CCIT Report #590 May 2006

A Local Algorithm for Ad Hoc Majority Voting Via Charge Fusion

Yitzhak Birk, Liran Liss, Ran Wolff and Assaf Schuster
Technion – Israel Institute of Technology
{birk@ee,liranl@tx,ranw@cs,assaf@cs}.technion.ac.il

June 24, 2006

Abstract

We present a local distributed algorithm for a general Majority Voting problem: different and time-variable voting powers and vote splits, arbitrary and dynamic interconnection topologies and link delays, and any fixed majority threshold. The algorithm combines a novel, efficient anytime spanning forest algorithm, which may also have applications elsewhere, with a "charge fusion" algorithm that roots trees at nodes with excess "charge" (derived from a node's voting power and vote split), and subsequently transfers charges along tree links to oppositely charged roots for fusion. At any instant, every node has an ad hoc belief regarding the outcome. Once all changes have ceased, the correct majority decision is reached by all nodes within a time that in many cases is independent of the graph size. The algorithm's correctness and salient properties are proved, and experiments with up to one million nodes provide further validation and actual numbers. To our knowledge, this is the first locality-sensitive solution to the Majority Vote problem for arbitrary, dynamically changing communication graphs.

1 Introduction

1.1 Background

Emerging large-scale distributed systems such as the Internet-based peer-to-peer systems, grid systems, ad hoc networks and sensor networks, impose uncompromising scalability requirements on (distributed) algorithms used for performing various functions. Clearly, for an algorithm to be perfectly scalable, i.e., O(1) complexity in problem size, it must be "local" in the sense that a node only exchanges information with nodes in its vicinity. Also, information must not need to flow across the graph. For some problems, there are local algorithms whose execution time is effectively independent of the graph size. Examples include Ring Coloring [1] and Maximum Independent Set [2].