TEL AVIV UNIVERSITY

Chasing Positive Bodies

Roie Levin

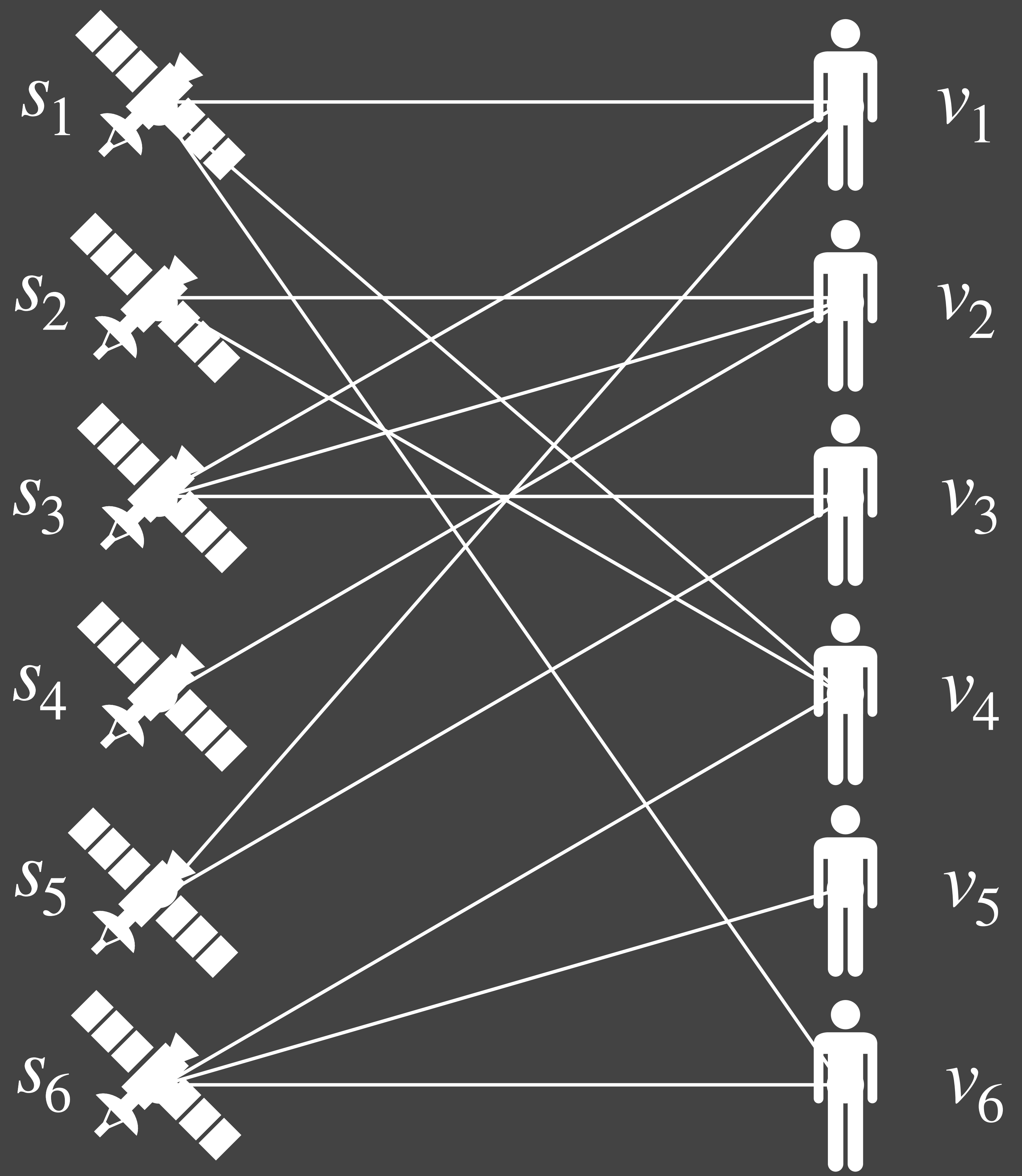
Sayan Bhattacharya
(U. of Warwick)

Niv Buchbinder
(Tel Aviv U.)

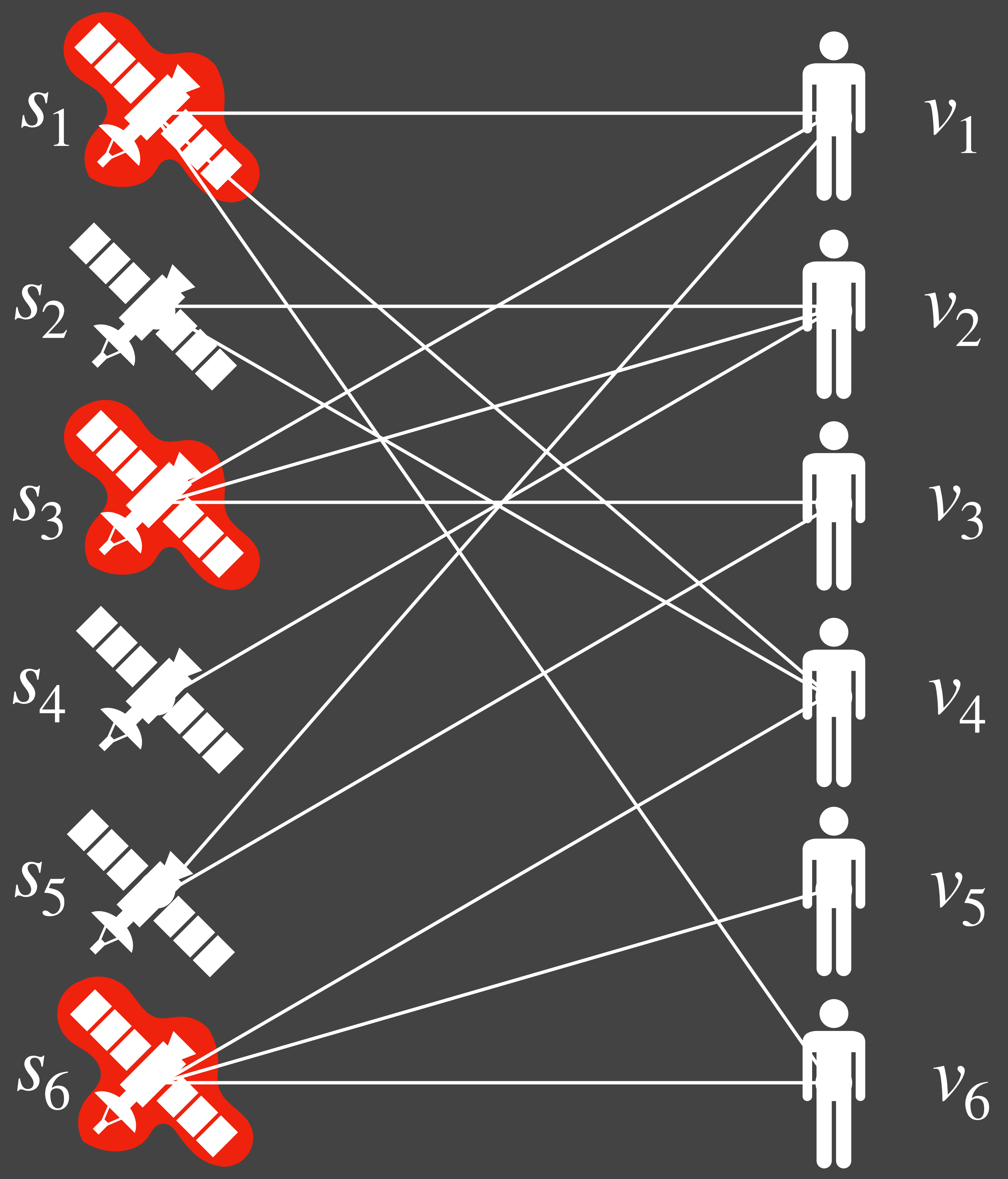
Thatchaphol Saranurak
(U. of Michigan)

