2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C) | 978-1-6654-4393-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ACSOS-C52956.2021.00036

Effect of Monotonic Filtering on Graph Collection Dynamics

Hunza Zainab, Giorgio Audrito, Soura Dasgupta and Jacob Beal

***Abstract*—Distributed data collection is a fundamental task in open systems. In such networks, data is aggregated across a network to produce a single aggregated result at a source device. Though self-stabilizing, algorithms performing data collection can produce large overestimates of aggregates in the transient phase. For example, in [1] we demonstrated that in a line graph, a switch of sources after initial stabilization may produce overestimates that are quadratic in the network diameter. We also proposed monotonic filtering as a strategy for removing such large overestimates. Monotonic filtering prevents the transfer of data from device A to device B unless the distance estimate at A is more than that at B at the previous iteration. For a line graph, [1] shows that monotonic filtering prevents quadratic overestimates. This paper analyzes monotonic filtering for an arbitrary graph topology, showing that for an N device network, the largest overestimate after switching sources is at most $2N$.**

***Index Terms*—edge computing, data aggregation, self-stabilization**

I. Introduction

This paper proposes and analyzes a strategy called *monotonic filtering*, for removing large overestimates in distributed data collection. Recent years have witnessed a proliferation of complex networked open systems comprising a plethora of heterogeneous devices like drones, smartphones, IoT devices, and robots. These systems mandate the formulation of new strategies of collective adaptation, with the ultimate goal of transforming these environments into a *pervasive computing fabric* where sensing, actuation, and computation are resilient and distributed across space [2]. The focus of this paper is on resilient *distributed sensing*, which could be of physical environmental properties or of digital or virtual characteristics of computing resources. By cooperation between physically proximate, interacting sets of mobile entities, distributed sensing can support complex situation recognition [3], monitoring [2], and observation and control of swarms of agents [4].

A defining coordination task in distributed sensing is data summarization from devices in a region. From this, one can perform many other operations like count, integrate, average, and maximize. Data summarization is like the *reduce* phase of MapReduce [5]. It is extended to agents communicating through their neighbors and spread across a region, e.g., in wireless sensor networks [6]. A common implementation of data summarization is by *distributed collection*, where information moves towards collector devices and aggregates *en route* to produce a unique result. Such self-organizing behavior (referred to as a "C" block in [7]), is a fundamental and widely used component of collective adaptive systems (CASs). It can be instantiated for values of any data type with a commutative and associative aggregation operator, and can be applied in many diverse contexts.

Several papers have characterized the dynamics of data summarization algorithms [7], [8] and on improving such dynamics [9]–[11]. These papers all show that, though self-stabilizing, these algorithms can give rise to large transient overestimates with potentially negative consequences. For example, if the goal is collect the net resources in a network of devices, then overestimates, however fleeting, may cause a leader to commit to more tasks than the network can perform.

For the example of a line graph, we showed in [1] that collection can give overestimates that are quadratic in the network diameter. This is observed in face of a particular source switch after stabilization has occurred. We presented the notion of monotonic filtering as a potential amelioration. This technique prevents collection across devices whose distance towards a source or collector device is decreasing. With line graphs, [1] showed that monotonic filtering prevents quadratic overestimates. In this paper, we analyze monotonic filtering for general graphs and the demonstrate that there are no quadratic overestimates in collection during the transient phase following a source switch. Rather, for an N-device network the largest overestimate is at most $2N$.

Section II gives preliminaries and Section III provides

Zainab and Dasgupta are with the Department of Electrical and Computer Engineering, University of Iowa, Iowa City, IA, USA, hunza-zainab@uiowa.edu, soura-dasgupta@uiowa.edu. Audrito is with the Computer Science Department and C3S, University of Torino, Torino, Italy, giorgio.audrito@unito.it. Beal is with Raytheon BBN Technologies, Cambridge, MA, USA, jakebeal@ieee.org Supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0049. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. Approved for public release, distribution unlimited (DARPA DISTAR case 34494, April 19, 2021). Dasgupta also has an appointment with the Shandong Adademy of Sciences, China.

results on distance estimates that are used to execute mononónic filtering. Section II-C defines monotonic filtering. Section IV, proves the main result. Section V gives simulations. Section VI concludes. Due to space constraints most proofs are in a version on arXiv, [12].

