

Unsplittable Flow in Paths and Trees and Column-Restricted Packing Integer Programs

Chandra Chekuri

Nitish Korula

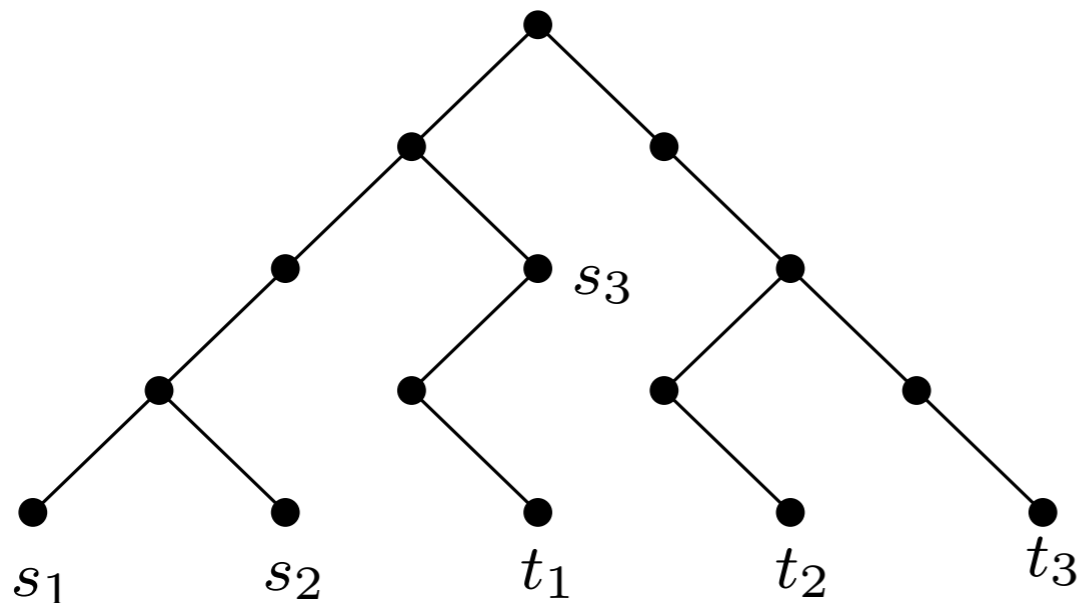
Alina Ene

University of Illinois at Urbana-Champaign

APPROX 2009

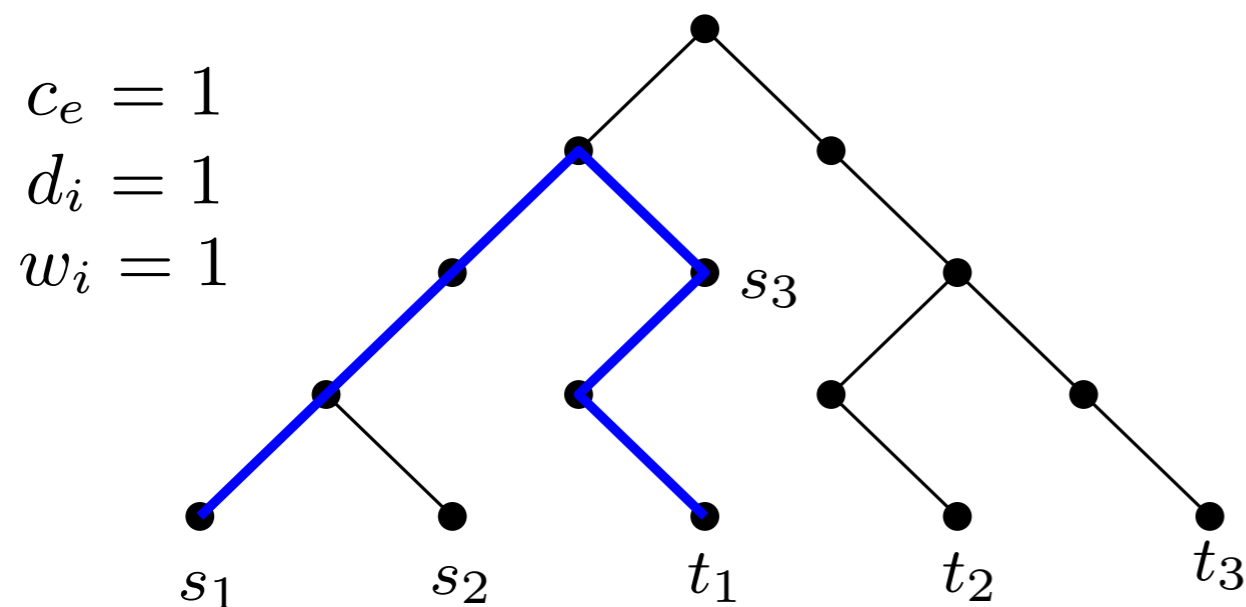
Unsplittable Flow Problem on a Tree

- Tree T with a capacity c_e on each edge
- Set \mathcal{R} of requests
- Request R_i : pair of vertices (s_i, t_i) , demand d_i , profit w_i
- Route R_i : send d_i units of flow along the path from s_i to t_i
- Find a feasible subset of requests with maximum profit



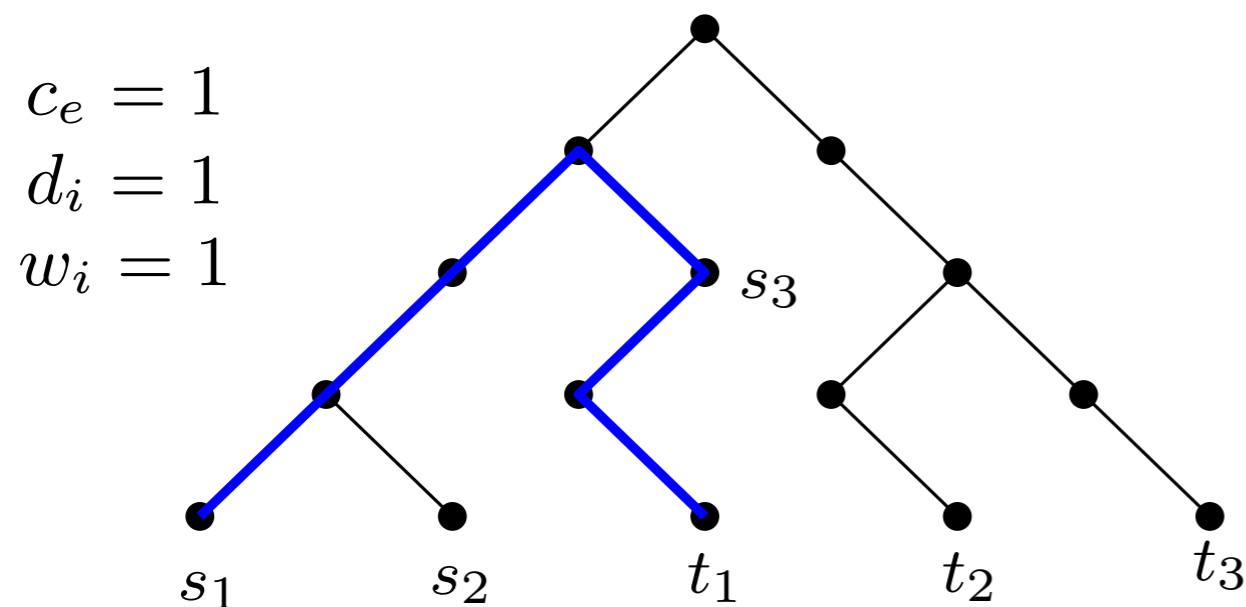
Unsplittable Flow Problem on a Tree

- Tree T with a capacity c_e on each edge
- Set \mathcal{R} of requests
- Request R_i : pair of vertices (s_i, t_i) , demand d_i , profit w_i
- Route R_i : send d_i units of flow along the path from s_i to t_i
- Find a feasible subset of requests with maximum profit



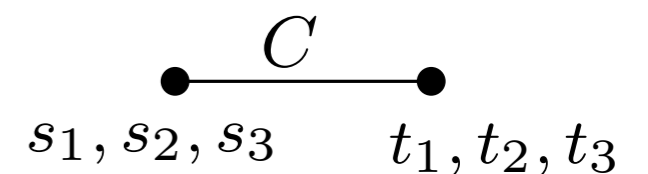
Unsplittable Flow Problem on a Tree

- Tree T with a capacity c_e on each edge
- Set \mathcal{R} of requests
- Request R_i : pair of vertices (s_i, t_i) , demand d_i , profit w_i
- Route R_i : send d_i units of flow along the path from s_i to t_i
- Find a feasible subset of requests with maximum profit



NP-hard

Knapsack:



APX-hard on trees with $d_i = 1$

LP Relaxation

- Standard LP relaxation for UFP on paths and trees

$$\text{Standard LP} \quad \max \sum_i w_i x_i$$

$$\sum_{i: e \in P_i} d_i x_i \leq c_e$$
$$x_i \in [0, 1]$$

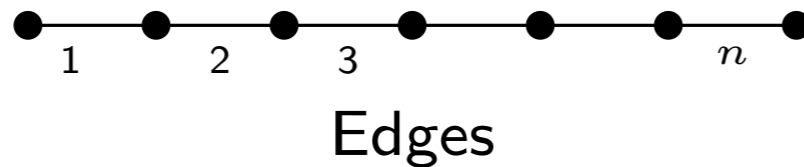
- Variable x_i : indicates whether request R_i is selected
- P_i : unique path between s_i and t_i
- Capacity constraints: total demand of selected requests on each edge is at most the capacity