Introduction

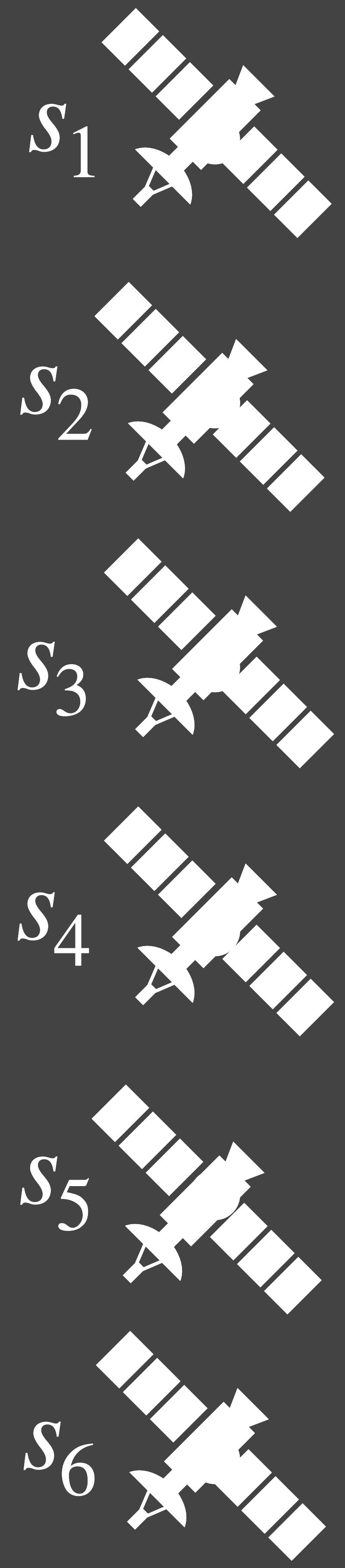
Motivating Problem



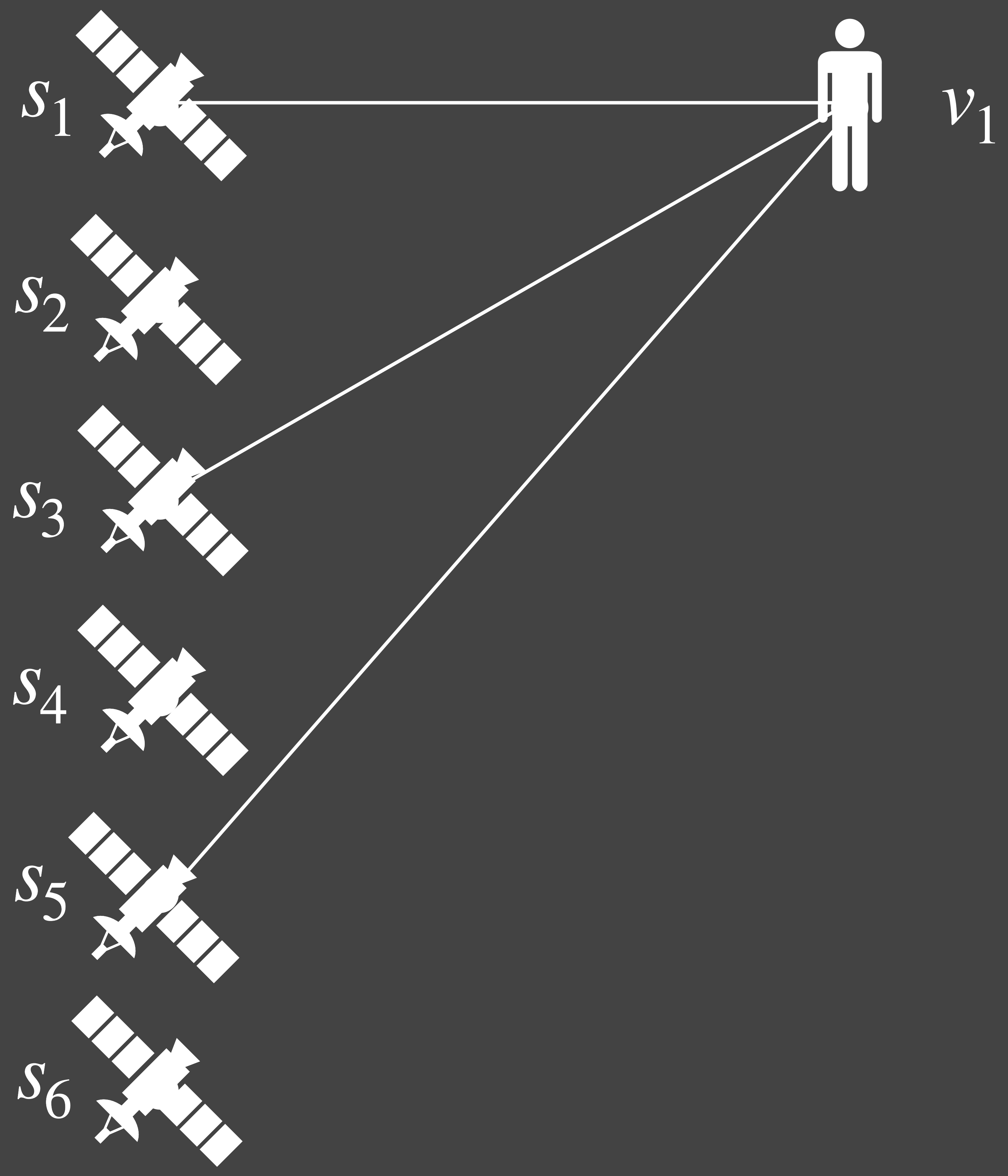
