Advanced Algorithms

Lecture 11: Local search (contd.), shortest paths

Announcements

- HW 3 out today [Wednesday after fall break].
- Mid-term exam (read HW 3!) ⟶ Next Thursday in class.
  - data structures
  - divide and conquer, recurrences
  - dynamic programming
  - greedy algorithms, local search
  - basics of graphs (spanning tree, shortest path)

Last few lectures

- Greedy algorithms
- Examples we saw: spanning tree, scheduling jobs, "set cover" (approximation alg)
- Analysis can be tricky
- Other standard e.g.s: finding a "basis" for a vector space (lin alg 101), Huffman codes, ...

(Matroid theory).

Local search

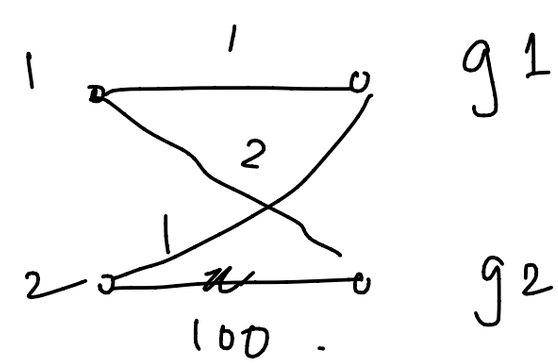
Key points

- Start with *any* solution, try improving by moving to "nearby" solution
- Stop if no nearby solution is better
- Any *local optimum* is actually "global" optimum (opt over domain)
- Convex optimization

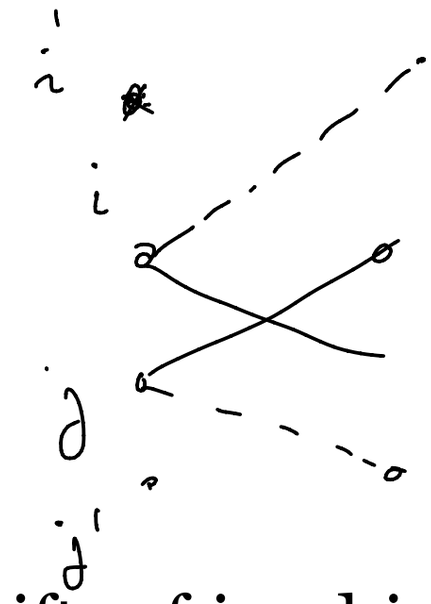
Matching problem

Problem: suppose we have *n* children and *n* gifts. Child *i* has some "happiness value" (V_{ij}) for gift *j*. Find an allocation (one gift per child) so that total happiness is maximized.

→ Greedy alg works poorly



Local search



- Start with *any* solution
- For every pair *i*,*j* of children, check if swapping the gifts of *i* and *j* improve solution; if so, make the swap
- Stop if no swap improves solution

$\longrightarrow$ $n^2 \times$ # of swaps.

Local search

- Start with *any* solution
- For every pair *i,j* of children, check if swapping the gifts of *i* and *j* improve solution; if so, make the swap
- Stop if no swap improves solution

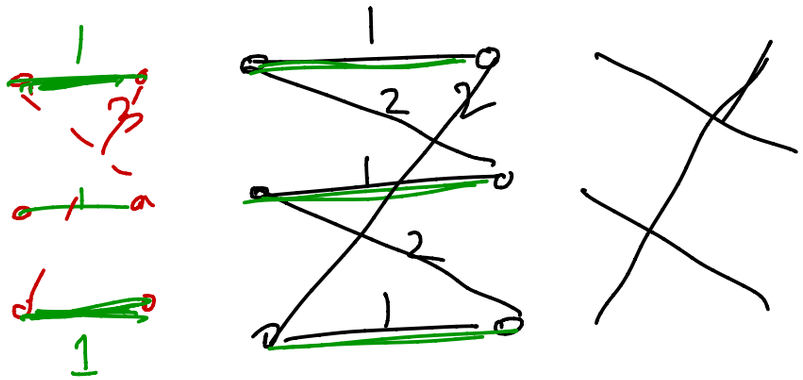
Formally…

$\sigma_1, \sigma_2, \ldots, \sigma_m$.

