Lecture 12: A new set of Techniques (Facility Location Problems)

So far we've seen:—

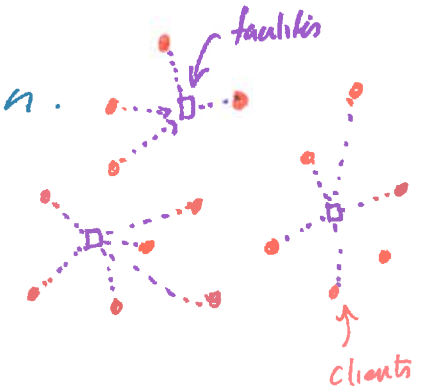
- Basic relax and round (LPs, SDPs)
- Greedy algos.
- Region growing (Leighton Rao) & Embeddings.
- Spectral Techniques.

Now let us consider a new collection of Ideas.

- "Primal Dual"
- "Local Search"
- And some more LP rounding beyond basic random rounding.
  & Greedy

Showcase on a new problem: Facility Location / k-median.

For both problems, basic setup —



- Metric space (V,d), usually finite $|V|=n$.
- a client set $C \subseteq V$.
- facility costs $f_i \geq 0 \;\forall i \in V$

Fac Loc: want to open some set $F \subseteq V$

s.t. $\min \sum_{c\in C} d(c,F) + \sum_{i\in F} f_i$

k median: open $|F|=k$ that

minimizes $\sum_{c\in C} d(c,F)$

- Problems seem related (and they are), more on that later
- Also, if $d(c,F)$ replaced by $[d(c,F)]^2$, then **k-means** (commonly studied).

Today we focus on <u>k-median</u> and <u>local search</u>

<u>Algo</u>: start with any "reasonable" solution

while ∃ a move that improves the solution, // if cost(new) < cost(old)

take it (or take the best such move).

what are <u>moves</u>? given a solution $F \subseteq V$

[swap(u,v) $u \in F$, $v \notin F$

move to solution $F - u + v$. ~~if it has lower~~]

can also consider p-swaps

$U \subseteq F$, $V \cap F = \emptyset$ $|U| = |V| = p$, then move to $(F \setminus U) \cup V$

Care about values of $p = O(1)$.

<u>Thm</u>:

Any local optimum wrt ① swaps has cost at most $5 \cdot OPT$

② p-swaps has cost at most $(3 + 2/p)$ OPT.

<u>Thm</u>:

$\forall \varepsilon > 0$ ∃ instances where the ~~[illegible]~~ locality gap is at least $3 - \varepsilon$ wrt any p-swaps

i.e. ∃ local optima whose cost $\geq (3 - \varepsilon) \cdot OPT$ p fixed.

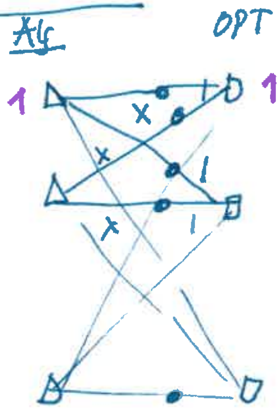
This theorem does not say anything about convergence.

However, by starting at a "reasonable" solution
and performing the best swap (or at least, any swap that makes "large" progress)
can give $3+\varepsilon$ approximation in $n^{O(1/\varepsilon)}$ time. [Thm]

Will see this given time (or will see as an exercise).

Let's first see how to get the 5apx and the lower bound of 3.

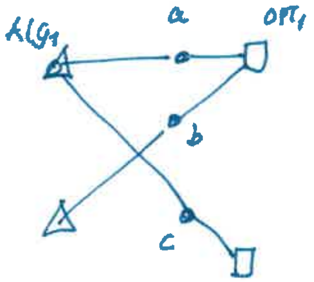
Lower bound.



Distances given by shortest path distances.

(Imagine a ~~bipa~~ complete bip. graph, on OPT × ALG, and put a client at distance x from ALG, 1 from OPT.)