Motivating Problem



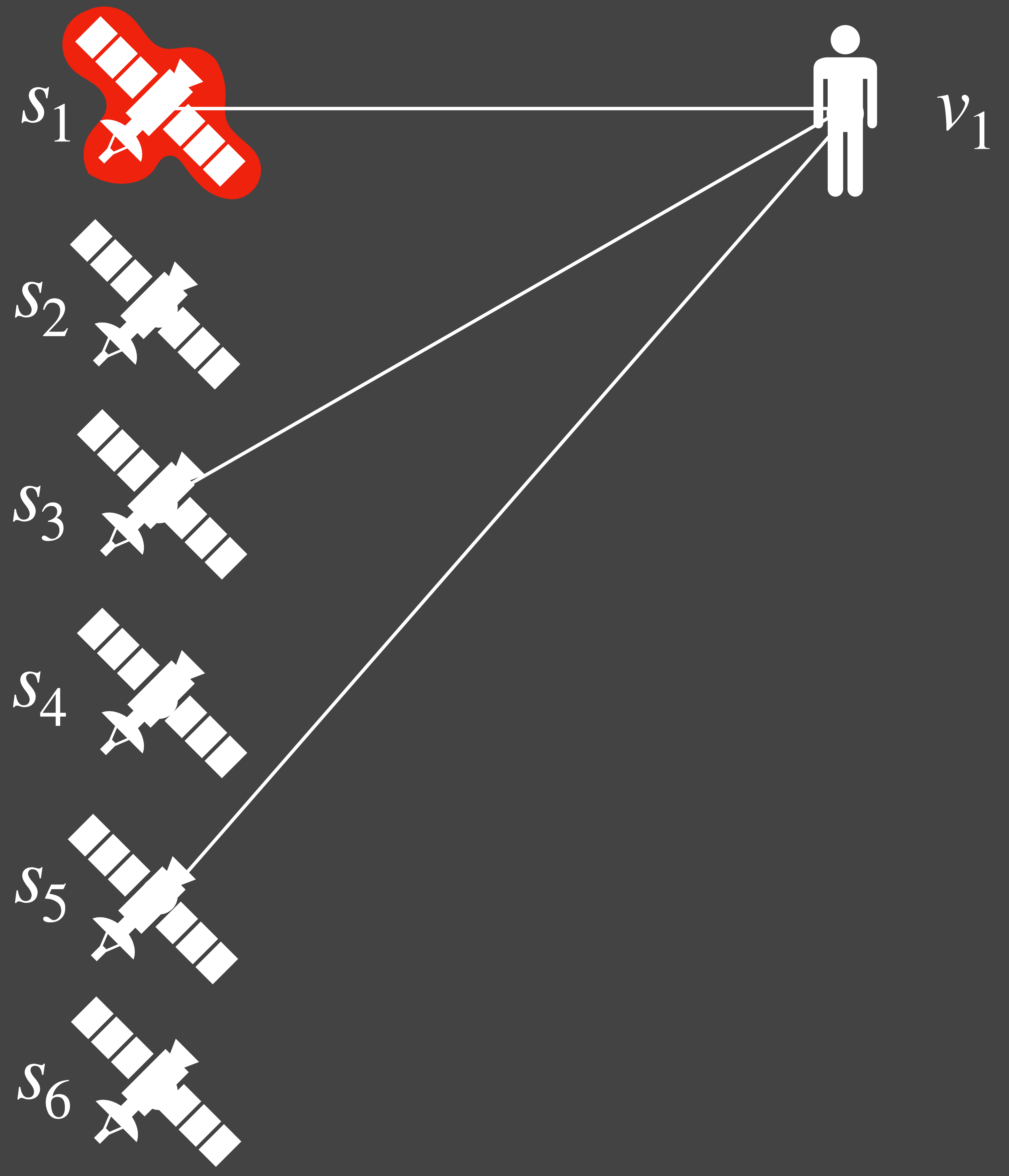
Motivating Problem



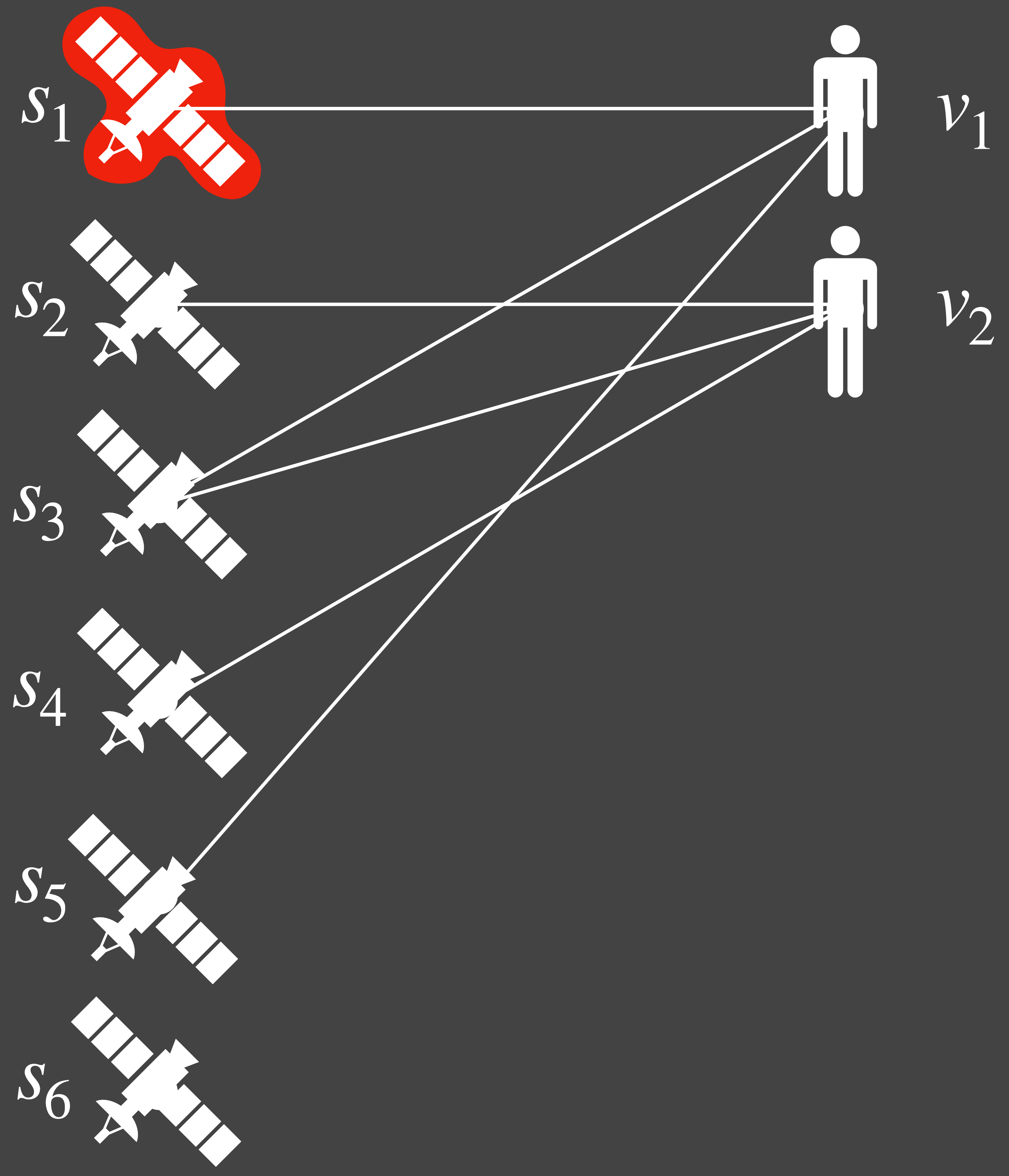
Motivating Problem



