COMPSCI 330 Lecture 4 Dynamic Programming

Monday, September 5, 2016 5:47 PM

- Fibonacci numbers (Basic Idea and Memorization)
- Shortest Path on Directed Acyclic Graphs (Ordering)
- Longest Common Subsequence (2-d tables)

- Fibonacci number

  $F(0)=1 \quad F(1)=1 \quad \forall n \geq 2 \quad F(n)=F(n-1)+F(n-2)$

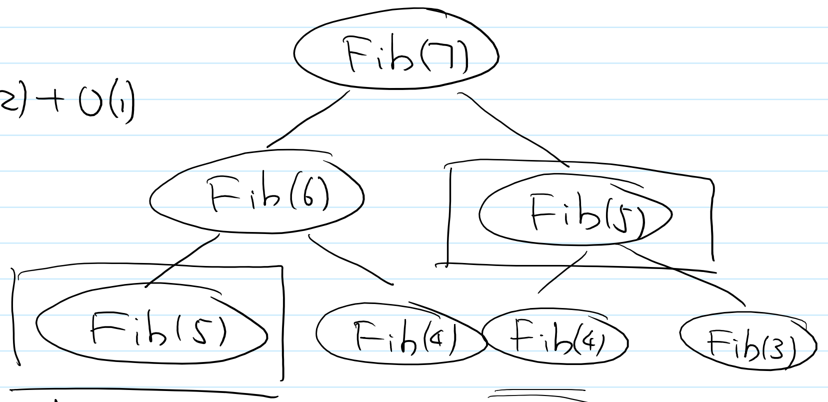
  - how to compute $F(n)$
    - recursive solution

```
Fib(n)
   if n≤1 then return 1
   return Fib(n-1) + Fib(n-2)
```

    - running time

$T(n) = T(n-1) + T(n-2) + O(1)$

$T(n) = O\left(\left(\frac{\sqrt{5}+1}{2}\right)^n\right)$



    - memorized search

```
Fib(n)
   if n≤1 return 1
 * if n is "solved", return solution[n]
   r = f(n-1) + f(n-2)
 * mark n as solved, solution[n] = r
   return r
```

	0	1	2	3	4	..	7	...
solution	1	1	2	3	5			

$T(n) = O(\#\text{ of elements in table} \times \text{amount of time per entry})$

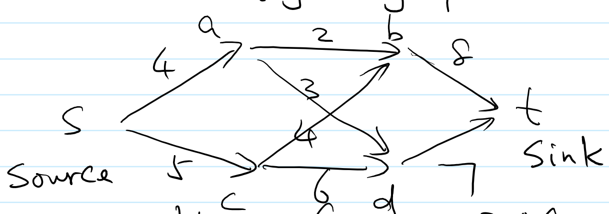
$= O(n \times 1) = O(n)$

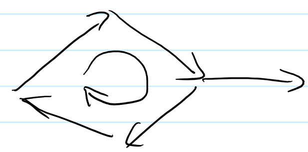
- iterative solution

```
Fib(n)
    if n ≤ 1 return 1
    F(0) = F(1) = 1
    for i = 2 to n
        F(i) = F(i-1) + F(i-2)
    return F(n)
```

- General idea of dynamic programming (DP)
    - save intermediate results to avoid repeated computation
- Design a DP algorithm
    1. identify important subproblems (make a table) list, matrix, ...
    2. fill in the entries of the table in a "good" order.

- shortest path in directed acyclic graphs
    - directed acyclic graph (DAG)

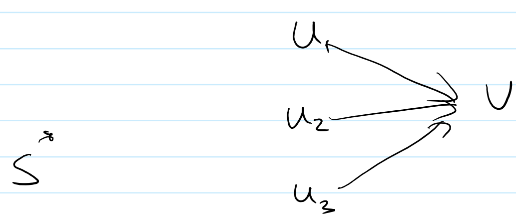




- problem: Given a DAG, edge (i,j) has length $w_{i,j}$
  want to find shortest path from s to t.
  (the length)

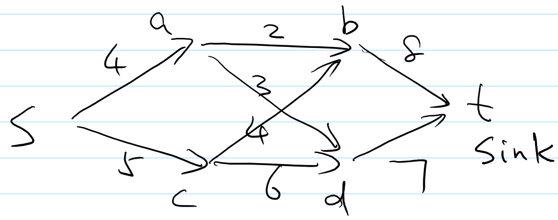
- recursive solution (think: what is the last step of the solution)

  shortest(v): length of shortest path from s to v
  if v = s return 0
  return $\min\limits_{\substack{u:(u,v)\text{ is}\\ \text{an edge}}}$ shortest(u) + $w_{u,v}$



- memorized search

```
shortest(v)
    if v = s  return 0
    if v is "solved" return distance[v]
    r = ∞
    for u = 1 to n
        if (u,v) is an edge, shortest(u) + W_{u,v} < r
            r = shortest(u) + W_{u,v}
    mark v as solved, distance[v] = r
    return r
```



	s	a	b	c	d	t
distance	0	4	6	5	7	14

- Example: Longest Common Subsequence (LCS)

Input: two sequences $a = 1,2,3,2,1$ $b = 2,3,1,4,1$

Subsequence: subset of elements in the same order (not necesarily continuous)

e.g. 1,2,3 ✓
1,3,2 ✓
1,2,1 ✓
1,1,2 ✗

$a = \underline{1}, 2, \underline{3}, \underline{2}, 1$

problem: find (the length of) the longest common subsequence of a, b.

(in this case 2, 3, 1)

(recall: look at the last step of the solution)

$a = 1, 2, 3, 2, 1$ $len(a) = n$

$b = 2, 3, 1, 4, 1$ $len(b) = m$

Q: Do a_n, b_m belong to the LCS.

case ① because $a_n = b_m$
it is possible both of them are in LCS

(if $a_n \neq b_m$ this case is impossible)

$LCS = \boxed{\ ?\ }\ 1$

$\boxed{\ ?\ } = LCS(a[1..n-1], b[1..m-1])$

case ② a_n is not in LCS

$LCS(a,b) = LCS(a[1..n-1], b[1..m])$

case ③ b_m is not in LCS

$$LCS(a, b) = LCS(a[1..n], b[1..m-1])$$