Primal Dual Algorithms (Lecture 14)

- Last lecture we saw how to use both primal & dual LPs to round a primal solution (after having solved the primal/dual LP pair).

- <u>today</u>: create solutions to the P/D pair by hand, ~~and~~ such that

  1. Primal and dual are feasible
  2. Primal is integer solution
  3. cost of primal $\leq \rho \cdot$ cost of dual

     for minimization (primal) problem

$\Rightarrow$ primal is ρ-apx.

(why? cost(primal) $\overset{(3)}{\leq}$ ~~opt LP primal~~ $\rho \cdot$ cost(dual)

$\leq \rho \cdot$ cost(optimal primal) $\leq \rho \cdot$ cost(opt integer)

weak duality relaxation

Let's warm up with a ~~simpler~~ more familiar problem.

<u>Vertex cover</u>. Pick set of vertices that hit every edge. $G = (V, E)$.

- HW1 gave greedy algo.

  Pick an edge uncovered until now. Pick both endpoints repeat.

- But what about the case when vertices are weighted? Want to pick least weight solution that hits all edges.

Let's give a primal-dual solution for this.

<u>Pf of 2apx</u>: opt soln $\geq$ ~~maximal~~ any matching in graph and we give matching & soln of cost ≤ 2 matching. ☺

LP: $\min \sum_{u \in V} c_u x_u$

st $x_u + x_v \geq 1 \quad \forall uv \in E$

$x_u \geq 0.$

dual $\max \sum y_e$

st $\sum_{e: v \in e} y_e \leq c_u \quad \forall u \in V$

$y_e \geq 0.$

Note: if $c_u = 1$ then any matching M is a feasible solution to dual (set $y_e = 1 \Leftrightarrow e \in M$)
and we have a solution (both endpoints of this maximal matching)
of cost $\leq 2M$.

So we are also being primal-dual.

More generally, for costs: start with $x = 0$, $y = 0$.

Pick an edge e, $\boxed{\text{raise } y_e}$ until some dual constraint becomes tight

Pick all vertices whose dual constraint is tight $\boxed{\text{set } x_v = 1}$

Until solution S is feasible

note: if $c_u = 1 \; \forall u$, gives the greedy algo above!

Clearly ① P/D feasible, ② P integral.

$$\text{Primal} = \sum_{v \in C} c_v = \sum_{v \in C} \sum_{e: v \in e} y_e = \sum_e y_e \cdot \sum_{\substack{v \in C: \\ v \in e}} 1 \leq 2 \cdot \sum_e y_e = 2 \cdot \text{dual}.$$

$\Rightarrow$ 2apx.

What's happening?

① hand crafting dual solution, raising "money",

and trying to ~~[illegible]~~ maintain some kind of complementary slackness

(if primal var $> 0 \Rightarrow$ dual constraint is tight).

Here we could raise a single dual var at a time.
For other problems, may help to raise many vars. simultaneously.

(2) each edge offers money to vertex. "Tight" vertices can open.

But each edge offers money to both its endpoints, so double dipping

that's the factor of 2. ☺.

Often the primal dual algos will have this double-dipping flavour, which is where the approx. arises.

Approx to facility location

$$\min \sum_i f_i y_i + \sum_{ij} d_{ij} x_{ij}$$

(P) st $x_{ij} \le y_i \quad \forall ij$

$\sum_i x_{ij} \ge 1 \quad \forall j \in C$

$x, y \ge 0.$

$$\max \sum_{j \in C} \alpha_j$$

(D) st $\alpha_j - \beta_{ij} \le d_{ij} \quad \forall ij$

$\sum_j \beta_{ij} \le f_i \quad \forall i$

$\alpha, \beta \ge 0.$

Interpretation: $\boxed{\alpha_j = \text{money that client } i \text{ is } \text{~~raising~~ offering to be connected.}}$

out of that same fraction (d_{ij}) is to be connected to facility @ j

and β_{ij} is to open the facility.

The total offering $\sum_j \beta_{ij}$ should never be more than the cost of the location i (f_i)

else we can open it.

What is the most we can raise this way? That's the dual.

So now we want to

(a) raise this money, and

(b) build a solution

} simultaneously!

Here's how we will do it ...

- Each client grows α_j (simultaneously)

  contribution $\beta_{ij} = (\alpha_j - d_{ij})^+ =: \max(\alpha_j - d_{ij}, 0)$.

- When $\sum_j \beta_{ij} = f_i$, "tentatively open" facility @ i.

- if j "touches" tentatively open facility (i.e. $\alpha_j \geq d_{ij}$ for some such i),

  stop growing α_j

Stop when all α_j stop growing.

---

do the animation via PPT

Immediate problem:

- Many facilities may be tentatively open. ☹

  E.g. single client, but many facilities at dist 1 of cost 1.

  All will tentatively open @ same time. ("double dipping" → "many times dipping")

- ☺ Use the metric property to close many of these.

~~Cleanup~~ Clean-up Phase

Consider tentatively opened facilities (call it F').

Put edge between i and $i' \in F'$ if $\exists$ client j that has $\beta_{ij} > 0$, $\beta_{i'j} > 0$

non-zero contribution to both i & i'.

Pick a maximal independent set of F', call it F.

Great! Removed double-dipping by construction, but what about cost of solution? Not too bad, we claim...

