Maximum Flow and Dynamic Programming

11-25-2003

Opening Discussion

- Who can describe a flow network to me. What are the properties it has and what types of systems can be modeled with it?
- Do you have any questions about the assignment? You should submit assignment #5 today.

Solving Maximum Flow

- We solve this with the Ford-Fulkerson method. There are many algorithms for implementing this method.
- The idea is that we repeatedly add "augmenting paths" to the flow. That is a path from source to sink that adds something to the current flow.
- Given a network and flow the residual is the difference between capacity and flow.

Augmenting Paths

- An augmenting path in a flow network is a path from the source to the sink along edges where the current flow is less than the capacity.
- Given such a path, the flow it can sustain is the edge on the path with the lowest difference between the capacity and the current flow.

The Basic Algorithm

- In the basic algorithm we initialize all the flows to zero first.
- While there is an augmenting path p, we find the edge with minimum additional capacity, $c_f(p)$. Then we go through all the edges, (u,v), in p and increment the flow from u to v by $c_f(p)$ while setting the reverse flow to that value.
- This runs in $O(E\,|f^*|)$ time.

Edmonds-Karp Algorithm

- We can enhance the basic algorithm by making it so that they choice of augmenting path is made with a breadth first search. This way we always pick the shortest augmenting path.
- This gives an algorithm that runs in $O(V\,E^2)$ time.

Basics of Dynamic Programming

- Dynamic programming is a technique typically applied to optimization problems. It can be used for the same types of problems we might normally use divide-and-conquer on, but it is more optimal if the sub-problems in divide and conquer would repeat work.
- The steps in doing it are
  - Characterize the structure of a solution
  - Make a recursive definition of solution
  - Compute value of optimal solution bottom-up
  - Construct solution from earlier information

Requirements and Sample Problems

- In order for dynamic programming to work, the problem has to have optimal substructure. That is to say that the optimal solution is built from optimal solutions to smaller problems.
- Fibonacci numbers are a prime example of where dynamic programming can be efficiently used.
- The 0/1 knapsack problem is another example.

Minute Essay

- Explain why dynamic programming can be much faster than normal recursive algorithms for some problems.
- The semester is quickly winding down. We will have quiz #6 on Tuesday after Thanksgiving.
- Have a Happy Thanksgiving!