take opt-score(i,j) to be the best score obtainable

parenthesizing $M_i \cdots M_j$

clarification
(from last lecture)

Properties enabling dynamic programming

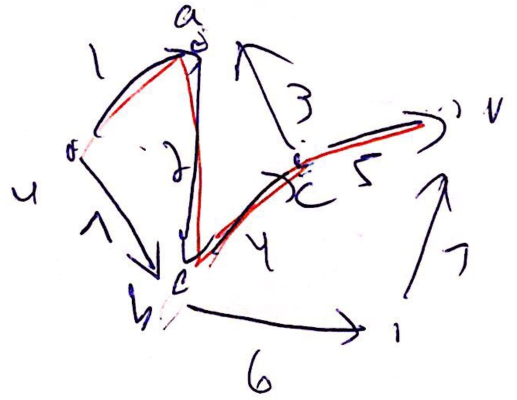
1. Divide-and-conquer applies
   - solutions to smaller problems are useful
   - solutions to " " can be found inside the larger problem solution
   - optimal substructure
2. memoization can use a polynomial size table
   - repeated subproblems (enough)

Allpairs shortest paths in graphs.

Given graph $G=(V,E)$ $V=\{v_1 --- v_n\}$

$E \subseteq V \times V$

$w(u,v) \in \mathbb{R}^+$ for $(u,v) \in E$



$$d[v_i, v_j] = \min_k d[v_i, v_k] + d[v_k, v_j]$$

$$d[v_i, v_i] = 0$$

infinite loop, no notion of problem size being reduced.

idea: bound # of hops to yield a problem "size"

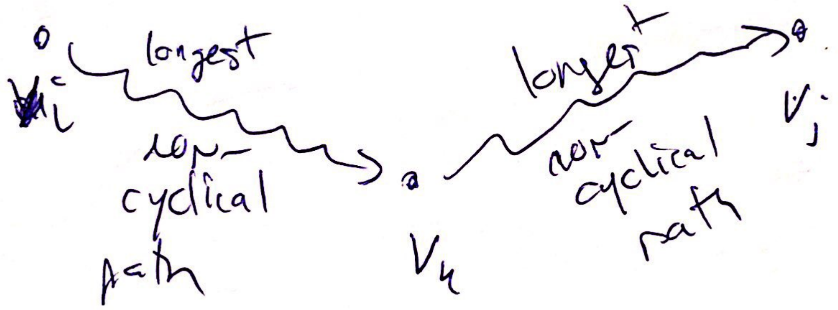
$d^h[i,j]$ = lowest cost to get from v_i to v_j in at most h hops

$1 \le h \le \lceil \lg n \rceil$ powers of 2
$1 \le i \le n$
$1 \le j \le n$ so $n^2 \lg n$ subproblems

then $d^h[i,j] = \min_k d^{h/2}[i,k] + d^{h/2}[k,j]$

$d^1[i,j] = $ ~~[illegible]~~, $w(v_i, v_j)$ or ∞ if $(v_i, v_j) \notin E$.

longest path "optimal substructure"



longest path problem is NP-hard!!

total path may be <u>cyclical</u>

— lost subproblem independence.

$$d_U^n[i,j] = \max_{k \in V} \max_{U' \subseteq V} d_{U'}^{n/2}(i,k) + d_{V-U'}^{n/2}(k,j)$$