

Brief Announcement: An Early-stopping Protocol for Computing Aggregate Functions in Sensor Networks*

Antonio Fernández Anta¹, Miguel A. Mosteiro^{1,2}, and Christopher Thraves³

¹ LADyR, GSyC, Universidad Rey Juan Carlos, 28933 Móstoles, Madrid, Spain

² Department of Computer Science, Rutgers University, Piscataway, NJ 08854, USA

³ Université Bordeaux I, LaBRI, domaine Universitaire, 33405 Talence, France

Introduction. Nodes in a Sensor Network can collaborate to process the sensed data but, due to unreliability, a monitoring strategy can not rely on individual sensors values. Instead, the network should use aggregated information from groups of sensor nodes [2, 3, 7]. The topic of this work is the efficient computation of aggregate functions in the highly constrained Sensor Network setting, where node restrictions are modeled as in [4], the random node-deployment is modeled as a geometric graph, and the resulting topology, node identifiers assignment and the assignment of input-values to be aggregated is adversarial.

While algebraic aggregate functions are well defined, the implementation of such computations in Sensor Networks has to deal with various issues that make even the definition of the problem difficult. First, the input-values at each node might change over time. Therefore, it is necessary to fix to which time step correspond those input-values. Second, arbitrary node failures make the design of protocols challenging. It has been shown [1] that the problem of computing an aggregate function among all nodes in a network where some nodes join and leave the network arbitrarily in time is intractable.

Results. The protocol proposed interleaves two algorithms, one following a tree-based approach and one following a mass-distribution approach. The tree-based algorithm will provide the correct result with low time and energy complexity in a failure-free setting. If the presence of failures prevents the tree-based computation from finishing, the mass-distribution algorithm will compute and disseminate an approximation of the result. The time taken by this algorithm is larger, but it is only incurred in presence of failures. Hence, the combined algorithm is *early stopping*.

The efficiency is measured in two dimensions: time and number of transmissions. These metrics are strongly influenced by collisions, especially because no collision detection mechanisms are available in this setting. In order to reduce collisions and energy consumption, a two-level hierarchy of nodes is used. The actual computation is done by a small set of nodes, called *delegate nodes*, that

* A full version of this work is available at [5]. This research was supported in part by Comunidad de Madrid grant S-0505/TIC/0285; Spanish MEC grant TIN2005-09198-C02-01; EU Marie Curie European Reintegration Grant IRG 210021; NSF grants CCF0621425, CCF 05414009, CCF 0632838; and French ANR Masse de Données project ALPAGE.

collect the sensed input-values from the non-computing nodes, called *slug nodes*. This structure has several advantages. First, collisions are reduced since they can only occur while the delegate nodes collect the sensed input-values from the slug nodes. After that, delegate nodes are able to communicate in a collision-free fashion. Second, energy is saved because the slug nodes can idle during the computation. Third, the subnetwork of delegate nodes has constant degree, which allows to easily build a constant-degree spanning tree. Finally, since the set of delegate nodes is small, there is a smaller probability that the tree-based algorithm will fail (since only failures of delegate nodes impact on it). Notice that, in presence of failures, the two-level structure may have to be reconstructed; fortunately, this can be done fast and locally.

The main contribution of this work is the presentation of a time-optimal early-stopping protocol that computes the average function in Sensor Networks under a harsh model of sensor restrictions⁴. More precisely, it is shown that, in a failure-free setting, with high probability, this protocol returns the exact value and terminates in $O(D + \Delta)$ steps, which is also shown to be optimal, and the overall number of transmissions is in $O(n(\log n + \Delta/\log n + \log \Delta))$ in expectation. On the other hand, in presence of failures, the protocol computes the average of the input-values of a subset of nodes that depends on the failure model. More precisely, it is shown that, after the last node fails and w.h.p., the protocol takes an extra additive factor of $O(\log(n/\varepsilon)/\Phi^2)$ in time and an extra additive factor of $O(n \log(1/\varepsilon)/\Phi^2)$ in the expected number of transmissions, where $\varepsilon > 0$ is the maximum relative error, and Φ is the conductance [6] of the network of delegates.

References

1. M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating aggregates on a peer-to-peer network. Technical report, Stanford Univ., Database group, 2003.
2. S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking*, 14(SI):2508–2530, 2006.
3. J.-Y. Chen, G. Pandurangan, and D. Xu. Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis. In *Proc. of the 4th Intl. Symp. on Information Processing in Sensor Networks*, page 46, 2005.
4. M. Farach-Colton, R. J. Fernandes, and M. A. Mosteiro. Bootstrapping a hop-optimal network in the WSM. *ACM Trans. on Algorithms*, 2008. In press.
5. A. Fernández Anta, M. A. Mosteiro, and C. Thraves. An early-stopping protocol for computing aggregate functions in sensor networks. Technical Report 3, LADyR, GSyC, Universidad Rey Juan Carlos, May 2008.
6. M. Jerrum and A. Sinclair. Conductance and the rapid mixing property for markov chains: the approximation of permanent resolved. In *Proc. of the 20th Ann. ACM Symp. on Theory of Computing*, pages 235–244, 1988.
7. D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proc. of the 44th IEEE Ann. Symp. on Foundations of Computer Science*, pages 482–491, 2003.

⁴ Other aggregate functions can be computed using a protocol for average without extra cost as described in [3, 7].