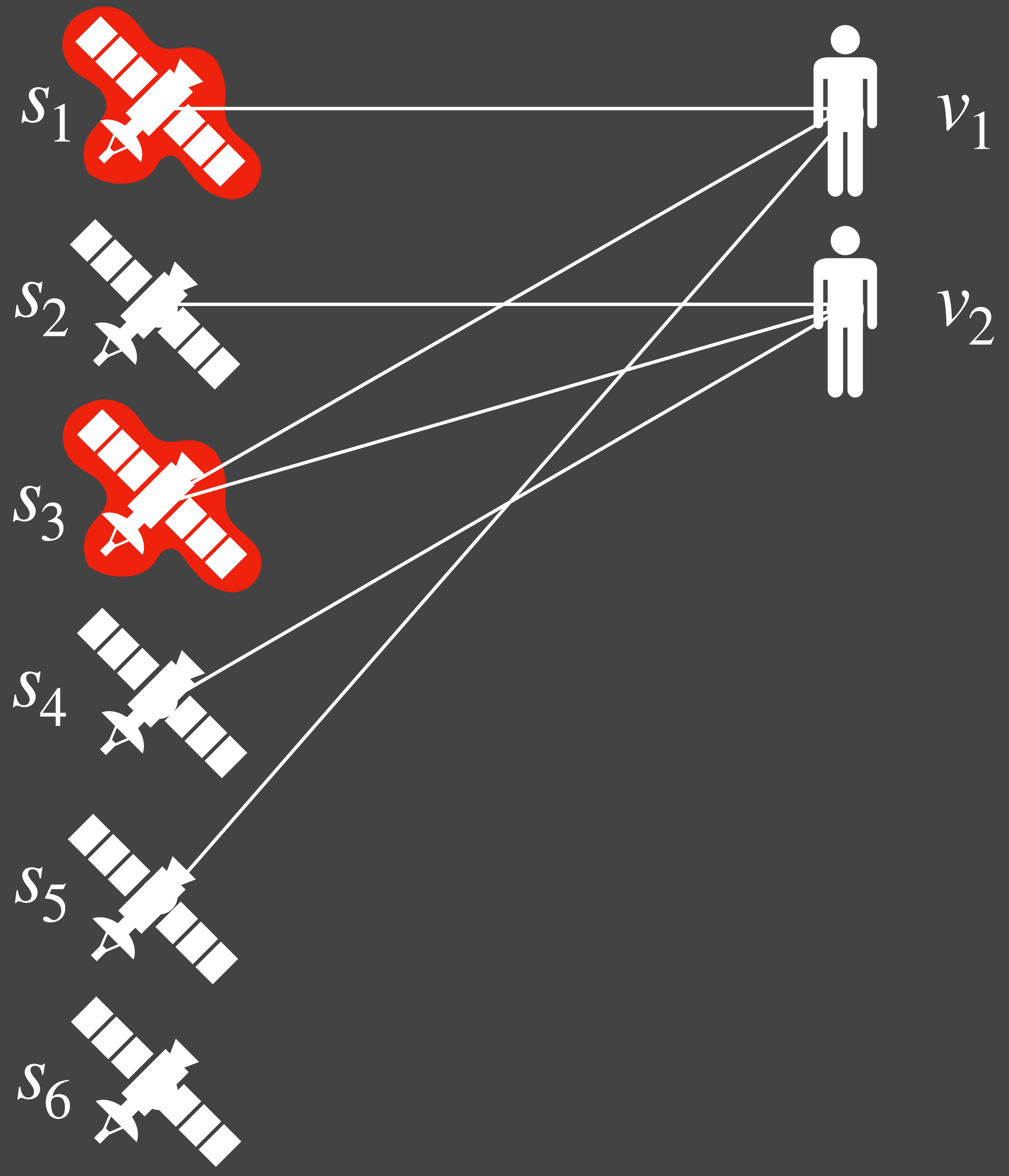
Motivating Problem



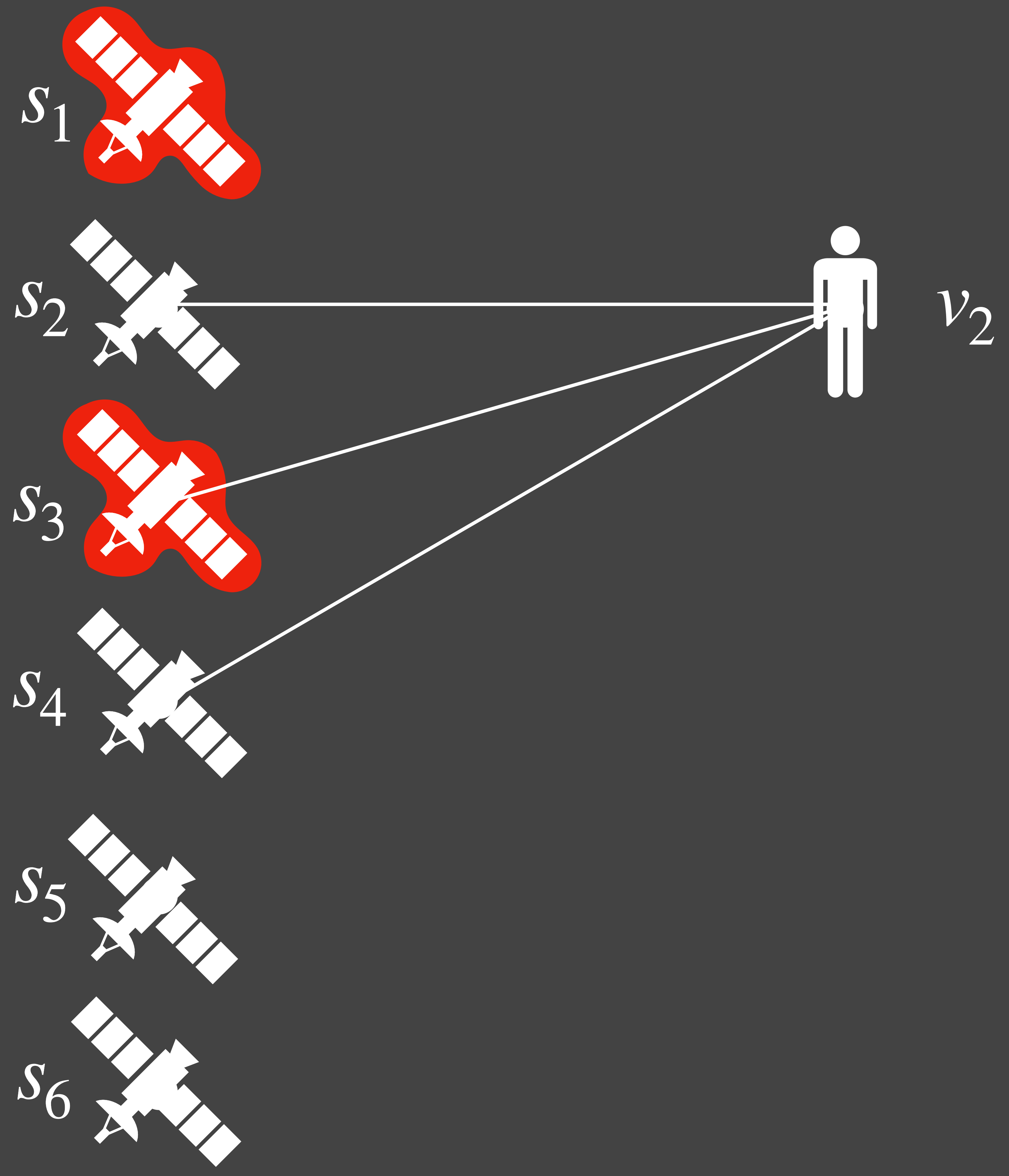
Motivating Problem



