Synchronizers, Spanners (1985; Awerbuch)

Michael Elkin, Ben-Gurion University of the Negev, www.cs.bgu.ac.il/˜elkinm

SYNONYMS: Network synchronization, low-stretch spanning subgraphs.

1 PROBLEM DEFINITION

Consider a communication network, modeled by an n-vertex undirected unweighted graph $G = (V, E)$, for some positive integer n. Each vertex of G hosts a processor of unlimited computational power; the vertices have unique identity numbers, and they communicate via the edges of G by sending messages of size $O(\log n)$ each.

In the *synchronous* setting the communication occurs in discrete *rounds*, and a message sent in the beginning of a round R arrives at its destination before the round R ends. In the *asynchronous* setting each vertex maintains its own clock, and clocks of distinct vertices may disagree. It is assumed that each message sent (in the asynchronous detting) arrives at its destination within a certain time τ after it was sent, but the value of τ is not known to the processors.

It is generally much easier to devise algorithms that apply to the synchronous setting (henceforth, synchronous algorithms) rather than to the asynchronous one (henceforth, asynchronous algorithms). In [1] Awerbuch initiated the study of simulation techniques that translate synchronous algorithms to asynchronous ones. These simulation techniques are called *synchronizers*.

To devise the first synchronizers Awerbuch [1] constructed a certain graph partition which is of its own interest. In particular, Peleg and Schäffer noticed [8] that this graph partition induces a subgraph with certain interesting properties. They called this subgraph a *graph spanner*. Formally, for an integer positive parameter k, a k-spanner of a graph $G = (V, E)$ is a subgraph $G' = (V, H)$, $H \subseteq E$, such that for every edge $e = (v, u) \in E$, the distance between v and u in H, $dist_{G'}(v, u)$, is at most k.

2 KEY RESULTS

Awerbuch devises three basic synchronizers, called α, β, and γ. The synchronizer α is the simplest one; using it results in a constant overhead in time, but in a very significant overhead in communication. Specifically, the latter overhead is linear in the number of edges of the underlying network. Unlike the synchronizer α, the synchronizer β requires a somewhat costly initialization stage. In addition, using it results in a significant time overhead (linear in the number of vertices n), but it is more communication-efficient than α. Specifically, its communication overhead is linear in n.

Finally, the synchronizer γ represents a tradeoff between the synchronizers α and β. Specifically, this synchronizer is parameterized by a positive integer parameter k. When k is small then the synchronizer behaves similarly to the synchronizer α, and when k is large it behaves similarly to the synchronizer β. A particularly important choice of k is $k = \log n$. At this point on the tradeoff curve the synchronizer γ has a logarithmic in n time overhead, and a linear in n communication overhead. The synchronizer γ has, however, a quite costly initialization step.

The main result of [1] concerning spanners is that for every $k = 1, 2, \ldots$, and every n-vertex unweighted undirected graph $G = (V, E)$, there exists an $O(k)$-spanner with $O(n^{1+1/k})$ edges. (This result was explicated by Peleg and Schäffer [8].)

3 APPLICATIONS

Synchronizers are extensively used for constructing asynchronous algorithms. The first applications of synchronizers are constructing the *breadth-first-search tree* and computing the *maximum flow*. These applications were presented and analyzed by Awerbuch in [1]. Later synchronizers were used for maximum matching [10], for computing shortest paths [7], and for other problems.

Graph spanners were found useful for a variety of applications in distributed computing. In particular, some constructions of synchronizers employ graph spanners [1, 9]. In addition, spanners were used for routing [4], and for computing almost shortest paths in graphs [5].

4 OPEN PROBLEMS

Synchronizers with improved properties were devised by Awerbuch and Peleg [3], and Awerbuch et al. [2]. Both these synchronizers have polylogarithmic time and communication overheads. However, the synchronizers of Awerbuch and Peleg [3] require a large initialization time. (The latter is at least linear in n.) On the other hand, the synchronizers of [2] are randomized. A major open problem is to obtain *deterministic* synchronizers with polylogarithmic time and communication overheads, and sublinear in n initialization time. In addition, the degrees of the logarithm in the polylogarithmic time and communication overheads in synchronizers of [2, 3] are quite large. Another important open problem is to construct synchronizers with imporved parameters.

In the area of spanners, spanners that distort large distances to a significantly smaller extent than they distort small distances were constructed by Elkin and Peleg in [6]. These spanners fall short from achieving a *purely additive distortion*. Constructing spanners with a purely additive distortion is a major open problem.

5 CROSS REFERENCES

Sparse graph spanners (2005; Elkin, Peleg).

Entry editors please feel free to add other cross references.

6 RECOMMENDED READING

[1] B. Awerbuch. Complexity of network synchronization. *J. ACM*, 4:804–823, 1985.

[2] B. Awerbuch, B. Patt-Shamir, D. Peleg, and M. E. Saks. Adapting to asynchronous dynamic networks. In *Proc. of the 24th Annual ACM Symp. on Theory of Computing*, pages 557–570, 1992.

[3] B. Awerbuch and D. Peleg. Network synchronization with polylogarithmic overhead. In *Proc. 31st IEEE Symp. on Foundations of Computer Science*, pages 514–522, 1990.

[4] B. Awerbuch and D. Peleg. Routing with polynomial communication-space tradeoff. SIAM J. Discrete Mathematics, 5:151-162, 1992.

[5] M. Elkin. Computing Almost Shortest Paths. In *Proc. 20th ACM Symp. on Principles of Distributed Computing*, 53-62, 2001.

[6] M. Elkin and D. Peleg. Spanner constructions for general graphs. In *Proc. of the 33th ACM Symp. on Theory of Computing*, pages 173–182, 2001.

[7] K.B. Lakshmanan and K. Thulasiraman and M.A. Comeau. An efficient distributed protocol for finding shortest paths in networks with negative cycles, *IEEE Trans. on Software Eng.*,15:639-644, 1989.

[8] D. Peleg and A. Schäffer. Graph spanners. *J. Graph Theory*, 13:99–116, 1989.

[9] D. Peleg and J. D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. on Comput.*, 18:740–747, 1989.

[10] B. Schieber and S. Moran. Slowing sequential algorithms for obtaining fast distributed and parallel algorithms: Maximum matchings. In *Proc. of 5th ACM Symp. on Principles of Distributed Computing*, pages 282-292, 1986.