$$\sigma_i := \text{gift to child } i$$

for all *i, j*

$$V_{i,\sigma_i} + V_{j,\sigma_j} \geq V_{i,\sigma_j} + V_{j,\sigma_i}$$



Local search

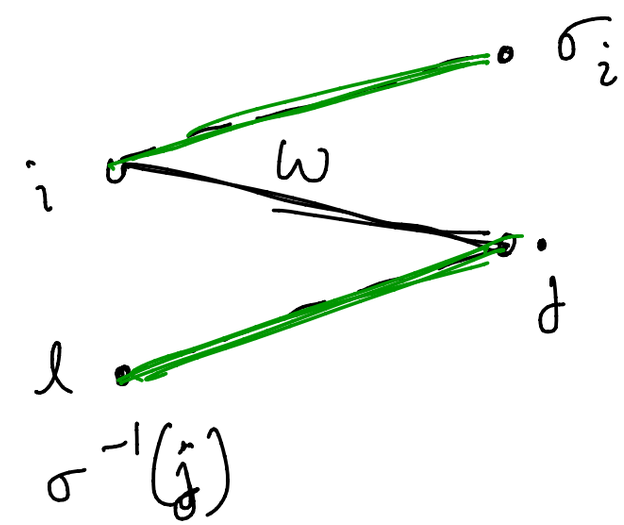
$S: (\sigma_1, \sigma_2, \ldots, \sigma_n)$.

some locally opt assignment.

Claim: take any solution S in which swaps do not increase value. Then total happiness of S >= (1/2) total happiness of OPT solution

How do we reason about OPT?

HINT: Can we "miss" a very high wt edge?



Claim: $V_{i,\sigma_i} + V_{l,\sigma_l} \geq w$.

Proof:

Local optimality gives that

$$V_{i,\sigma_i} + V_{l,j} \geq V_{ij} + \underbrace{V_{l,\sigma_i}}_{\geq 0} \geq w$$

$\sigma_l = j$.

2 approximation – proof

$\sigma_1, \sigma_2, \ldots, \sigma_n$.

Claim: take any solution S in which swaps do not increase value. Then total happiness of S >= (1/2) total happiness of OPT solution

$\gamma_1, \gamma_2, \ldots, \gamma_n$

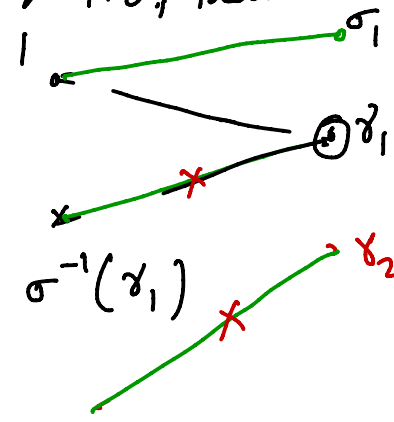
(in OPT solution, child i is assigned gift γ_i).

$\Downarrow$. Claim:

$$2\left(V_{1,\sigma_1} + V_{2,\sigma_2} + \ldots + V_{n,\sigma_n}\right) \geq \cancel{\frac{1}{2}}\left[V_{1,\gamma_1}, V_{2,\gamma_2}, \ldots, V_{n,\gamma_n}\right]$$

if all wts are between 0.1 & 1.0, then.

$$V_{1,\sigma_1} + V_{\sigma^{-1}(\gamma_1),\gamma_1} \geq V_{1,\gamma_1} + \dots$$



$$V_{2,\sigma_2} + V_{\sigma^{-1}(\gamma_2),\gamma_2} \geq V_{2,\gamma_2}$$

$$\vdots$$

$$\sum_i V_{i,\sigma_i} + \sum_i V_{\sigma^{-1}(\gamma_i),\gamma_i} \geq \sum_i V_{i,\gamma_i} \iff 2\sum_i V_{i,\sigma_i} \geq \sum_i V_{i,\gamma_i}$$

Basic tools – a review

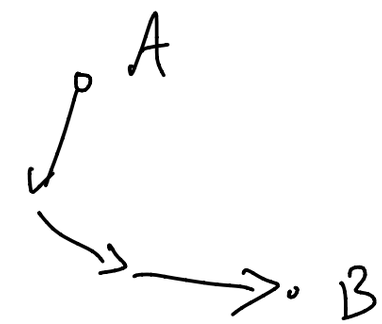
Review

- Divide and conquer / recursion — inductive analysis, recurrences
- Dynamic programming (store answers to sub-problems)
- Greedy algorithms (easy to design + implement, often hard to reason), local search.