Integrality gap for UFP on paths

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

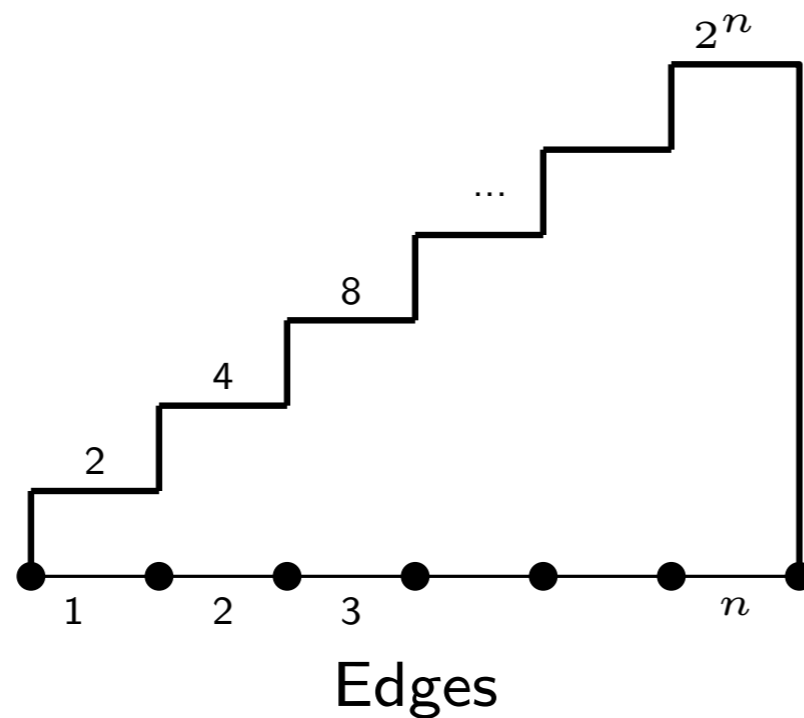
Staircase Example



Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

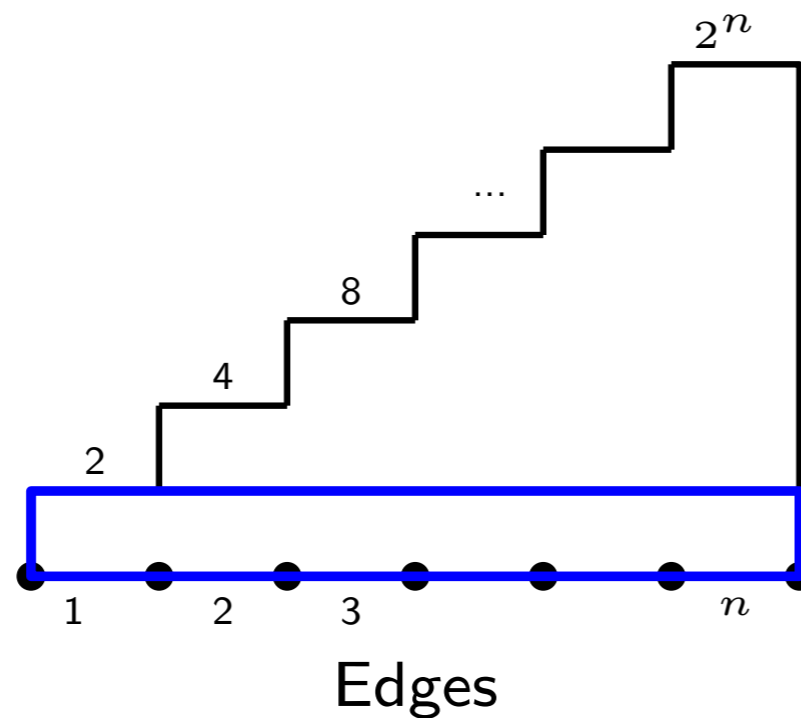
Staircase Example



Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example

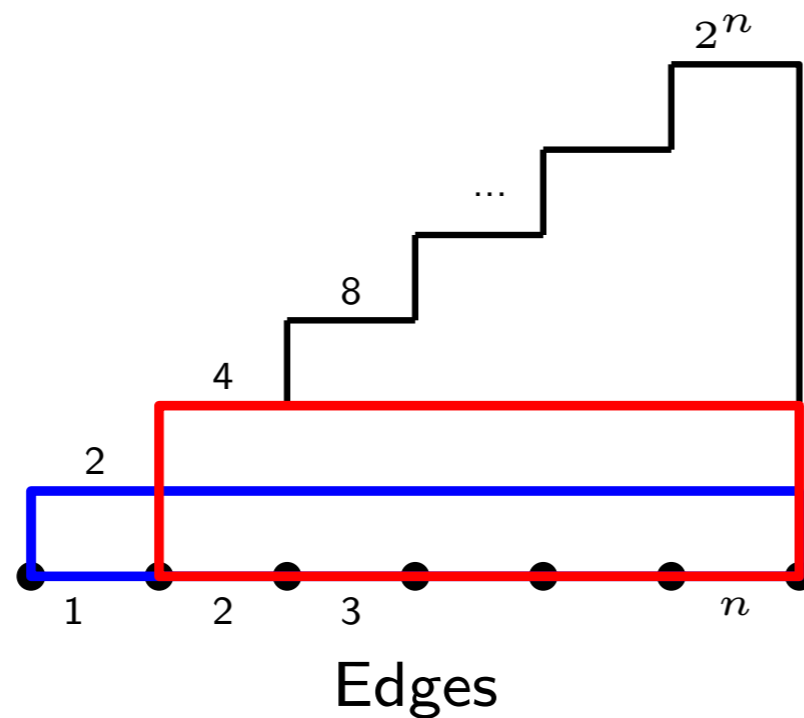


$$d_1 = 2 \quad w_1 = 1$$

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example

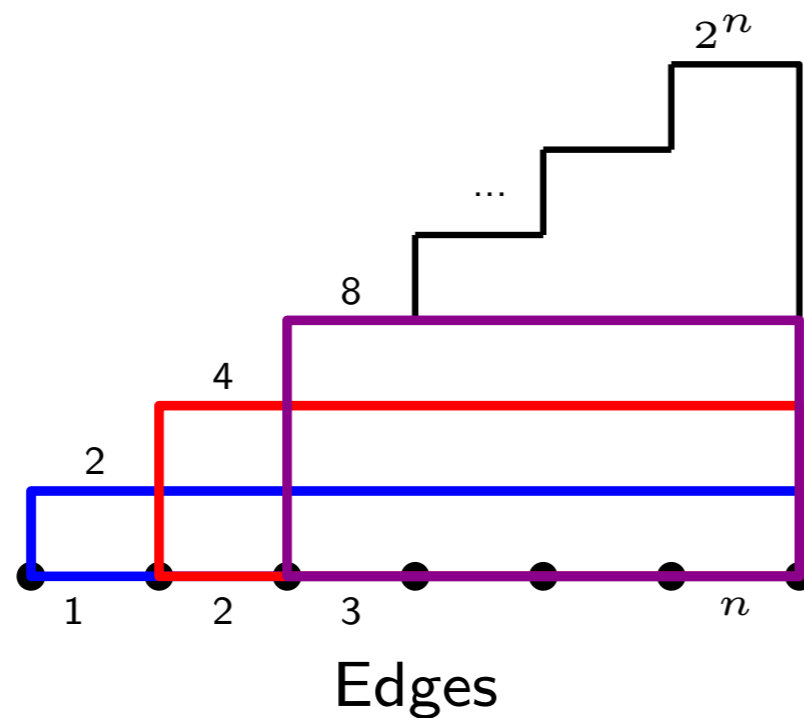


$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \end{array}$$

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example

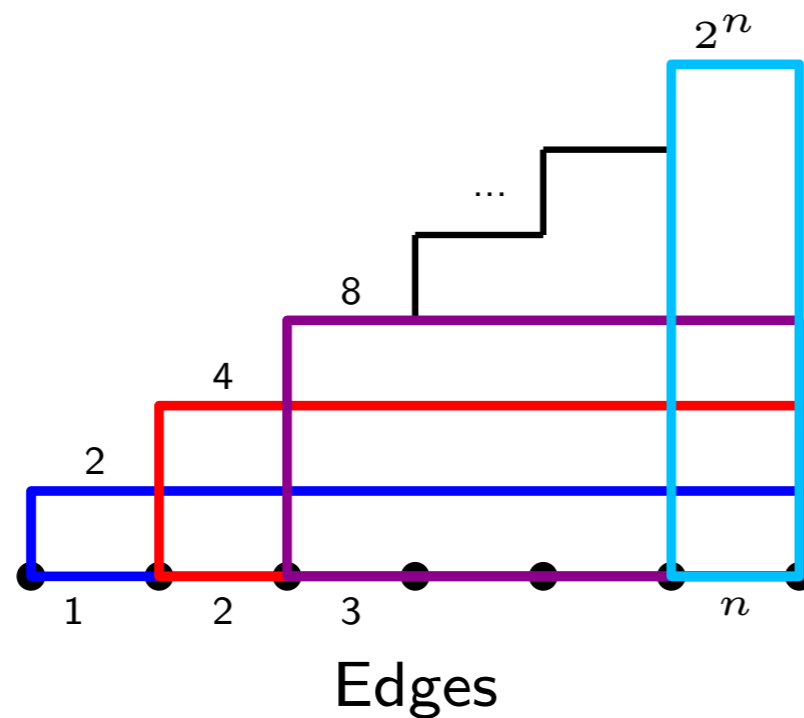


$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \end{array}$$

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example

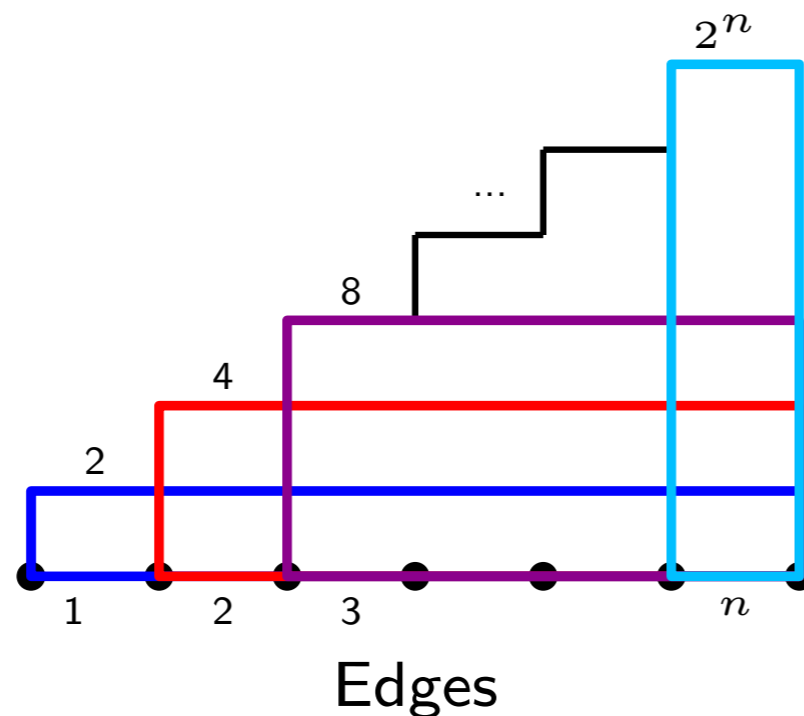


$$\begin{aligned} d_1 &= 2 & w_1 &= 1 \\ d_2 &= 4 & w_2 &= 1 \\ d_3 &= 8 & w_3 &= 1 \\ & \dots & & \\ d_n &= 2^n & w_n &= 1 \end{aligned}$$

