CMPS 201 5-24-10

We showed:

Thm At least $\lfloor \frac{n}{2} \rfloor \lceil \frac{n}{2} \rceil$ 'adjacency Questions' must be asked (in worst case) to determine if a graph G on n vertices is connected.

In fact:

Thm': At least $\binom{n}{2}$ adjacency Questions are necessary (in worst case) to determine the Connectivity of a graph on n Vertices.

Recall $\lfloor \frac{n}{2} \rfloor \cdot \lceil \frac{n}{2} \rceil = \frac{1}{4} n^2 + o(n^2)$

$$\binom{n}{2} = \frac{n(n-1)}{2} = \frac{1}{2} n^2 - \frac{1}{2} n$$

Proof:

Consider some Algorithm that solves this problem, start it on an unspecified Graph $G = (V, E)$ with $|V| = n$.

Daemon's strategy: answer no to any edge probe, unless that answer proves G to be disconnected

More Precisely, Daemon maintains Two sets X, Y where initially $Y = \phi$ and $X = E(K_n)$ so $|X| = \binom{n}{2}$

Daemon Performs following Procedure when edge e is Probed.

Probe(e)

1.) if $X - e$ is Conn.
2.) $X \leftarrow X - e$
3.) answer NO
4.) else
5.) $Y \leftarrow Y + e$
6.) answer yes

Note: we think of X (and Y) as identified with $(V(K_n), X)$.

Notice:

- AT All times $Y \subseteq X$.
- $X - Y$ consists of exactly those edges of K_n which have not yet been probed.
- Both X, Y are consistent with Daemon's seq. of answers. Since if answer yes, add to Y & already in X, while if ans. no, remove from X & already not in Y.

claim: the following invariants are maintained over any seq. of edge probes.

(a) subgraph X is connected.
obvious from Probe() algorithm.

(b) If X contains a cycle, then none of its edges belongs to Y.

Proof: Deleting an edge from such a cycle would leave X connected, so such an edge would be removed from X and not added to Y.

(c) It follows from (b) that Y is acyclic at all times.

(d) If $Y \neq X$, then Y is disconnected.

Proof: Assume, to get a ~~X~~, that Y is connected. Being acyclic then, Y is a tree. Since $Y \neq X$, there exists an edge $e \in X - Y$, i.e. $e \in X$ but $e \notin Y$. If e were added to Y, it would form a cycle with the other edges of Y.

$\therefore$ X contains a cycle consisting of e, along with some edges of Y. This contradicts (b).

$\therefore$ we conclude that Y is disconnected.

Now suppose the Algorithm Halts and returns a verdict (conn/discon.) after doing fewer than $\binom{n}{2}$ edge probes. Then at least one edge of K_n was not probed, hence $X - Y \neq \emptyset$, hence $X \neq Y$.

By (d) Y is disconnected, while By (a) X is connected, and both are consistent with the seq. of answers. $\therefore$ Algorithm cannot be correct, $\therefore$ any correct Algorithm must do at least $\binom{n}{2}$ edge probes (in worst case.) ///.

Dynamic Programming

- See 15.1 - 15.5 2nd ed.
- also Brassard & Bratley.