II. Preliminaries

We model a network of devices as an undirected graph $G = (V, E)$, with $V = \{0, \cdots, N-1\}$ being the set of devices that are the nodes in the graph and E being the edge set of connections between devices. We assume that device $i \in V$ carries the value v_i, and that there is a designated *source* set of nodes $S(t) \subset V$. The goal is to aggregate the accumulation of the values v_i at the source set, i.e., generate accumulates a_i such that

$$\sum_{i \in S} a_i = \sum_{i \in V} v_i.$$

In particular, if there is only one source, then its accumulate should be the sum of all of the v_i. This is a special case of the data collection block (C-block) of [7].

A. The Basic Approach

There are many ways to achieve this objective depending on the circumstances [10], [11], [13], [14]. In this paper, we adaptively determine a spanning tree and accumulate values from children to parents. The spanning tree is determined by the *Adaptive Bellman-Ford* (ABF) algorithm [7], [15] a special case of what is called a G-block in [7]; ABF estimates distances. Two nodes are neighbors if they share an edge. Define edge length between any two neighbors to be 1 and $\mathcal{N}(i)$ to be the set of neighbors of i. Then with $\hat{d}_i(t)$ the distance and S the source set, ABF proceeds as

$$\hat{d}_i(t) = \begin{cases} 0 & \textit{if } i \in S(t) \\ \min_{j \in \mathcal{N}(i)}\{\hat{d}_j(t-1) + 1\} & \text{otherwise} \end{cases}. \tag{1}$$

The minimizing j in the second bullet is called a current constraining or simply, constraining node, of i. More precisely the constraining node $c_i(t)$ obeys:

$$c_i(t) = \begin{cases} i & \textit{if } i \in S(t) \\ \underset{j \in \mathcal{N}(i)}{\text{argmin}}\{\hat{d}_j(t-1) + e_{ji}\} & \text{otherwise} \end{cases}. \tag{2}$$

These constraining nodes set up the spanning tree: The set of children of i are the nodes they constrain:

$$C_i(t) = \{j |\ c_j(t-1) = i\}. \tag{3}$$

Then in keeping with the strategy outlined at the beginning of this section, we can update the accumulate at node i through the recursion:

$$a_i(t) = \sum_{j \in C_i(t)} a_j(t-1) + v_i. \tag{4}$$

B. Quadratic Overestimates

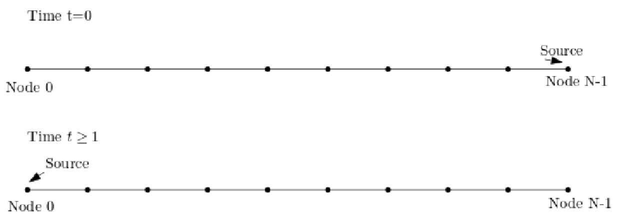

Figure 1. Representation of an N-node line graph (N-line) with a source switch from time $t = 0$ to $t = 1$.

From [15], one knows that ABF is self-stabilizing. This means in particular that both $c_i(t)$ and $C_i(t)$ must acquire steady state values and thus the recursion (4) must also converge. For example, consider the line graph in Figure 1. Suppose the source is the rightmost node with index $N-1$, and the nodes to the left of 0 are indexed in sequence as $1, \cdots, N-1$. Assume that

$$v_i = 1 \ \forall \ i \in V, \tag{5}$$

i.e., all values are 1 and distances are hop lengths. Then at steady state one has

$$\hat{d}_i(t) = N - 1 - i \text{ and } a_i = i + 1. \tag{6}$$

Now suppose these values have been acquired at $t = 0$, but the source switches from $N-1$ to zero at $t = 1$. Then the following theorem from [1] shows that *en route* to self-stabilization $a_i(t)$ suffer from quadratic overestimates in the transient phase.