- Using randomness in algorithm design
- Optimization and linear programming

Finding the shortest path

Problem



n m

Problem: given a (directed) graph $G = (V, E)$ with edge lengths (> 0), find shortest length path from u to v.

(*)

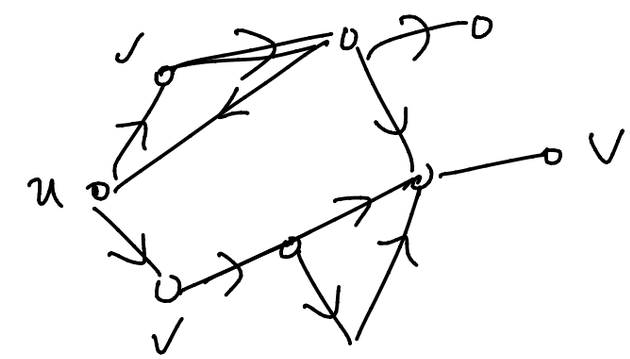
- Brute force — exponentially many paths
- Saw a dynamic programming solution — paths with L "hops" —> O(L(m+n)) time [Bellman-Ford alg.]

Sub-problem: length of shortest k-hop path from w to v (for all w)

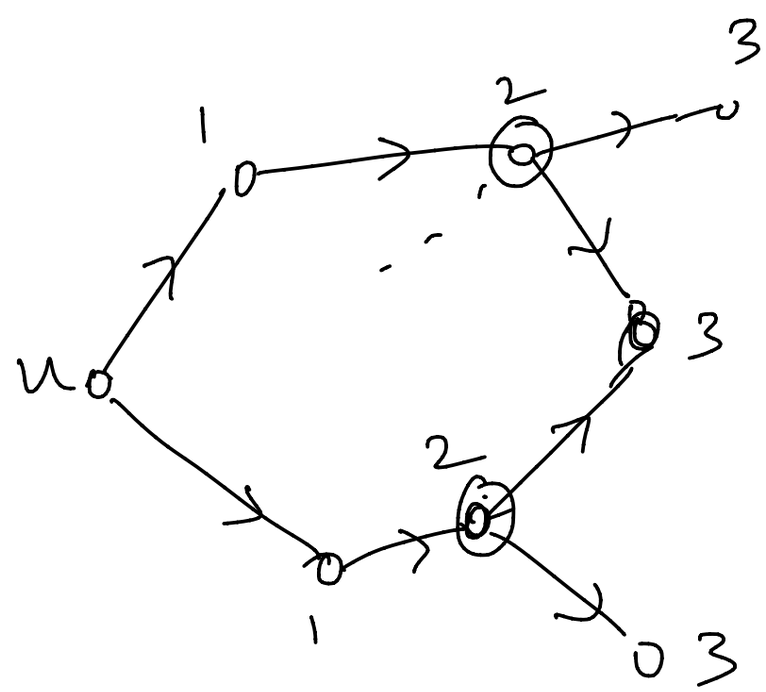
$n(m+n)$ if used for problem (*).

Can we do better?

Problem: given a (directed) graph $G = (V, E)$ with edge lengths (> 0), find shortest length path from u to v.



What about un-weighted graphs?



Growing a region around u of increasing radius.