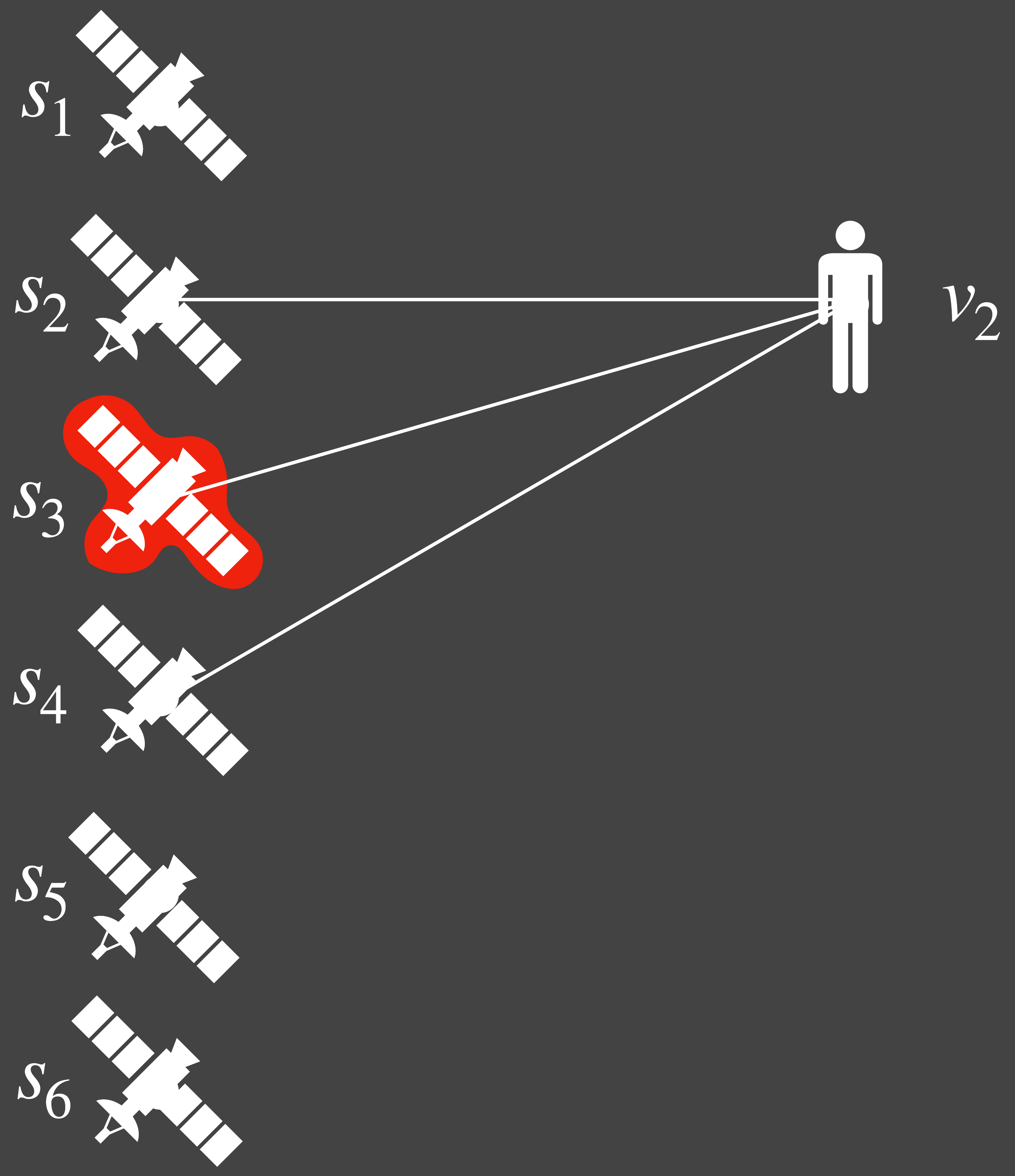
Motivating Problem



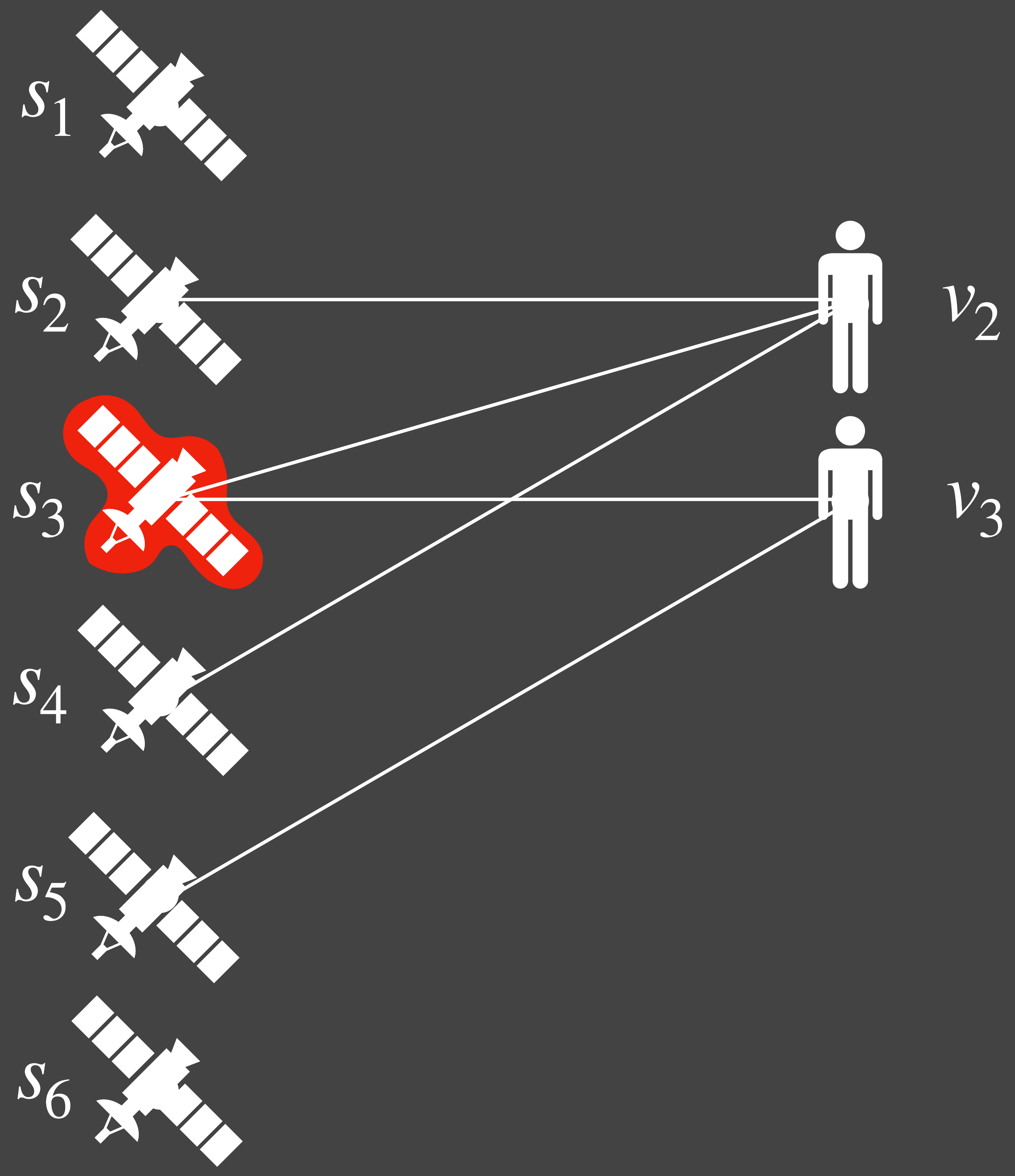
Motivating Problem