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example



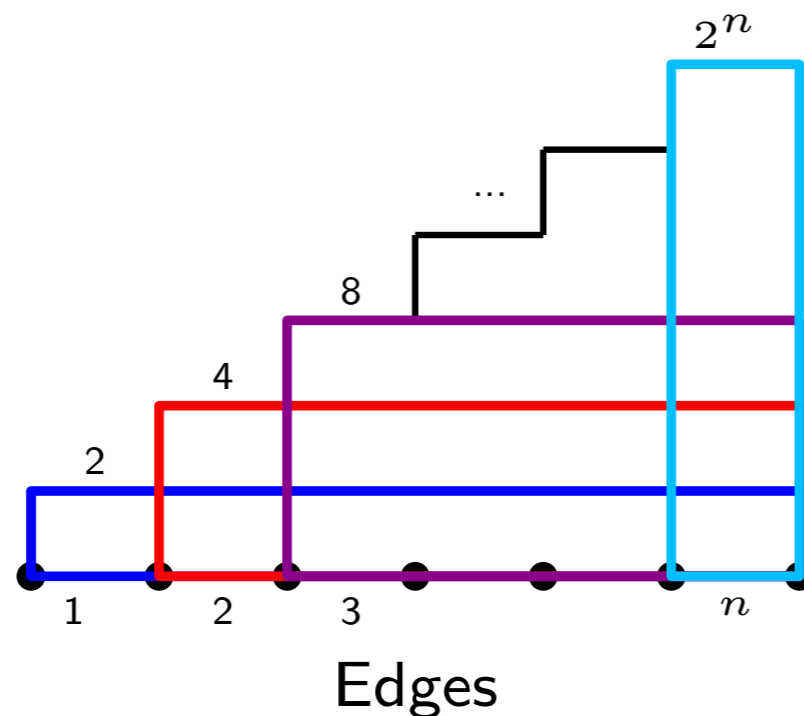
$$\begin{aligned} d_1 &= 2 & w_1 &= 1 \\ d_2 &= 4 & w_2 &= 1 \\ d_3 &= 8 & w_3 &= 1 \\ & \dots & & \\ d_n &= 2^n & w_n &= 1 \end{aligned}$$

LP solution has profit at least $n/2$: $x_i = 1/2$ is feasible

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example



$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \\ & \dots \\ d_n = 2^n & w_n = 1 \end{array}$$

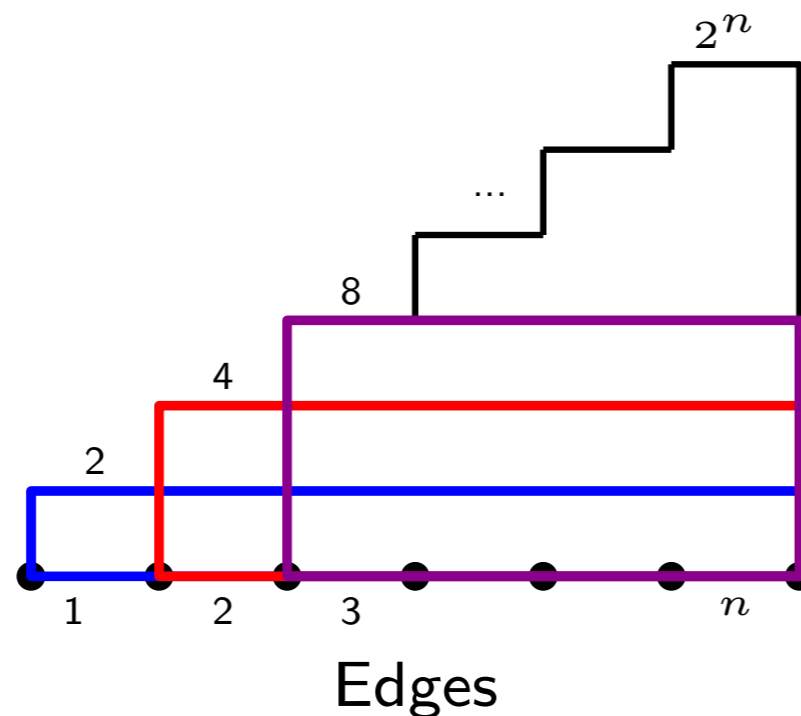
LP solution has profit at least $n/2$: $x_i = 1/2$ is feasible

Integral sol has profit 1: any integral sol routes at most one request

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example



$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \end{array}$$

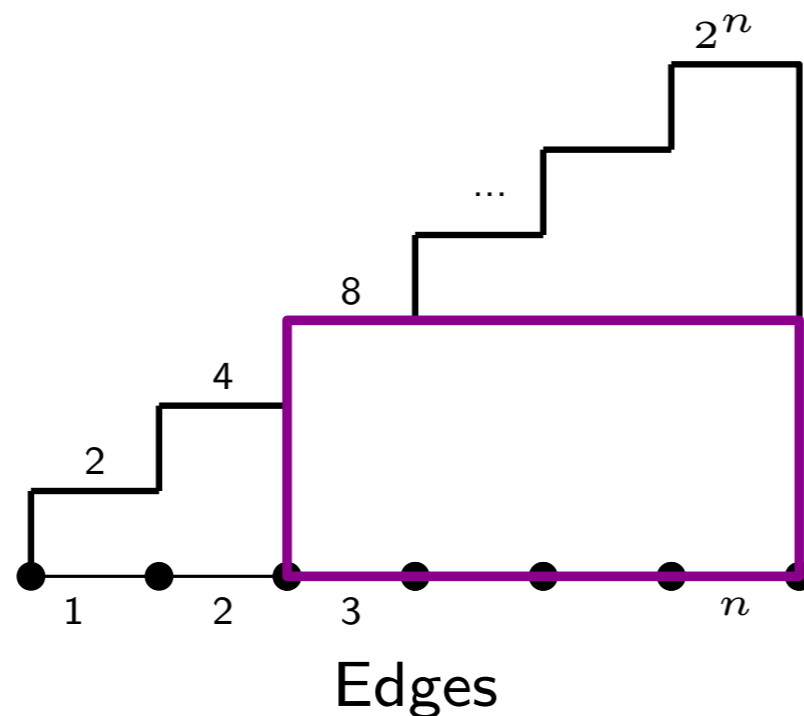
LP solution has profit at least $n/2$: $x_i = 1/2$ is feasible

Integral sol has profit 1: any integral sol routes at most one request

Integrality gap for UFP on paths

The standard LP has $\Omega(n)$ integrality gap [Chakrabarti et al., 2002]

Staircase Example



$$d_3 = 8 \quad w_3 = 1$$

LP solution has profit at least $n/2$: $x_i = 1/2$ is feasible

Integral sol has profit 1: any integral sol routes at most one request

UFP with no-bottleneck assumption

- No-bottleneck assumption (NBA)

$$\max_i d_i \leq \min_e c_e$$

- Standard LP has $O(1)$ gap for UFP-NBA on paths
[Chakrabarti, Chekuri, Gupta, Kumar 2002]
- Standard LP has $O(1)$ gap for UFP-NBA on trees
($2 + \epsilon$)-approx for paths using LP and dynamic programming
 ~ 48 -approx for trees using LP
[Chekuri, Mydlarz, Shepherd 2003]
- This talk: UFP (without NBA)

Unsplittable Flow Problem

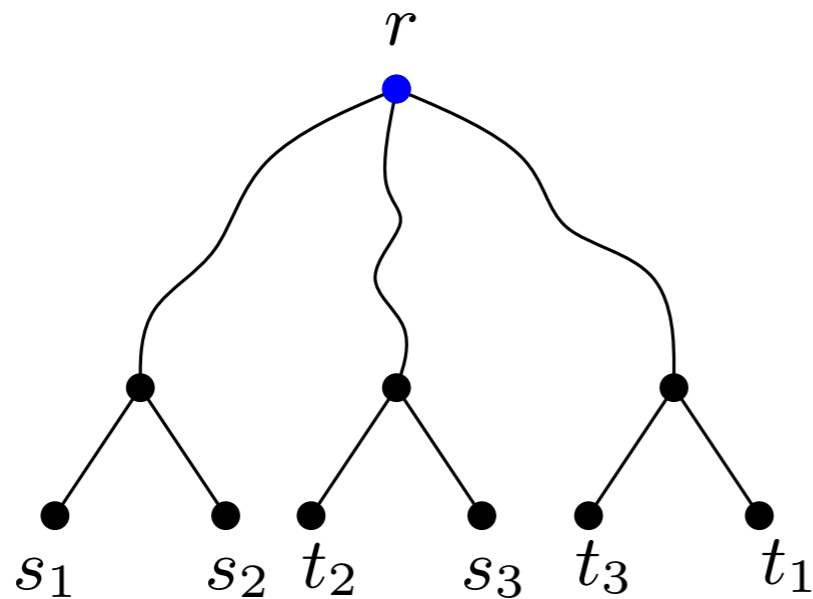
- $(1 + \epsilon)$ approx in $n^{O(\log n/\epsilon)}$ time if capacities and demands are at most $2^{\text{polylog}(n)}$ [Bansal et al, 2006]
- $O(\log n)$ for UFP on paths [Bansal et al, 2009]
- Open problems
 - $o(n)$ approx for UFP on trees
 - LP relaxation for UFP on paths/trees with $o(n)$ gap

