Abstract

This paper considers the basic $\mathcal{PULL}$ model of communication, in which in each round, each agent extracts information from few randomly chosen agents. We seek to identify the smallest amount of information revealed in each interaction (message size) that nevertheless allows for efficient and robust computations of fundamental information dissemination tasks.

We focus on the *Majority Bit Dissemination* problem that considers a population of n agents, with a designated subset of *source agents*. Each source agent holds an *input bit* and each agent holds an *output bit*. The goal is to let all agents converge their output bits on the most frequent input bit of the sources (the *majority bit*). Note that the particular case of a single source agent corresponds to the classical problem of *Broadcast* (also termed *Rumor Spreading*). We concentrate on the severe fault-tolerant context of *self-stabilization*, in which a correct configuration must be reached eventually, despite all agents starting the execution with arbitrary initial states. In particular, the specification of who is a source and what is its initial input bit may be set by an adversary. We first design a general compiler which can essentially transform any self-stabilizing algorithm with a certain property (called "the *bitwise-independence property*") that uses ℓ-bits messages to one that uses only $\log \ell$-bits messages, while paying only a small penalty in the running time. By applying this compiler recursively we then obtain a self-stabilizing *Clock Synchronization* protocol, in which agents synchronize their clocks modulo some given integer T, within $\tilde{O}(\log n \log T)$ rounds w.h.p., and using messages that contain 3 bits only.

We then employ the new Clock Synchronization tool to obtain a self-stabilizing Majority Bit Dissemination protocol which converges in $\tilde{O}(\log n)$ time, w.h.p., on every initial configuration, provided that the ratio of sources supporting the minority opinion is bounded away from half. Moreover, this protocol also uses only 3 bits per interaction.