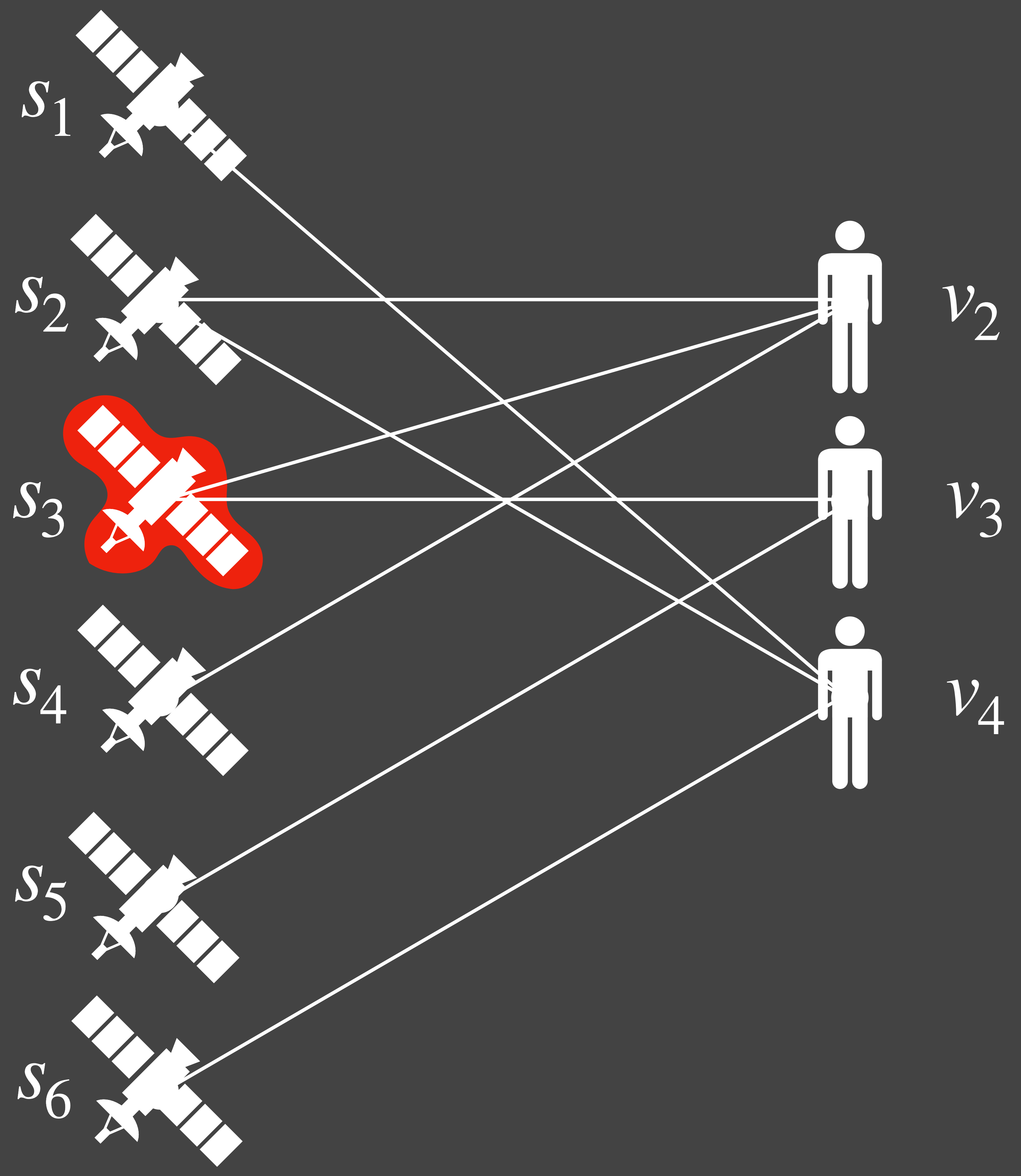
Motivating Problem



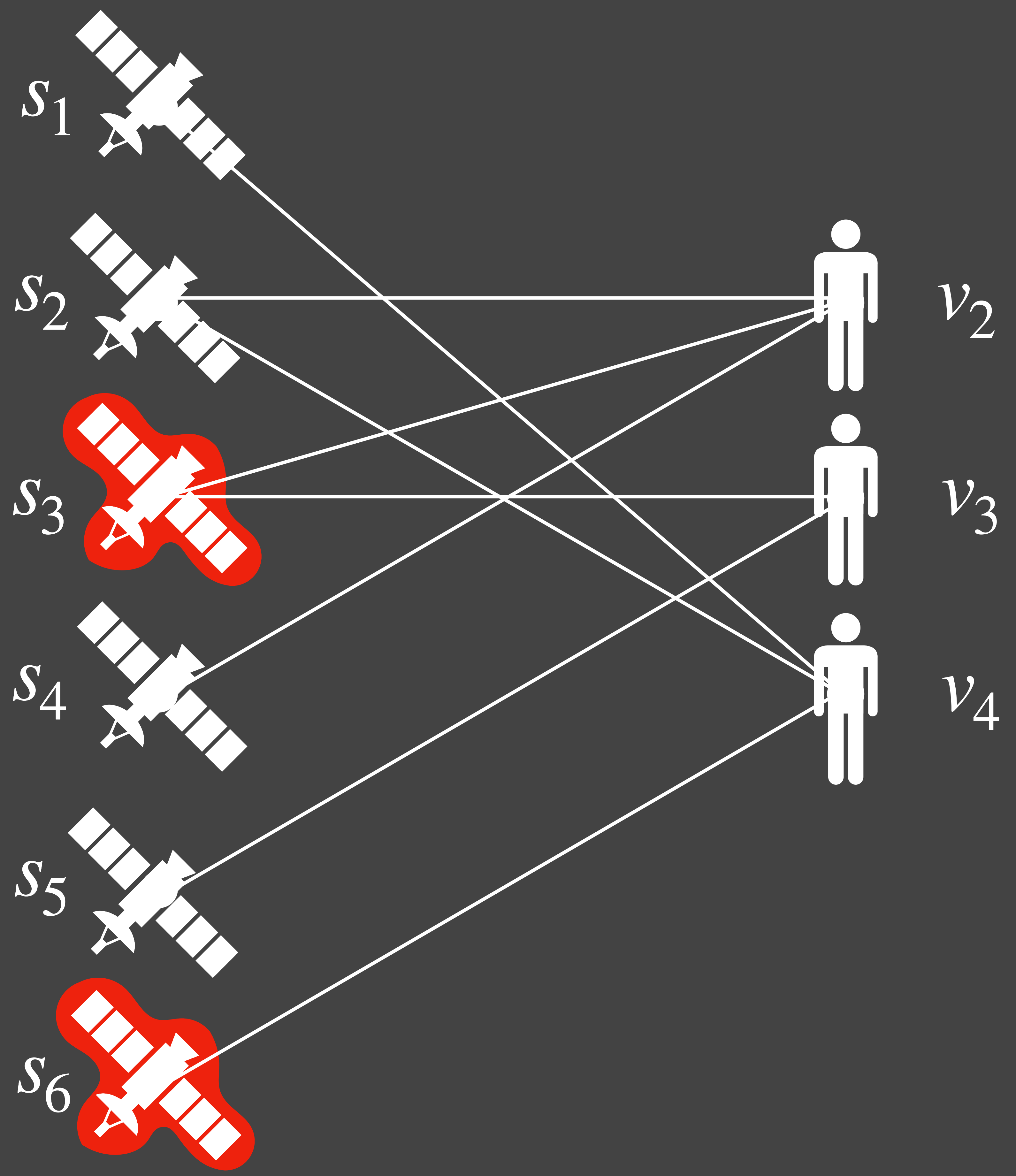
Motivating Problem