Theorem 1. *Consider the line graph in Figure 1, with (5) in force. Suppose at $t = 0$ $a_i(0)$ and $\hat{d}_i(0)$ are as in (6). Suppose for all $t \geq 1$, $S(t) = \{0\}$. Then under (1) and (4), the maximum partial accumulate $a_i(t)$ reached by the source is obtained at time $t = 2N-2$ and is:*

$$a_0(2N-2) = \left\lceil \frac{N-1}{2} \right\rceil N + N - 1 \geq \frac{N(N+1)}{2} - 1$$

before reaching the correct value at time $t = 2N-1$ i.e.,

$$a_0(2N-1) = N \tag{7}$$

C. Monotonic Filtering

Monotonic filtering was proposed in [1] as a remedy for such quadratic overestimation. Specifically, this device permits only a subset of the nodes that i constrains to be a valid children: only those $c_i(t)$ whose distance estimate in the previous iteration was one more than $\hat{d}_i(t)$ are allowed to be children. Thus the set of children in (3) is replaced by

$$C_i(t) = \{j | i = c_j(t-1) \wedge \hat{d}_j(t-1) = \hat{d}_i(t) + 1\}. \tag{8}$$

Neither the definition of constraining nodes, nor the underlying accumulation equation changes. The latter in particular remains as (4). It should be noted that the self-stabilizing nature of ABF, per [15], ensures that all distance estimates converge to their correct values. In such a steady state, under (5), every node that constrains another automatically satisfies the restriction on distance estimates given in (8). As proved in [1], this additional restriction is all that is needed to remove any overestimate from happening in a line graph. In this paper we show that this amelioration persists for general undirected graphs. The next subsection describes the analytical framework.

D. Definitions and Assumptions

We now make some definitions to set up the assumptions that underlie our analysis of monotonic filtering. Suppose d_i is the hop count of i from the source set S. Then from Bellman's Principle of Optimality it obeys

$$\begin{cases} d_i = 0 & i \in S \\ \min_{j\in\mathcal{N}(i)} d_j + 1 & \text{otherwise} \end{cases}. \tag{9}$$

This leads to the definition of a *true constraining node.*

Definition 1 (True constraining node)**.** A k that minimizes the right hand side of (9) is a true constraining node of i. As there may be two neighbors j and k of i such that $d_j = d_k$, a node may have multiple true constraining nodes while the true constraining node of a source is itself.

We now define the notion of *effective diameter* introduced in [15].

Definition 2 (Effective Diameter)**.** Consider a sequence of nodes in a graph such that each node is a true constraining node of its successor. The effective diameter D is defined as the longest length such a sequence can have in the graph.

Thus, if (5) holds then the effective diameter is

$$1 + \max_{i\in V} d_i.$$

We also define some important sets that are critical for our analysis.

Definition 3. Define d_{ij} to be the minimum distance between nodes i and j. Further define $\mathcal{F}_k(m)$ to be the set of nodes whose minimum distance from node k is m, i.e.,

$$\mathcal{F}_k(m) = \{i \mid d_{ik} = m\}.$$

If a graph $G = (V, E)$ has an effective diameter D with a single source node 0, then as depicted in Figure 2 there is a sequence of nodes, without loss of generality $i = 0, 1, \ldots D-1$, such that node i is the true constraining node of node $i+1$. Henceforth we assume the graph is as depicted in this figure. In fact, the following assumption holds.

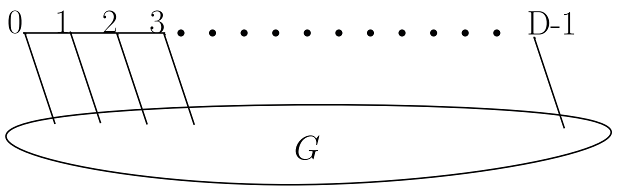

Figure 2. Arbitrary graph G with a sequence highlighted.

Assumption 1. The graph $G = (V, E)$, is as in Figure 2. Further (5) holds and the shortest path from 0 to $D-1$ has D hops. The algorithm used is defined by (1, 2, 8). For $t \le 0$, $D-1$ is the source, and for $t > 0$, 0 is the source. Further $\hat{d}_i(0)$ and $a_i(0)$ are steady state values of the algorithm assuming $D-1$ is the source. In particular,

$$\hat{d}_i(0) = m, \ \forall\ i \in \mathcal{F}_{D-1}(m). \tag{10}$$

Also, for every integer m

$$\sum_{i\in\mathcal{F}_{D-1}(m)} a_i(0) \le N. \tag{11}$$

Further, the effective diameter when 0 is the only source is D; when $D-1$ is the source, the effective diameter is D_0.

