

MA/CSSE 473

Day 30



**Dynamic
Programming**

Binomial Coefficients

Warshall's algorithm

MA/CSSE 473 Day 30

- HW 12 due Thursday
- Exam 2 is Friday
 - Covers through Chapter 7 and HW12.
- **Student Questions**
- Dynamic Programming
- Binomial Coefficients
- Warshall's Algorithm



Dynamic programming

- Used for problems with overlapping subproblems
- Typically, we save (memoize) solutions to the subproblems, to avoid recomputing them.
- Example: Fibonacci numbers
 - Standard recursive formula is too expensive, because of repeated calculation of smaller fibonacci numbers.
 - Caching previously-computed values changes the algorithm from exponential to linear.



Dynamic Programming Example

- Binomial Coefficients:
- $C(n,k)$ is the coefficient of x^k in the expansion of $(1+x)^n$
- $C(n,0) = C(n,n) = 1$.
- If $0 < k < n$, $C(n,k) = C(n-1, k) + C(n-1, k-1)$
 - How do we know it's true?
- Can show by induction that the "usual" factorial formula for $C(n, k)$ is correct.
- If we don't cache values as we compute them, this can take a lot of time, due to duplicate computation



Computing a binomial coefficient b

Binomial coefficients are coefficients of the binomial formula:

$$(a + b)^n = C(n,0)a^n b^0 + \dots + C(n,k)a^{n-k}b^k + \dots + C(n,n)a^0 b^n$$

Recurrence: $C(n,k) = C(n-1,k) + C(n-1,k-1)$ for $n > k > 0$

$$C(n,0) = 1, \quad C(n,n) = 1 \text{ for } n \geq 0$$

Value of $C(n,k)$ can be computed by filling a table:

	0	1	2	...	$k-1$	k
0	1					
1	1	1				
.						
.						
.						
$n-1$					$C(n-1,k-1)$	$C(n-1,k)$
n						$C(n,k)$



Computing $C(n, k)$:

ALGORITHM *Binomial*(n, k)

//Computes $C(n, k)$ by the dynamic programming algorithm

//Input: A pair of nonnegative integers $n \geq k \geq 0$

//Output: The value of $C(n, k)$

for $i \leftarrow 0$ **to** n **do**

for $j \leftarrow 0$ **to** $\min(i, k)$ **do**

if $j = 0$ **or** $j = i$

$C[i, j] \leftarrow 1$

else $C[i, j] \leftarrow C[i - 1, j - 1] + C[i - 1, j]$

return $C[n, k]$

Time efficiency: $\Theta(nk)$

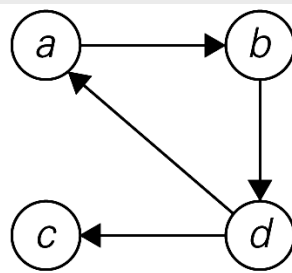
Space efficiency: $\Theta(nk)$

Exercise 8.1.7 asks you to compare the efficiency of this approach with some other approaches



Transitive closure of a directed graph

- For each pair of vertices, (A,B) , in the directed graph G , is there a path from A to B in G ?
- Start with the boolean adjacency matrix A for the n -node graph G . $A[i][j]$ is 1 iff there is a directed edge from node i to node j in G .
- The **transitive closure** of G is the boolean matrix T such that $T[i][j]$ is 1 iff there is a nontrivial directed path from node i to node j in G .
- If we use boolean adjacency matrices, what does M^2 represent? M^3 ?
- In boolean matrix multiplication, $+$ stands for **or**, and $*$ stands for **and**



(a)

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

(b)

$$T = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix}$$

(c)

FIGURE 8.2 (a) Digraph. (b) Its adjacency matrix. (c) Its transitive closure.

Transitive closure *via* multiplication

- Again, using + for **or**, we get

$$T = I + A + A^2 + A^3 + \dots$$

- Can we limit it to a finite operation?
- We can stop at A^n .
 - How do we know this?
- Number of numeric multiplications for solving the whole problem?



