that this is also sufficient (see figure 7). Since $h = log_\Delta(n(\Delta-1)+1)-1$, the theorem is proved. □

Conjecture 1. *Let G be a dynamic graph with n nodes of degree at most Δ. Let $m = (\Delta-1)\log_\Delta(n(\Delta-1)+1) - \Delta + 2$, we have:*

$$FCTD^m(G,s,0) \geq MDAT_{Opt}(G,s)$$

We observe that the conjecture is proved for $\Delta = n - 1$ and is trivial if $\Delta = 1$.

5. Impossibility Results

In this section we present several classes of dynamic graphs where only a centralized algorithm that knows the future can compute optimal and "good" approximate solutions. A hierarchy of classes of dynamic graphs has been identify in previous work [10]. Here we present only the few we are interested in.

- $\mathcal{RC}$ *(Recurrent connectivity)*: $\forall u, v \in V,\ \forall t \in \mathbb{N}$:

$$\mathcal{J}_{(u,v)}^{[t,+\infty)} \neq \emptyset$$

- $\mathcal{BRC}$ *(Time-bounded recurrent connectivity)*: There exists a bound T such that, $\forall u, v \in V,\ \forall t \in \mathbb{N}$:

$$\mathcal{J}_{(u,v)}^{[t,t+T]} \neq \emptyset$$

- $\mathcal{P}$ *(Periodic)*: the graph is connected and there exists $T \in \mathbb{N}$ such that:

$$\big(\exists t,\ (u,v) \in E_t\big) \Rightarrow (\forall k \in \mathbb{N},\ (u,v) \in E_{t+kT})$$

The two following impossibility results are for the class of periodic graphs, and naturally extend to larger classes such as recurrent connected. The first impossibility result concerns distributed algorithms. A distributed algorithm is a set of local algorithms that are each executed independently by each node. It is also assumed that nodes no *not* have knowledge about the global topology (or future global topology) of the graph; they may only have knowledge about *adjacent* edges and nodes (or their future). In general, when two nodes interact, we assume that they are allowed and able to exchange their local knowledge (for instance, about their respective local future if they are aware of it) and use this information for future interactions. For our problem, a distributed algorithm has to decide, whenever an interaction occurs, whether the node sends its data or not.

Proposition 1. *In $\mathcal{P}$, the dynamic MDAT problem does not have a distributed optimal algorithm, even if each node knows its own future.*

Proof. We define $G = (V, \{E_i\})$ and $G' = (V', \{E'_i\})$ as follow (see figure 8):

$$\begin{array}{lll} V = V' = \{s, 1, 2, 3\}, \quad \forall i \in \mathbb{N}, & & \\ E'_i = \{(1, s), (2, 3)\},\ E_i = \{(1, s)\} & \text{if } i \equiv 0 \mod 3 \\ E'_i = E_i = \{(1, 2), (3, s)\} & \text{if } i \equiv 1 \mod 3 \\ E'_i = E_i = \{(1, s), (2, 3)\} & \text{if } i \equiv 2 \mod 3 \end{array}$$

At time 0 a distributed algorithm $\mathcal{A}$ do the same thing for node 1 and s on G and G', because they have the same neighbors now and in the future. If $\mathcal{A}$ decide that node 1 transmits its data to s at time 0, then $\mathcal{A}$ is not optimal on G (since its faster to wait the data from node 2 at time 1 and then transmit at time 2). Otherwise, $\mathcal{A}$ is not optimal on G' (since all data can be aggregated at time 1). □

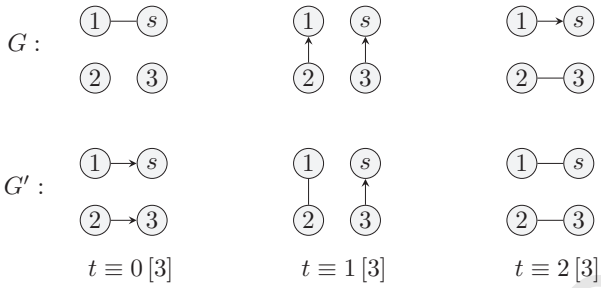


Figure 8: Optimal data aggregation in graph G and G'. In G, node 1 transmits at time 0 and in G', node 1 does not transmits at time 0, even though node 1 has the same future in both G and G'.

