MA/CSSE 473 Day 27 Dynamic programming intro

1. In what situation can dynamic program speed up the runtime of an algorithm?

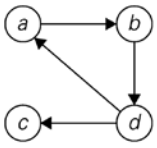
2. How does Dynamic Programming improve the performance of Fibonacci computation (compared to just using the recursive formula?

3. What is the recursive formula used by the dynamic programming algorithm for binomial coefficients?

4. Transitive closure of a directed graph
    a. How to compute it using matrix multiplication

    b. Number of integer multiplications required for this approach.

    c. Warshall's algorithm
        i. Number the vertices: $v_1, v_2, \ldots, v_n$
        ii. Graph represented by a boolean adjacency matrix M.
        iii. Numbering is arbitrary, but is fixed throughout the algorithm.
        iv. Define the boolean matrix $R^{(k)}$:
        v. $R^{(k)}[i][j]$ is 1 iff there is a path from v_i to v_j in the directed graph that has the form $v_i = w_0 \rightarrow w_1 \rightarrow \ldots \rightarrow w_s = v_j$, where
            1. s >=1, and
            2. for all t = 1, …, s-1, the w_t is v_m for some $m \leq k$
            i.e, none of the intermediate vertices are numbered higher than k
        vi. What is $R^{(0)}$?
        vii. Note that the transitive closure T is $R^{(n)}$
        viii.



$$A = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

        ix. A quicker way to calculate $R^{(k)}$ from $R^{(k-1)}$

5. Optimal linked list order (if we know the probability of search for each item)
    a. Item x_i in list has probability p_i. What is expected number of probes for search?

    b. Example: $p_1 = ½$, $p_2 = ¼$, …, $p_{n-1} = 1/2^{n-1}$, $p_n = 1/2^{n-1}$
    Expected # of probes for best case, worst case:

    c. What if we do not know the probabilities?

6. Optimal binary search tree (for case where we know the probabilities (or frequencies)
    a. For today, we only deal with successful searches.
    b. If $P(x_i) = p_i$, what is the expected number of probes for a search?

    c. Guiding principle for optimization

7. How many different BSTs with n nodes (containing numbers 1, 2, …, n)?

8. **Example:** consider only successful searches, with probabilities A(0.2), B(0.3), C(0.1), D(0.4).

worst opposite greedy

better best?