Warshall's algorithm

- Similar to binomial coefficients algorithm
- Assumes that the vertices have been numbered 1, 2, ..., n
- Define the boolean matrix $R^{(k)}$ as follows:
 - $R^{(k)}[i][j]$ is 1 iff there is a path in the directed graph $i=v_0 \rightarrow v_1 \rightarrow \dots \rightarrow v_s=j$, where
 - $s > 1$, and
 - for all $t = 2, \dots, n-1, v_t \leq k$
i.e, none of the intermediate vertices are numbered higher than k.
- Note that T is $R^{(n)}$



$R^{(k)}$ example

- $R^{(k)}_{[i][j]}$ is 1 iff there is a path in the directed graph

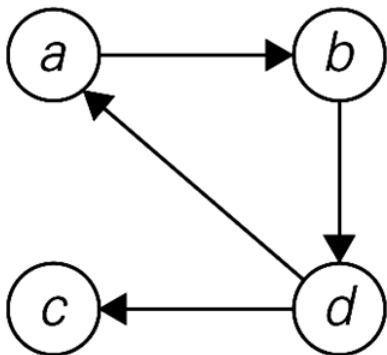
$i=v_0 \rightarrow v_1 \rightarrow \dots \rightarrow v_s=j$, where

– $s > 1$, and

– for all $t = 2, \dots, n-1, v_t \leq k$

- assuming that the nodes are numbered in alphabetical order, calculate $R^{(0)}$ and $R^{(1)}$

You can find a larger example in a book that is available at Safari Books on-line, through the Logan Library Web page (in the Databases section near the top of the page). The book is Sedgwick, *Algorithms Part 5*. See section 19-3



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



Quickly Calculating $R^{(k)}$

- Back to the matrix multiplication approach:
 - How much time did it take to compute $A^k[i][j]$, once we have A^{k-1} ?
- Can we do better when calculating $R^{(k)}[i][j]$ from $R^{(k-1)}$?
- How can $R^{(k)}[i][j]$ be 1?
 - either $R^{(k-1)}[i][j]$ is 1, or
 - there is a path from i to k that uses no vertices higher than $k-1$, and a similar path from k to j .
- Thus $R^{(k)}[i][j] = R^{(k-1)}[i][j]$ **or** ($R^{(k-1)}[i][k]$ **and** $R^{(k)}[k][j]$)
- Note that this can be calculated in constant time
- Time for calculating $R^{(k)}$ from $R^{(k-1)}$?
- Total time for Warshall's algorithm?
- How does this time compare to using DFS?

Code and example on next slides



ALGORITHM *Warshall*($A[1..n, 1..n]$)

//Implements Warshall's algorithm for computing the transitive closure

//Input: The adjacency matrix A of a digraph with n vertices

//Output: The transitive closure of the digraph

$R^{(0)} \leftarrow A$

for $k \leftarrow 1$ **to** n **do**

for $i \leftarrow 1$ **to** n **do**

for $j \leftarrow 1$ **to** n **do**

$R^{(k)}[i, j] \leftarrow R^{(k-1)}[i, j]$ **or** ($R^{(k-1)}[i, k]$ **and** $R^{(k-1)}[k, j]$)