If we consider time as a measure of complexity, the following proposition shows that the best competitive ratio a distributed algorithm with knowledge of (local) future can achieve is unbounded.

Proposition 2. *In $\mathcal{P}$, the competitive ratio of a distributed algorithm is unbounded for the dynamic MDAT problem, when considering time as a measure of complexity, even if each node knows its own future.*

Proof. We construct two graphs, G and G', so that there is an arbitrary delay between the optimal solution and any other solution, resulting in unbounded competitive ratio. In more details, for all $K > 2$, we define $G_K = (V, \{E_i\})$ and $G'_K = (V', \{E'_i\})$, periodic with period $T = 2K^2$, as follow :

$$
\begin{array}{ll}
V = V' = \{s, 1, 2, 3\}, \quad \forall i \in \mathbb{N}, & \\
E'_i = \{(1, s), (2, 3)\},\ E_i = \{(1, s)\} & \text{if } i \equiv 0 \mod T \\
E'_i = E_i = \{(1, 2), (3, s)\} & \text{if } i \equiv 1 \mod T \\
E'_i = E_i = \{(1, s), (2, 3)\} & \text{if } i \equiv 2K \mod T \\
E'_i = E_i = \emptyset & \text{otherwise}
\end{array}
$$

So that, for an algorithm $\mathcal{A}$, either (i) $\mathcal{A}$ chooses that node 1 transmits at time 0 and the time to aggregate all the data is greater than $2K^2$ compared to $2K$ for an optimal centralized algorithm (*i.e.* a ratio of K) or (ii) $\mathcal{A}$ chooses that node 1 does not transmit at time 0 and the time to aggregate all the data is greater than $2K$ compared to 2 for an optimal centralized algorithm (*i.e.* a ratio of K). In both case, the ratio K between the time to aggregate the data with $\mathcal{A}$ compared to an optimal centralized algorithm can be made arbitrary big. □

Proposition 3. *In $\mathcal{P}$, without knwoledge of the future, the dynamic MDAT problem does not allow a centralized optimal algorithm.*

Proof. Let $k \in \{1, 2\}$. We define G_k as follow (see figure 9):

$$
V = \{s, 1, 2\}, \quad \forall i \geq 0, \quad E_i = \begin{cases} (1, 2) & \text{if } i \equiv 0 \mod 2 \\ (k, s) & \text{if } i \equiv 1 \mod 2 \end{cases}
$$

Let $\mathcal{A}$ be an algorithm that does not know the future. At time 0, $\mathcal{A}$ is executed the same way on G_1 and G_2. If $\mathcal{A}$ decides that node 1 should transmit its data to node 2 at time

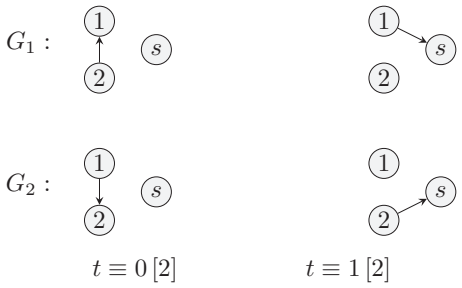


Figure 9: Optimal data aggregation in graph G_1 and G_2. In G_1, node 2 transmits at time 0 whereas in G_2, node 1 does.

0, $\mathcal{A}$ cannot solve the problem on G_1. If $\mathcal{A}$ decides that 2 should transmit to 1 at time 0, then $\mathcal{A}$ cannot solve the problem on G_2. Otherwise, if $\mathcal{A}$ choose to do nothing at time 0, the solution given by $\mathcal{A}$ is not optimal. □

Again, if we consider the time as a measure of complexity, the following proposition shows that the best competitive ratio a centralized algorithm without the knowledge of future (called online algorithm) can achieve is unbounded.

