DP and Greedy Algorithms

12-4-2003

Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignment?

Coding DP

- Let's look at code to solve the assembly line problem and the 0/1 knapsack problem.
- Here is the correct recursion for 0/1 knapsack.

$$v(i,w)=\begin{cases}0 & i<1 \vee w\leq 0\\ v(i-1,w) & w<w_i\\ \max\left(v(i-1,w),v(i-1,w-w_i)+v_i\right) & otherwise\end{cases}$$

Memoization

- Because we have trained our brains to think in a top-down approach a lot of the time, it can often be helpful to use a top-down style like we would with recursion, but simply save the intermediate answers. For this we have an array like in DP that we pass through the recursive calls. When we need something we try looking in the array before making the recursive call.

Longest Common Subsequence

- For this problem, we want to find the longest sequence of characters that is found in two strings. The sequence must be found in both in order, but doesn't have to be consecutive.
- For example, "Happy Birthday" and "Have a nice day" share the common subsequence "Ha iday"

Solution to LCS

- Does this problem have optimal substructure?
- How can we characterize a solution?
- What does the recursive function look like for finding the length of the LCS? To answer this last question let's think of what is it a function of, and what the possibilities are for a given value.

Greedy Algorithms

- This is another type of algorithm design technique that is used to solve optimization problems. This method can be faster than DP on simple problems where it is applicable.
- The idea with a greedy algorithm is that you always pick what looks like the best choice at each step. There is never backtracking or second guessing.

Room Scheduling Problem

- Imagine you have a room that you need to schedule events for. You have a list of events, each with a start time and a finish time and your objective is to get the maximum number of events into that room.
- This can be done in a greedy way. If we sort by finishing time, we always pick the next event to finish that doesn't overlap with an earlier event.

Minute Essay

- What order will greedy algorithms tend to be (assuming you can find the optimal pick quickly)? How many possibilities do they have to test out?
- You should turn in test code for your Dijkstra's algorithm today. Tuesday is the last day of class and the due date for assignment #6.