Observe that $D_0 \ge D$. Also note that (11) is a trivial consequence of the steady state values generated by (4) and the fact that for all i, $v_i = 1$.

III. Evolution of distance estimates

To understand the behavior of the accumulates after the source switch, one must first understand how distance estimates evolve subsequent to the switch. To this end we have the following preparatory lemma describing the true distances of neighbors from the old and new source, using Assumption 1, which ensures that all distances are hop lengths.

Lemma 1. *Suppose Assumption 1 holds, $D > 2$, and $\mathcal{F}_k(m)$ are as in Definition 3. (i) Then for all $k \in V$, $0 < m \le D-1$ and $i \in \mathcal{F}_k(m)$ all neighbors of i are in $\mathcal{F}_k(m-1) \bigcup \mathcal{F}_k(m) \bigcup \mathcal{F}_0(m+1)$. If $\mathcal{F}_k(m) \ne \varnothing$, i has at least one neighbor in $\mathcal{F}_0(m-1)$. (ii) Moreover, 0 and $D-1$ are not neighbors.*

We now provide a lemma that describes the evolution of distance estimates for $t > 0$. To this end, we partition V into three sets $\mathcal{I}_i(t)$ for $i = 0, 1, 2$. $\mathcal{I}_0(t)$ comprises nodes that are less than t hops away from 0. In particular, the distance estimates at these nodes have converged to their true distances from 0. The set $\mathcal{I}_1(t)$ comprises nodes that are not in $I_0(t)$ but have felt the effect of a change in source. The set $\mathcal{I}_2(t)$ consists of the remaining nodes, whose distance estimates and accumulates are identical to what they were at $t = 0$.

Lemma 2. *Suppose Assumption 1 holds and $D > 1$. For all $t \geq 1$ consider the following partitioning of V:*

$$\mathcal{I}_0(t) = \bigcup_{m=0}^{t-1} \mathcal{F}_0(m), \tag{12}$$

$$\mathcal{I}_1(t) = \left(\bigcup_{m=0}^{t-1} \mathcal{F}_{D-1}(m) \right) \setminus \mathcal{I}_0(t) \tag{13}$$

and

$$\mathcal{I}_2(t) = V \setminus \{\mathcal{I}_0(t) \cup \mathcal{I}_1(t)\}, \tag{14}$$

of V. Then under (1,2), the following hold.

$$\hat{d}_i(t) = m \;\forall\; i \in \mathcal{F}_0(m) \text{ and } 0 \leq m < t, \tag{15}$$

$$\hat{d}_i(t) = m \;\forall\; i \in \mathcal{F}_{D-1}(m) \bigcap \mathcal{I}_2(t). \tag{16}$$

and

$$\hat{d}_i(t) \in \{t, t+1\} \;\forall\; i \in \mathcal{I}_1(t). \tag{17}$$

IV. Accumulates Under Monotonic Filtering

We now compute the partial accumulates under the new definition of children given in (8), using the distance estimates characterized in Lemma 2, and the evolution of accumulates defined in (4). The added constraint in (8), which intuitively ensures that data is collected by always descending distances, is satisfied by every node in a stable state; however, it may not be satisfied during transients. We will show that monotonic filtering suffices to eliminate quadratic overestimates in a general graph.

We consider the partial accumulates in each of our partitioned sets individually. The following lemma characterizes the accumulates in $\mathcal{I}_1(t)$.

Lemma 3. *Under Assumption 1, the partial accumulates in $\mathcal{I}_1(t)$ defined in (13), obey*

$$a_i(t) = 1 \;\forall\; i \in \mathcal{I}_1(t) \tag{18}$$

To characterize the accumulates of nodes in $\mathcal{I}_0(t)$, we need information on how accumulates evolve in the neighboring nodes that are part of the set $\mathcal{I}_2(t)$. The following lemma gives an upper bound on that.

Lemma 4. *Under Assumption 1, nodes $j \in C_i(t) \bigcap \mathcal{I}_2(t)$ for all $i \in \mathcal{I}_0(t)$ obey:*

$$\sum_{i \in \mathcal{I}_0(t)} \sum_{j \in C_i(t) \bigcap \mathcal{I}_2(t)} a_j(t-1) \leq N \tag{19}$$

Given the information regarding neighbors of $\mathcal{I}_0(t)$ in other sets, we can now characterize the accumulates in $\mathcal{I}_0(t)$ according to the following lemma.

