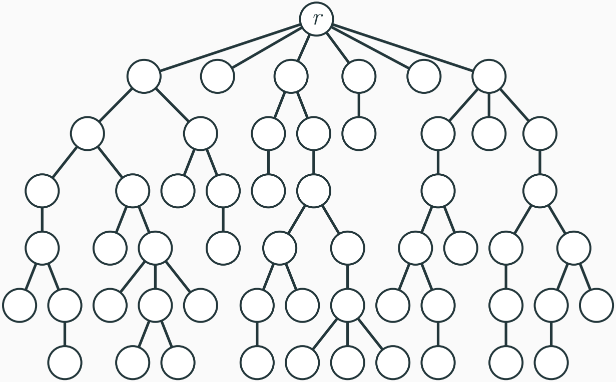
Firefighting on Trees Beyond Integrality Gaps

David Adjiashvili, Andrea Baggio, **Rico Zenklusen**

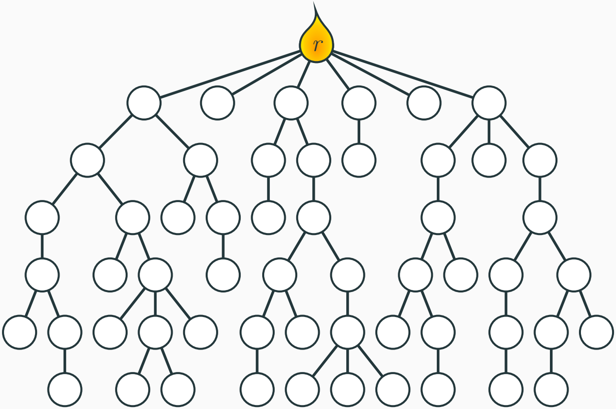
ETH Zurich

Introduction

Fire Spreading Model

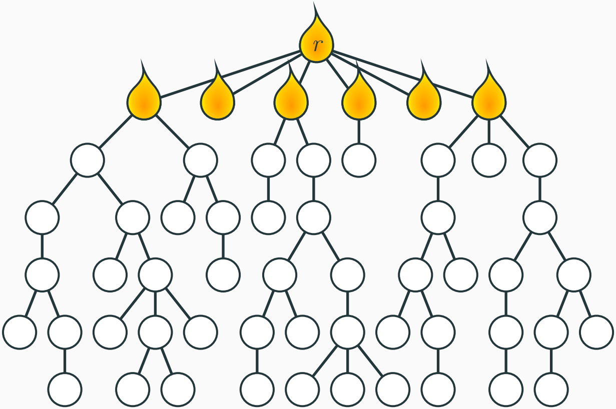


Fire Spreading Model



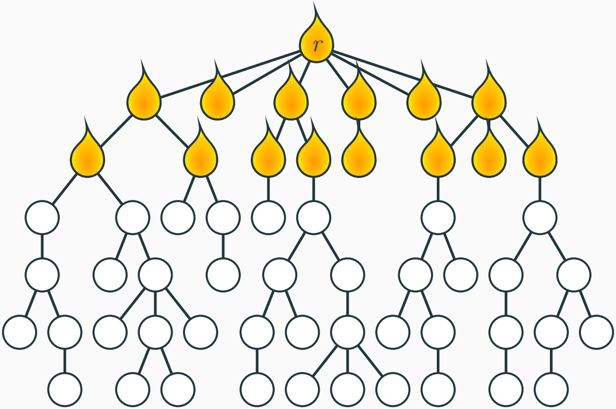
$t = 0$

Fire Spreading Model



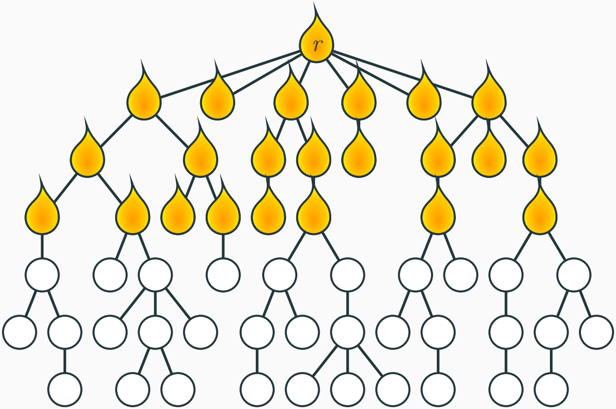
$t = 1$

Fire Spreading Model



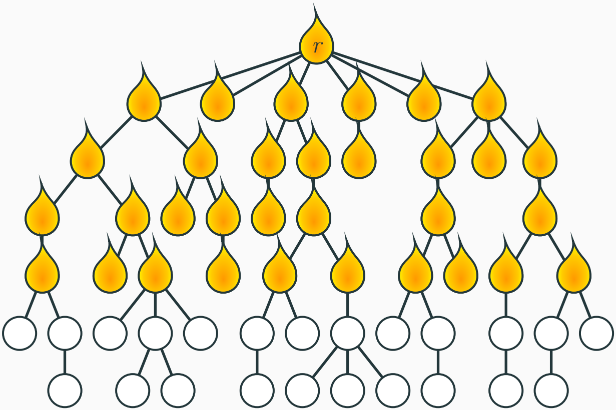
$t = 2$

Fire Spreading Model



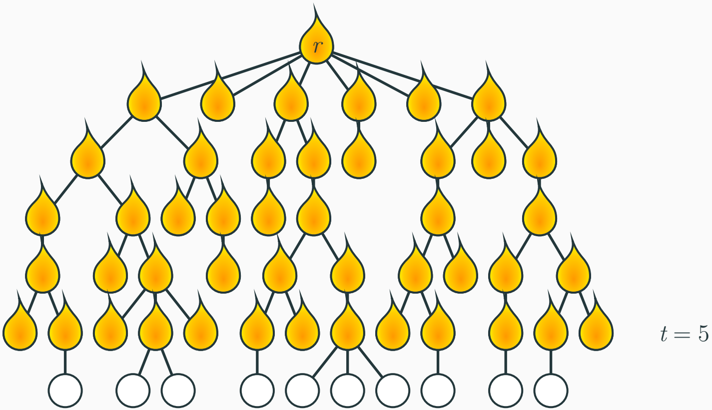
$t = 3$

Fire Spreading Model

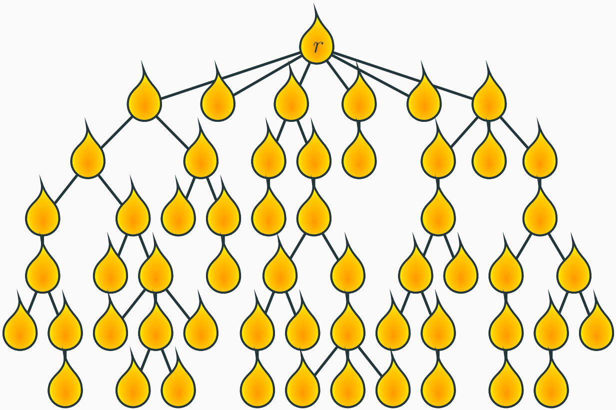


$t = 4$

Fire Spreading Model