Our Results

- $O(\log^2 n)$ approx for UFP on trees
first algorithm with $o(n)$ approx ratio
- LP relaxation for UFP on paths with $O(\log^2 n)$ gap
first LP with $o(n)$ integrality gap
recently improved to $O(\log n)$ gap
- Related results for UFP and column restricted packing IPs

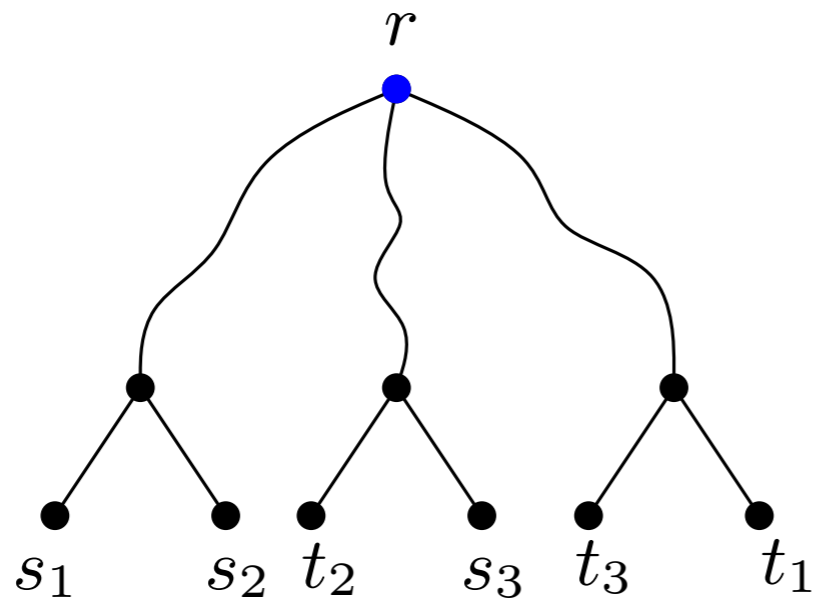
UFP on trees: special case

- Unit-weight instance
- Intersecting instance: request paths go through the root



UFP on trees: special case

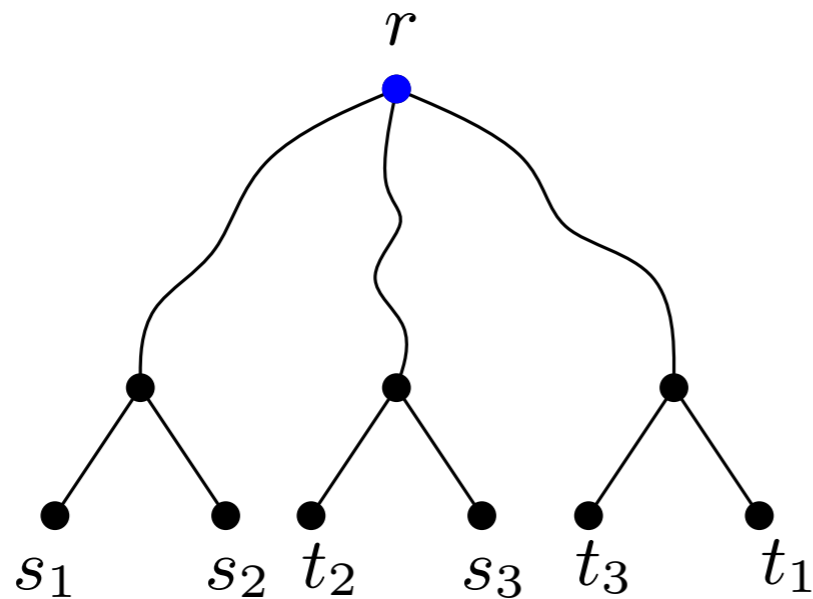
- Unit-weight instance
- Intersecting instance: request paths go through the root



- Sort demands: $d_1 \leq d_2 \leq \dots \leq d_k$

UFP on trees: special case

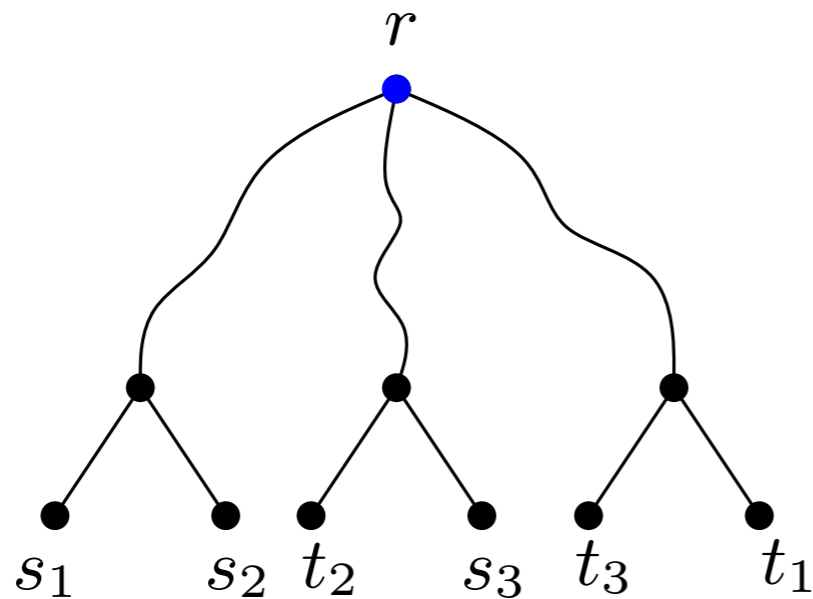
- Unit-weight instance
- Intersecting instance: request paths go through the root



- Sort demands: $d_1 \leq d_2 \leq \dots \leq d_k$
- Greedily add requests while maintaining feasibility

UFP on trees: special case

- Unit-weight instance
- Intersecting instance: request paths go through the root

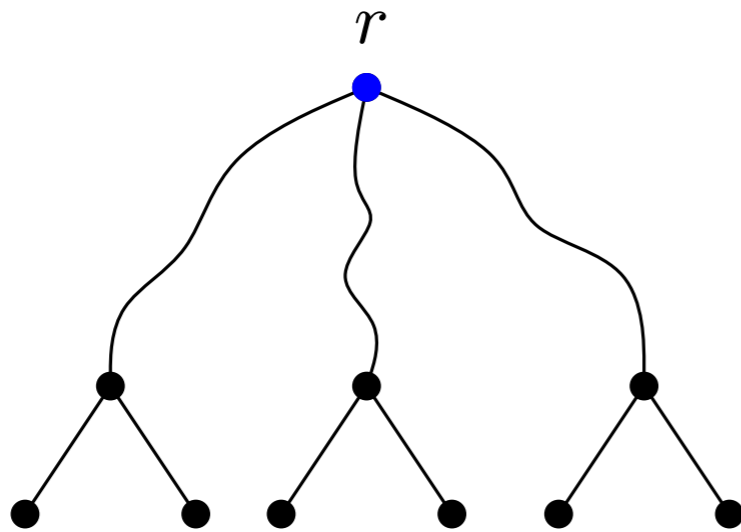


2-approx

- Sort demands: $d_1 \leq d_2 \leq \dots \leq d_k$
- Greedily add requests while maintaining feasibility

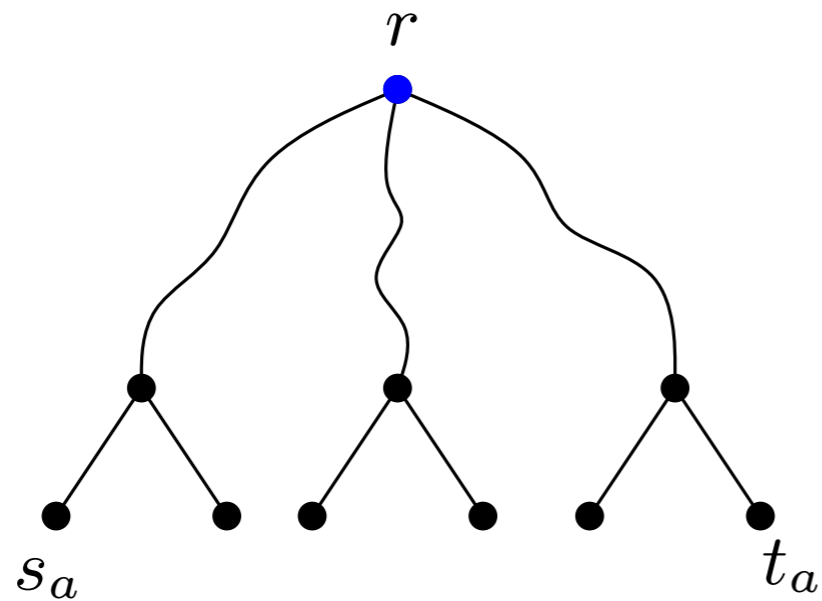
UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



UFP on trees: special case

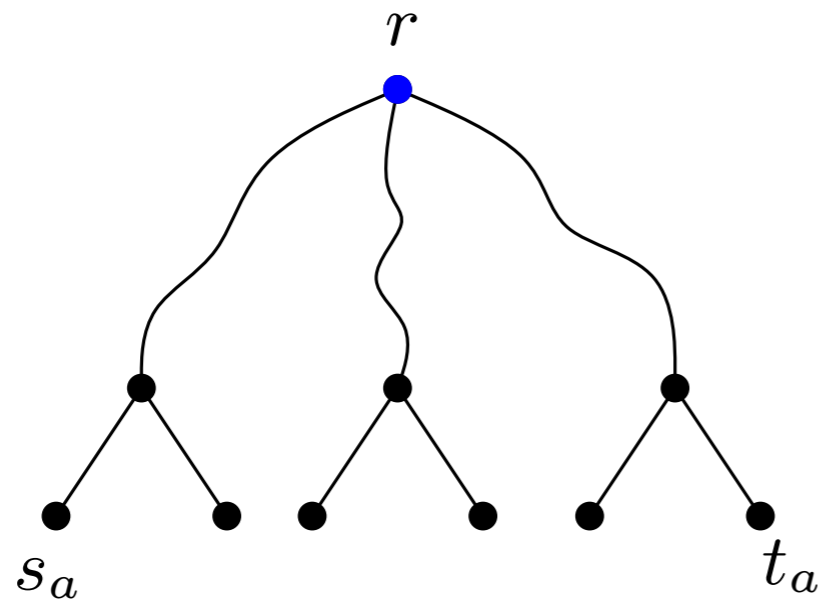
Lemma. Greedy achieves a 2-approximation.