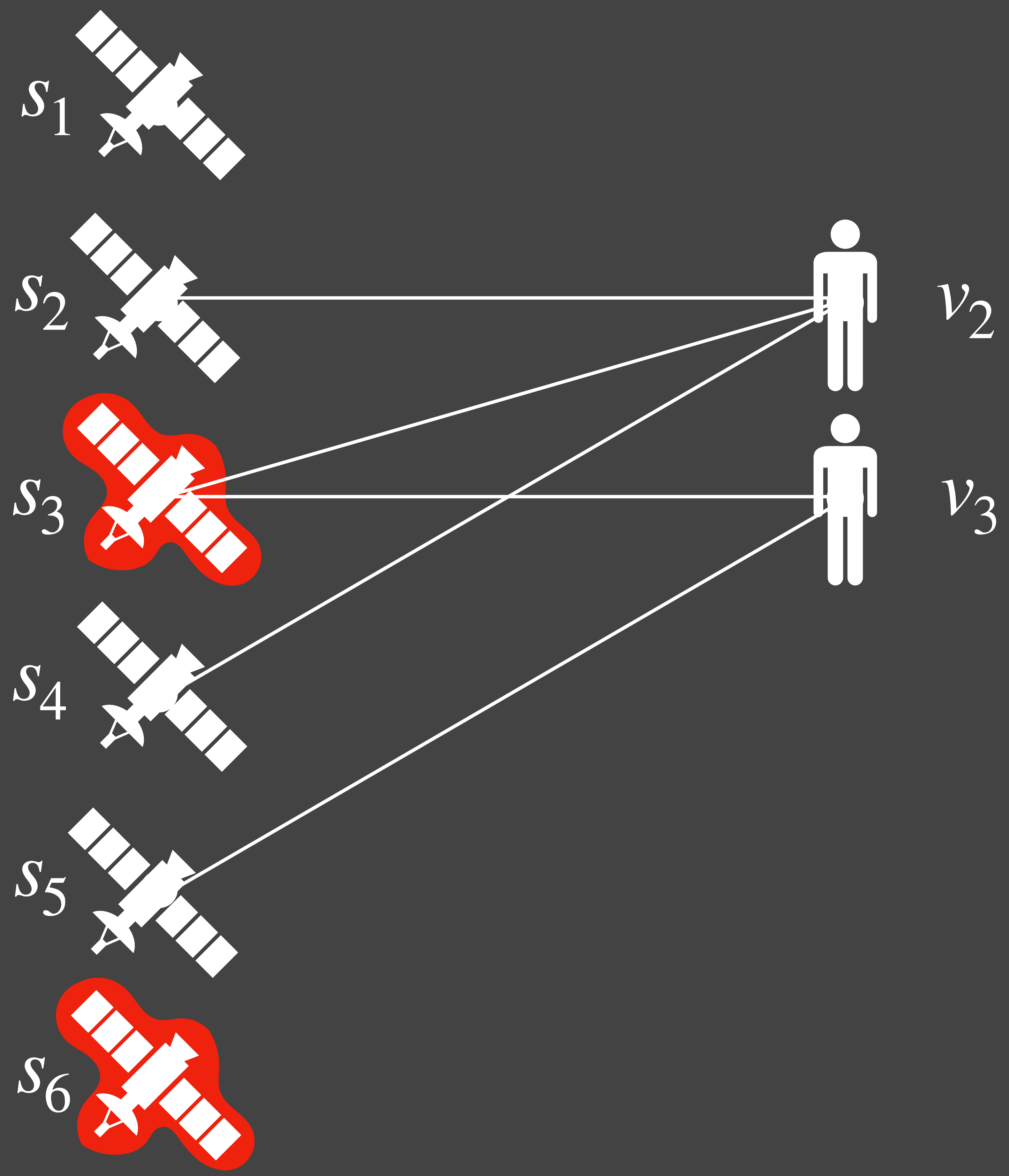
Motivating Problem



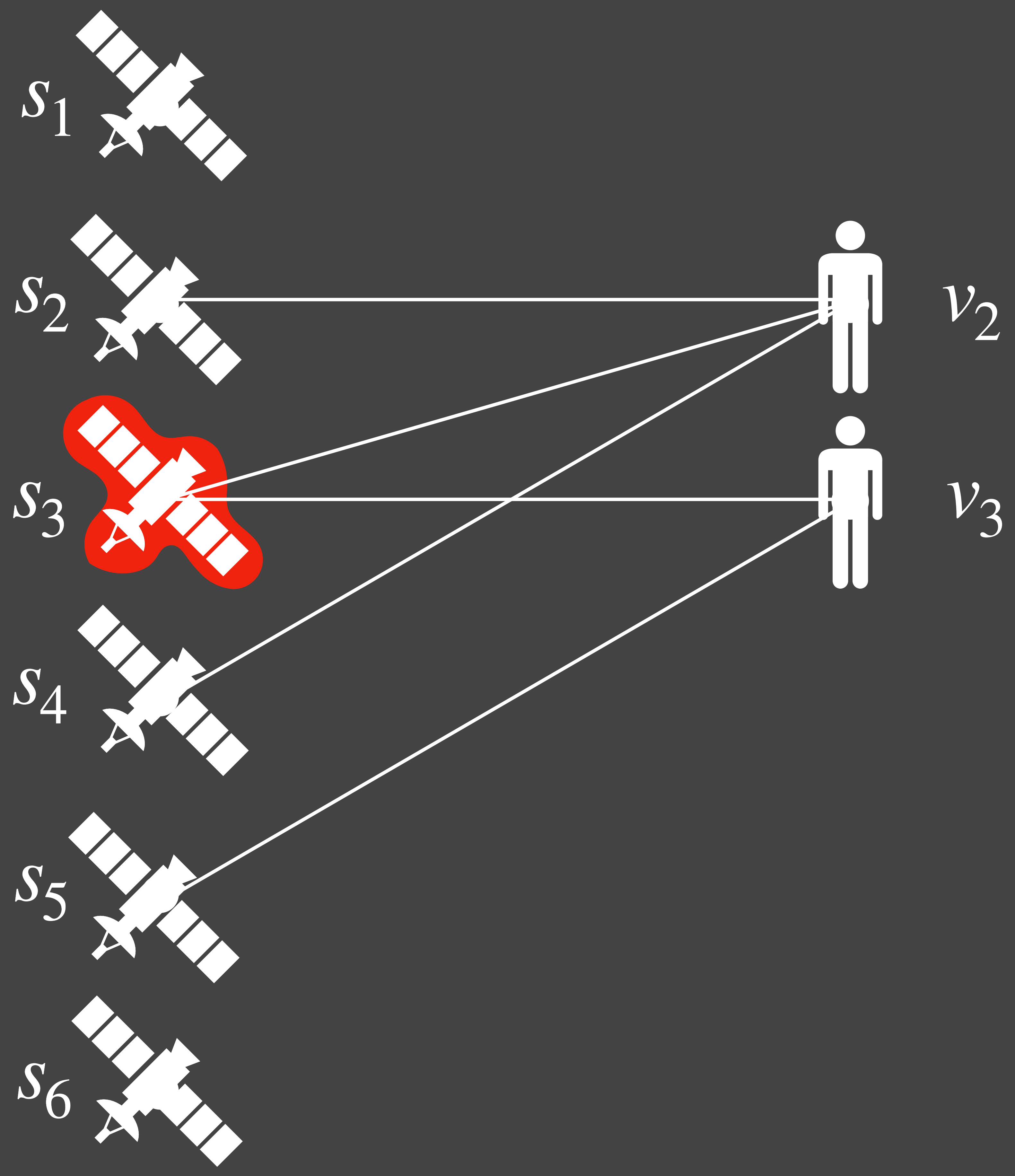
Motivating Problem