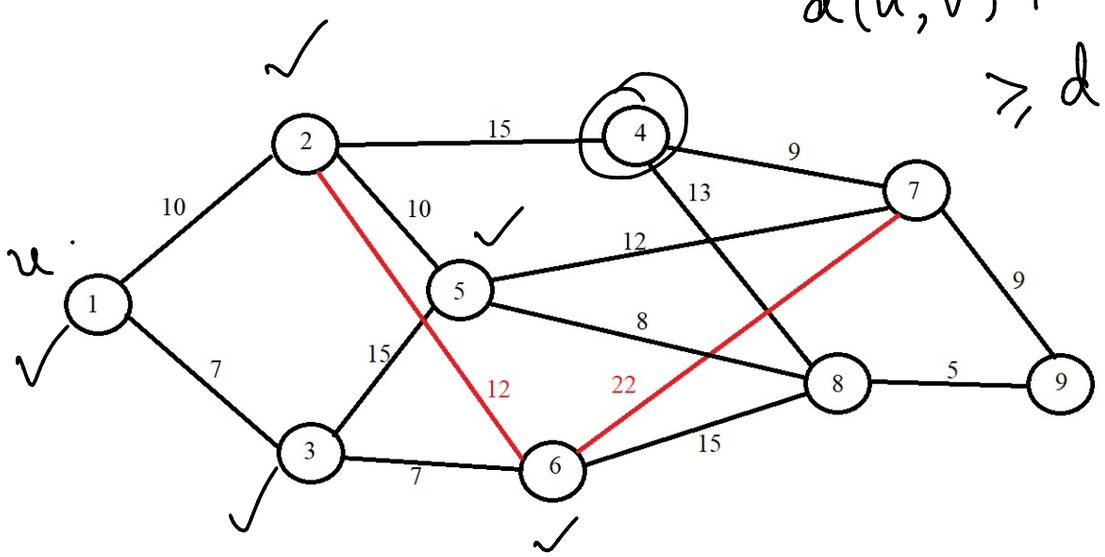
$O(m+n)$.

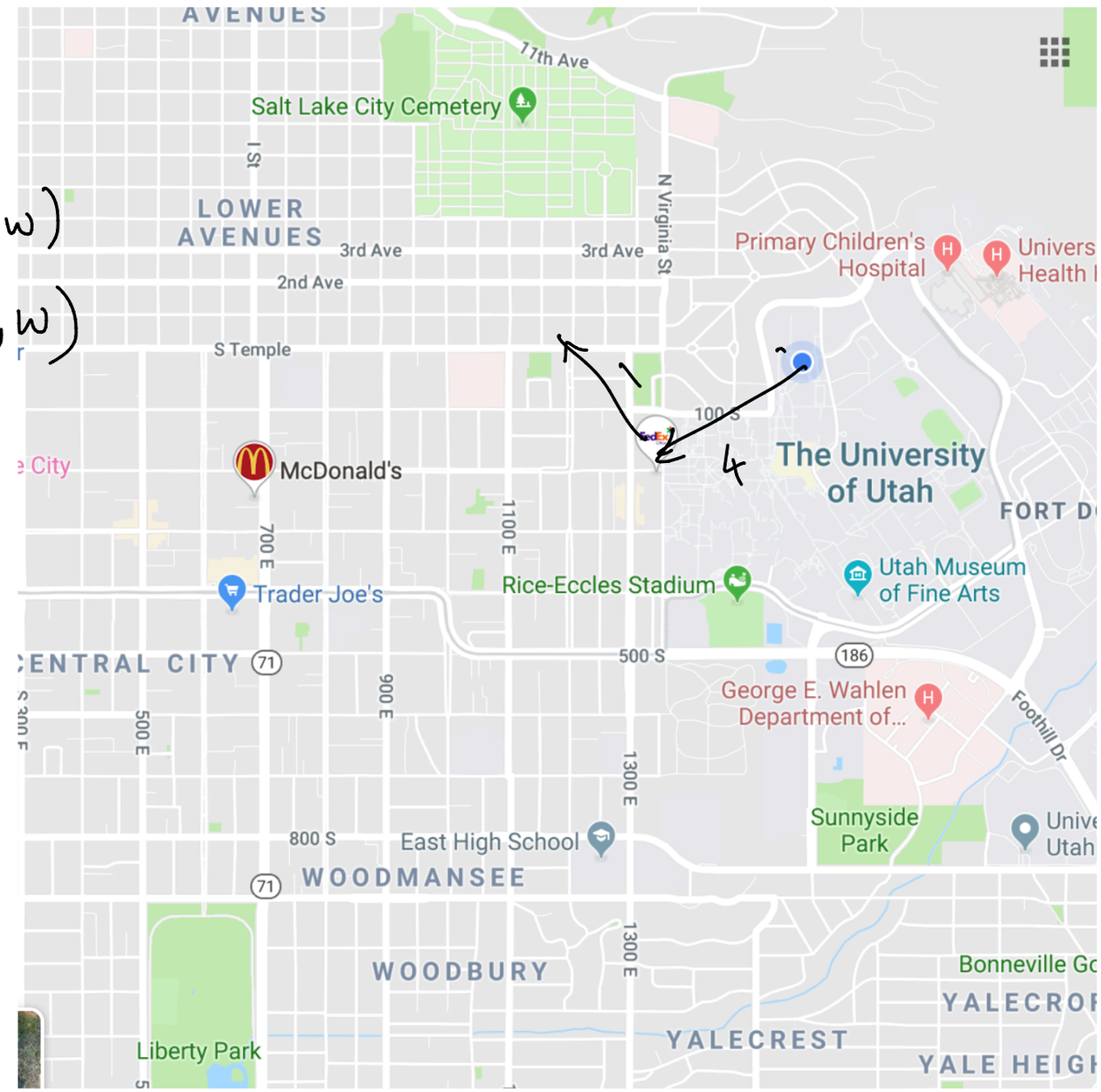
Construct ball of radius ‘r’?