Greedy routes the request A with smallest demand

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



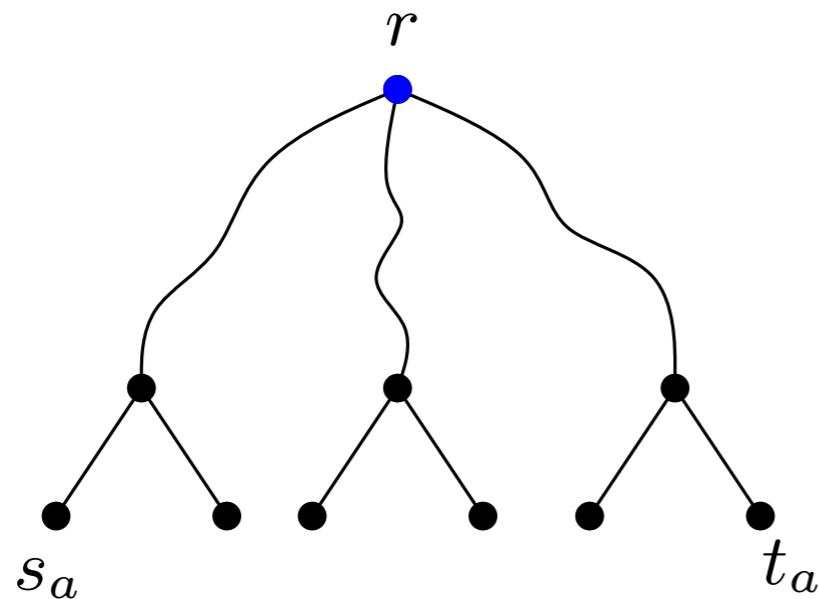
Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

win by induction

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

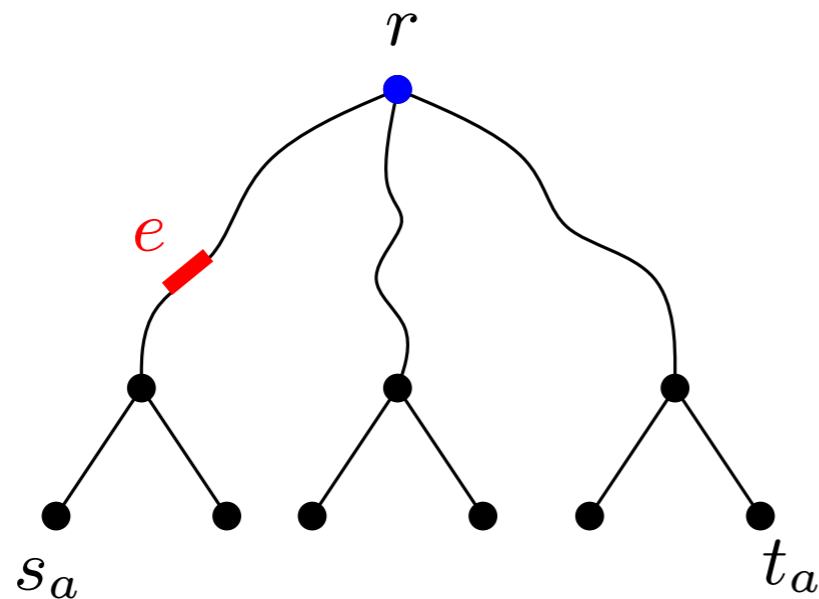
win by induction

Case 2: A is not in the optimal solution

swap two requests B and C in optimal solution with A

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

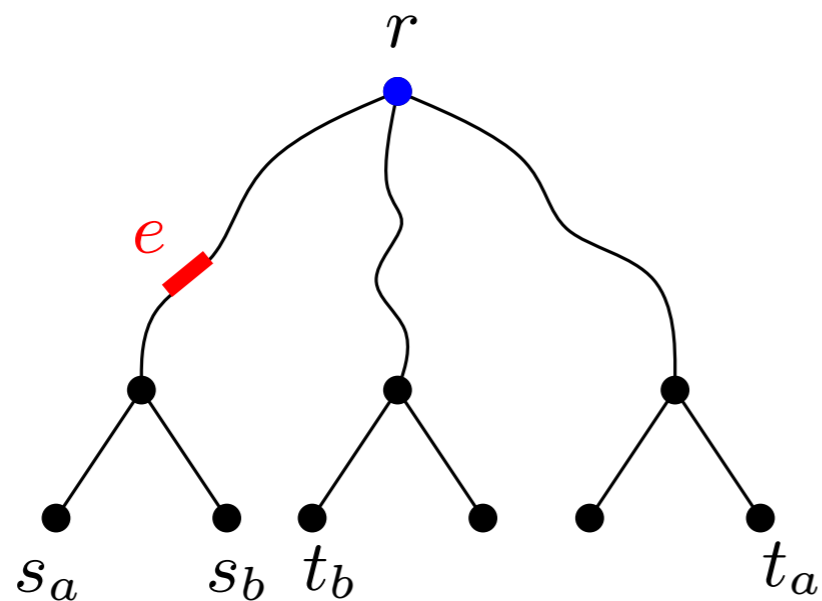
win by induction

Case 2: A is not in the optimal solution

swap two requests B and C in optimal solution with A

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



B uses e

Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

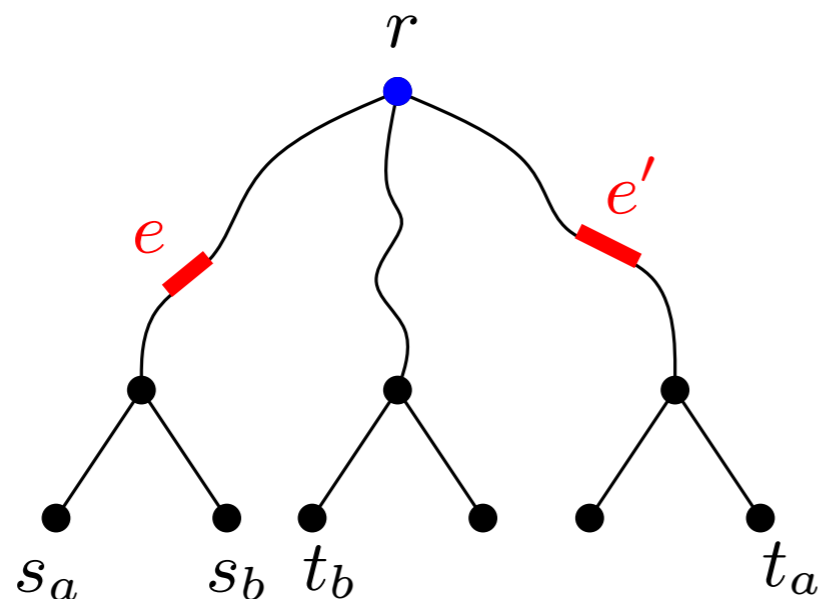
win by induction

Case 2: A is not in the optimal solution

swap two requests B and C in optimal solution with A

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



B uses e

Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

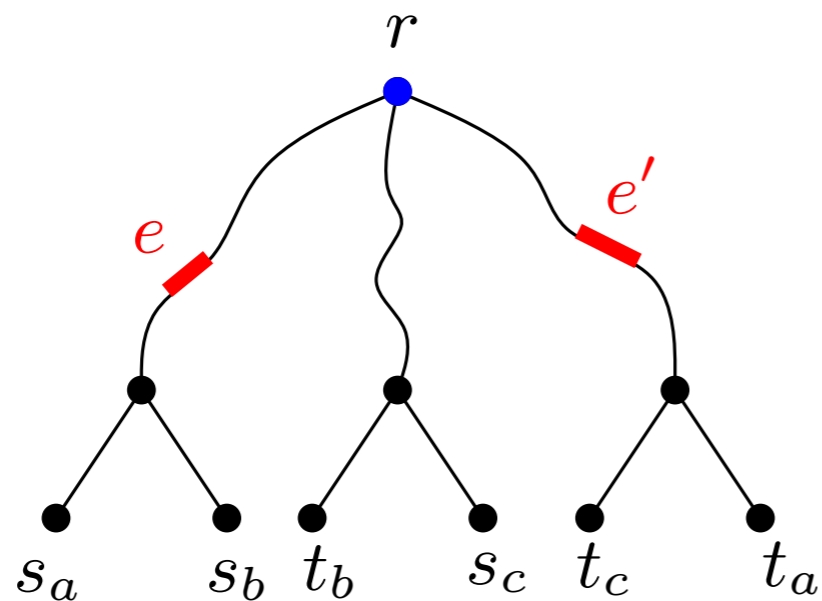
win by induction

Case 2: A is not in the optimal solution

swap two requests B and C in optimal solution with A

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



B uses e

C uses e'

Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

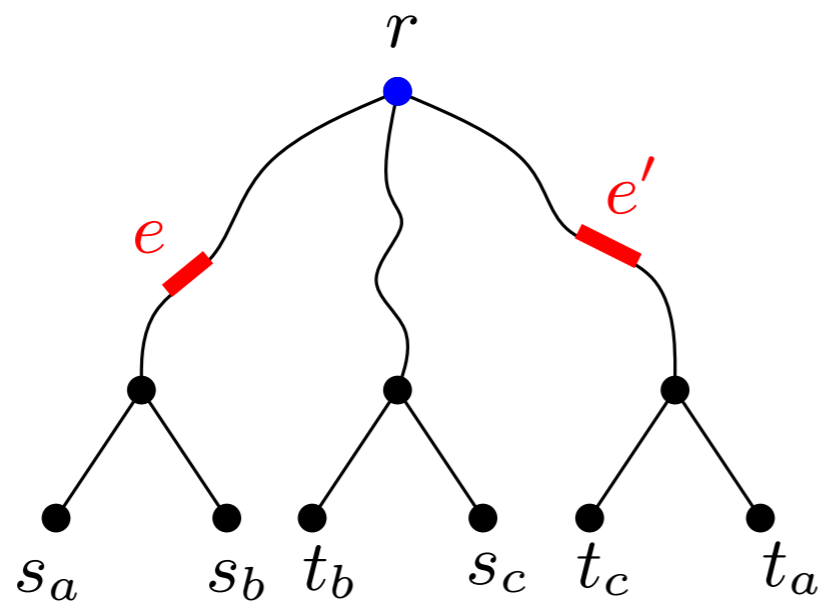
win by induction

Case 2: A is not in the optimal solution

swap two requests B and C in optimal solution with A

UFP on trees: special case

Lemma. Greedy achieves a 2-approximation.



B uses e

C uses e'

$$d_a \leq d_b$$

$$d_a \leq d_c$$

Greedy routes the request A with smallest demand

Case 1: A is in the optimal solution

win by induction

Case 2: A is not in the optimal solution

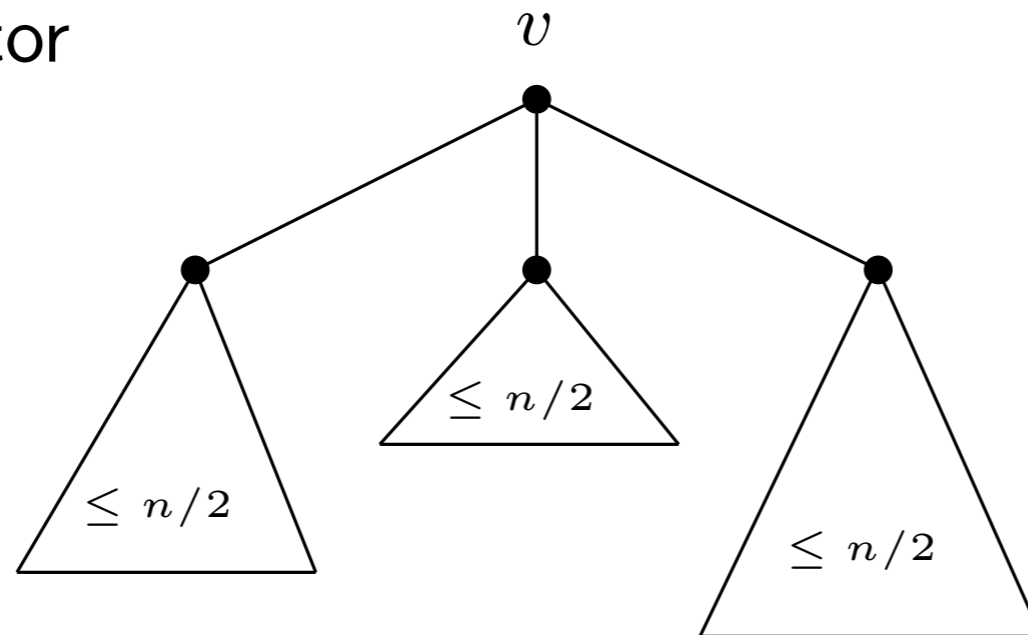
swap two requests B and C in optimal solution with A

UFP on trees

UFP on trees

- Reduction to intersecting instances

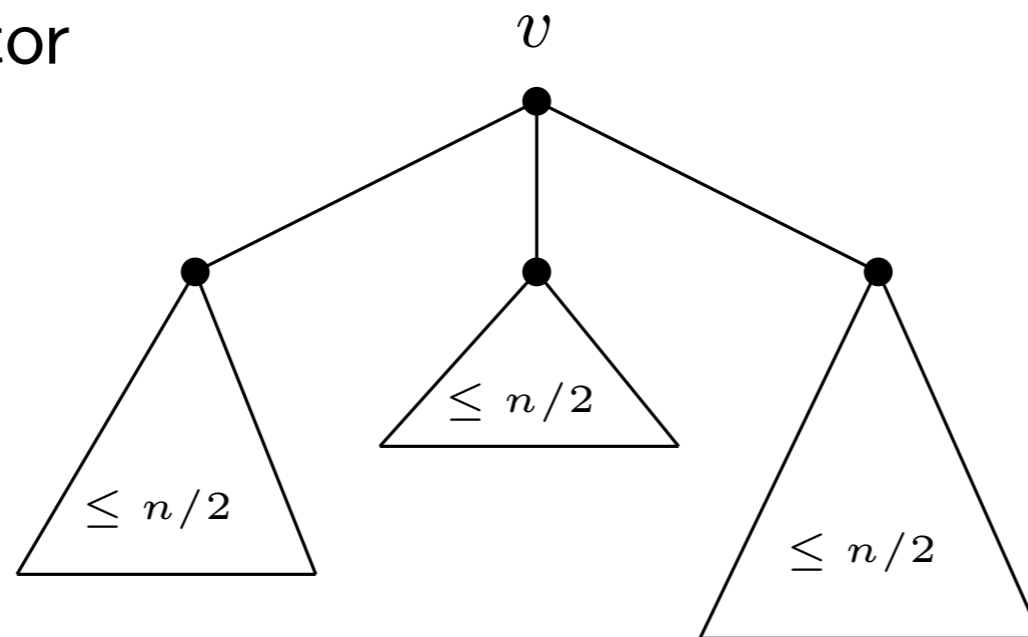
v : separator



UFP on trees

- Reduction to intersecting instances

v : separator

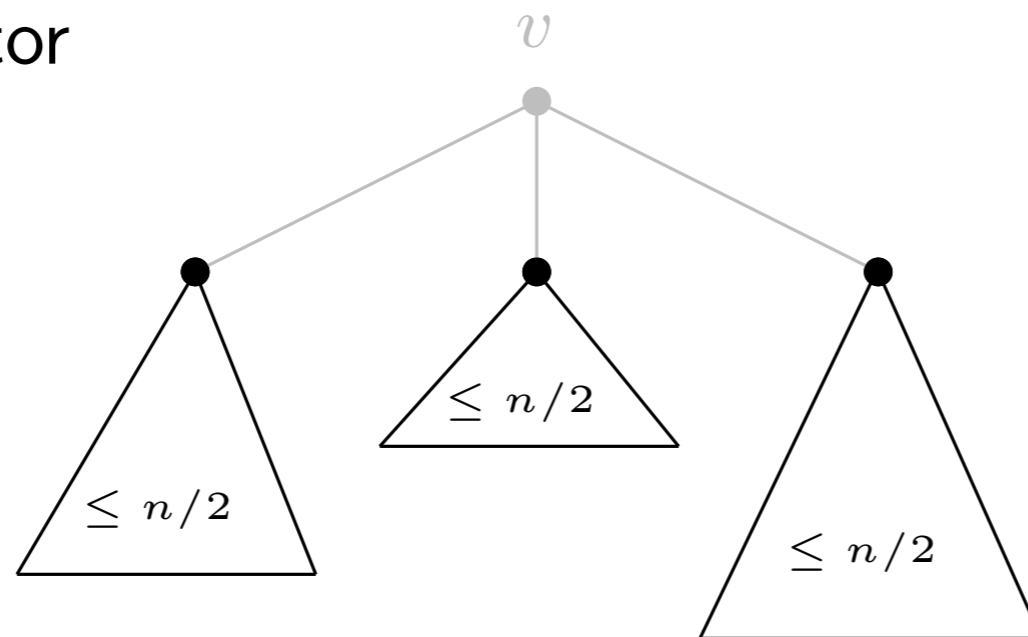


- Many requests pass through v : return these requests

UFP on trees

- Reduction to intersecting instances

v : separator

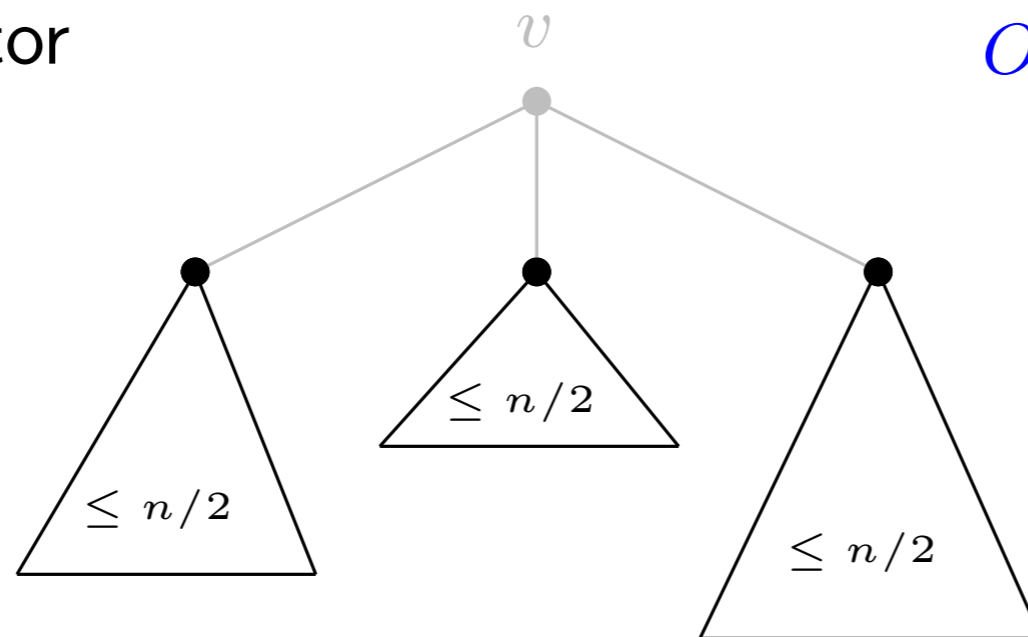


- Many requests pass through v : return these requests
- Many requests are contained in the components of $T \setminus v$: recurse