Proposition 4. *In $\mathcal{P}$, the competitive ratio of an online algorithm (that is, a centralized algorithm without knowledge of the future) is unbounded for the dynamic MDAT problem, considering time as a measure of complexity.*

Proof. Let $G^\infty(V, E^\infty)$ be defined as follow:

$$V = \{s, 1, 2\}, \quad \forall i \geq 0, \quad E_i^\infty = \begin{cases} (1,2) & \text{if } i \equiv 0 \mod 2 \\ (1,s) & \text{if } i \equiv 1 \mod 2 \end{cases}$$

Let $\mathcal{A}$ be an algorithm that does not know the future. When executing $\mathcal{A}$ on G^∞, either *(i)* $\mathcal{A}$ decides never to transmit, and the corresponding competitive ratio of $\mathcal{A}$ is infinite, *(ii)* the first node to transmit is node 1, and the competitive ratio of $\mathcal{A}$ is infinite, or *(iii)* at some time t, $\mathcal{A}$ decides that node 2 transmits to node 1. Let $G^{t,T}(V, E^{t,T})$, with $T > t$, be defined as follow:

$$\forall i \geq 0, \quad E_i^{t,T} = \begin{cases} (1,2) & \text{if } i \equiv 0 \mod 2 \\ (1,s) & \text{else, if } i \equiv j \mod 3T, \text{ with } 0 < j < t \\ (2,s) & \text{otherwise} \end{cases}$$

Then, when executing $\mathcal{A}$ on $G^{t,T}$, node 2 transmits to node 1 at time t, and node 1 has to wait until time $3T + 1$ to transmit its data to s. Since the optimal offline algorithm executed on $G^{t,T}$ terminates in 2 steps if $t > 0$, and 3 steps otherwise, the competitive ratio of $\mathcal{A}$ on $G^{t,T}$ is greater than T.

□

6. Approximation Algorithm

In this section, we present an approximation algorithm for the dynamic MDAT problem. The maximum duration of a solution output by this algorithm reaches the theoretical upper bound given in Section 4. The complexity of our algorithm is better than the one appearing in the preliminary version of this paper [19], as it needs to loop two times over the number of edges of all snapshots between time 0 and time t_f (where t_f is the duration of the found solution), instead of the square of this number for the previous algorithm.

During the first inner *while* loop, the algorithm finds the arrival time of a foremost convergecast tree of the nodes in $remainingNodes$, starting at time t_s. The arrival time, denoted t_f, becomes the starting point to find a collision-free schedule between time t_f and time t_s in a backward manner for the nodes in $remainingNodes$. If some nodes in $remainingNodes$ are not able to transmit between time t_s and time t_f, we start a new iteration and compute the duration of the foremost convergecast tree consecutive to the one found in the previous iteration (its departure time is the arrival time of the previous foremost convergecast tree).

The last *for* loop converts the sequence S that contains the senders over the time to a dynamic data aggregation schedule. With this method, if a node is in $S_{t_1} \cap S_{t_2}$, then only the last transmission is taken into account. The algorithm loops over all the edges, two times, for each snapshot of the dynamic graph between time 0 and the duration of the found solution.

Our algorithm uses Procedure $canTransmit(u, Senders, Receivers)$, that returns `true` if and only if Node u can transmit its data to a node in $Receivers$ without interfering with other nodes in $Senders$.

Lemma 4. *If Algorithm GDAS terminates, the sequence $\{S_i\}_{i \leq t_f}$ is a valid DAS.*

Proof. If GDAS terminates, then the sequence S satisfies $S_0 = V$, because each node that has been removed from $remainingNodes$ line 21 has been added to S_t line 23. Moreover, for each time t, nodes in $S_t \setminus S_{t+1}$ (if not empty) have been added to S_t line 23 after Function $canTransmit$ returned `true`, thus their data are successfully received by nodes in S_{t+1}. This implies $(G, S_t, t) \to (G, S_{t+1}, t+1)$. □

Theorem 5. *If a graph G is in $\mathcal{RC}$, algorithm GDAS finds a valid dynamic data aggregation schedule such that*

$$duration(GDAS(G, s)) \leq FCTD^{n-1}(G, s, 0)$$

Proof. First, we observe that the inner *while* loop simulates a multicast of every node in the network and stops when the sink node receives data from the nodes in $remainingNode$. The duration $t_f - t_s$ is smaller than the duration of a foremost convergecast tree starting at t_s (and it terminates in finite time since the graph is recurrent connected).

We prove that after each iteration of the main *do ... while* loop, the cardinal of the set $remainingNodes$ strictly decreases.

