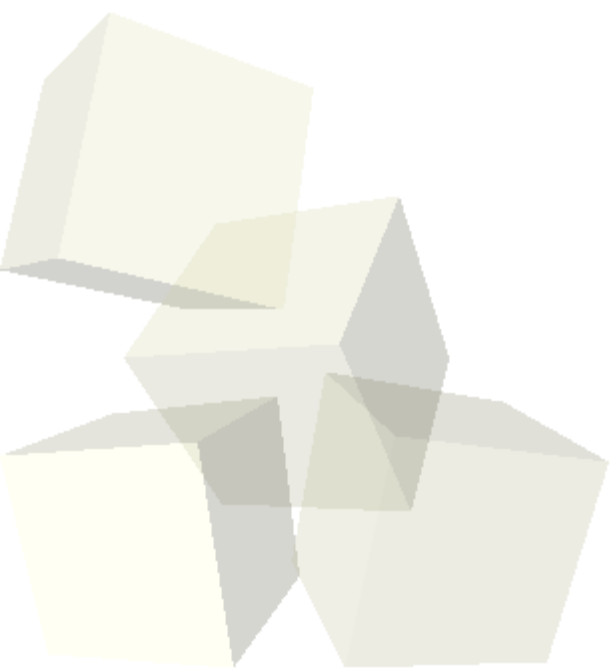




# Approximation Algorithms

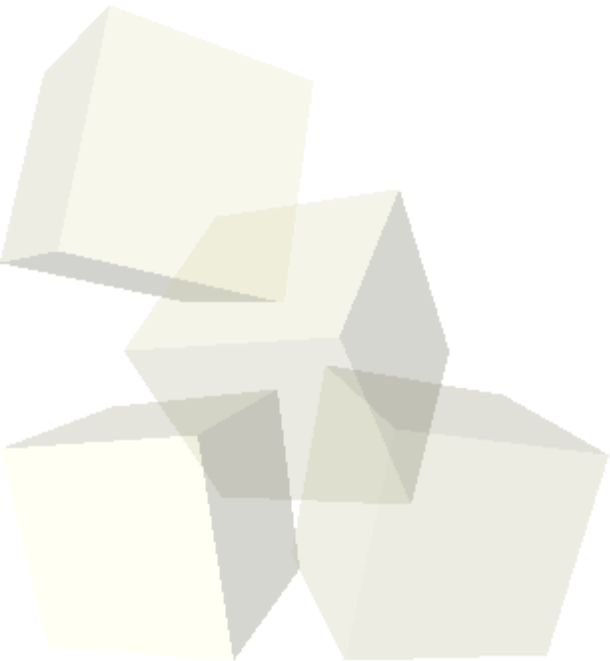
4-25-2006





# Opening Discussion

- What did we talk about last class?





# Randomized MAX-3-CNF

- The MAX-3-CNF is an optimization form of the NP 3-CNF. Instead of asking whether there is a solution from which all terms are true, we ask what set of inputs gives us the most true terms.
- We can get an  $8/7$ -approximation algorithm simply by selecting the values for the inputs at random. This works because each term has three variables (or their negations) in it so each one will have a  $1/2$  chance of being true. The three together have a  $1/8$  chance they are all false or  $7/8$  chance at least one is true.
- If we treat true as 1, our expected value for each term is  $7/8$  for each one that can be satisfied.



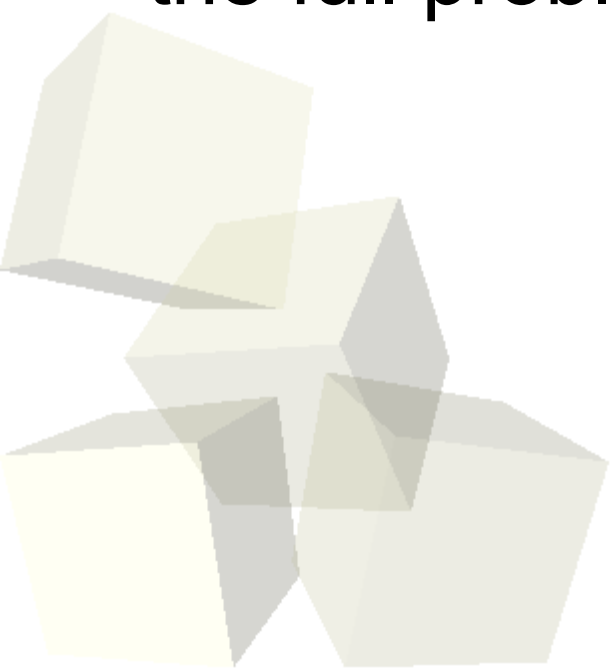
# Weighted Vertex Covering

- The weighted vertex covering adds a weight to each of the vertices in an undirected graph and we have to find a covering set with a minimal sum for the weights.
- We can't approximate this the way we could with a standard minimal vertex covering and hope to get a good approximation.
- We can approximate this with linear programming. Ideally our variables are  $x(v)$  with a value of 0 or 1. However, that doesn't help us since integer linear programming isn't tractable. Instead, we allow  $x(v)$  to vary between 0 and 1.



# LP Weighted Vertex Covering

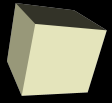
- This gives us the following linear programming problem.
  - ◆ Minimize  $\sum(w(v)*x(v))$
  - ◆  $x(u)+x(v) \geq 1$  for each  $(u,v)$  in  $E$
  - ◆  $x(v) \leq 1$
  - ◆  $x(v) \geq 0$
- The solution to this will be a 2-approximation to the full problem.





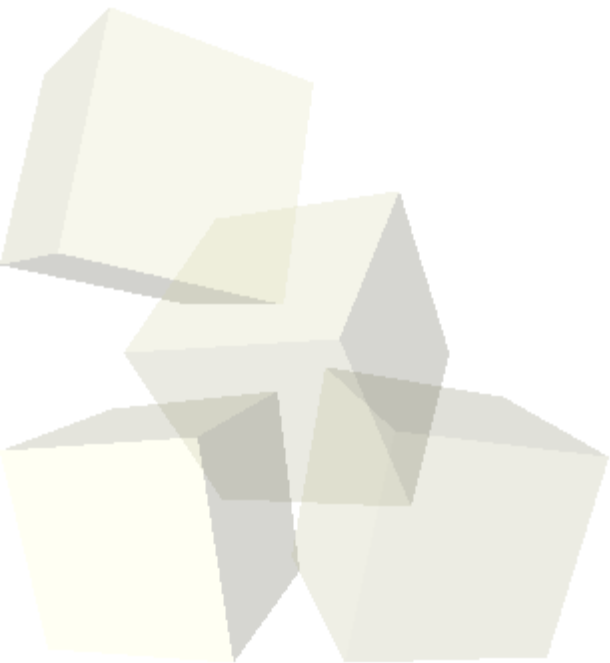
# Subset-Sum Problem

- Here is our last approximation algorithm. Given a set of non-negative numbers,  $S$ , we want to find the subset that is as close to a value,  $t$ , without exceeding it.
- A general solution can be made by continually building a list that contains the sums of all subsets of  $S$  iteratively putting in more of the elements of  $S$ . We can prune the list by taking out any elements greater than  $t$ . This is exponential.
- We can get an approximation by keeping the list short through trimming. This process removes any element  $y$  if there is an element  $z$  such that  $y/(1+d) \leq z \leq y$ .



# Number Theory Stuff

- Cryptography – factoring is hard.
- Modular arithmetic





# Reminders

- Remember to turn in test 7 next class.