UFP on trees

- Reduction to intersecting instances

v : separator



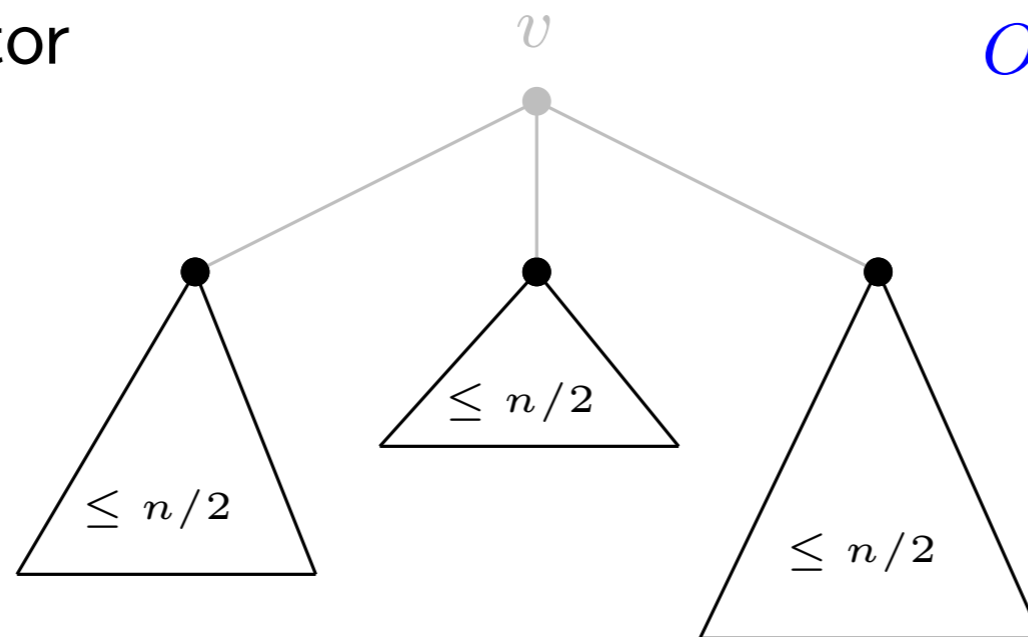
$O(\log n)$ approx for unit-weight

- Many requests pass through v : return these requests
- Many requests are contained in the components of $T \setminus v$: recurse

UFP on trees

- Reduction to intersecting instances

v : separator



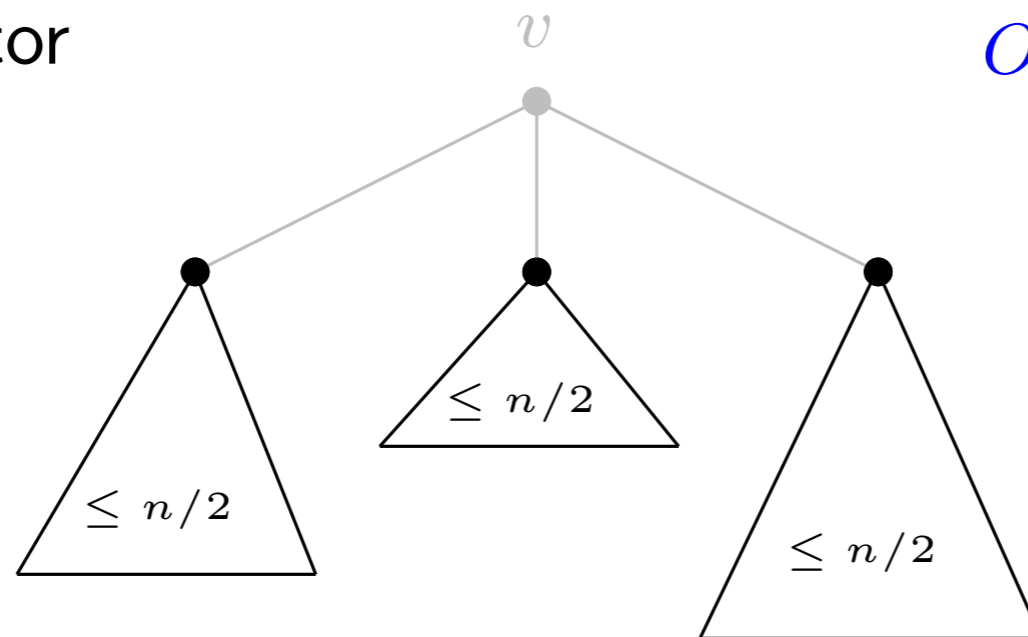
$O(\log n)$ approx for unit-weight

- Many requests pass through v : return these requests
- Many requests are contained in the components of $T \setminus v$: recurse
- Reduction to unit-weight: standard grouping and scaling

UFP on trees

- Reduction to intersecting instances

v : separator



$O(\log n)$ approx for unit-weight

$O(\log^2 n)$ approx

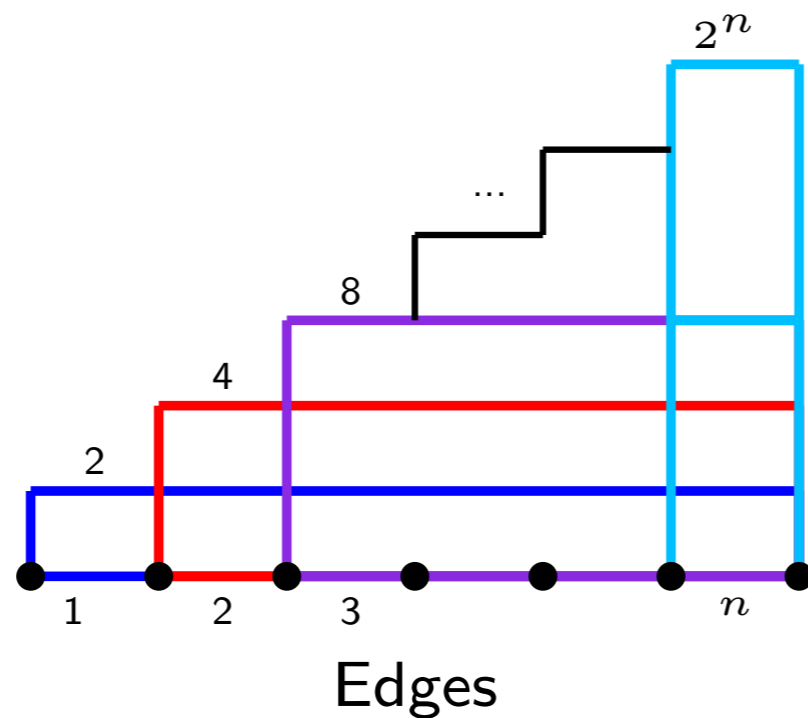
- Many requests pass through v : return these requests
- Many requests are contained in the components of $T \setminus v$: recurse
- Reduction to unit-weight: standard grouping and scaling

LP relaxation for UFP on paths?

LP relaxation for UFP on paths

LP relaxation for UFP on paths

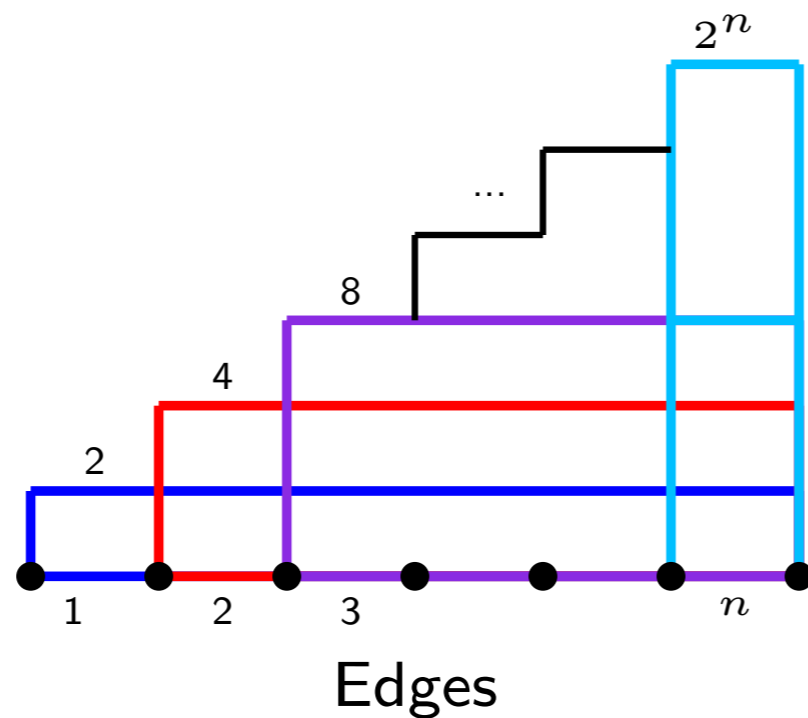
- Recall integrality example for Standard LP



$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \\ & \dots \\ d_n = 2^n & w_n = 1 \end{array}$$