return $R^{(n)}$

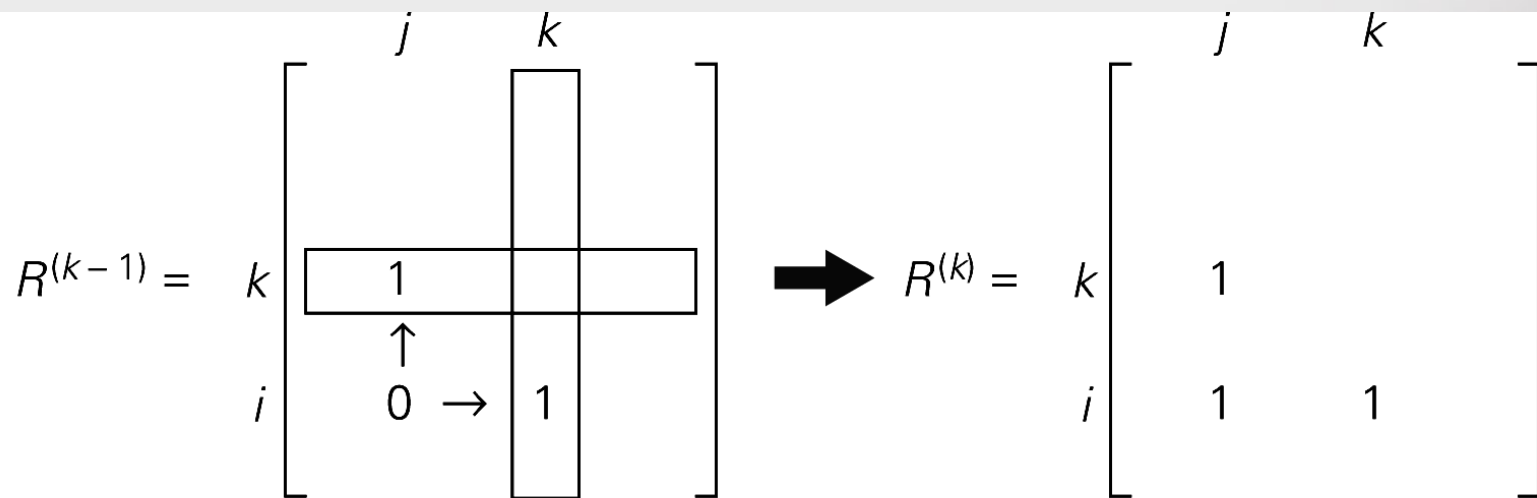
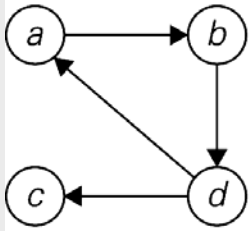


FIGURE 8.3 Rule for changing zeros in Warshall's algorithm



$$R^{(0)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{matrix}$$

Ones reflect the existence of paths with no intermediate vertices ($R^{(0)}$ is just the adjacency matrix); boxed row and column are used for getting $R^{(1)}$.

$$R^{(1)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & \boxed{1} & 0 & 0 \\ \boxed{0} & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & \mathbf{1} & 1 & 0 \end{bmatrix} \end{matrix}$$

Ones reflect the existence of paths with intermediate vertices numbered not higher than 1, i.e., just vertex a (note a new path from d to b); boxed row and column are used for getting $R^{(2)}$.

$$R^{(2)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 1 & \boxed{0} & \mathbf{1} \\ 0 & 0 & 0 & 1 \\ \boxed{0} & 0 & 0 & 0 \\ 1 & 1 & \boxed{1} & \mathbf{1} \end{bmatrix} \end{matrix}$$

Ones reflect the existence of paths with intermediate vertices numbered not higher than 2, i.e., a and b (note two new paths); boxed row and column are used for getting $R^{(3)}$.

$$R^{(3)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 1 & 0 & \boxed{1} \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \boxed{1} & 1 & 1 & \boxed{1} \end{bmatrix} \end{matrix}$$

Ones reflect the existence of paths with intermediate vertices numbered not higher than 3, i.e., a , b , and c (no new paths); boxed row and column are used for getting $R^{(4)}$.

$$R^{(4)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} \mathbf{1} & 1 & \mathbf{1} & 1 \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{matrix}$$

Ones reflect the existence of paths with intermediate vertices numbered not higher than 4, i.e., a , b , c , and d (note five new paths).

FIGURE 8.4 Application of Warshall's algorithm to the digraph shown. New ones are in bold.