Lemma 5. *Under Assumption 1, the partial accumulates at nodes in the set $\mathcal{I}_0(t)$ obey*

$$a_i(t) \leq 2N \;\forall\; i \in \mathcal{I}_0(t) \tag{20}$$

Together, the previous lemmas imply that the graph does not have overestimates above $2N$ during convergence of the collection algorithm. We now have the main result.

Theorem 2. *In a general graph, with monotonic filtering the partial accumulations $a_i(t)$ in sets $\mathcal{I}_0(t)$, $\mathcal{I}_1(t)$ and $\mathcal{I}_2(t)$ for any time t, have an upper bound that is given as:*

$$a_i(t) \leq 2N \;\forall\; i \in \mathcal{I}_0(t) \tag{21}$$

$$a_i(t) = 1 \;\forall\; i \in \mathcal{I}_1(t) \tag{22}$$

and

$$a_i(t) \leq N \;\forall\; i \in \mathcal{I}_2(t) \tag{23}$$

Proof. The first two expressions follow directly from lemma 5 and 3 respectively.

The proof of (23) is trivial as nodes in the set $\mathcal{I}_2(t)$ have not been affected yet by the source switch. They maintain the converged accumulates they had at the previous steady state, and these values never exceed N, the total number of nodes in the graph. □

V. Simulation

In order to evaluate the performance of monotonic filtering in a more general setting, we simulated a network of 100 to 1000 nodes, randomly displaced in a square, with a connection range such that the average number of neighbours per node is about 10. We performed 1000 simulations and averaged the results. The simulation is publicly available online.[1]

A sample screenshot of the simulation is depicted in Figure 3, while synthetic plots of the average collection results in source nodes are given in Figure 4 (after bibliography). When the position of the nodes is fixed (*speed* is zero), monotonic filtering can prevent any overestimate from occurring. The estimate still converges to the correct result as fast as with the basic approach, which instead suffers from high peaks during reconfiguration. When nodes are steadily moving (*speed* is positive), both algorithms start underestimating the true count as the speed increases and number of devices increase. However, the underestimates are much more pronounced with monotonic filtering, while the basic approach is able to tolerate small speeds.

[1]https://github.com/Harniver/monotonic-filtering-dynamics

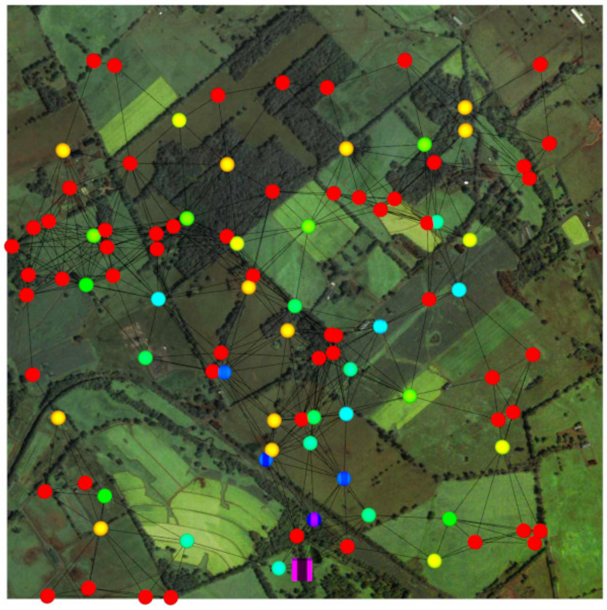

Figure 3. Simulation of collection algorithms on a random bidimensional arrangement. Colors of nodes are tuned from red (estimate 1) to magenta (correct estimate) to black (infinitely large estimate); and the central color corresponds to the basic approach while the color of the sides corresponds to monotonic filtering. The screenshot is taken during a transient recovery, in which the source (big square) has a very large overestimate with the basic approach (black central color) while it is almost correct with monotonic filtering (magenta sides).