Sps close ALG_1 open OPT_1



type a: old cost x new cost 1
type b: —— x new cost 1
type c: —— x —— x+2.

so k clients give improvement of $(x-1)$
$(k-1)$ ——— loss of $(x+2)-x = 2$

$\Rightarrow$ total ~~change~~ improvement is $k(x-1) - (k-1)2$

$= k(x-3)+2$

$\Rightarrow$ if $x = 3 - 2/k$ this is a local optimum. ☺

[Thm] $\forall \varepsilon, \exists k$ large enough s.t. local opt wrt 1-swaps has cost $\geq 3-\varepsilon$.
(similarly can set ——— wrt p-swaps also cost $\geq 3-\varepsilon'$)

Now for the algorithm's performance

Sps ALG at a local OPT, what is the cost?

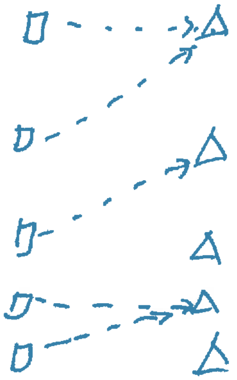
All swaps are non-improving.

generic ideas used in most local search analysis.

[So show a set of swaps, that, because of non-improvement, give bounds on the cost of ALG vs OPT.

- Simpler "bijective" case

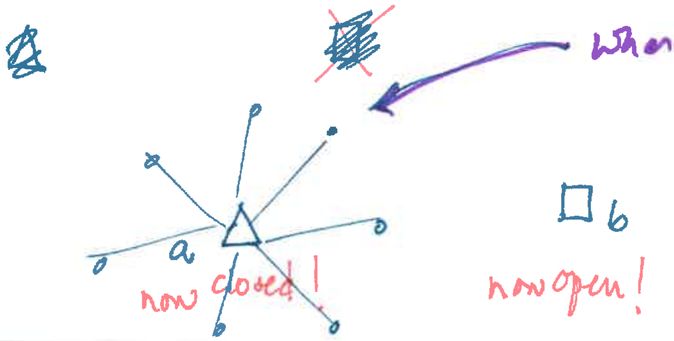
for each fac b ∈ OPT, let $\pi(b)$ be closest ALG facility to b.



bijective case when this map π: OPT → ALG is a bijection.

Consider the swaps where we open some b ∈ OPT and close $\pi(b) = a \in$ ALG.

What does non-improvement tell us?



Where do these clients go? to b? to someone else?

All clients assigned to b in OPT, let's assign them to b.

Notation: $C^*(b)$: clients assigned to b in OPT's soln.

$C(a)$: clients assigned to $a \in$ ALG's solution

O_j : cost of client j in OPT

A_j : ―――――――― ALG

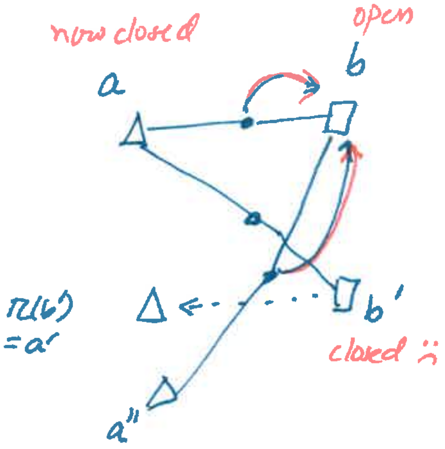
$\Rightarrow$ All clients in $C^*(b)$, assign them to b.

Δ COST $= (O_j - A_j)$

All clients in $C(a) \setminus C^*(b)$ where to assign them?

