Compact, Provably-Good LP Relaxations for Orienteering ~~and RVRP~~

Zachary Friggstad

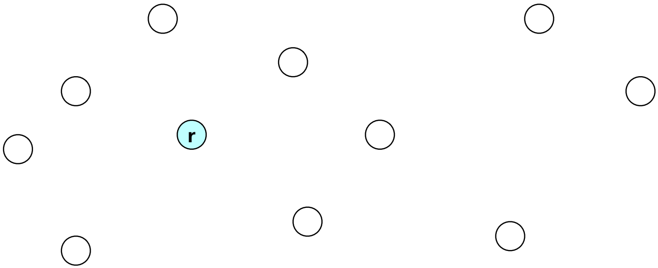
with

Chaitanya Swamy

BIRS TSP Workshop - 2018

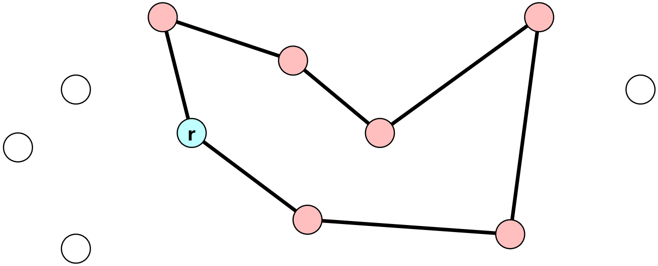
Orienteering

Starting from **Corbet Hall**, visit as many sights in Banff during the Wednesday break.



Orienteering

Starting from **Corbet Hall**, visit as many sights in Banff during the Wednesday break.

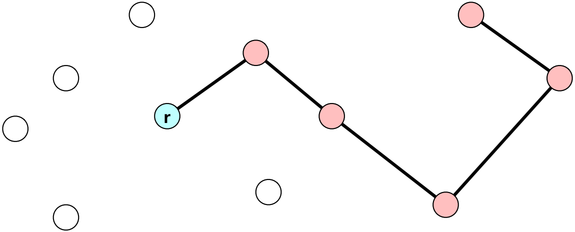


Rooted Orienteering

Given metric distances $d()$ over points $V \cup \{r\}$ where:

- V - clients
- r - depot

Each $v \in V$ has a reward $\rho(v) \geq 0$. Also have a distance bound $D \geq 0$.

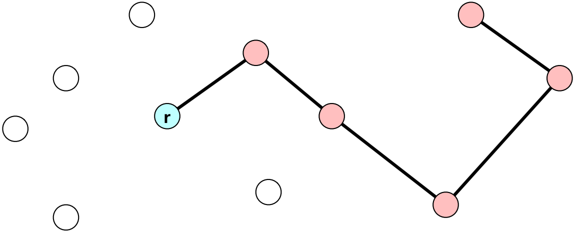


Rooted Orienteering

Given metric distances $d()$ over points $V \cup \{r\}$ where:

- V - clients
- r - depot

Each $v \in V$ has a reward $\rho(v) \geq 0$. Also have a distance bound $D \geq 0$.



Objective

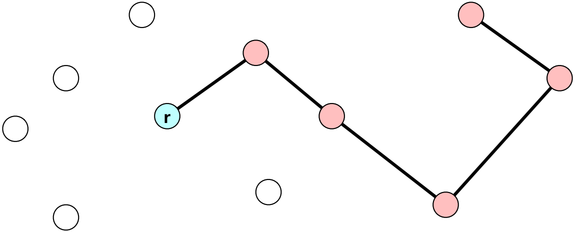
Find an r-rooted path P with $d(P) \leq D$ of maximum reward $\rho(P)$.

Rooted Orienteering

Given metric distances $d()$ over points $V \cup \{r\}$ where:

- V - clients
- r - depot

Each $v \in V$ has a reward $\rho(v) \geq 0$. Also have a distance bound $D \geq 0$.



Objective

Find an r-rooted path P with $d(P) \leq D$ of maximum reward $\rho(P)$.

If an end vertex t is also specified (could be $t = r$), we call this **Point-to-Point Orienteering**.

A Brief History

- First, a 4-approximation for rooted orienteering [Blum et al, 1994].
- Then, a 3-approximation for Point-to-Point Orienteering. [Bansal et al, 2004].
- The best is a $(2 + \epsilon)$ for Point-to-Point. [Chekuri, Korula, and Pal, 2012].