VI. Conclusion

In this paper, we investigated the effect of a monotonic filtering condition on the transient values of a single-path collection algorithm during recovery from a source switch. In a static graph, the monotonic filtering condition is proved to bound overestimates to at most $2N$, while single-path collection without it is showed to reach quadratic overestimates in some cases. By evaluating the algorithms in simulation, we show that in practice transient large overestimates do occur without filtering, while no overestimate at all is present with monotonic filtering. Finally, we also simulate the behavior of the algorithms under persistent perturbations, i.e., steady movement of the nodes. In this scenario, both algorithms degrade their quality towards underestimates; however, with monotonic filtering the degradation occurs much sooner, with lower movement speeds and lower number of devices.

In future work, monotonic filtering should be compared with existing strategies [16], in its ability to avoid overestimates under persistent perturbations.

References

[1] H. Zainab, G. Audrito, S. Dasgupta, and J. Beal, "Improving collection dynamics by monotonic filtering," *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, 2020.

[2] N. Bicocchi, M. Mamei, and F. Zambonelli, "Self-organizing virtual macro sensors," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 7, no. 1, pp. 2:1–2:28, 2012.

[3] J. Coutaz, J. L. Crowley, S. Dobson, and D. Garlan, "Context is key," *Communications of the ACM*, vol. 48, no. 3, pp. 49–53, 2005.

[4] M. Viroli, D. Pianini, A. Ricci, and A. Croatti, "Aggregate plans for multiagent systems," *International Journal of Agent-Oriented Software Engineering*, vol. 4, no. 5, pp. 336–365, 2017.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[6] A. K. Talele, S. G. Patil, and N. B. Chopade, "A survey on data routing and aggregation techniques for wireless sensor networks," in *International Conference on Pervasive Computing (ICPC)*. IEEE, 2015, pp. 1–5.

[7] M. Viroli, G. Audrito, J. Beal, F. Damiani, and D. Pianini, "Engineering resilient collective adaptive systems by self-stabilisation," *ACM Transactions on Modeling and Computer Simulation*, vol. 28, no. 2, pp. 16:1–16:28, 2018.

[8] Y. Mo, J. Beal, and S. Dasgupta, "Error in self-stabilizing spanning-tree estimation of collective state," in *2nd IEEE International Workshops on Foundations and Applications of Self* Systems (FAS*W)*. IEEE Computer Society, 2017, pp. 1–6.

[9] G. Audrito and S. Bergamini, "Resilient blocks for summarising distributed data," in *1st Workshop on Architectures, Languages and Paradigms for IoT (ALP4IoT)*, ser. EPTCS, vol. 264, 2017, pp. 23–26.

[10] G. Audrito, S. Bergamini, F. Damiani, and M. Viroli, "Effective collective summarisation of distributed data in mobile multi-agent systems," in *18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1618–1626. [Online]. Available: http://dl.acm.org/citation.cfm?id=3331882

[11] ——, "Resilient distributed collection through information speed thresholds," in *22th International Conference on Coordination Models and Languages (COORDINATION)*, ser. Lecture Notes in Computer Science. Springer, 2020, to appear.

[12] H. Zainab, G. Audrito, S. Dasgupta, and J. Beal, "Monotonic filtering for distributed collection," Jul 2021. [Online]. Available: https://arxiv.org/abs/2107.05791

[13] Q. Liu, A. Pruteanu, and S. Dulman, "Gradient-based distance estimation for spatial computers," *Comput. J.*, vol. 56, no. 12, pp. 1469–1499, 2013.

[14] G. Audrito, F. Damiani, and M. Viroli, "Optimal single-path information propagation in gradient-based algorithms," *Science of Computer Programming*, vol. 166, pp. 146–166, 2018.

[15] Y. Mo, S. Dasgupta, and J. Beal, "Robustness of the adaptive bellman -ford algorithm: Global stability and ultimate bounds," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 4121–4136, 2019.

[16] G. Audrito, S. Bergamini, F. Damiani, and M. Viroli, "Resilient distributed collection through information speed thresholds," in *Coordination Models and Languages*, ser. Lecture Notes in Computer Science, vol. 12134. Springer, 2020, pp. 211–229.

