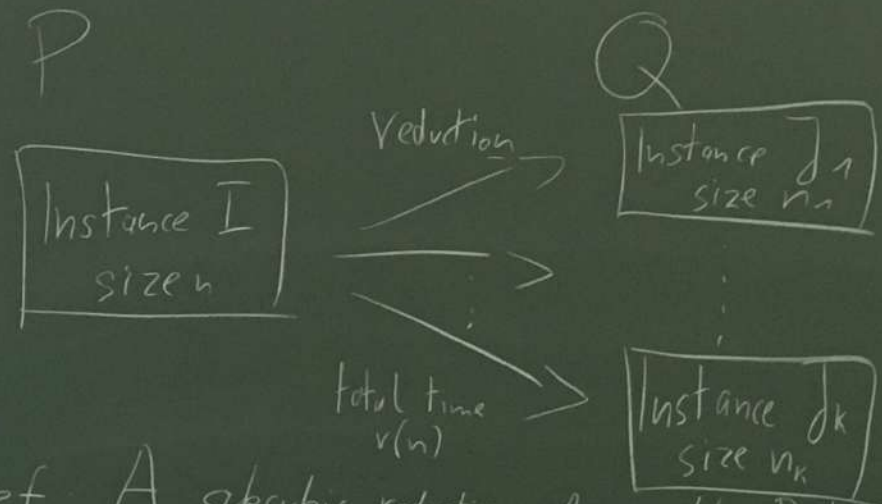
Lemma (Transitivity): If $A \leq B$ and $B \leq C$, then $A \leq C$.

If $A \leq B$, $B \leq C$, $C \leq A$, then A, B, C are "subcubic equivalent"

Consequence: If one of A, B, C admits an $O(n^{3-\varepsilon})$-time algorithm, then all of A, B, C do



Def.: A subcubic reduction from problem P to problem Q exists ($P \leq Q$) if there is an alg. $\mathcal{A}$ with oracle acces to Q s.t.

- For every instance x of P, $\mathcal{A}(x)$ solves problem P on x
- Exluding oracle calls, $\mathcal{A}$ runs in time $O(n^{3-\gamma})$ on instances of size n
- For every $\varepsilon > 0$ there is a $\delta > 0$ s.t. for every instance x of P of size n $\sum_i n_i^{3-\varepsilon} \leq n^{3-\delta\varepsilon}$. If $n = n_1 + \ldots + n_k$

n_i: size of instance of Q in i-th oracle call

Neg Triangle $\to$ APSP

$G \to G'$

$= (V,E)$ $\quad = (V^{(1)} \dot\cup V^{(2)} \dot\cup V^{(3)} \dot\cup V^{(4)}, E^{(1)} \dot\cup E^{(2)} \dot\cup E^{(3)})$

$V^{(i)} = \{v^{(i)} : v \in V\}$

$E^{(i)} = \{(u^{(i)}, v^{(i+1)}) : (u,v) \in E\}$

$w(u^{(i)}, v^{(i+1)}) = w(u,v)$

Observation $\forall v \in V$: $dist_{G'}(v^{(1)}, v^{(4)}) < 0$

$\iff$ G contains a negative-weight triangle including v

Reduction

- Construct G'
- Compute APSP on G'
- Look at $dist_{G'}(v^{(1)}, v^{(4)})$ for every node v
- Output 'yes' if $\downarrow < 0$ for some node v
- O.w. output 'no'

Given weighted directed graph ⊛

Task: $\forall$ nodes u,v compute (length of) the shortest path from u to v

Given $n \times n$ matrices A, B ⊛

Compute matrix C

$C[i,j] = \min_k (A[i,k] + B[k,j])$

$C = A \oplus B$

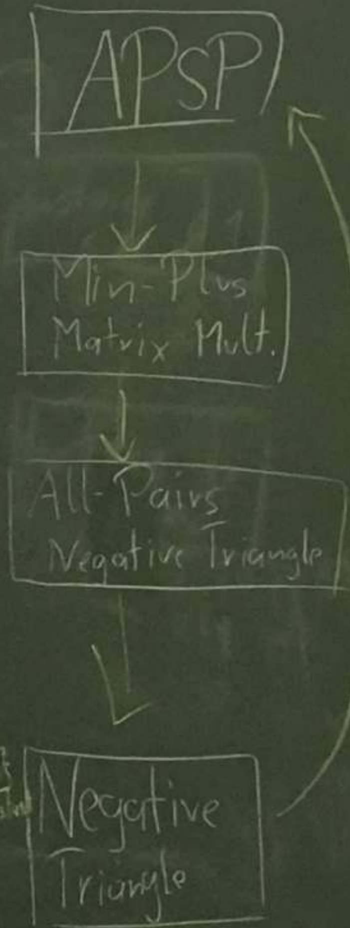
Given w.d. graph with node set

$V = I \dot\cup J \dot\cup K$

Task: for every $i \in I$, $j \in J$ decide if there is a neg triangle $i \to k \to j \to i$