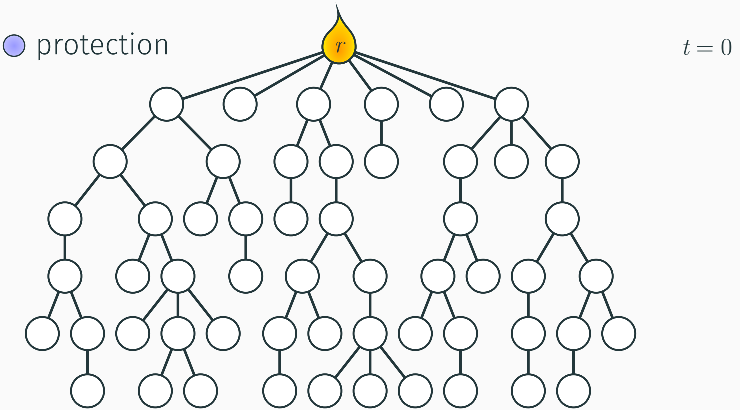


Fire Spreading Model



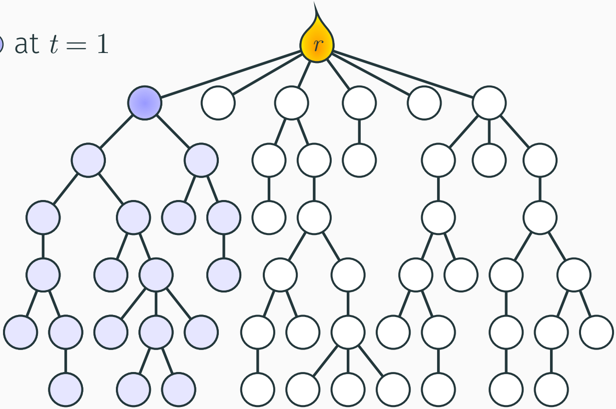
$t = 6$

FireFighter Problem - FFP



FireFighter Problem - FFP

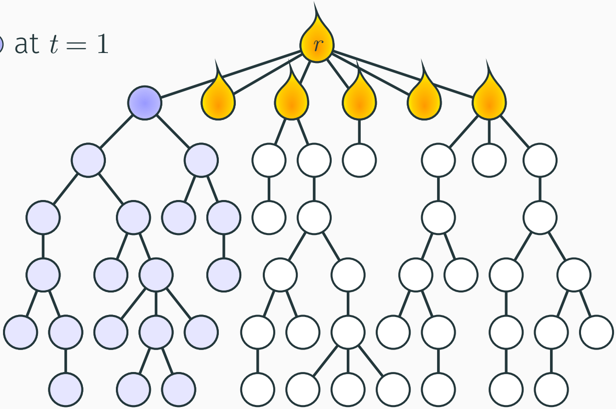
1 ● at $t = 1$



$t = 1$

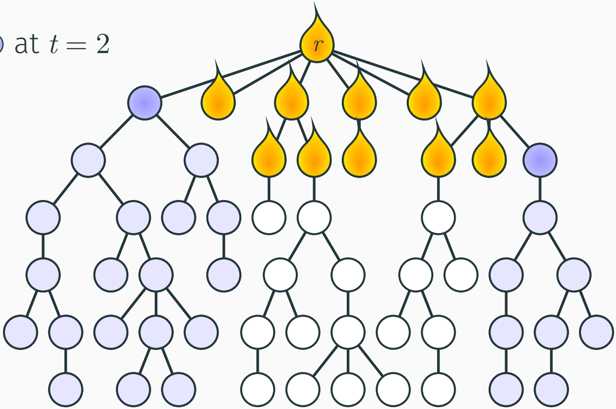
FireFighter Problem - FFP

1 ○ at $t = 1$

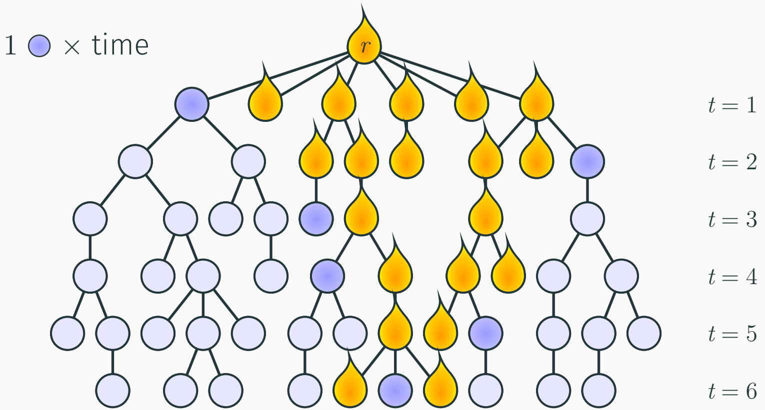


$t = 1$

1 ◯ at $t = 2$

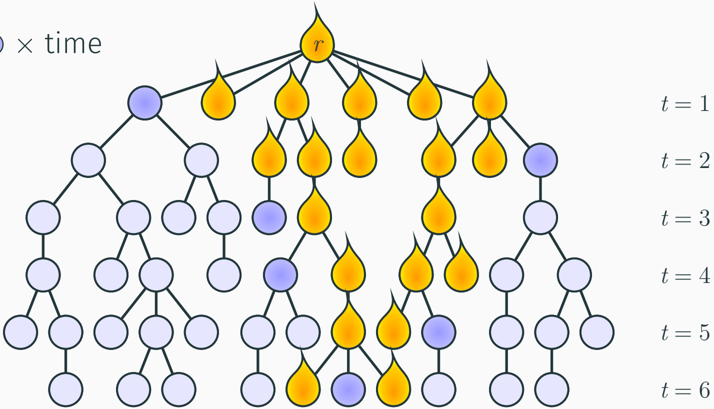


$t = 2$


1 × time
r
t = 1
t = 2
t = 3
t = 4
t = 5
t = 6

FireFighter Problem - FFP

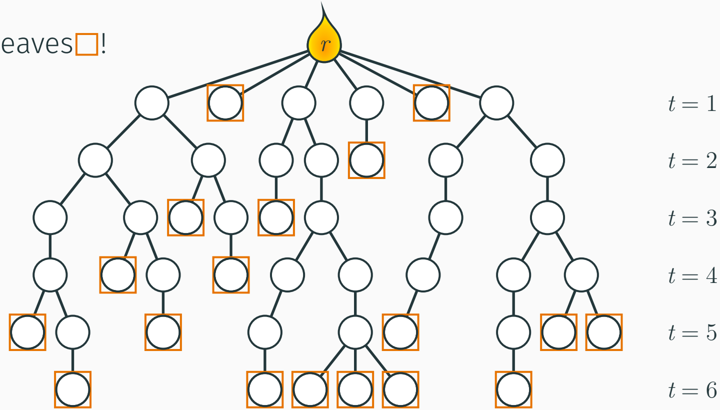
1 ● × time



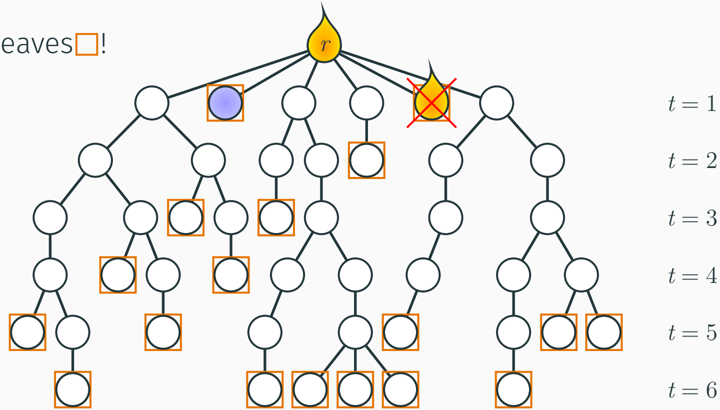
GOAL

Allocate one ● × time as to maximize saved nodes.

save all leaves□!