A Brief History

- First, a 4-approximation for rooted orienteering [Blum et al, 1994].
- Then, a 3-approximation for Point-to-Point Orienteering. [Bansal et al, 2004].
- The best is a $(2 + \epsilon)$ for Point-to-Point. [Chekuri, Korula, and Pal, 2012].

Briefly, the asymmetric version is also studied.

- An $O(\log^2 OPT)$-approximation. [Chekuri, Korula, and Pal, 2007].
- An $O(\rho \cdot \log n)$-approximation: $\rho =$ ATSP integrality gap. [Nagarajan and Ravi, 2007].
  At the time, $\rho = O(\log n)$ but now we know better!

A Brief History

- First, a 4-approximation for rooted orienteering [Blum et al, 1994].
- Then, a 3-approximation for Point-to-Point Orienteering. [Bansal et al, 2004].
- The best is a $(2 + \epsilon)$ for Point-to-Point. [Chekuri, Korula, and Pal, 2012].

Briefly, the asymmetric version is also studied.

- An $O(\log^2 OPT)$-approximation. [Chekuri, Korula, and Pal, 2007].
- An $O(\rho \cdot \log n)$-approximation: $\rho =$ ATSP integrality gap. [Nagarajan and Ravi, 2007].
  At the time, $\rho = O(\log n)$ but now we know better!

Notice: The improved integrality gap bound for ATSP led to an improved approximation for a different problem!

Specific Results

Poly-size LP relaxations with the following integrality gap bounds.

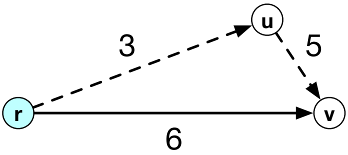
- **Rooted Orienteering**: 3
- **Point-to-Point Orienteering**: 6
- **RVRP**: A natural relaxation with a gap of 27, an unnatural relaxation with a gap of 15.
  This beats a 28.86-approximation that used a large configuration LP [F. and Swamy, 2014].

The Regret Metric

We shift focus to a new metric called the **regret metric**.

$$d^{reg}(u, v) := d(r, u) + d(u, v) - d(r, v).$$

How much longer is $r \to u \to v$ than $r \to v$ directly?



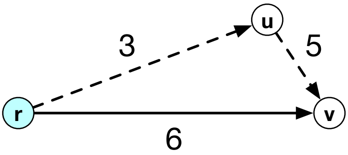
$$d^{reg}(u, v) = 3 + 5 - 6 = 2$$

The Regret Metric

We shift focus to a new metric called the **regret metric**.

$$d^{reg}(u, v) := d(r, u) + d(u, v) - d(r, v).$$

How much longer is $r \to u \to v$ than $r \to v$ directly?



$$d^{reg}(u, v) = 3 + 5 - 6 = 2$$

Key Properties:

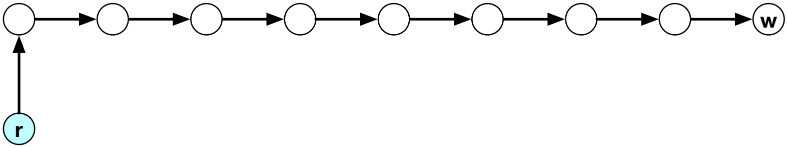
- For any $v \in V$, $d^{reg}(r, v) = 0$.
- For any $r \to v$ path P, $d^{reg}(P) = d(P) - d(r, v)$.

Pruning w.r.t. Regret

Before presenting the LP, we briefly discuss a slightly weaker goal.

Observe a rooted $r - w$ path P is a feasible orienteering solution iff $d^{reg}(P) \leq D - d(r, v)$.

Now suppose P is an $r - w$ path with $d^{reg}(P) \leq \alpha \cdot (D - d(r, w))$.

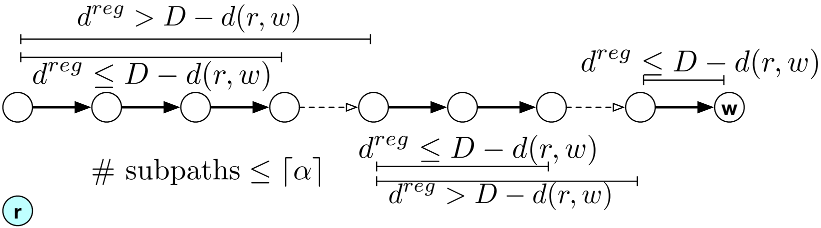


Claim

If w has maximum distance from r among all clients, we can *chop* P to a feasible solution with value $\geq \rho(P)/\lceil \alpha \rceil$.