Let t_f^i be the value of t_f at the end of the i-th iteration. Suppose we have already executed the $i-1$ iteration of the main loop. At the beginning of the i-th iteration, we compute in the inner *while* loop the minimum time to aggregate (maybe with collision)

Algorithm GDAS: Greedy Data Aggregation Schedule

Input: MDAT Instance (G, s)

$S \leftarrow$ empty sequence
$remainingNodes \leftarrow V \setminus \{s\}$
$t \leftarrow 0$
do
  $t_s \leftarrow t$
  for $u \in V$ **do**
    $data_t[u] \leftarrow \{u\}$
  while $remainingNodes \not\subset data_t[s]$ **do**
    $data_{t+1} \leftarrow data_t$
    for $(u, v) \in E_t$ **do**
      $data_{t+1}[v] \leftarrow data_t[u] \cup data_t[v]$
      $data_{t+1}[u] \leftarrow data_t[u] \cup data_t[v]$
    $t \leftarrow t + 1$
  $t_f \leftarrow t$
  $marked \leftarrow \{s\}$
  for $t = t_f - 1, \ldots, t_s$ **do**
    $S_t \leftarrow \emptyset$
    $previouslyMarked \leftarrow marked$
    for $v \in \{N_t(u) | u \in previouslyMarked\}$ **do**
      if $canTransmit(v, S_t, marked) \wedge data_{t-1}[v] \cap remainingNodes \neq \emptyset$ **then**
        $remainingNodes \leftarrow remainingNodes \setminus \{v\}$
        $marked \leftarrow marked \cup \{v\}$
        $S_t \leftarrow S_t \cup \{v\}$
  $t \leftarrow t_f$
while $remainingNodes \neq \emptyset$;
for $t = t_f - 1, \ldots, 0$ **do**
  $S_t \leftarrow S_t \cup S_{t+1}$
return S

the data of the nodes in $remainingNodes$. Indeed, for a node u, $v \in data_t[u]$ implies that there exists a journey from v to u ending before time t. Since $data_{t_f^i}[s]$ contains $remainingNodes$, for each node $u \in remainingNode$, there exists a journey from u to s with departure greater than $t_f^{i-1} = t_s^i$ and arrival smaller that t_f^i.

In the *for* loop starting Line 16, we create a collision-free schedule that aggregates the data of at least one node in $remainingNodes$. Indeed, for each time $t \in [t_s^i..t_f^i]$ (in decreasing order), a node is chosen to transmit if the transmission does not create a collision (with the help of the function $canTransmit$) and the node is in a journey from a node in $remainingNodes$ to s (since $data_{t-1}[v] \cap remainingNodes \neq \emptyset$). Since a node can be marked at most once, there exists a time t, when the only nodes (not marked) on a journey from a node in $remainingNodes$ to s are themselves in $remainingNodes$. One of these nodes is chosen to transmit at time t and is removed from the set $remainingNodes$.

This implies that there can be at most $n - 1 = \#(V \setminus \{s\})$ iterations of the main loop. After each iteration, the duration $t_f - t_s$ is smaller than the duration of a foremost convergecast tree starting at t_s. Since for each iteration, the computation restart from the end of the previous one, after i iterations of the main loop, t_f^i is smaller than the duration of i consecutive foremost convergecast trees. Since there can be at most $n - 1$ iterations of the main loop, the duration of the dynamic data aggregation schedule is at most $LTJD^{n-1}(G, s, 0)$. □

If the graph is T-time-bounded recurrent connected, then a FCT duration is smaller than T. Thus, we can derive an absolute bound and the following approximation factor for the dynamic MDAT problem.

Corollary 1. *Algorithm GDAS, in $\mathcal{BRC}$ with bound T, satisfies:*

$$duration(GDAS(G; s)) \leq T(n - 1)$$

Thus, it is an approximation of factor $T(n - 1)$, for the dynamic MDAT problem.

7. Conclusion

We studied the complexity of the minimum data aggregation time problem in wireless sensor networks. We proved that the problem is NP-complete in a static WSN of degree at most three, and NP-complete in a dynamic WSN of degree at most two. The degree constraint is crucial, as a smaller one induces a trivial solution in both cases. Then we gave tight lower and upper bounds for the minimum data aggregation time problem in dynamic networks and the first approximation scheme for the problem. Also, in a dynamic graph with n nodes of degree at most Δ, we conjecture a more accurate upper bound of l time-independent foremost convergecast trees (with $l = (\Delta - 1) \log_\Delta (n (\Delta - 1) + 1) - \Delta + 2$).