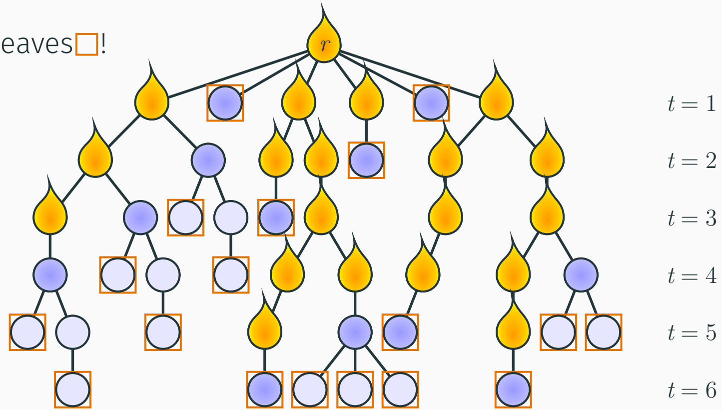


save all leaves□!



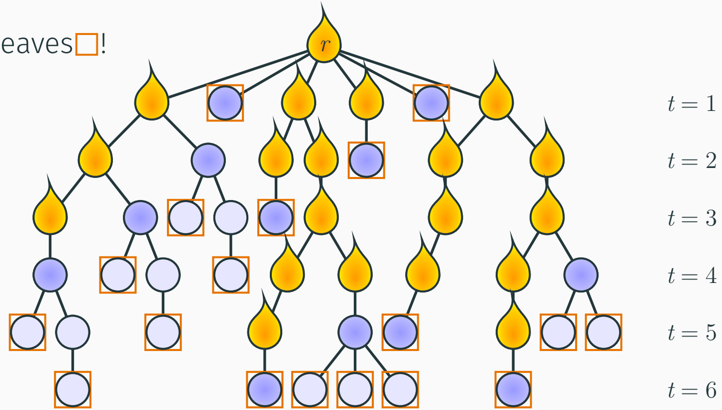
Using only one ● × time, impossible!

save all leaves□!



With two ● × time, saving all □ is possible!

save all leaves□!



GOAL

Minimize number of ● × time needed to save all leaves.

Previous Results - FireFighter Problem On Trees

Hardness - Finbow, King, MacGillivray, and Rizzi (2007)

The problem is NP-hard.

Approximation - Hartnell and Li (2000)

Simple greedy algorithm achieves $1/2$-approximation.

Approximation - Cai, Verbin, and Yang (2008)

$(1-1/e)$-approximation (LP based!).

Approx. - Anshelevich, Chakrabarty, Hate, and Swamy (2012)

$(1-1/e)$-approx. via monotone submodular function maximization subject to a partition matroid constraint.

Hardness - Finbow, King, MacGillivray, and Rizzi (2007)

NP-hard to approximate within any factor better than 2.

$O(\log n)$-approx. via constant-factor approximation for FFP.

Approximation - Chalermsook and Chuzhoy (2010)

$O(\log^* n)$-approximation (LP based!).

Do Integrality Gaps Reflect Approximation Hardness?

Current best algorithms for FFP and RMFC are LP based.

FFP

$(1-1/e)$ matches integr. gap.

(Chalermsook and Vaz (2016))

RMFC

$O(\log^* n)$ matches integr. gap up to constant-factor.

(Chalermsook & Chuzhoy (2010))

Answer not clear before!

Current best algorithms for FFP and RMFC are LP based.

FFP

$(1-1/e)$ matches integr. gap.

(Chalermsook and Vaz (2016))

RMFC

$O(\log^* n)$ matches integr. gap up to constant-factor.

(Chalermsook & Chuzhoy (2010))

Answer not clear before!

FFP

Special case of monotone submodular max. subject to partition matroid.
(no $(1-1/e+\epsilon)$-approx Fisher, Nemhauser and Wolsey [1978], Feige [1998]).

RMFC

Similar to the asymmetric k-center problem.
(no $o(\log^* n)$-approx, unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\log\log n})$ (Chuzhoy, Guha, Halperin, Khanna, Kortsarz, Krauthgamer, and Naor (2005))).

Theorem - Adjiashvili, Baggio, and Zenklusen (2016)

PTAS for the FireFighter Problem on trees.

Theorem - Adjiashvili, Baggio, and Zenklusen (2016)

$O(1)$-approximation for RMFC on trees.

Theorem - Adjiashvili, Baggio, and Zenklusen (2016)

PTAS for the FireFighter Problem on trees.

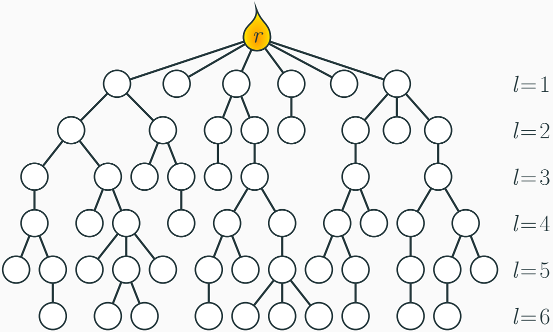
Theorem - Adjiashvili, Baggio, and Zenklusen (2016)

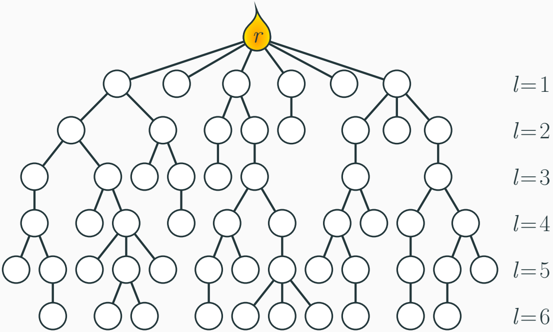
$O(1)$-approximation for RMFC on trees.

Both algorithms are guided by the same known LPs!

12-Approximation for the RMFC Problem

The Linear Program



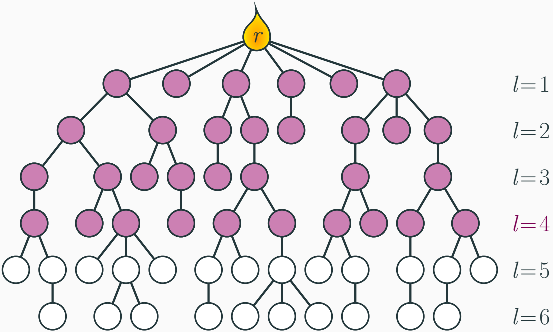


$$x_u = \begin{cases} 1 & \text{if } \bullet \\ 0 & \text{ow.} \end{cases}$$

$$\min_{B \in \mathbb{R}_{\geq 1}} \quad B$$

$$\text{s. t.}$$

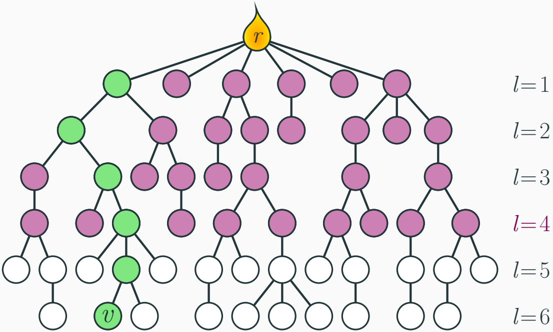
$$x_u \in \{0, 1\} \qquad \forall\ u \in V$$



$$x_u = \begin{cases} 1 & \text{if } \bullet \\ 0 & \text{ow.} \end{cases}$$

$V_{\leq l}$ = levels $\leq l$

$$\begin{aligned} \min_{B \in \mathbb{R}_{\geq 1}} \quad & B \\ \text{s.t.} \quad & \\ & x(V_{\leq l}) \leq B \cdot l && \forall\, l \in \text{levels} \\ & x_u \in \{0,1\} && \forall\, u \in V \end{aligned}$$