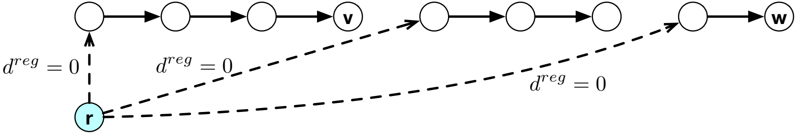
Pruning w.r.t. Regret

First, break $P - \{r\}$ into $\lceil \alpha \rceil$ subpaths, each having d^{reg}-distance $\leq D - d(r, w)$.



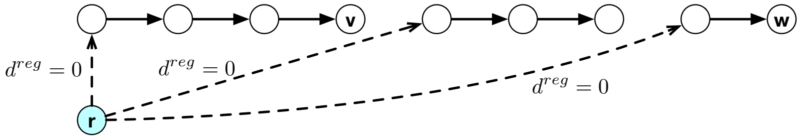
Pruning w.r.t. Regret

Make each subpath a rooted path by prepending r. Recall $d^{reg}(r, x) = 0$ for all $x \in V$.



Pruning w.r.t. Regret

Make each subpath a rooted path by prepending r. Recall $d^{reg}(r, x) = 0$ for all $x \in V$.

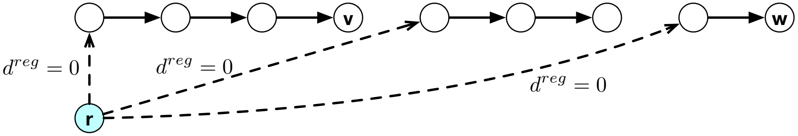


Any of these r-rooted subpaths P' ending at, say, v has length

$$d^{reg}(P') + d(r, v) \leq (D - d(r, w)) + d(r, v) \leq D.$$

Pruning w.r.t. Regret

Make each subpath a rooted path by prepending r. Recall $d^{reg}(r, x) = 0$ for all $x \in V$.



Any of these r-rooted subpaths P' ending at, say, v has length

$$d^{reg}(P') + d(r, v) \leq (D - d(r, w)) + d(r, v) \leq D.$$

So the most profitable path has value $\geq \rho(P)/\lceil \alpha \rceil$.

The LP

Before forming the LP, guess the node w on the optimum path that is furthest from r and discard farther nodes.

The LP

Before forming the LP, guess the node w on the optimum path that is furthest from r and discard farther nodes.

Variables

We deal with a **bidirected-cut** relaxation of the problem.

- z_v - indicating we visit v.
- x_e - indicating we use edge/arc e.

The LP

Before forming the LP, guess the node w on the optimum path that is furthest from r and discard farther nodes.

Variables

We deal with a **bidirected-cut** relaxation of the problem.

- z_v - indicating we visit v.
- x_e - indicating we use edge/arc e.

$$
\begin{array}{llcll}
\max: & \sum_v \rho(v) \cdot z_v & & & \\
\text{s.t.}: & x(\delta^{in}(v)) & \geq & x(\delta^{out}(v)) & v \in V & \text{(preflow)} \\
& x(\delta^{in}(S)) & \geq & z_v & v \in S \subseteq V & \text{(clients reachable)} \\
& x(\delta^{out}(r)) & = & 1 & & \text{(one path)} \\
& z_w & = & 1 & & \text{(visits } w\text{)} \\
& \sum_e d(e) \cdot x_e & \leq & D & & \text{(distance bound)} \\
& x, z & \geq & 0 & &
\end{array}
$$

The LP

Before forming the LP, guess the node w on the optimum path that is furthest from r and discard farther nodes.

Variables

We deal with a **bidirected-cut** relaxation of the problem.

- z_v - indicating we visit v.
- x_e - indicating we use edge/arc e.

$$
\begin{array}{lrcll}
\max: & \sum_v \rho(v) \cdot z_v & & & & \\
\text{s.t.}: & x(\delta^{in}(v)) & \geq & x(\delta^{out}(v)) & v \in V & \text{(preflow)} \\
 & x(\delta^{in}(S)) & \geq & z_v & v \in S \subseteq V & \text{(clients reachable)} \\
 & x(\delta^{out}(r)) & = & 1 & & \text{(one path)} \\
 & z_w & = & 1 & & \text{(visits } w\text{)} \\
 & \sum_e d(e) \cdot x_e & \leq & D & & \text{(distance bound)} \\
 & x, z & \geq & 0 & &
\end{array}
$$

Notes: Can "fold" the guess into the LP to avoid guessing. *i.e.* (x^w, z^w) variables. Can make poly-size using flow variables.

