**Discrete Math in CS** **A First Example: Voting Systems**
**CS 280 Fall 2005 (Kleinberg)**

A typical way in which computer scientists approach problems is to write down some properties that a solution should satisfy, and then reason about whether these properties can be achieved.

There are a number of areas where the introduction of this style of thinking had revolutionary consequences; as one illustration, and a nice first example for this course, we look at Arrow's Impossibility Theorem (1951) from mathematical economics. (Arrow won the Nobel Prize in Economics in 1972, cited in part for this result.) The background of Arrow's Theorem is voting, and in particular some "paradoxical" properties of voting outcomes that were discovered by French philosophers in the 1700's.

**Condorcet's Principles.** The Marquis de Condorcet was one of these philosophers, and he has influenced a lot of thinking on the design of voting systems.

Suppose we have a list of candidates, and a set of voters, and each voter has a ranking of all candidates by preference.

- Example: Suppose the candidates are $x$, $y$, $z$. The voters produce rankings
  - Voter 1 : $x, y, z$
  - Voter 2 : $x, z, y$
  - Voter 3 : $z, x, y$
  - Voter 4 : $y, x, z$
- Who should be the winner if the above voters get together to choose a candidate? $x$ has the property that in head-to-head comparison it defeats $y$ (3 to 1) and defeats $z$ (3 to 1). We will call a candidate a *Condorcet winner* if it has this property: it wins in each head-to-head comparison.

Notice that what we're doing already has a fairly computer-science flavor to it. We have a set of "inputs" — the rankings of all voters — and we wish to design a function that will produce a desired "output" — the winning candidate. We now want to reason about what properties we would like the output to have, and what sorts of functions might guarantee these properties.

Let's discuss some of Condorcet's principles.

- The *Condorcet criterion* says that whenever there is a Condorcet winner, the voting system should declare it the winner. (Could there be two different Condorcet winners? No, since one would have to defeat the other.)
- What happens in U.S. Presidential elections? We just take the candidate who appears at the front of the most lists. (This is called *pluarilty voting.*) Does this guarantee that we obey the Condorcet criterion? No. For example:

  - 10 voters favor $x, z, y$.
  - 2 voters favor $y, z, x$.
  - 9 voters favor $z, x, y$.

  The winner under plurality voting is $x$, even though a majority of voters prefer $z$ to $x$. Here, candidate $y$ acts as a "spoiler"; if they hadn't been in the election, then $z$ would have won.

- Is the Condorcet criterion a good idea? Generally yes, though there are examples that make you wonder. Suppose

  - 10 voters favor $x, y, z$.
  - 2 voters favor $y, x, z$.
  - 9 voters favor $z, y, x$.

  Then $y$ is the Condorcet winner, but intuitively $y$ acts sort of as the "compromise candidate" that no one's really excited about.

- Another problem: there might not be a Condorcet winner.

  - 10 voters favor $x, y, z$.
  - 10 voters favor $y, z, x$.
  - 10 voters favor $z, x, y$.

  For any candidate, there is another candidate that beats it 20-10. This example is called the *Condorcet paradox*: it's not at all clear how to choose a winner in such a situation.

**What do we mean by a voting system?** These kinds of issues left people feeling very confused about how best to design a voting system, but it was only 150 years later that Kenneth Arrow took the step of formalizing what a voting system was trying to accomplish, and encapsulating the crux of these debates in an elegant *impossibility theorem*. Here's how he did this.

We assume a set of $k$ *options* (the candidates), and a set of $n$ voters. Each voter $i$ has a preference list $P_i$.

Informally, what is a voting system? To think about this, let's get back to our analogy with a function that takes an input and produces an output. Roughly, a voting system is something that takes all these preferences, analyzes them somehow, and comes up with a "consensus" order on the options, ranking one option first in the overall consensus, another second, and so forth. So formally, it's a function $f$ that takes a sequence of preference orders $P_1, P_2, \ldots, P_n$ and outputs a consensus order $P = f(P_1, P_2, \ldots, P_n)$. (Note that in this definition, we'll be asking for the output to consists of a ranking of all options, rather than just a selection of the "winning" one.)

**What properties should a voting system satisfy?** Arrow proposed two desirable properties. The first is *unamimity*:

- If all voters rank option $a$ ahead of option $b$, then the consensus order ranks $a$ ahead of $b$ too. (Formally, if $a$ is ahead of $b$ in $P_i$ for every $i$, then $a$ is ahead of $b$ in $f(P_1, \ldots, P_n)$.

The second property is *independence of irrelevant alternatives*:

- Informally, the consensus order of $a$ and $b$ should depend only on the relative orders of $a$ and $b$ in each ranking. Formally, if $a$ is ahead of $b$ in a consensus $f(P_1, P_2, \ldots, P_n)$, and then we change each ranking $P_i$ to $P_i'$ simply by sliding the position of $c \neq a, b$ forward or backward, then $a$ is also ahead of $b$ in the consensus $f(P_1', P_2', \ldots, P_n')$.

Can we think of any voting systems that satisfy these two properties? Actually, here's one: just take voter number 1's ranking and return it as the consensus. This satisfies unanimity (if everyone favors $a$ to $b$, then so does voter 1) and independence of irrelevant alternatives (sliding the position of option $c$ doesn't change how voter 1 feels about $a$ and $b$). We'll refer to this voting systems as a *dictatorship*, since voter 1 simply acts as a dictator.

For that matter, there are $n$ such voting systems that satisfy these two properties: one in which we take voter $i$ as the dictator for each possible choice of $i = 1, 2, \ldots, n$.

Thus, we'll refer to a voting system as a *dictatorship* if there is a "hard-wired" voter $i$ such that the voting system consists simply of returning $i$'s ranking as the consensus for any input.

**Arrow's Theorem.** Arrow's striking result is that dictatorships are the only voting systems that satisfy our two properties.

- *Arrow's Theorem*: If a voting system $f$ satisfies unanimity and independence of irrelevant alternatives, then it is a dictatorship.

While it's now clear what Arrow's Theorem says, it's not at all clear why it's true. For that, we need a proof. We won't go into the proof in this lecture, but, time permitting, we will try to come back to the theorem later in the semester and describe a proof.

**Other Voting Systems.** Arrow's Theorem has never really meant that voting is "impossible," simply that any voting system you invent will exhibit some potentially undesirable pathologies. And a large number of voting systems have been proposed over the years. Here we mention two of them as examples.

In the *Borda Count* system, with $k$ options, each voter gives $k$ points to their first-place option, $k-1$ to their place option, and in general $k-j+1$ points to their $j^{\text{th}}$ favorite option. For each option, the total number of points given by all voters is then added up, and the options are ranked in order of their point totals.

In *Instant Runoff Voting*, the option with the fewest first-place votes is "eliminated" and placed last in the consensus ranking. This option is then deleted from each voter's ranking, and the procedure is applied recursively to the remaining options.

**Some computer science context.** The setting of Arrow's Theorem has appeared recently in the context of search engine rankings. Suppose you have a service that tries to "combine" the results of several search engines: it issues the same query to Google, Yahoo, MSN Search, Teoma, and so forth, and then tries to identify a "consensus" among their rankings. Such services are sometimes called *meta-search engines*. For this task, voting systems provide ways of combining the individual search engine rankings into a consensus, and Arrow's Theorem provides powerful limitations on what such a consensus can look like.