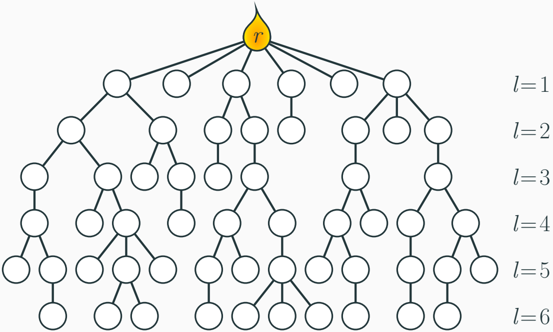


$x_u = \begin{cases} 1 & \text{if } \bullet \\ 0 & \text{ow.} \end{cases}$

$V_{\leq l}$ = levels $\leq l$

P_v = path $r \to v$

$$\begin{aligned} \min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\ \text{s. t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\ & x(V_{\leq l}) \leq B \cdot l & \forall\, l \in \text{levels} \\ & x_u \in \{0,1\} & \forall\, u \in V \end{aligned}$$



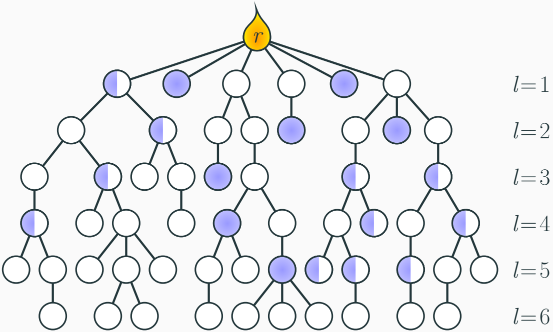
$$x_u = \begin{cases} 1 & \text{if } \bigcirc \\ 0 & \text{ow.} \end{cases}$$

$V_{\leq l}$ = levels $\leq l$

P_v = path $r \to v$

$$(LP) \qquad \begin{aligned} \min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\ \text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\ & x(V_{\leq l}) \leq B \cdot l & \forall\, l \in \text{levels} \\ & x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \end{aligned}$$

The Linear Program



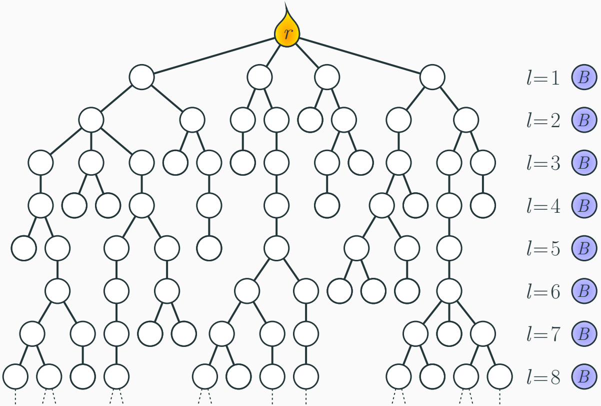
$$x_u = \begin{cases} 1 & \text{if } \bullet \\ 0 & \text{ow.} \end{cases}$$

$V_{\leq l}$ = levels $\leq l$

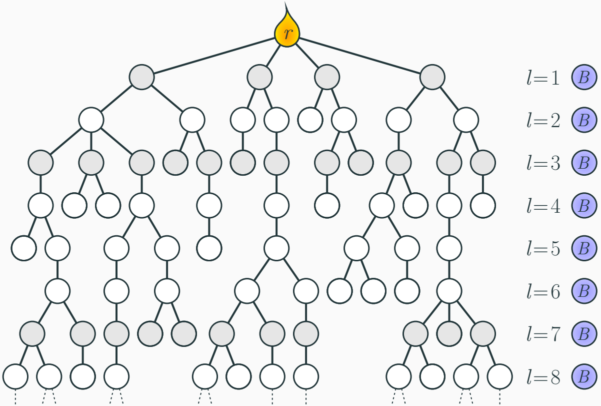
P_v = path $r \to v$

$$(LP) \quad \begin{aligned} \min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\ \text{s. t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\ & x(V_{\leq l}) \leq B \cdot l & \forall\, l \in \text{levels} \\ & x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \end{aligned}$$

Compression - Reducing Depth to $O(\log N)$ levels

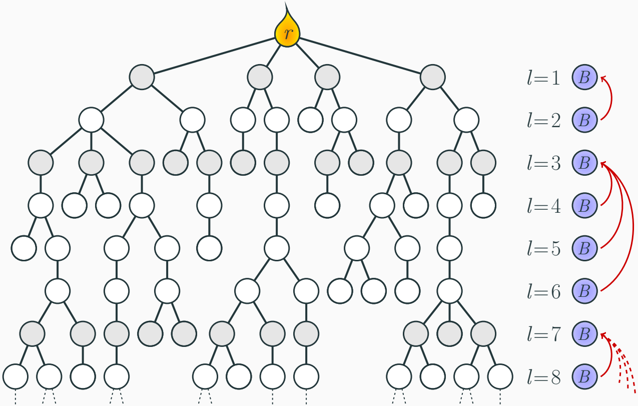


Compression - Reducing Depth to $O(\log N)$ levels



$l = 2^i - 1,\ i = 1, 2, \ldots$

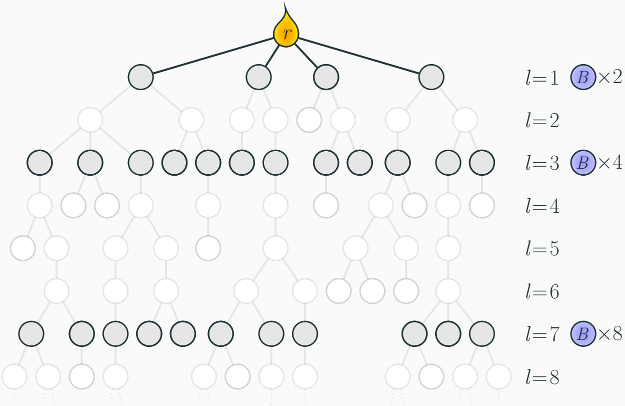
Compression - Reducing Depth to $O(\log N)$ levels