Finally we observed that only a centralized algorithm with full knowledge can compute the optimal solution of the problem. Thus, we gave a simple approximate algorithm giving a solution whose time matches the theoretical upper bound.

One can observe that allowing nodes to transmit several times their data, instead of only once, does not change the results concerning centralized algorithms that are aware of the future. Indeed, a schedule that contains multiple transmissions by node can be converted to a schedule where each node transmits only once, by keeping only the last transmission. Moreover, we also believe that the impossibility results concerning

distributed algorithm, and online algorithms, still hold even if a constant number of transmissions are allowed by node.

References

[1] B. N. Clark, C. J. Colbourn, D. S. Johnson, Unit disk graphs, Discrete Mathematics 86 (1) (1990) 165 – 177.

[2] X. Chen, X. Hu, J. Zhu, Minimum data aggregation time problem in wireless sensor networks, in: Mobile Ad-hoc and Sensor Networks, Springer, 2005, pp. 133–142.

[3] V. Annamalai, S. K. Gupta, L. Schwiebert, On tree-based convergecasting in wireless sensor networks, in: Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE, Vol. 3, IEEE, 2003, pp. 1942–1947.

[4] B. Yu, J. Li, Y. Li, Distributed data aggregation scheduling in wireless sensor networks, in: INFOCOM 2009, IEEE, IEEE, 2009, pp. 2159–2167.

[5] X. Xu, M. Li, X. Mao, S. Tang, S. Wang, A delay-efficient algorithm for data aggregation in multihop wireless sensor networks, Parallel and Distributed Systems, IEEE Transactions on 22 (1) (2011) 163–175.

[6] M. Ren, L. Guo, J. Li, A new scheduling algorithm for reducing data aggregation latency in wireless sensor networks., International Journal of Communications, Network & System Sciences 3 (8).

[7] T. D. Nguyen, V. Zalyubovskiy, H. Choo, Efficient time latency of data aggregation based on neighboring dominators in wsns, in: Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, IEEE, 2011, pp. 1–6.

[8] E. Fasolo, M. Rossi, J. Widmer, M. Zorzi, In-network aggregation techniques for wireless sensor networks: a survey, Wireless Communications, IEEE 14 (2) (2007) 70–87.

[9] A. Casteigts, S. Chaumette, A. Ferreira, Characterizing topological assumptions of distributed algorithms in dynamic networks, in: Structural Information and Communication Complexity, Springer, 2010, pp. 126–140.

[10] A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro, Time-varying graphs and dynamic networks, in: Ad-hoc, Mobile, and Wireless Networks, Springer, 2011, pp. 346–359.

[11] F. Kuhn, R. Oshman, Dynamic networks: Models and algorithms, SIGACT News 42 (1) (2011) 82–96.

[12] A. Casteigts, P. Flocchini, B. Mans, N. Santoro, Shortest, fastest, and foremost broadcast in dynamic networks, Int. Journal of Foundations of Computer Science 26 (4) (2015) 499–522.

[13] A. Casteigts, P. Flocchini, B. Mans, N. Santoro, Building fastest broadcast trees in periodically-varying graphs, arXiv preprint arXiv:1204.3058.

[14] A. Clementi, R. Silvestri, L. Trevisan, Information spreading in dynamic graphs, in: Proceedings of the 2012 ACM symposium on Principles of distributed computing, ACM, 2012, pp. 37–46.

[15] S. Abshoff, F. Meyer auf der Heide, Structural information and communication complexity: 21st international colloquium, sirocco 2014, takayama, japan, july 23-25, 2014. proceedings, Springer International Publishing, Cham, 2014, pp. 194–209.

[16] A. Cornejo, S. Gilbert, C. Newport, Aggregation in dynamic networks, in: Proceedings of the 2012 ACM symposium on Principles of distributed computing, ACM, 2012, pp. 195–204.

[17] T. Biedl, G. Kant, A better heuristic for orthogonal graph drawings, Computational Geometry 9 (3) (1998) 159–180.

[18] D. Lichtenstein, Planar formulae and their uses, SIAM journal on computing 11 (2) (1982) 329–343.

[19] Q. Bramas, S. Tixeuil, The complexity of data aggregation in static and dynamic wireless sensor networks, in: Stabilization, Safety, and Security of Distributed Systems, Vol. 9212 of Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 36–50.