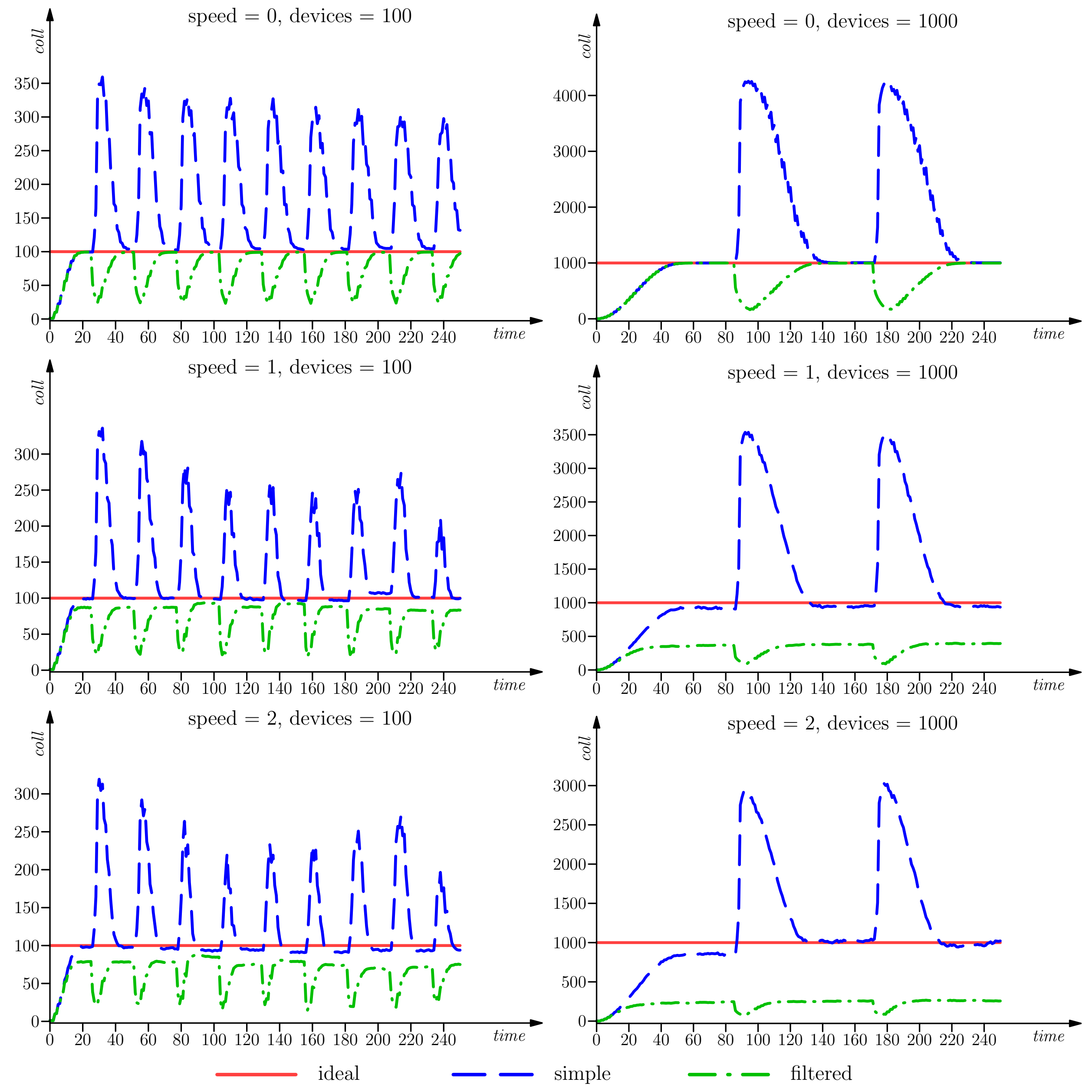

Figure 4. Average collection results (counting device number) in source nodes over simulated time, where the source is periodically switching to different devices (at times matching the periodic spikes). The lines correspond to the number of devices (*ideal*, in red), to the basic approach (*simple*, in blue), and to monotonic filtering (*filtered*, in green). The basic approach suffers from significant overestimates at every source change, while monotonic filtering provides underestimates instead. In mobile networks (*speed* greater than zero) with a sufficient number of devices, the underestimates given by monotonic filtering become systematic, while the basic approach is still able to reach a reasonably correct value.