Sps. $j \in \nearrow$ and $j \in C^*(b')$.

so assign to a' (which is open, by bijective)
$= \pi(b') \neq \pi(b)$



$$\Delta \text{COST} = d(j, a') - A_j$$

$$\leq d(j, b') + d(b', \pi(b')) - A_j$$

$$\leq d(j, b') + d(b', a) - A_j \quad \text{b/c } \pi(b') \text{ is closest to } b' \text{ in ALG}$$

$$\leq d(j, b') + d(b', j) + d(j, a) - A_j \quad \text{by } \Delta\text{-ineq.}$$

$$= 2O_j + \cancel{A_j} - \cancel{A_j}$$

$$= 2O_j$$

$$\Rightarrow \underset{\text{of }(a,b)\text{ swap}}{0 \leq \Delta\text{Cost}} = \sum_{j \in C^*(b)}' (O_j - A_j) + \sum_{j \in C(a) \setminus C^*(b)}' 2O_j$$

$$\leq \sum_{j \in C(b)}' (O_j - A_j) + \sum_{j \in C(a)}' 2O_j$$

And this must be non-negative

Now sum over all $(\overset{\pi(b)}{\overset{\|}{i}}, b)$ swaps, (and use that $j \in C^*(b)$ for some b and $j \in C(\pi(b))$ for some b)

$$\sum_j (O_j - A_j) + \sum_j 2O_j \geqslant 0.$$

$$\Rightarrow 3OPT - ALG \geqslant 0.$$ ☺

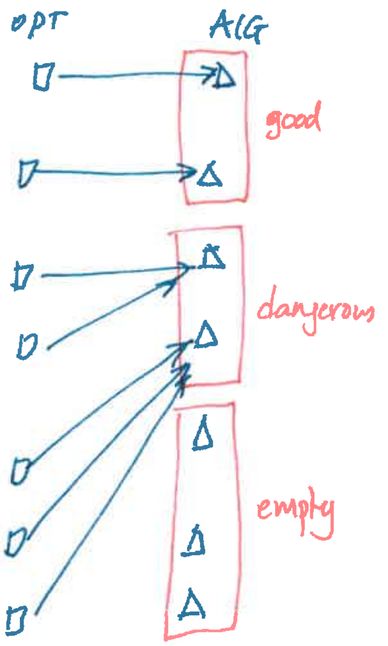
But used bijective property. How to handle case where $\pi(b) = \pi(b')$ for some $b \neq b'$

~~Call an alg facility~~ Draw the graph on OPT x ALG of pairs $(b, \pi(b))$

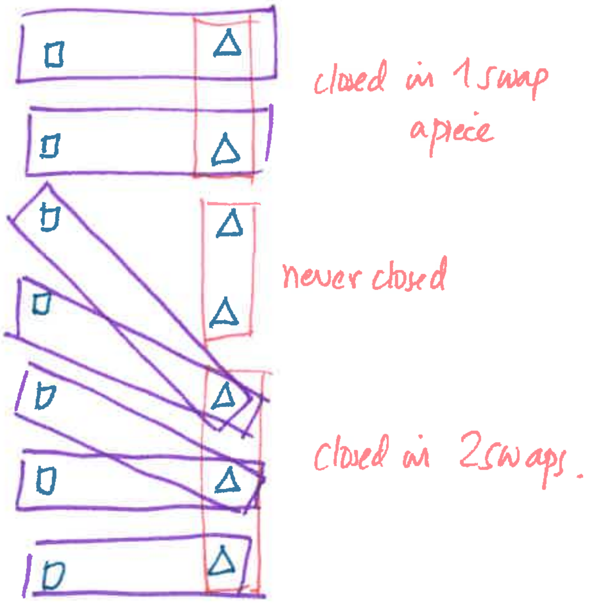
- has k edges.
- every node on left has degree 1. (good)
- so some nodes on right have degree 1.
- but # of nodes on right with degree 0 (empty) $\geqslant$ # nodes on right with degree $\geqslant 2$. (dangerous)