What's the cost?

Let's assign clients to open facilites in F.

(a) if $\beta_{ij} > 0$ for $i \in F$ then j "contributes" to facility $i \in F \Rightarrow$ assign j to $N(i)$.

<u>Note</u>: by construction of F from F', each client contributes to only one $i \in F$.

(b) if j not assigned above, and if $d_j = d_{ij}$ for some $i \in F$

then again assign j to $N(i)$.

For remaining clients j, neither facilities i s.t. j contributed to them nor —— i s.t. $d_j = d_{ij}$ are open.

Call these lonely clients.

<u>Lemma (3hop)</u>: For a lonely client j, $\exists$ open facility at distance at most $3d_j$ in F. (Proof soon)

<u>Given this lemma, let's prove 3-approx:</u>

- Cost of non-lonely clients + open facilites

$$= \sum_{i \in F} \Big(f_i + \sum_{j \in N(i)}' d_{ij} \Big) = \sum_{i \in F} \Big(\sum_{\substack{j \in N(i): \\ j \text{ contributes to } i}}' \beta_{ij} + \sum_{j \in N(i)}' d_{ij} \Big)$$

$$= \sum_{i \in F}' \sum_{j \in N(i)}' d_j = \sum_{j \text{ non-lonely}}' d_j$$

- Cost of lonely clients

$$\sum_{j \text{ non-lonely}}' d(j, F) \leq 3 \cdot \sum_{j \text{ lonely}}' d_j \qquad \Rightarrow \text{total cost} \leq 3 \cdot \sum_j' d_j$$

$$\leq 3 \cdot OPT$$

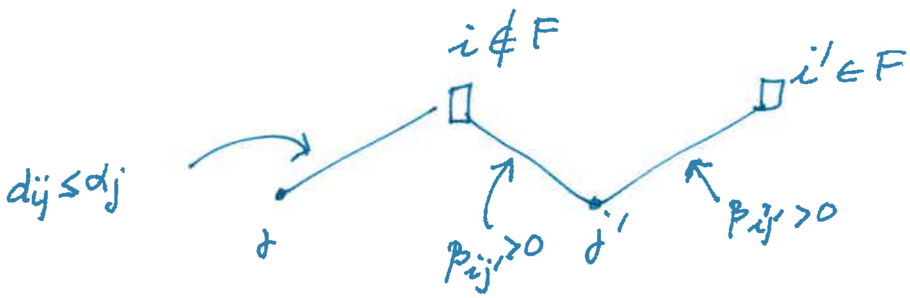
Proof of 3-hop Lemma

Sps j lonely. Consider the dual increase procedure.

~~Either~~ We stopped raising α_j because of some ~~open~~ facility i that was tentatively opened (in F').

Now i must be closed in F (because j is lonely).

So i closed because of j' contrib to i and i' both and i' open in F.



$$d(j,F) \le d(j,i) + d(i,j') + d(j',i')$$

$$\le \alpha_j + \alpha_{j'} + \alpha_{j'}.$$

Now: we claim that $\alpha_{j'} \le \alpha_j$

Since we raise all duals simultaneously, and α_j stopped raising b/c of i, $\alpha_{j'}$ would stop b/c of i (~~if~~ if it had not stopped earlier)

(was tentatively opened @ time α_j if not before)

$\Rightarrow \alpha_{j'} \le \alpha_j$

This completes the proof of 3-hop lemma

$\square$

Finishes fac location proof.

Shows that cost of primal solution $\leq 3.\sum' \alpha_j = 3$ Dual solution

$\leq 3.$ OPT

weak duality

<u>Aside:</u> can use to show slightly stronger guarantee —

that $$3\sum'_{i \in F} f_i + \sum'_{j \in C} d(j,F) \leq 3.\sum'_{j \in C} \alpha_j$$

!

In other words $$\sum'_{j \in C} d(j,F) \leq 3\left[\sum'_{j \in C} \alpha_j - \sum'_{i \in F} f_i\right]$$

<u>Why is this interesting?</u>

- Can give an $O(1)$-apx for k median this way [Jain Vazirani 99]

- Method of Lagrangian-Multiplier preserving algos.

  Some other time, or see ~~[WW 10] for~~ the [WS 10] book for details.

- Allows relating constrained problems to Lagrangian versions

$$\min \sum' d_{ij} x_{ij} \qquad \qquad \min \sum' d_{ij} x_{ij} + \boxed{f\left(\sum' y_i - k\right)}$$

$$\sum' x_{ij} = 1, \quad x_{ij} \leq y_i \quad \Rightarrow \quad \sum' x_{ij} = 1, \quad x_{ij} \leq y_i$$

$$\boxed{\sum' y_i \leq k}$$

k median! facility location!

Recap:

Primal-dual technique —

- raise primal and dual in coordinated ways,
  basically construct primal solution, and
  - raise dual "proof" that optimal solution must cost that much
  - or just view dual as money that pays for primal.

- Can also view as approximate complementary slackness.

Several problems:

- Steiner tree/forest
- facility location and k-median
- network design problems

all have primal-dual algorithms.

(Actually can also get P-D algos for problems in P).

Do not use LP solver

⇒ can be faster than LP rounding methods.

— X —

Next week, some hardness,

then we'll be back the week after with some more algos.