$l=2^i$-1, $i=1,2,\ldots$

up-push

Compression - Reducing Depth to $O(\log N)$ levels

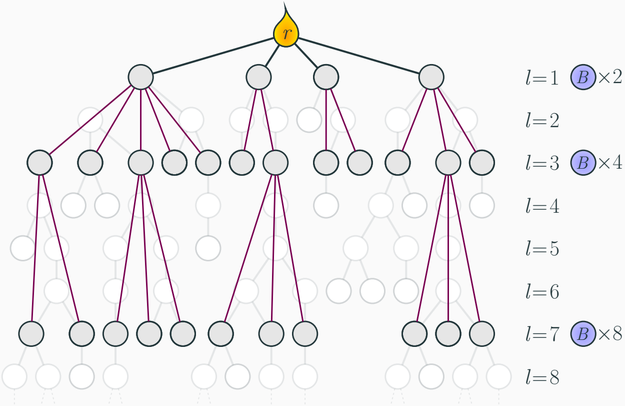


$l = 2^i\text{-}1,\ i = 1,2,\ldots$

up-push

erase

Compression - Reducing Depth to $O(\log N)$ levels



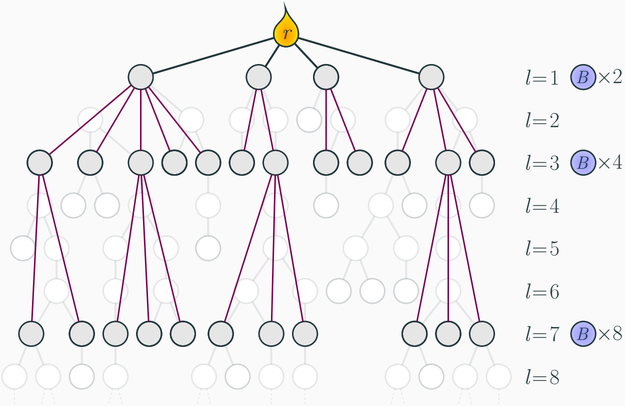
$l = 2^i - 1,\ i = 1,2,\ldots$

up-push

erase

connect

Compression - Reducing Depth to $O(\log N)$ levels



$l = 2^i$-1, $i = 1,2,\ldots$

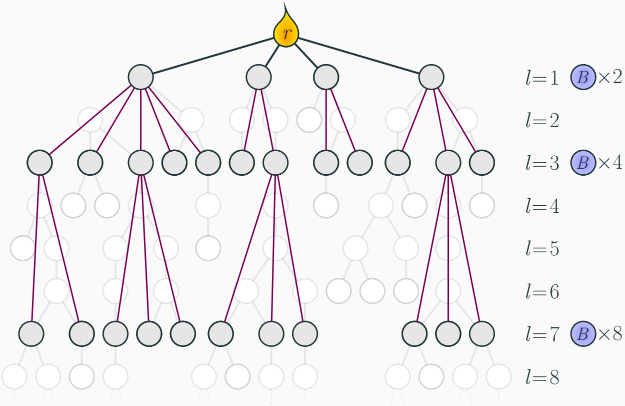
up-push

erase

connect

Compressed tree has $L = O(\log N)$ levels.

Compression - Reducing Depth to $O(\log N)$ levels



$l = 2^i$-1, i=1,2,...

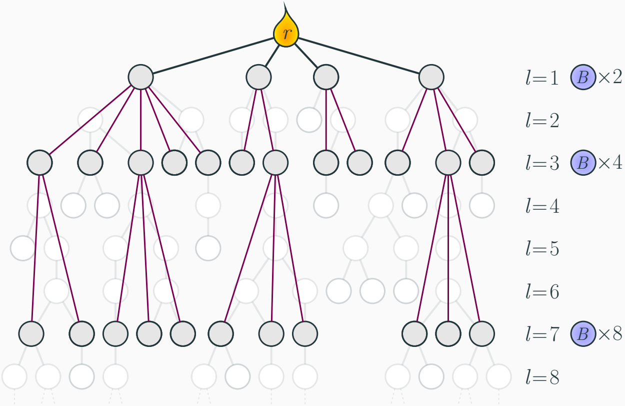
up-push

erase

connect

- If RMFC_{orig} is feasible for budget B
  $\Rightarrow \mathrm{RMFC}_{comp}$ is feasible for $B_{l'} = 2^{l'} \cdot B$.
- Solution to RMFC_{comp} with $B_{l'} = 2^{l'} \cdot B$
  $\rightarrow$ solution to RMFC_{orig} with $2B$.

Compression - Reducing Depth to $O(\log N)$ levels



$l = 2^i$-1, $i = 1,2,\ldots$

up-push

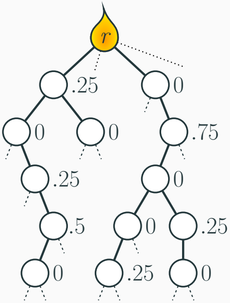
erase

connect

Compression + Dynamic Programming

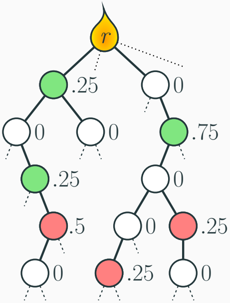
$\rightarrow$ 2-approximation in quasi-poly time.

Unclear how to make this algo efficient!



Let x^* be a vertex solution of (LP).

Deeper into the LP: Loose and Tight Nodes



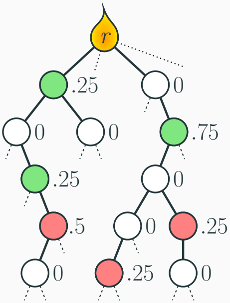
Let x^* be a vertex solution of (LP).

A node u is loose w.r.t. x^* if

- $x_u^* > 0$
- $x^*(P_u) < 1$

A node u is tight w.r.t. x^* if

- $x_u^* > 0$
- $x^*(P_u) = 1$



Let x^* be a vertex solution of (LP).

A node u is loose w.r.t. x^* if

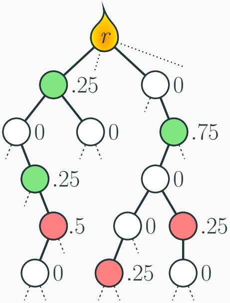
- $x^*_u > 0$
- $x^*(P_u) < 1$

A node u is tight w.r.t. x^* if

- $x^*_u > 0$
- $x^*(P_u) = 1$

Constraint matrix on tight vertices is TU.

Deeper into the LP: Loose and Tight Nodes



Let x^* be a vertex solution of (LP).

A node u is loose w.r.t. x^* if

- $x^*_u > 0$
- $x^*(P_u) < 1$

A node u is tight w.r.t. x^* if

- $x^*_u > 0$
- $x^*(P_u) = 1$

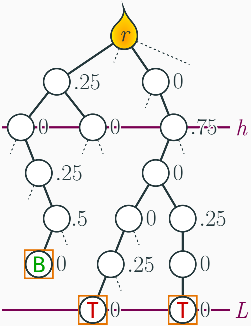
Constraint matrix on tight vertices is TU.

Sparsity Lemma

Last h levels contain $\leq h$ loose nodes.

$\rightarrow$ Easy to get solution with LP-value if, on top of budget, depth-many (loose) vertices can be protected for free.

Even deeper into the LP: Bottom and Top Heavy Leaves



Let x^* be a vertex solution of (LP).

Fix level $h = \lfloor \log L \rfloor$.

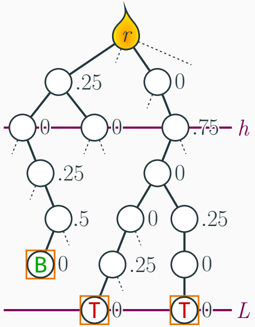
A leaf v is bottom-heavy if

- $x^*(P_v \cap V_{>h}) \geq 0.5$

A leaf v is top-heavy if

- it is not bottom-heavy

Even deeper into the LP: Bottom and Top Heavy Leaves



Let x^* be a vertex solution of (LP).

Fix level $h = \lfloor \log L \rfloor$.

A leaf v is bottom-heavy if

- $x^*(P_v \cap V_{>h}) \geq 0.5$

A leaf v is top-heavy if

- it is not bottom-heavy

Bottom-heavy leaves can be saved with budget $2B + 1$

- Protect all loose nodes in $V_{>h}$: # loose vertices $\leq L$.
  Can be done by replacing B by $B + 1$:
  Each level $\ell > h$ gets $2^\ell \geq L$ more budget.
- Solve TU-system over tight vertices in $V_{>h}$ to save all T.
  $2x$ is feasible for this LP → suffices to double budget.

Fix some level $h = \lfloor \log^{(q)} L \rfloor$ and $\mu \in (0, 1]$.