A Decomposition Theorem

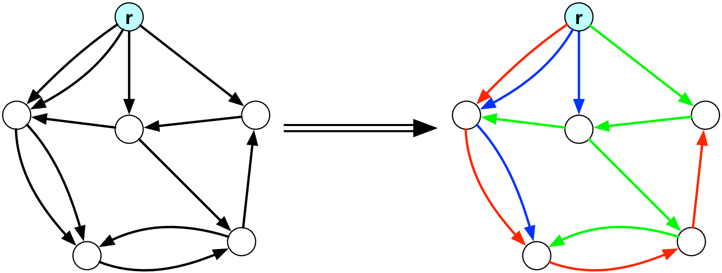
Let $D = (V + r, A)$ be a multi-digraph satisfying preflow conditions at each $v \in V$:

$$|\delta^{in}(v)| \geq |\delta^{out}(v)|.$$

Let λ_v be the $r - v$ edge connectivity.

Theorem (Bang-Jensen, Frank, and Jackson, 1995)

For any $K > 0$, there are K arc-disjoint r-branchings where each vertex v lies on $\min\{K, \lambda_v\}$ *branchings.*

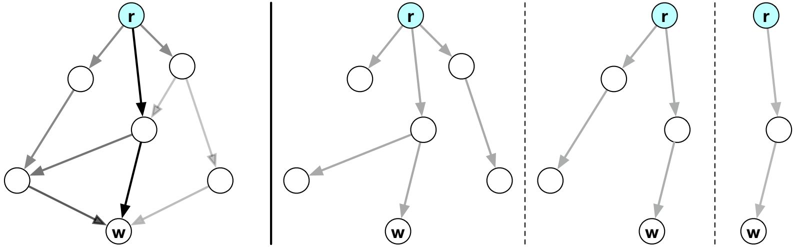


The fractional version:

Theorem

The preflow x dominates a convex combination of r-branchings where each $v \in V$ lies on a z_v-weight of these branchings.

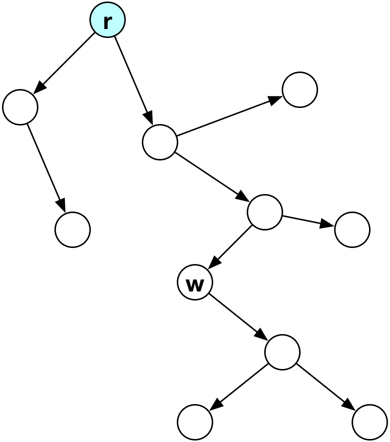
Note, w lies on each branching.



Can be found in poly-time [Post and Swamy, 2015].

The Rounding Algorithm

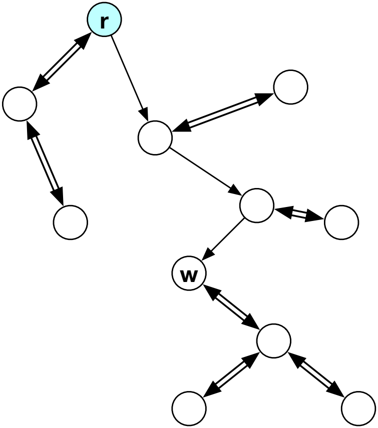
Sample a random branching B in the decomposition.



The expected $d()$-cost of B is $\leq D$.

The Rounding Algorithm

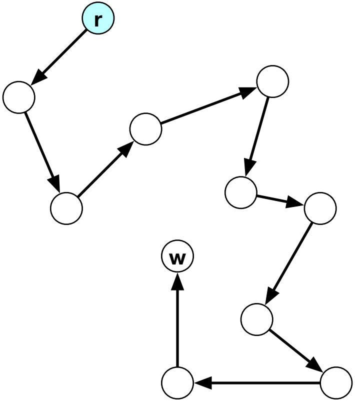
Double edges not on the $r - w$ path.



The expected $d()$-cost is $\leq D + (D - d(r, w))$.

The Rounding Algorithm

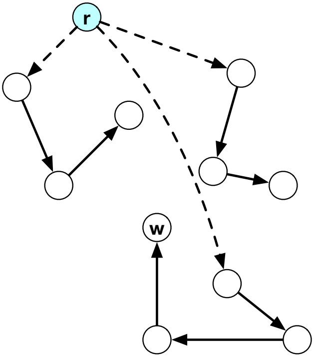
Of course, shortcut the resulting Eulerian walk to an $r - w$ path.