- Can we find shortest path to all vertices of “distance” < r from u?
- Graphs as “metrics”

$d(u,v)$

$d(u,v) + d(v,w) \geq d(u,w)$

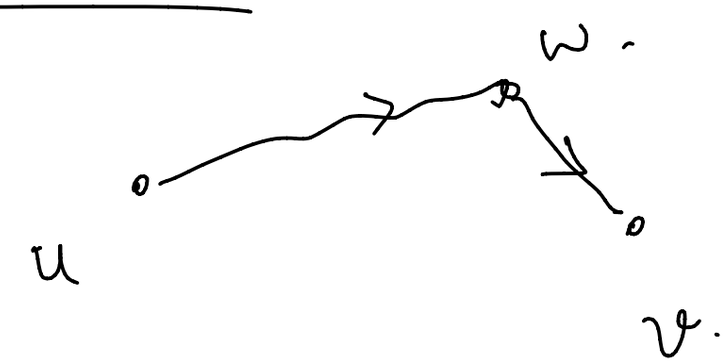




What all vertices are at dist ≤ 20 from u?

Construct ball of radius ‘r’?

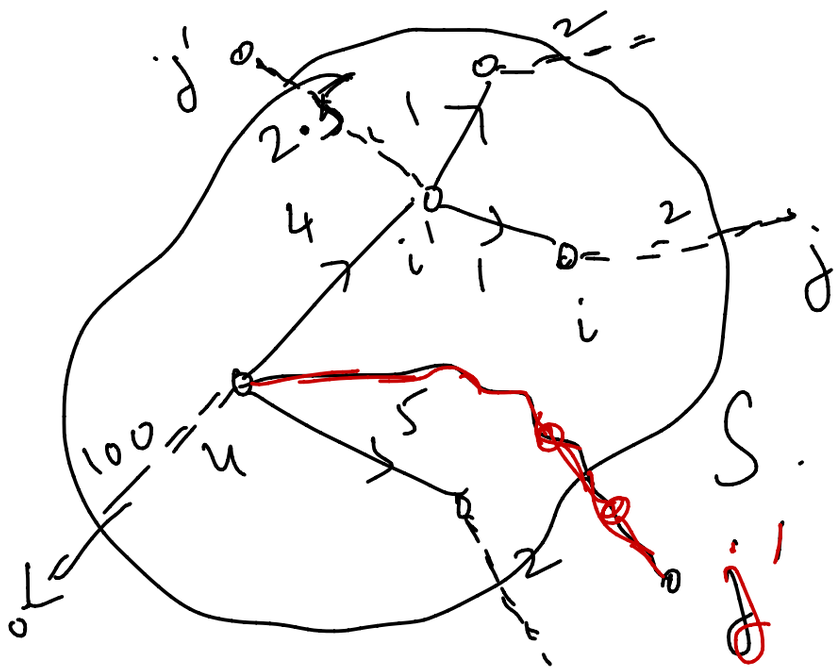
- Can we find shortest path to all vertices of “distance” < r from u?



- Only way to have path of length r ==
    - direct edge of length r to v
    - path to vertex at distance < r, followed by edge

Incrementing the ball

- What is the "next closest" vertex?



→ Let S be the current ball

→ For all the edges ij out of S, compute $d(u,i) + l_{ij}$

→ Pick the j that has the least such value, add j to S.

<u>Claim:</u> j is the closest vertex outside S to u.

Proof by contradiction

- Let j' be a closer point to u (if poss.).
- Look at path from $u \to j'$.

Dijkstra’s algorithm

Running time

Deja vu – Prim's algorithm