A leaf v is bottom-heavy if

- $x^*(P_v \cap V_{>h}) \geq \mu$

A leaf v is top-heavy if

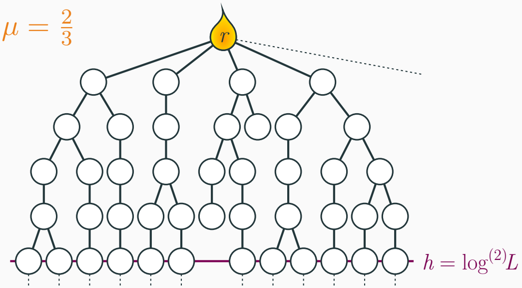
- it is not bottom-heavy

Theorem

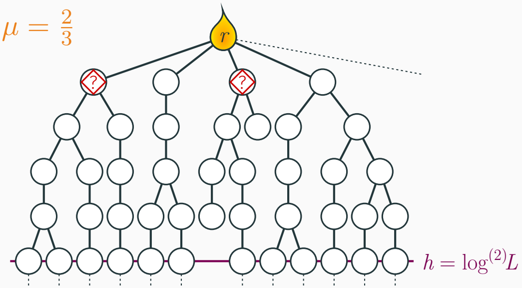
Using budget $\frac{q}{\mu} B + 1$ only on the bottom part,
all bottom-heavy leaves can be saved.

If $\mu = 1$ and $q = \log^* L$ $\rightarrow$ $O(\log^* N)$-approximation.

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



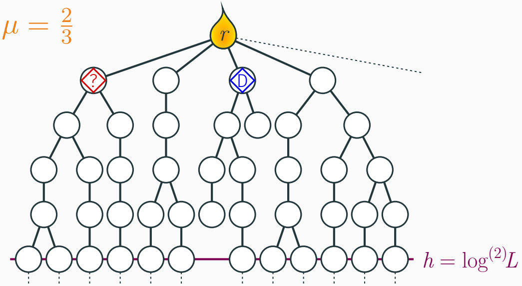
Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP and find exploration set ◇.

◇: topmost nodes covering Ⓣ leaves

$$(LP) \qquad \begin{array}{lll} \min\limits_{B \in \mathbb{R}_{\geq 1}} & B & \\ \text{s.t.} & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\ & x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\ & x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \end{array}$$



Run LP and find exploration set ?.

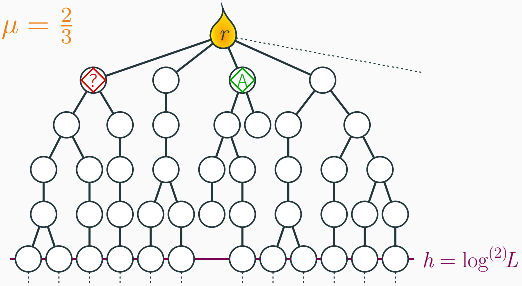
?: topmost nodes covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$
(LP) \quad \begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



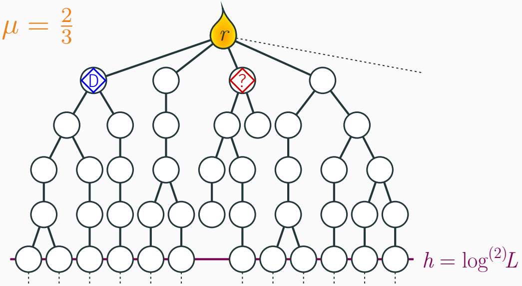
Run LP and find exploration set ◇?.

◇?: topmost nodes covering (T) leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP) \qquad \begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B \\
\text{s.t.} \quad & x(P_v) \geq 1 && \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l && \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} && \forall\, u \in V
\end{aligned}
$$



Run LP and find exploration set ◇?.

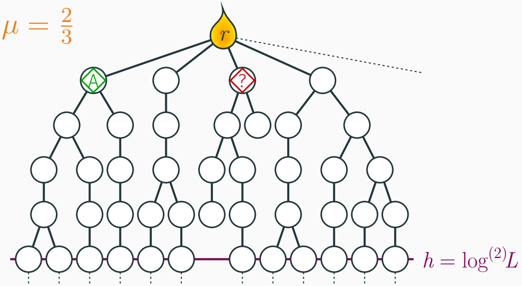
◇?: topmost nodes covering (T) leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP) \quad \begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP and find exploration set ?.

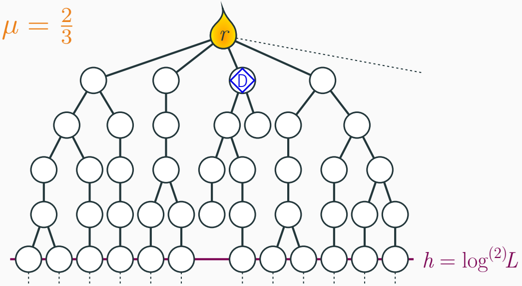
?: topmost nodes covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$(LP) \quad \begin{array}{lll} \min_{B \in \mathbb{R}_{\geq 1}} & B & \\ \text{s.t.} & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\ & x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\ & x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \end{array}$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP and find exploration set ?.

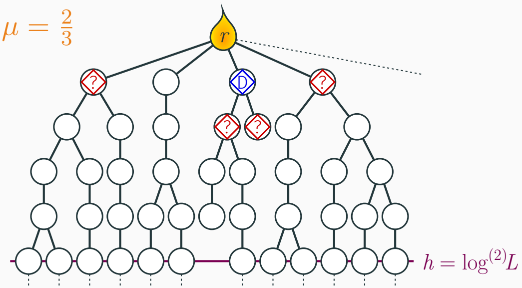
?: topmost nodes covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$
(LP) \quad \begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\ v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\ l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\ u \in V
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

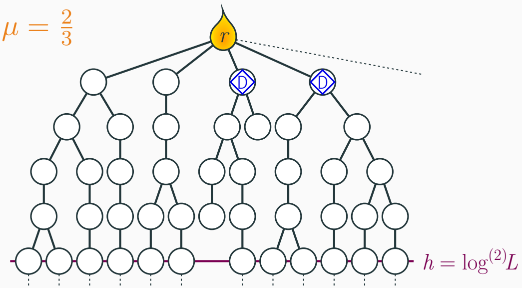
> ◇?: topmost nodes + ◇A + ◇D covering (T) leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(\text{◇A}, \text{◇D})) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in \text{◇A} \\
& x_u = 0 & \forall\, u \in \text{◇D}
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(A,D) and find exploration set ?.

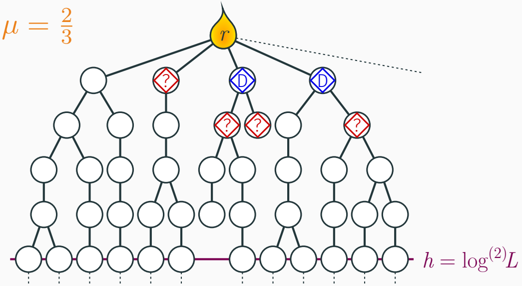
?: topmost nodes + A + D covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$
(LP(A, D)) \quad \begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B \\
\text{s.t.} \quad & x(P_v) \geq 1 && \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l && \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} && \forall\, u \in V \\
& x_u = 1 && \forall\, u \in A \\
& x_u = 0 && \forall\, u \in D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

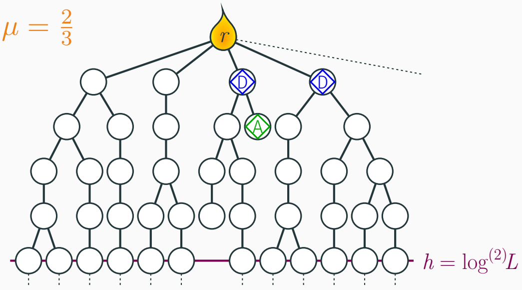
◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

“Guess”/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(\Diamond_A, \Diamond_D)) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in \Diamond_A \\
& x_u = 0 & \forall\, u \in \Diamond_D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



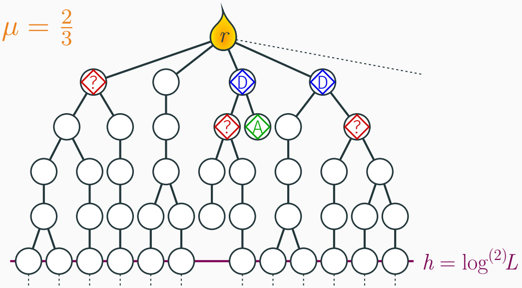
Run LP(A, D) and find exploration set ?.