The expected $d()$-cost of these paths is still $\leq D + (D - d(r, w))$.

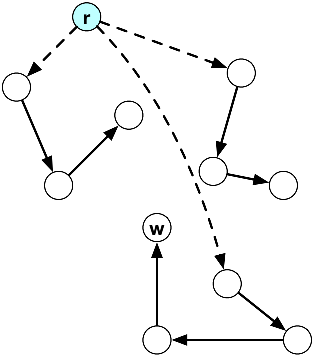
Equivalently: The expected $d^{reg}()$-cost is $\leq 2 \cdot (D - d(r, w))$.

Chop into rooted paths with $d^{reg}()$-distance $\leq D - d(r, w)$.
i.e. Feasible orienteering solutions!



If the original path P had regret $\alpha_P \cdot (D - d(r, w))$, this creates $\leq \lceil \alpha_P \rceil \leq \alpha_P + 1$ paths.

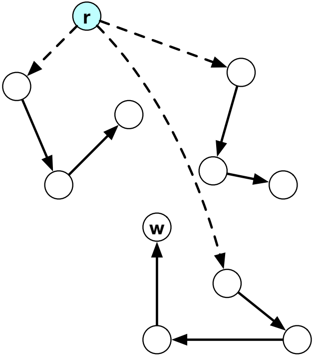
Chop into rooted paths with $d^{reg}()$-distance $\leq D - d(r, w)$.
i.e. Feasible orienteering solutions!



If the original path P had regret $\alpha_P \cdot (D - d(r, w))$, this creates $\leq \lceil \alpha_P \rceil \leq \alpha_P + 1$ paths.

This creates ≤ 3 subpaths **in expectation** as $\mathbf{E}[\alpha_P] \leq 2$.

Chop into rooted paths with $d^{reg}()$-distance $\leq D - d(r, w)$.
i.e. Feasible orienteering solutions!



If the original path P had regret $\alpha_P \cdot (D - d(r, w))$, this creates $\leq \lceil \alpha_P \rceil \leq \alpha_P + 1$ paths.

This creates ≤ 3 subpaths **in expectation** as $\mathbf{E}[\alpha_P] \leq 2$.

Some subpath created this way has value $\geq OPT_{LP}/3$.

Algorithm Summary

1. Guess the furthest node w.
2. Solve the LP.
3. Decompose (x, z) into branchings.
4. For each branching:
   - Double edges not on the $r - w$ path.
   - Shortcut the Eulerian path.
   - Chop into feasible solutions.
5. Return the best subpath created.

Algorithm Summary

1. Guess the furthest node w.
2. Solve the LP.
3. Decompose (x, z) into branchings.
4. For each branching:
   - Double edges not on the $r - w$ path.
   - Shortcut the Eulerian path.
   - Chop into feasible solutions.
5. Return the best subpath created.

Again, the guesswork can be avoided by folding w into the LP and performing this rounding for each (x^w, z^w)-family of variables.

Algorithm Summary

1. Guess the furthest node w.
2. Solve the LP.
3. Decompose (x, z) into branchings.
4. For each branching:
   - Double edges not on the $r - w$ path.
   - Shortcut the Eulerian path.
   - Chop into feasible solutions.
5. Return the best subpath created.

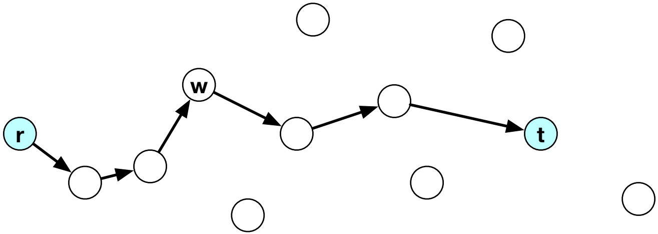
Again, the guesswork can be avoided by folding w into the LP and performing this rounding for each (x^w, z^w)-family of variables.

Comment

Without the guess, the gap is very bad. Even if we just guess the furthest distance but not the node itself!