LP relaxation for UFP on paths

- Recall integrality example for Standard LP

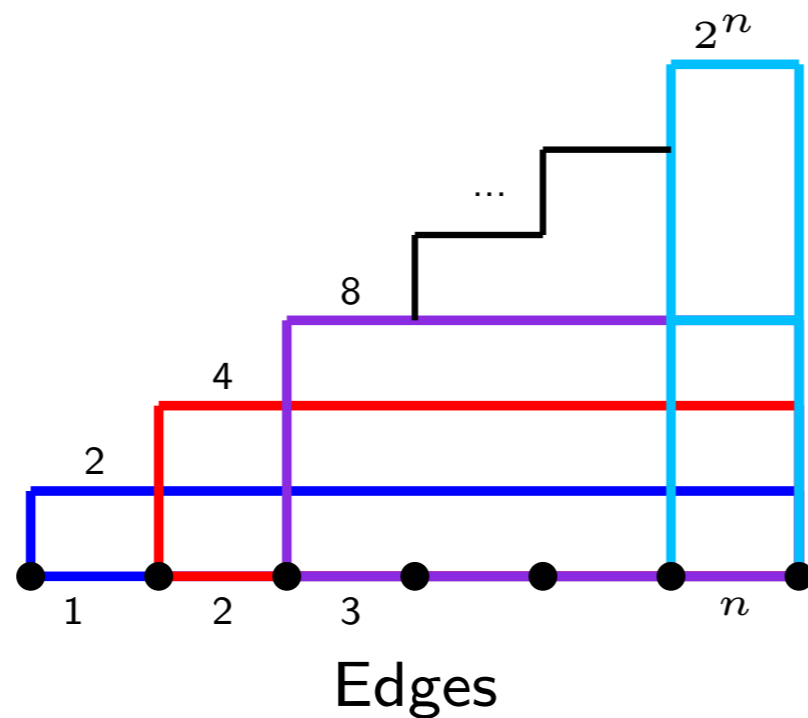


$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \\ & \dots \\ d_n = 2^n & w_n = 1 \end{array}$$

- Each request blocks all other requests

LP relaxation for UFP on paths

- Recall integrality example for Standard LP



$$\begin{array}{ll} d_1 = 2 & w_1 = 1 \\ d_2 = 4 & w_2 = 1 \\ d_3 = 8 & w_3 = 1 \\ & \dots \\ d_n = 2^n & w_n = 1 \end{array}$$

- Each request blocks all other requests
- Constrain the LP to fractionally select at most one request

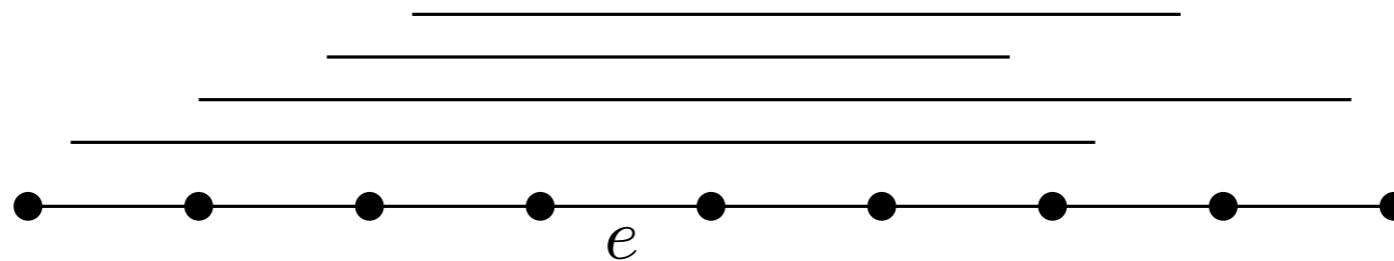
Strengthening the Standard LP

Strengthening the Standard LP

- Introduce constraints for some sets with only one routable request

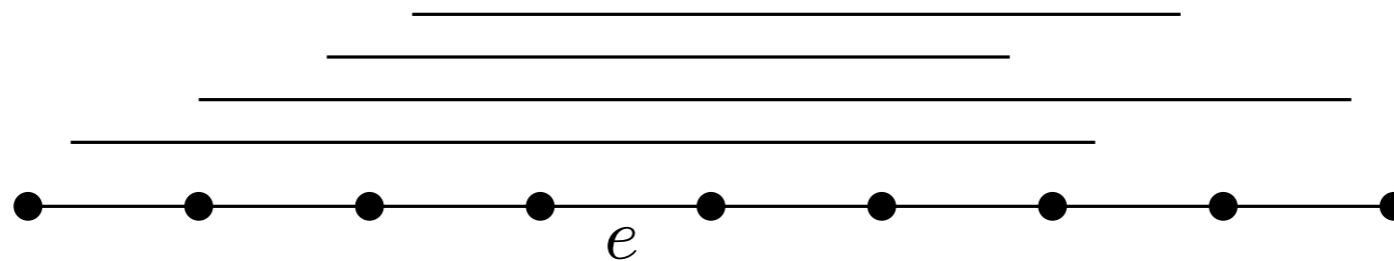
Strengthening the Standard LP

- Introduce constraints for some sets with only one routable request
- Focus on intersecting instances



Strengthening the Standard LP

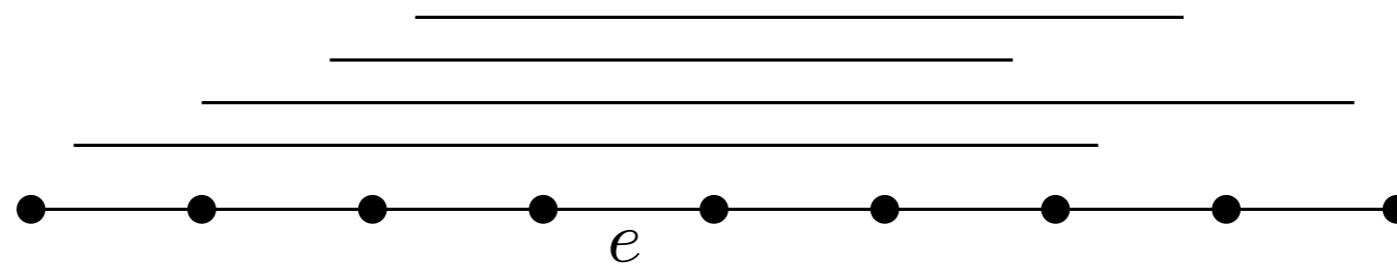
- Introduce constraints for some sets with only one routable request
- Focus on intersecting instances



- Request R_j **blocks** R_i if $d_j > d_i$ and R_i, R_j are not both routable

Strengthening the Standard LP

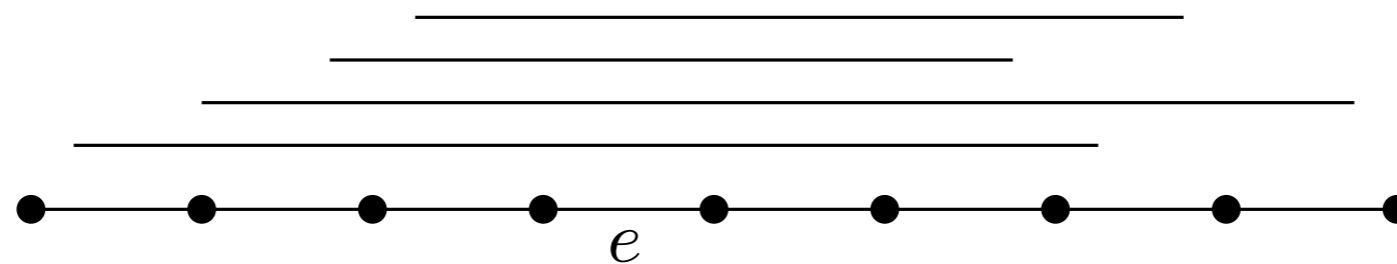
- Introduce constraints for some sets with only one routable request
- Focus on intersecting instances



- Request R_j **blocks** R_i if $d_j > d_i$ and R_i, R_j are not both routable
- Blocking set corresponding to R_i
Block(i) : R_i together with all requests that block R_i

Strengthening the Standard LP

- Introduce constraints for some sets with only one routable request
- Focus on intersecting instances



- Request R_j **blocks** R_i if $d_j > d_i$ and R_i, R_j are not both routable
- Blocking set corresponding to R_i
Block(i) : R_i together with all requests that block R_i
- Introduce constraints for all the blocking sets

A new relaxation

LP relaxation for intersecting instances

$$\mathbf{UFP-LP} \quad \max \sum_i w_i x_i$$

$$\begin{aligned} \sum_{i: e \in P_i} d_i x_i &\leq c_e && \forall e \\ \sum_{j \in \text{Block}(i)} x_j &\leq 1 && \forall i \\ x_i &\in [0, 1] \end{aligned}$$

A new relaxation

LP relaxation for intersecting instances

$$\begin{array}{ll} \mathbf{UFP-LP} & \max \sum_i w_i x_i \\ & \sum_{i: e \in P_i} d_i x_i \leq c_e \quad \forall e \\ & \sum_{j \in \text{Block}(i)} x_j \leq 1 \quad \forall i \\ & x_i \in [0, 1] \end{array}$$

Theorem. Integrality gap for UFP-LP is $O(\log n)$ for intersecting instances of UFP on paths.

A new relaxation

LP relaxation for non-intersecting instances

A new relaxation

LP relaxation for non-intersecting instances

Consider only intersecting sets

A new relaxation

LP relaxation for non-intersecting instances

Consider only intersecting sets

Theorem. Integrality gap for UFP-LP is $O(\log^2 n)$ for general instances of UFP on paths.

Remark. The paper has a more elaborate LP relaxation.

Remarks and Open Problems

- Open problems for paths/trees
 - $O(1)$ approx algorithm for UFP on paths or trees?
 - Does UFP-LP have $O(1)$ gap for paths or trees?
 - PTAS for UFP on paths?
- Recent progress
 - $O(\log n)$ approx for paths via UFP-LP

Thank You
Questions?