. so pair each b to $\pi(b)$ if $\pi(b)$ good

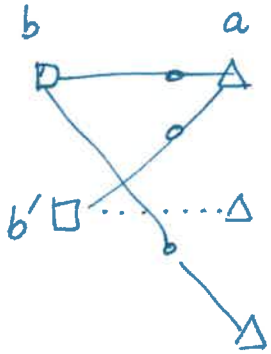
. pair all other b to some empty alg facility st. each empty used in 2 pairs



so. in this example,



For each swap above, open b, close a



, $\forall j \in C^*(b)$, change is $O_j - A_j$

, $\forall j \in C(a) \setminus C^*(b)$, assign to ~~[illegible]) which is never~~

say $j \in C^*(b')$ $\quad \pi(b')$ which by our construction is never a.

and so is open.

Again same argument says change is $\leq (2O_j + A_j) - A_j \leq 2O_j$

$$\text{so.} \sum_{(a,b)\,\text{swaps}} \Big(\overbrace{\sum_{j \in C^*(b)} (O_j - A_j) + \sum_{j \in C(a) \setminus C^*(b)} 2O_j}^{\text{non } \cancel{\text{zero}} \text{ negative b/c non improving}} \Big) \geq 0.$$

$$\Rightarrow \sum_{\substack{j \in C^*(b):\, b \text{ opened in} \\ \text{swap list}}}' (O_j - A_j) + \sum_{\substack{j \in C(a): \\ a \text{ closed in} \\ \text{swap list}}} 2O_j \geq 0.$$

but each b opened once <u>exactly</u>

each a closed $\leq$ twice

$$\Rightarrow \sum_{j \in C}' (O_j - A_j) + 2\sum_{j \in C}' 2O_j \geq 0$$

$$\Rightarrow Alg = \sum_{j \in C}' A_j \leq 5 \sum_{j \in C}' O_j = 5\,OPT$$

How to get poly time algorithm:

2 pieces

(I) make big steps when possible

(II) start at reasonable sol^n.

(I) Same, kind of argument says:

Generic Ideas: used in most local search

Sps. $Alg \geq (5+\delta) OPT$

$\Rightarrow$ at least one of the k swaps we show have improvement $\geq \delta/k$ OPT.

if we take best swap.

$\Rightarrow$ improve by at least $\left(\frac{ALG - 5OPT}{k}\right)$

i.e. $ALG(new) \leq ALG(old) - \left(\frac{ALG_{old} - 5OPT}{k}\right)$

~~hence can get from~~

Hence decrease "distance to 5OPT" by a constant factor in k steps.

and hence get

$$ALG \text{ after } t \text{ steps} \leq 5OPT + (ALG_{initial} - 5OPT)\left(1-\frac{1}{k}\right)^t$$

$$\leq 5OPT + ALG_{init}\left(1-\frac{1}{k}\right)^t$$

(II) Reasonable solution:

Pick some solution with cost $ALG_{init} \leq n \cdot OPT$

then get $(5+\varepsilon)OPT$ after $t = \frac{k}{\varepsilon} \log n$ steps at most.

How? Pick any ~~solution~~ facility f_1 to start. $F_1 \leftarrow \{f_1\}$

for ~~each~~ $i = 2 \ldots k$

Choose f_i at largest distance from F_{i-1}

Exercise: n approximation.

Other, better starting solutions exist. (Exercises)

Recap :

Local search

- simple algo idea
- define set of moves such that
  - (a) can find improving move in poly time (or best impr. move)
  - (b) local optima wrt this set of moves is good.

Widely used in practice

- often no guarantees of performance of local optima
- often convergence time may be unbounded / exponential
  (even to get to near-optimum solns).

In theoretical analyses

- show some subset of moves that help compare ALG to OPT.
- several nice applications
  - even for exponentially large swap sets. (capacitated fac. loc.)
    (labeling problems in vision)
  - first algos for bounded-degree spanning trees, k-median w/ outliers.
- clean analyses for clean algorithms.