Point-to-Point Orienteering

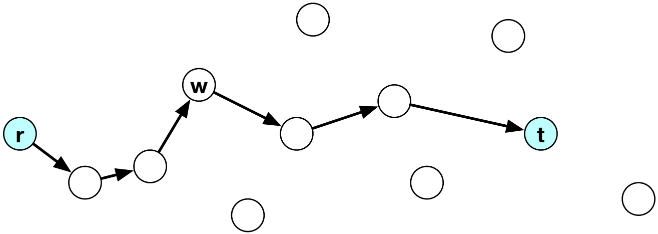
Suppose we want an $r - t$ path of bounded length.



Guess the node w on opt. with largest $d(r, w) + d(w, t)$.

Point-to-Point Orienteering

Suppose we want an $r - t$ path of bounded length.



Guess the node w on opt. with largest $d(r, w) + d(w, t)$.

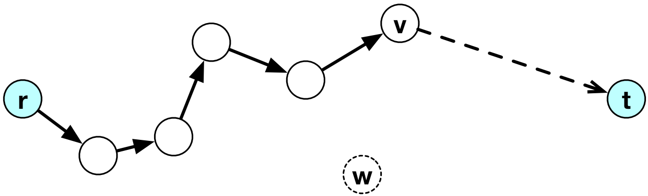
LP: one unit of $r - w$ flow x^L and one unit of $w - t$ flow x^R.

Also z_v^L and z_v^R variables indicating if v is visited before w or after w, respectively.

To round it, the x^L-flow is a preflow from r with cost at most $D - d(w,t)$, so do as before.

This produces a path ending at some v with length

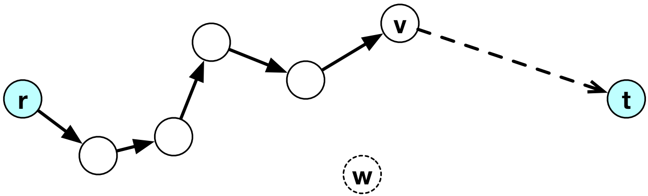
$$\leq D - d(w,t) + d(r,v) - d(r,w).$$



To round it, the x^L-flow is a preflow from r with cost at most $D - d(w, t)$, so do as before.

This produces a path ending at some v with length

$$\leq D - d(w, t) + d(r, v) - d(r, w).$$



Extending from v to an $r - t$ path yields a path with distance

$$D + [d(r, v) + d(v, t)] - [d(r, w) + d(w, t)] \leq D$$

with at least $1/3$ the value of z^L.

Similarly, the reverse of x^R is a preflow out of t so we can get a feasible solution with at least $1/3$ the value of z^R.

The best solution overall has value $\geq OPT_{LP}/6$.

Similarly, the reverse of x^R is a preflow out of t so we can get a feasible solution with at least $1/3$ the value of z^R.

The best solution overall has value $\geq OPT_{LP}/6$.

Wrapping Up

Recent: One can even avoid solving the LP; consider **Rooted Orienteering** again.

Similarly, the reverse of x^R is a preflow out of t so we can get a feasible solution with at least $1/3$ the value of z^R.

The best solution overall has value $\geq OPT_{LP}/6$.

Wrapping Up

Recent: One can even avoid solving the LP; consider **Rooted Orienteering** again.

Post and Swamy describe a combinatorial, primal-dual algorithm for the Prize-Collecting Arborescence problem that finds a solution with cost $\leq$ the optimum prize-collecting path solution..

Similarly, the reverse of x^R is a preflow out of t so we can get a feasible solution with at least $1/3$ the value of z^R.

The best solution overall has value $\geq OPT_{LP}/6$.

Wrapping Up

Recent: One can even avoid solving the LP; consider **Rooted Orienteering** again.

Post and Swamy describe a combinatorial, primal-dual algorithm for the Prize-Collecting Arborescence problem that finds a solution with cost $\leq$ the optimum prize-collecting path solution..

Using Lagrangian relaxation, one can find a bipoint "solution": two branchings spanning the guess w with "average" cost $\leq D$ and average profit $\geq OPT$.

Similarly, the reverse of x^R is a preflow out of t so we can get a feasible solution with at least $1/3$ the value of z^R.

The best solution overall has value $\geq OPT_{LP}/6$.

Wrapping Up

Recent: One can even avoid solving the LP; consider **Rooted Orienteering** again.

Post and Swamy describe a combinatorial, primal-dual algorithm for the Prize-Collecting Arborescence problem that finds a solution with cost $\leq$ the optimum prize-collecting path solution..

Using Lagrangian relaxation, one can find a bipoint "solution": two branchings spanning the guess w with "average" cost $\leq D$ and average profit $\geq OPT$.

All done!

Thank You