Distributed Computing Made Secure: A New Cycle Cover Theorem

Abstract:

In the area of distributed graph algorithms a number of network's entities with local views solve some computational task by exchanging messages with their neighbors. Quite unfortunately, an inherent property of most existing distributed algorithms is that throughout the course of their execution, the nodes get to learn not only their own output but rather learn quite a lot on the inputs or outputs of many other entities. This leakage of information might be a major obstacle in settings where the output (or input) of network's individual is a private information (e.g. distributed networks of selfish agents, decentralized digital currency such as Bitcoin, voting systems).

While being quite unfamiliar notion in the classical distributed setting, the notion of secure multi-party computation (MPC) is one of the main themes in the Cryptography community. Yet despite all extensive work in the area, no existing algorithm fits the framework of classical distributed models in which there are no assumptions on the graph topologies and only messages of bounded size are sent on the edges in each round.

In this work, we introduce a new framework for \textit{secure distributed graph algorithms} and provide the first \textit{general compiler} that takes any "natural" non-secure distributed algorithm that runs in \$r\$ rounds, and turns it into a secure algorithm that runs in \$\widetilde{O}(r \cdot D \cdot \text{poly}(\Delta))\$ rounds where \$\Delta\$ is the maximum degree in the graph and \$D\$ is its diameter. This round complexity is nearly optimal for bounded degree graphs.

The main technical part of our compiler is based on a new cycle cover theorem: We show that the edges of every bridgeless graph \$G\$ of diameter \$D\$ can be covered by a collection of cycles such that each cycle is of length \$\widetilde{O}(D)\$ and each edge of the graph \$G\$ appears in \$\widetilde{O}(1)\$ many cycles. This provides the basis for additional combinatorial constructions required by our compiler and might be of independent combinatorial and algorithmic interest.

Joint work with Merav Parter.