?: topmost nodes + A + D covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$
(LP(A, D)) \quad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in A \\
& x_u = 0 & \forall\, u \in D
\end{aligned}
$$



Run LP(A, D) and find exploration set ?.

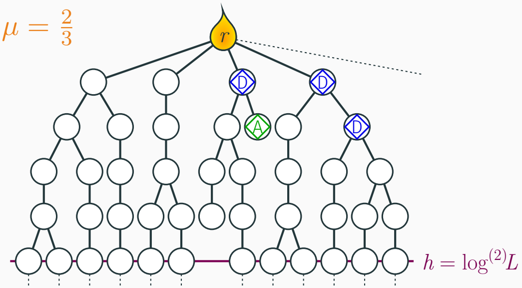
?: topmost nodes + A + D covering T leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

$$
(LP(A, D)) \quad \min_{B \in \mathbb{R}_{\geq 1}} B
$$

$$
\begin{aligned}
\text{s.t.} \quad & x(P_v) \geq 1 && \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l && \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} && \forall\, u \in V \\
& x_u = 1 && \forall\, u \in A \\
& x_u = 0 && \forall\, u \in D
\end{aligned}
$$



Run LP(◊A, ◊D) and find exploration set ◊?.

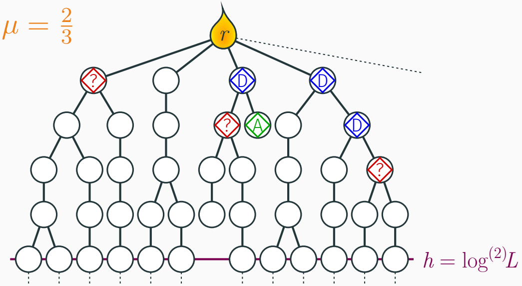
◊?: topmost nodes + ◊A + ◊D covering Ⓣ leaves

"Guess"/explore ◊?:

- ◊A if protected by OPT
- ◊D if not protected

$$
(LP(\Diamond A, \Diamond D)) \quad
\begin{array}{llll}
\min\limits_{B \in \mathbb{R}_{\geq 1}} & B & \\
\text{s.t.} & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
 & x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
 & x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
 & x_u = 1 & \forall\, u \in \Diamond A \\
 & x_u = 0 & \forall\, u \in \Diamond D
\end{array}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A,◇D) and find exploration set ◇?.

> ◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

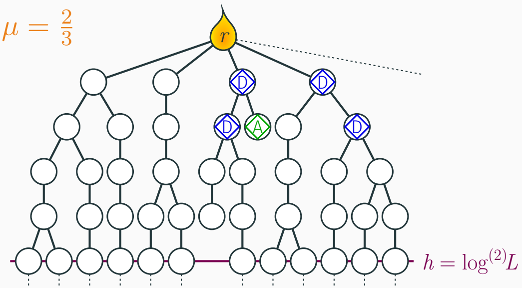
“Guess”/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

(LP(◇A, ◇D))

$$\min_{B \in \mathbb{R}_{\geq 1}} \quad B$$

s. t.	$x(P_v) \geq 1$	$\forall\, v \in$ leaves
	$x(V_{\leq l}) \leq B \cdot 2^l$	$\forall\, l \in$ levels
	$x_u \in \mathbb{R}_{\geq 0}$	$\forall\, u \in V$
	$x_u = 1$	$\forall\, u \in$ ◇A
	$x_u = 0$	$\forall\, u \in$ ◇D



Run LP(◇A,◇D) and find exploration set ◇?.

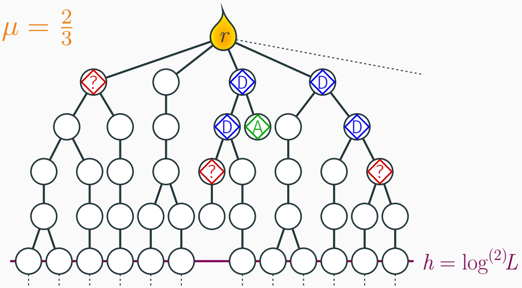
> ◇?: topmost nodes + ◇A + ◇D covering (T) leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(\lozenge A, \lozenge D)) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in \lozenge A \\
& x_u = 0 & \forall\, u \in \lozenge D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

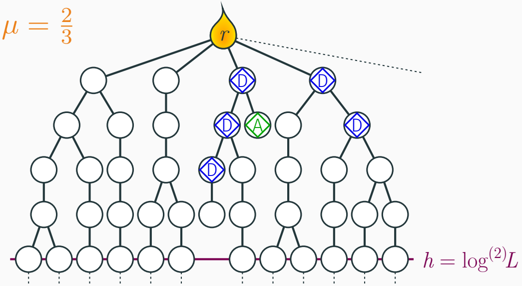
◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(◇A, ◇D)) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in ◇A \\
& x_u = 0 & \forall\, u \in ◇D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

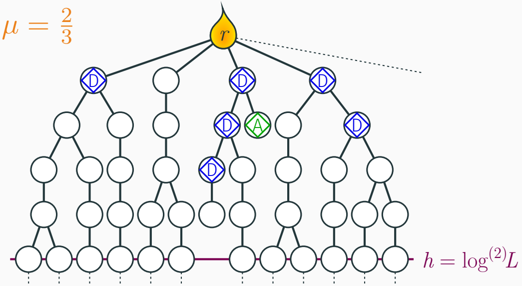
◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(\text{◇A}, \text{◇D})) \quad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in \text{◇A} \\
& x_u = 0 & \forall\, u \in \text{◇D}
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

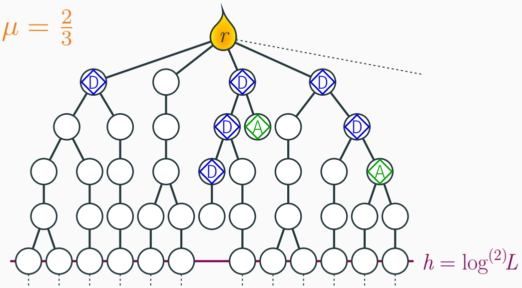
◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(◇A, ◇D)) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in ◇A \\
& x_u = 0 & \forall\, u \in ◇D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



Run LP(◇A, ◇D) and find exploration set ◇?.