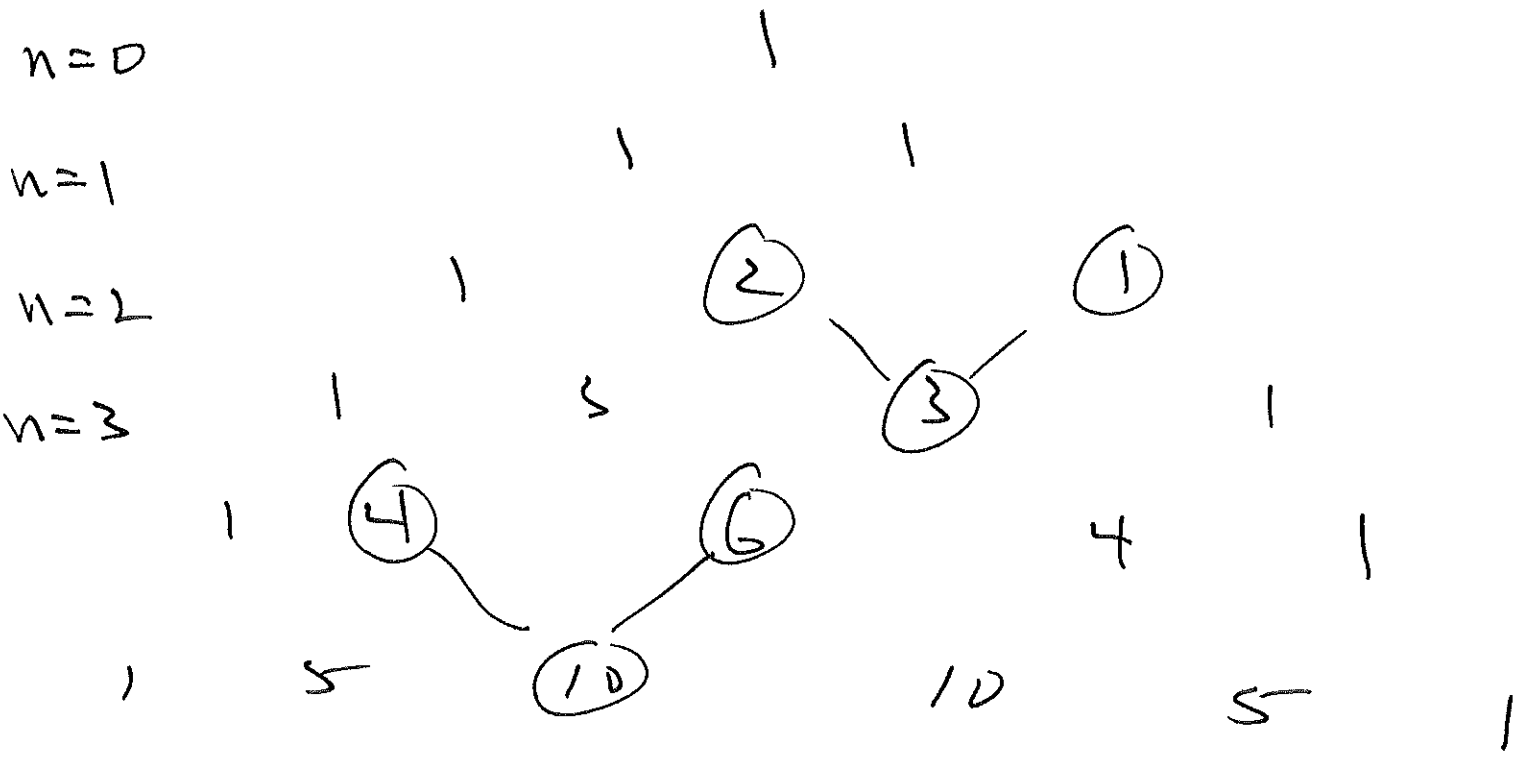
Ex.

Consider problem of computing $\binom{n}{k}$ where $0 \le k \le n$, using Pascals identity

$$\binom{n}{k} = \begin{cases} 1 & \text{if } k=0 \text{ or } k=n \\ \binom{n-1}{k-1} + \binom{n-1}{k} & \text{if } 0<k<n \\ 0 & \text{otherwise.} \end{cases}$$

Exercise: Prove this using $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

Pascals Triangle:

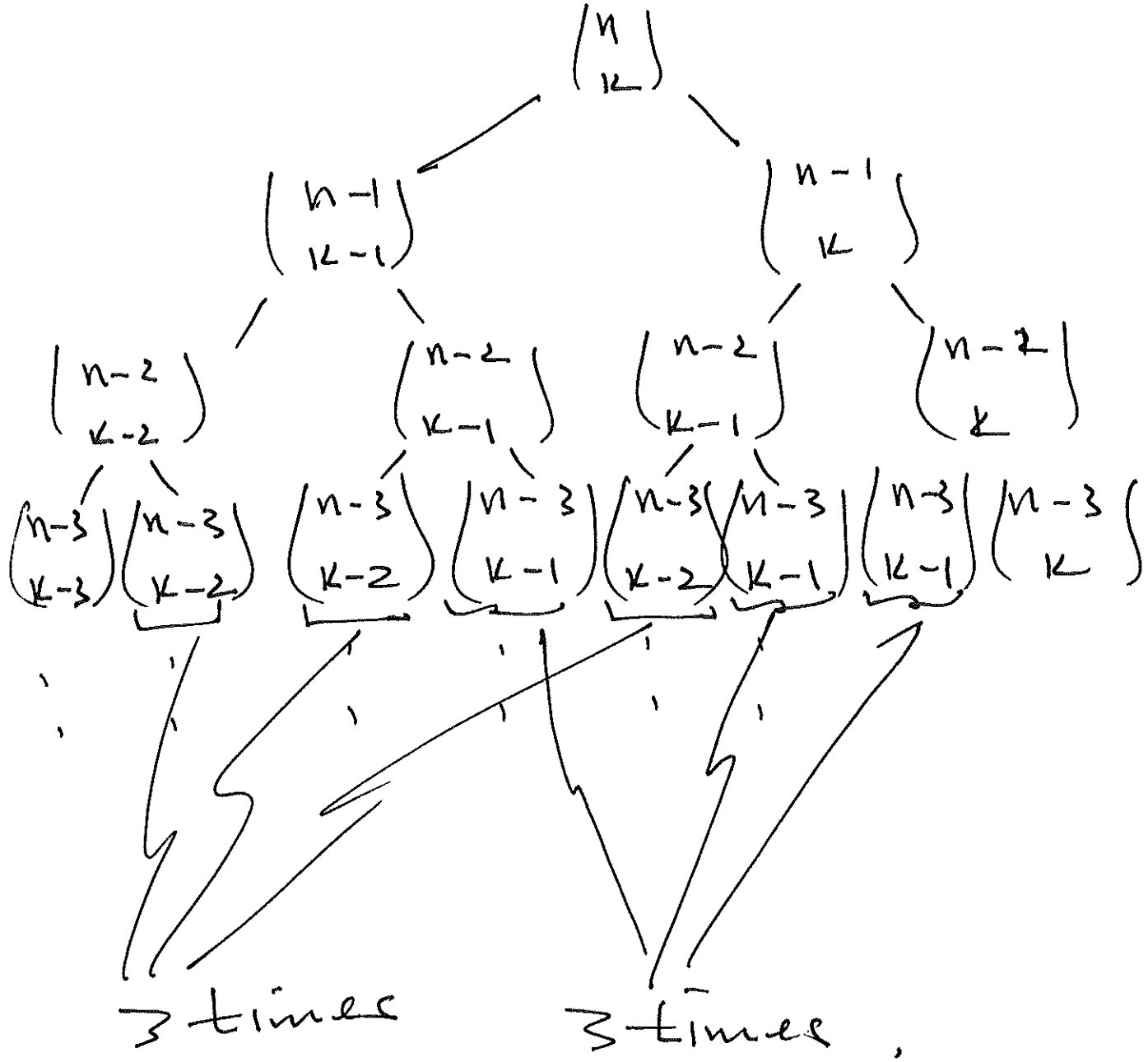


OBvious Recursive algorithm.

BinCoef(n, k)

1.) if k=0 or n
2.) return 1
3.) else if 0<k<n
4.) return BinCoef(n-1, k-1)+BinCoef(n-1, k)
5.) else
6.) return 0

the Problem is that subproblems are being solved multiple times



Notice that

$$\text{cost} = \text{Const.}(\#\text{recursive invocations})$$

$$= \Theta\left(\binom{n}{k}\right)$$

If $n = 2k$ then

$$\binom{n}{k} = \binom{2k}{k} = \Theta\left(\frac{4^k}{\sqrt{k}}\right)$$

↑ by Stirlings formula.

$\therefore$ RunCost(n, k) may be exponential in run time.

	0	1	2	3	.....	K-1	K
0	1	0	0	0	- . . .	0	0
1	1	1	0	0	. . . -	0	0
2	1	2	1	0	.....	0	0
3	1	3	3	1	- - - -	0	0
⋮	⋮	⋮	⋮	⋮		⋮	⋮
n-1	.....					$\binom{n-1}{k-1}$	$\binom{n-1}{k}$
n	- - - - - - - - - - - - - - - - - -						$\binom{n}{k}$

DynBinCoef(n, k)

1.) C[0] ← 1
2.) for i ← 1 to K
3.) C[i] ← 0
4.) for j ← 1 to n
5.) for i ← K down to 1
6.) C[i] ← C[i-1] + C[i]
7.) return C[K]

Exercise: refine this so as to gain efficiency by not adding zeros's leave out other unneeded calculations.

run time: $\Theta(n \cdot k)$

better than $\Theta\left(\binom{n}{k}\right)$

the Coin Changing Problem

Suppose we have coins in n denominations $\{d_1, d_2, d_3, \ldots, d_n\}$ where each $d_i \geq 1$ is an integer.

We wish to disburse an amount N using the fewest possible number of coins.

Assume: we have unlimited supply of coins in each denomination.

Two Questions:

- what is least # of coins needed to pay N units
- Exactly which coins (How many in each denomination) Are to be disbursed.