A Study on Aggregation by Averaging Algorithms

[Poster Abstract]

Paulo Jesus
MSc Student
University of Minho
Portugal
paulo.c.o.jesus@gmail.com

Carlos Baquero
University of Minho
Portugal
cbm@di.uminho.pt

Paulo Sérgio Almeida
University of Minho
Portugal
psa@di.uminho.pt

1. ABSTRACT

With the advent of multihop ad-hoc networks, sensor networks and large scale overlay networks, there is a demand for tools that can abstract meaningful system properties from given assemblies of nodes. Distributed aggregation algorithms allow the evaluation of properties such as: network size; total storage capacity; average load; and majorities.

A useful class of aggregation algorithms is based on averaging techniques. Such algorithms start from a set of input values spread across the nodes, and iteratively average the values in neighbour nodes that are able to communicate. Eventually all nodes will converge to the same value and can estimate some useful metric. For instance, if one node starts with input 1 and all other nodes with input 0, eventually all nodes will end up with the same value v and the aggregate property *network size* can be estimated by the fraction $1/v$.

In this evaluation we will consider three representative averaging algorithms and compare their behaviour in a common simulation environment. The selected algorithms are *push-sum* [3], *push-pull* [2] and *DRG* [1]. To our knowledge this the first extensive comparison of averaging algorithms.

Previous analysis by simulation have considered a sequence of rounds where, in each round, pairs of nodes average their values atomically. We will show that such oversimplified simulations mask a latent problem in the *push-pull* algorithm. When pairwise averaging is more accurately simulated by decomposing the interactions into sequences of message exchanges, potential interleaving of executions is exposed. We observed that *push-pull* can loose "mass" in some interleavings (violating the invariant that the sum of all values in the system should remain constant). This makes the *push-pull* algorithm converge to a wrong value. Interestingly, this problem does not occur in the *push-sum* algorithm that preceded it.

The corrections to this problem are fairly easy but introduce additional delays in the convergence speed, since atomicity is restored at the expense of some waiting or of canceled exchanges. We propose two potential solutions (*push-pull-ordered-wait* and *push-pull-back-cancellation*) and show how these improved variants of *push-pull* compare with the two other algorithms.

Our evaluation will consider two topologies: a fixed random network and a mesh network. We choose a random network, where each node is connected to a random small set of neighbours; we consider this to be more realistic than using the assumption that, for each message exchange, each node chooses a random node in the whole network to communicate with; here the chosen node is restricted to the set of neighbours to which links are available. The mesh network simulation is intended to evaluate how these algorithms behave in a geographically restricted connectivity scenario (i.e. proximity based).

Our analysis shows convergence speed as a function of both the number of rounds (see Figure 1) and the number of message transmissions. This gives an idea of how convergence evolves with respect to time and energy expenditure.

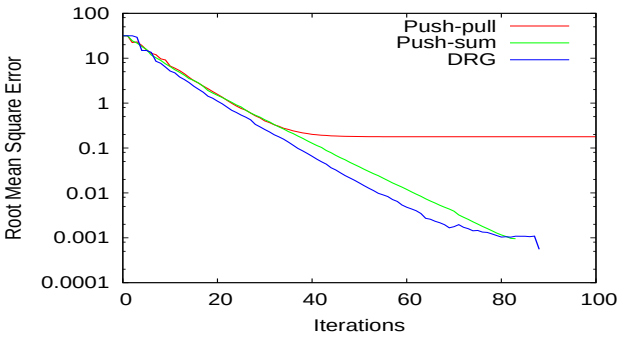

Figure 1: Convergence speed by number of rounds (random network).

2. REFERENCES

[1] J.-Y. Chen, G. Pandurangan, and D. Xu. Robust computation of aggregates in wireless sensor networks: Distributed randomized algorithms and analysis. *IEEE Transactions on Parallel and Distributed Systems*, 17(9):987–1000, 2006.

[2] M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3):219–252, 2005.

[3] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, page 482, Washington, DC, USA, 2003. IEEE Computer Society.