5

In a generalized voting system representing a collective decision made by some number of voters between n candidates, there are many possible vote tallying schemes. Although plurality tallying schemes, in which each voter casts a single vote for their favorite candidate, are by far the most popular in practice, the Condorcet tally method is of much more theoretical interest, as it is provably the only scheme which satisfies a number of desirable properties. In a Condorcet tally, for each possible pair of candidates, each voter casts a vote for one candidate or the other. In each of these $n(n-1)$ smaller tallies, the winner is the candidate which garners the most votes. The Condorcet winner, then, is the candidate which successfully defeats every other candidate in the head-to-head tallies. It is worth noting that there is not always such a candidate. In these cases, there is a cycle of candidates $C_i, \ldots, C_k$, called a top cycle, with the property that $C_1 > C_2 > \ldots > C_k > C_1$, all of which beat all other candidates head-to-head.

One particular domain in which Condorcet tallies are of particular interest is that of graphs. In particular, much of the recent work of Dr. Whitman Richards has focused on using graphs of daemons, termed "anigrafs," as models of cognition. In these models, each vertex represents an atomic subunit of cognition, akin to Minsky's "agents." When a decision needs to be made by the cognitive structure in question, each daemon suggests a possible outcome, and provides a numerical assessment of how strongly its observations support its assertion. These numbers are the weights of the nodes, and the weights of the edges are given by some notion of "distance" between the daemons (that is to say, the daemon "Hunger" might be closer to the daemon "Thirst" than to the daemon "Fear"). In the anigraf model, a Condorcet tally is performed, in which, for each pair of nodes, each other node casts its votes for alternative which is closer (i.e. has a shorter shortest path to it).

We seek to develop, analyze, and possibly implement efficient randomized algorithms for solving the problem of finding the Condorcet winner of a graph (or identifying the top cycle, if no winner exists). The naive deterministic algorithm runs in $O(n^3)$ time, by simply generating the all pairs shortest path matrix of the graph, then making $n^2(n-1)$ queries to this matrix to determine the outcome of every possible head-to-head matching. A slightly more clever randomized algorithm can identify the Condorcet winner, if it exists, in $O(n(n \log n + E))$, by running a randomly seeded single elimination

I collaborated with Sarah Lieberman and Dan Ports on all problems.

tournament among the nodes, and using Dijkstra with a Fibonacci heap to determine the outcome of each pairing. For sparse graphs, this algorithm runs in $O(n^2 \log n)$ time, but it also runs in $O(n^3)$ for dense graphs. To the best of our (and Dr. Richards's) knowledge, no better algorithms for solving this problem are currently known. We propose to apply the principles learned in this class to develop a more efficient algorithm. In particular, we will explore the possibility of using random sampling on a dense graph to produce a sparser graph, on which even the relatively simple randomized algorithm outlined above outperforms the naive deterministic algorithm.

Sounds intriguing. Be sure to have a plan for what to do if you don't succeed in making progress with the research idea. There hasn't been much success with sampling for shorter paths, but perhaps the special form of the problem you aim to solve makes it more susceptible.

4/22/2005
I collaborated with Sarah Lieberman and Dan Ports on all problems.