Given weighted directed graph ⊛ weights: $\{-n^c, \dots, n^c\}$

Task: Decide if there is a triangle (c const.) of total negative weight



From All-Pairs Neg Triangle to Neg Triangle

First assume neg. triangle alg. also outputs one neg triangle (if it exists)

- Initialize C as $n \times n$ all-zeroes matrix

- Split each of I, J, K into $\leq \frac{n}{s}$ parts of size s

$I = I_1 \cup \dots \cup I_{n/s}$, $J = J_1 \cup \dots \cup J_{n/s}$, $K = K_1 \cup \dots \cup K_{n/s}$

$\longrightarrow \left(\frac{n}{s}\right)^3$ triples of the form (I_x, J_y, K_z)

- For each triple (I_x, J_y, K_z)

  While $G[I_x \cup J_y \cup K_z]$ contains a neg triangle

  Find a neg triangle $(i,k), (k,j), (j,i)$

  Set $C[i,j] = 1$

  Remove (j,i) from the graph

<u>Correctness</u>
- algorithm terminates (can remove at most n^2 edges)
- if $i\,k\,j$ is neg triangle, it will be found

Running Time

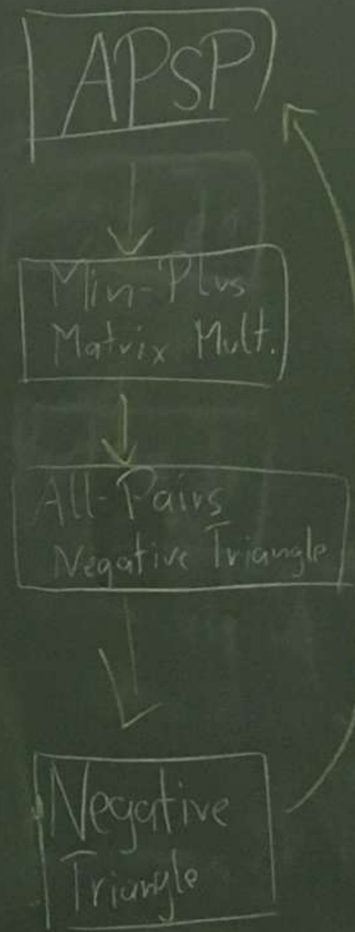
$\#\text{oracle calls} \cdot T_{\text{FindNT}}(s)$

$\leq (\#\text{triples} + \#\text{triangles found}) \cdot \text{Find}_{NT}(s)$

$\leq \left(\left(\frac{n}{s}\right)^3 + n^2\right) \cdot \text{Find}_{NT}(s)$

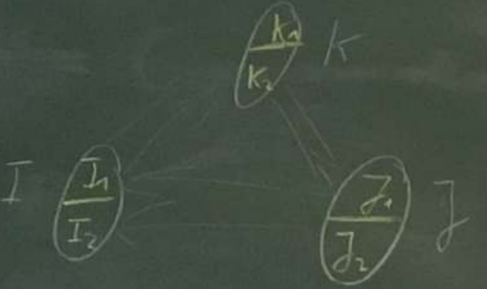
[Assume $\text{Find}_{NT}(n) = n^{3-\varepsilon}$
Set $s = n^{1/3}$]

$= O(n^2 \cdot (n^{1/3})^{3-\varepsilon}) = O(n^{3-\varepsilon/3})$



From All-Pairs Neg Triangle to Neg Triangle

Now: Finding Neg Triangle using decision algorithm.



Partition each node set into two sets of same size each

$I = I_1 \cup I_2 \qquad J = J_1 \cup J_2 \qquad K = K_1 \cup K_2$

For each triple (I_a, J_b, K_c)

check if subgraph $G[I_a \cup J_b \cup K_c]$ contains neg $\triangle$.

For one triple (I_a, J_b, K_c) that contains a neg triangle

Recurse on $G[I_a \cup J_b \cup K_c]$

[Base case: If I, J, K constant size then use "brute-force" approach to find and output neg $\triangle$]

Running Time

oracle calls $\cdot\ T_{FindNT}(s)$

$\le$ (#triples + #triangles found) $\cdot\ Find_{NT}(s)$

$\le \left(\left(\frac{n}{s}\right)^3 + n^2 \right) \cdot Find_{NT}(s)$

[Assume $Find_{NT}(n) = n^{3-\varepsilon}$

Set $s = n^{1/3}$]

$= O\left(n^2 \cdot (n^{1/3})^{3-\varepsilon}\right) = O\left(n^{3-\varepsilon/3}\right)$

Correctness

Running Time

$T_{FindNT}(n) \le 2^3 \cdot T_{DecideNT}(n)$

$+ Find_{NT}\left(\frac{n}{2}\right)\ O(n^2)$

$= O\left(\sum_{0 \le i \le \log n} T_{DecideNT}\left(\frac{n}{2^i}\right)\right) = O\left(T_{DecideNT}(n) + n^2\right)$