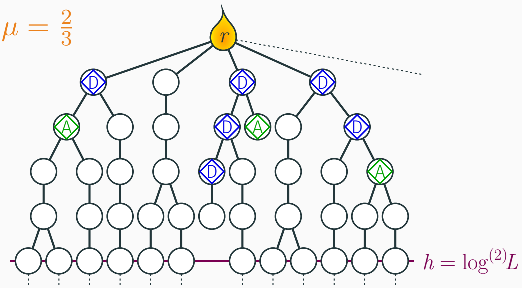
> ◇?: topmost nodes + ◇A + ◇D covering (T) leaves

“Guess”/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(◇A, ◇D)) \quad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in ◇A \\
& x_u = 0 & \forall\, u \in ◇D
\end{aligned}
$$

Saving Top-Heavy Leaves: Iterative Guessing Guided by LP



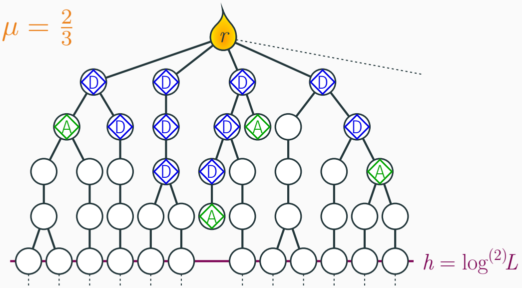
Run LP(◇A, ◇D) and find exploration set ◇?.

◇?: topmost nodes + ◇A + ◇D covering (T) leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

$$
(LP(\text{◇A}, \text{◇D})) \qquad
\begin{aligned}
\min_{B \in \mathbb{R}_{\geq 1}} \quad & B & \\
\text{s.t.} \quad & x(P_v) \geq 1 & \forall\, v \in \text{leaves} \\
& x(V_{\leq l}) \leq B \cdot 2^l & \forall\, l \in \text{levels} \\
& x_u \in \mathbb{R}_{\geq 0} & \forall\, u \in V \\
& x_u = 1 & \forall\, u \in \text{◇A} \\
& x_u = 0 & \forall\, u \in \text{◇D}
\end{aligned}
$$



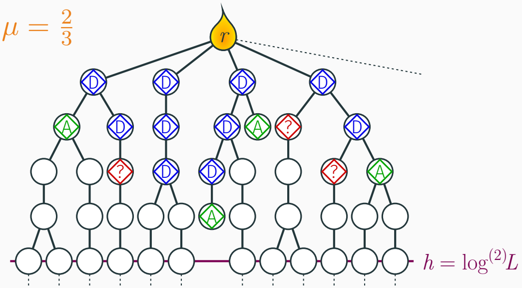
Run LP(A,D) and find exploration set ?.

?: topmost nodes + A + D covering (T) leaves

"Guess"/explore ?:

- A if protected by OPT
- D if not protected

- r + A + D forms a subtree with no D below an A.
- Exaustive search: $\Rightarrow$ At each recursion depth, at least one pair (A,D) is compatible with OPT.
- $|?| = O\left(\log^2 L\right)$.



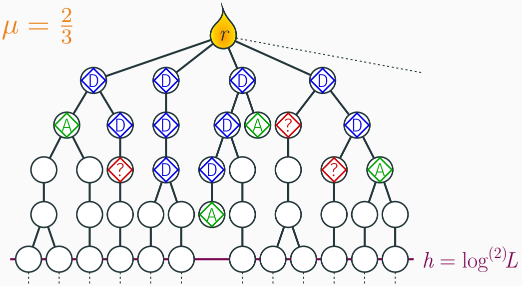
Run LP([A],[D]) and find exploration set [?].

[?]: topmost nodes + [A] + [D] covering (T) leaves

"Guess"/explore [?]:

- [A] if protected by OPT
- [D] if not protected

Saving Top-Heavy Leaves: When Having Good Guess



Run LP(◇A, ◇D) and find exploration set ◇?.

◇?: topmost nodes + ◇A + ◇D covering Ⓣ leaves

"Guess"/explore ◇?:

- ◇A if protected by OPT
- ◇D if not protected

Lemma

If ◇? does not cover any ●$_{\mathrm{OPT}}$ within $V_{\leq h}$:

- Problem reduces to bottom part!
- $\rightarrow LP($◇A$, V_{\leq h} \setminus$ ◇A$)$ has value $B \leq \frac{5}{2} B_{\mathrm{OPT}}$.
- Using approach for bottom part:
  $\rightarrow$ Solution to compressed RMFC with $B \leq 6 B_{\mathrm{OPT}}$.

Enumerating Triples ($\diamond_A$, $\diamond_D$, $\diamond_?$): a Recursive Algorithm

Enum($\diamond_A$, $\diamond_D$, γ): Enumerating explored triples.

1. Get opt vertex sol (x, B) to $LP(\diamond_A, \diamond_D)$ and find $\diamond_?$.
2. If $B > \log L$: **stop**
3. Add ($\diamond_A$, $\diamond_D$, $\diamond_?$) to the triples to be considered.
4. If $\gamma \neq 0$:
   - For $u \in \diamond_?$:
     - $\text{Enum}(\diamond_A + u, \diamond_D, \gamma - 1)$
     - $\text{Enum}(\diamond_A, \diamond_D + u, \gamma - 1)$

Enumerating Triples (◇A,◇D,◇?): a Recursive Algorithm

Enum(◇A, ◇D, γ): Enumerating explored triples.

1. Get opt vertex sol (x, B) to LP(◇A, ◇D) and find ◇?.
2. **If** $B > \log L$: **stop**
3. Add (◇A,◇D,◇?) to the triples to be considered.
4. **If** $\gamma \neq 0$:
    - **For** $u \in$ ◇?:
        - Enum(◇A $+ u$, ◇D, $\gamma - 1$)
        - Enum(◇A, ◇D $+ u$, $\gamma - 1$)

If γ small enough:

→ poly-number of triples.

If γ large enough:

→ ∃ (◇A, ◇D, ◇?) s.t. ◇? does not cover $\bigcirc_{\mathrm{OPT}}$ within $V_{\leq h}$.

Finding the Right γ $\rightarrow$ $\gamma = O(\log^2 L \log^{(2)} L)$

Algorithm is efficient

Recall $|\diamond| = O(\log^2 L)$.

$\rightarrow$ Number of recursive calls

$$\leq O(|\diamond|^\gamma) = (\log L)^{O(\log^2 L \log^{(2)} L)} = 2^{o(L)} = o(N).$$

Finding the Right γ $\rightarrow$ $\gamma = O(\log^2 L \, \log^{(2)} L)$

Algorithm is efficient

Recall $|$◇?$| = O(\log^2 L)$.

$\rightarrow$ Number of recursive calls

$$\leq O(|◇?|^\gamma) = (\log L)^{O(\log^2 L \, \log^{(2)} L)} = 2^{o(L)} = o(N).$$

Enumeration finds good triple (◇A,◇D,◇?)

$\Phi(A, D):$ Sum of distances from each ●$_{\mathrm{OPT}}$ in $V_{\leq h}$ to $r +$ ◇A $+$ ◇D.

Assume (◇A, ◇D) is compatible with ●$_{\mathrm{OPT}}$:

$\rightarrow$ If some ●$_{\mathrm{OPT}}$ in $V_{\leq h}$ below ◇? $\Rightarrow$ Φ decreases for good guess.

A good triple (◇A,◇D,◇?) is found if $\gamma \geq \Phi(\emptyset, \emptyset)$, and

$$\Phi(\emptyset, \emptyset) < h \cdot \underbrace{2^{h+1} B_{\mathrm{OPT}}}_{\text{total budget in } V_{\leq h}} \leq O(\log^{(2)} L \, \log^2 L).$$

Conclusions

Theorem

$O(1)$-approximation for RMFC on trees.

We use:

- Compression.
- LP-guided recursive enumeration algorithm to guess super-constant set of vertices to protect.

OPEN: 2-approximation for RMFC.
(We have a quasi-polynomial 2-approximation.)

Theorem

PTAS for the FireFighter Problem on trees.