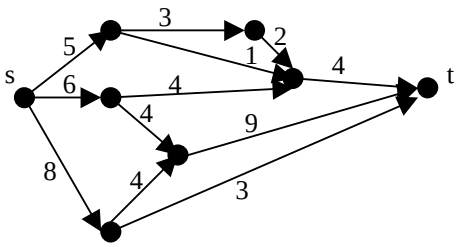
Quiz #6 Answers

1. Describe one of the methods we talked about for finding the all-pairs shortest path.

In class we talked about two methods. Both of them use a matrix representation of the graph and both are based on dynamic programming ideas. The first one used something similar to matrix multiplication. Instead of multiplying elements and adding them up, we added elements and took the minimum. This was done repeatedly until we got our final matrix of distances. It was made slightly more intelligent by having the procedure use "repeated squaring".

The second method was built on the idea of only allowing intermediate paths between vertices to include certain subsets of the vertices. At first this subset was empty so the only paths were from direct connections. Then vertex one was added and optimal paths were found that were either direct or only had vertex one between them. Next, vertex two as added and the same thing was repeated. Once the final vertex was put in we had all the shortest paths.

2. What is the maximum flow of the following graph?



The answer here is 15. We get this by repeatedly adding flow along paths where the flow isn't yet at capacity. For example, we might pick a first path along the bottom that includes the edges with capacity 8 and 3. This path can only bear a flow of 3, so each element along it is assigned a flow of 3. We might then take the next path up that goes across edges with capacities of 8, 4, and 9. The 8 edge already has a flow of 3 so it could sustain 5 more. The others have no flow yet so the limiting edge is the 4 and we add a flow of 4 to each of the edges. We repeat this until there is no path from the source to the sink that can bear a higher flow. In this case that ends when the flow going from source to sink is 15. Equivalently all the edges leaving the source together have a flow of 15 and all the edges entering the sink have a flow of 15.

Extra Credit: When can we use dynamic programming to solve a problem?

For full credit the answer is when there is optimal substructure. Using it in any other situation produces the wrong results. Partial credit was given for a description of when we want to use it which is when work was repeated many times with calls that were calculating the same values.