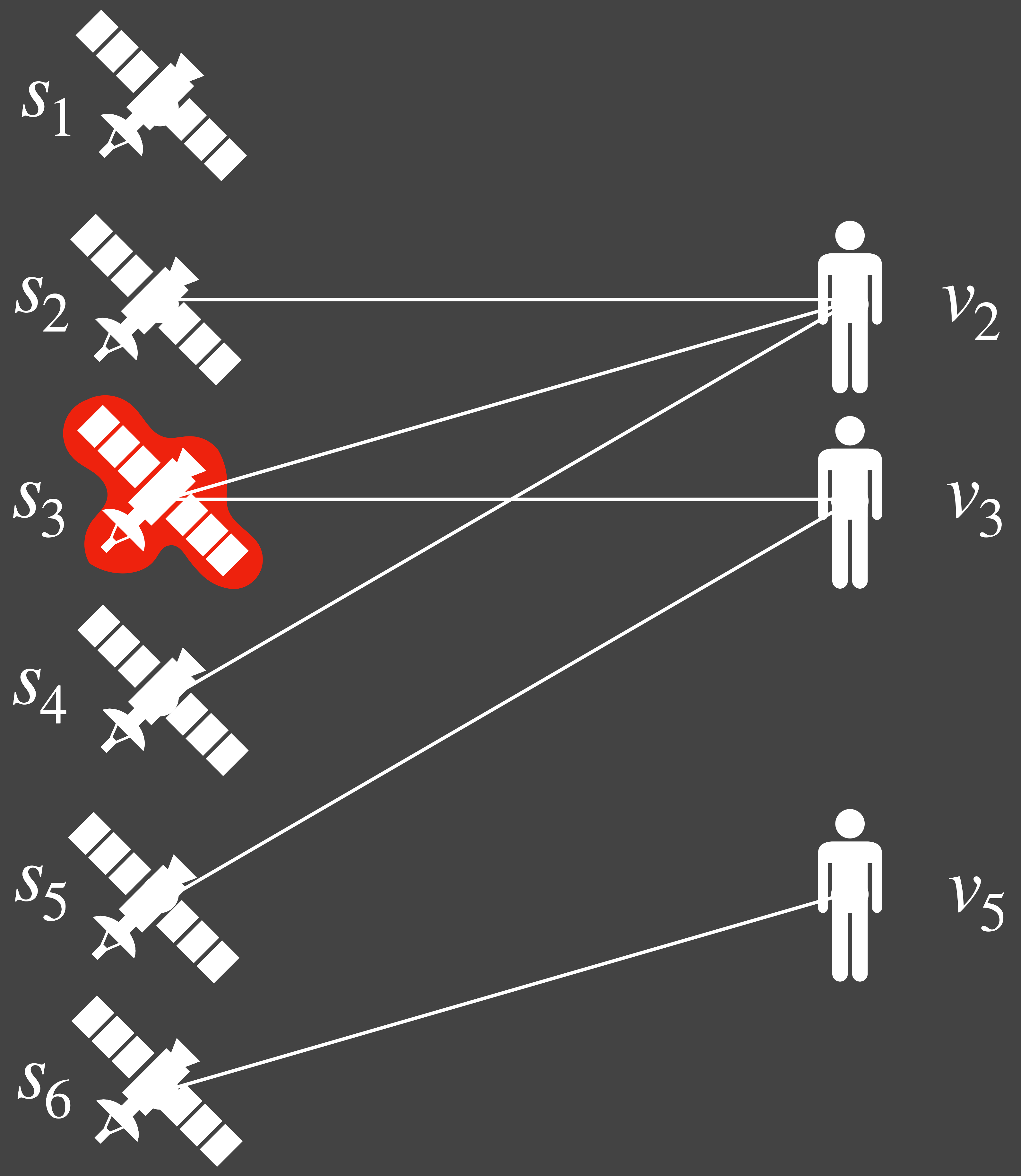
Motivating Problem



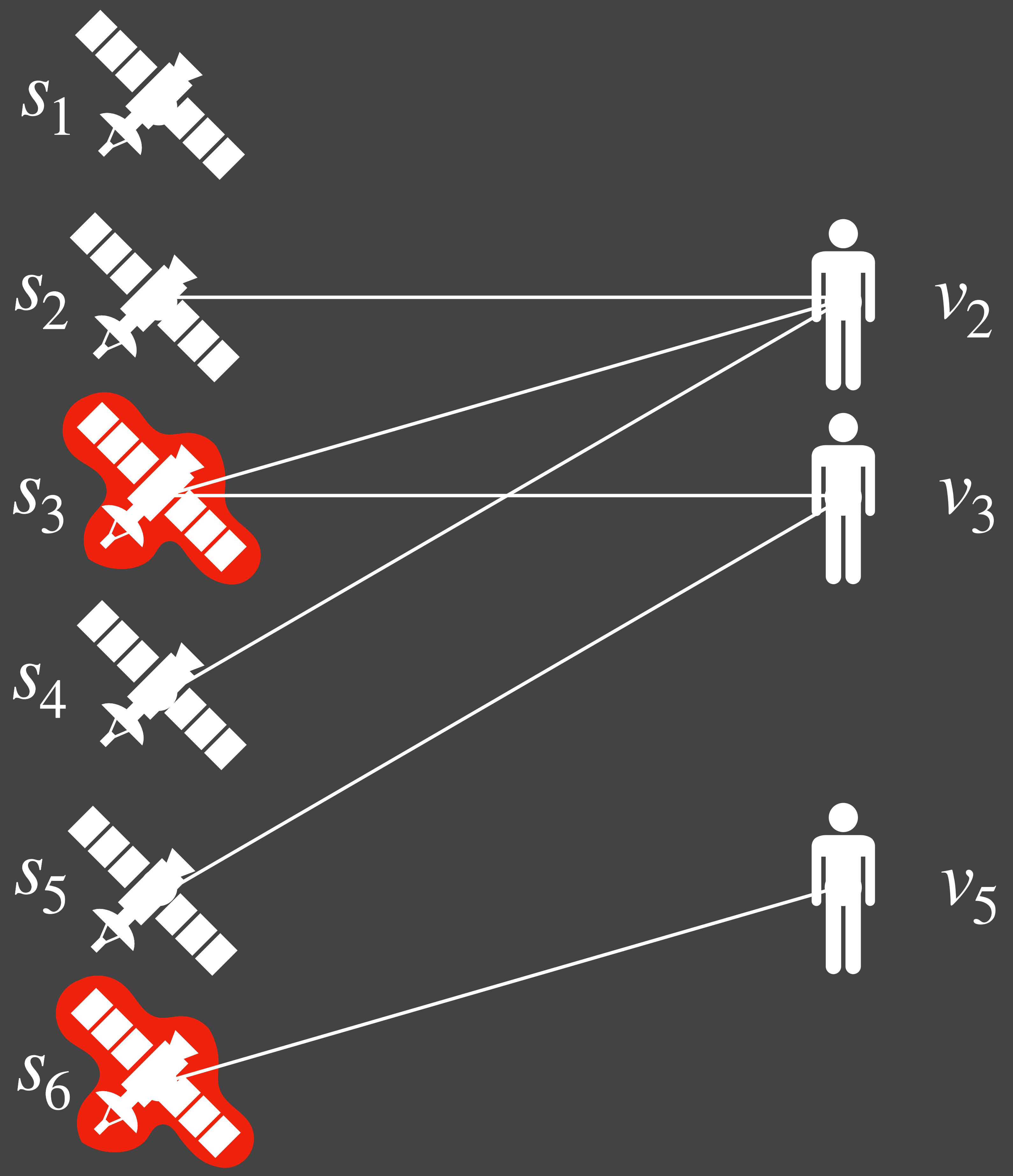
Motivating Problem



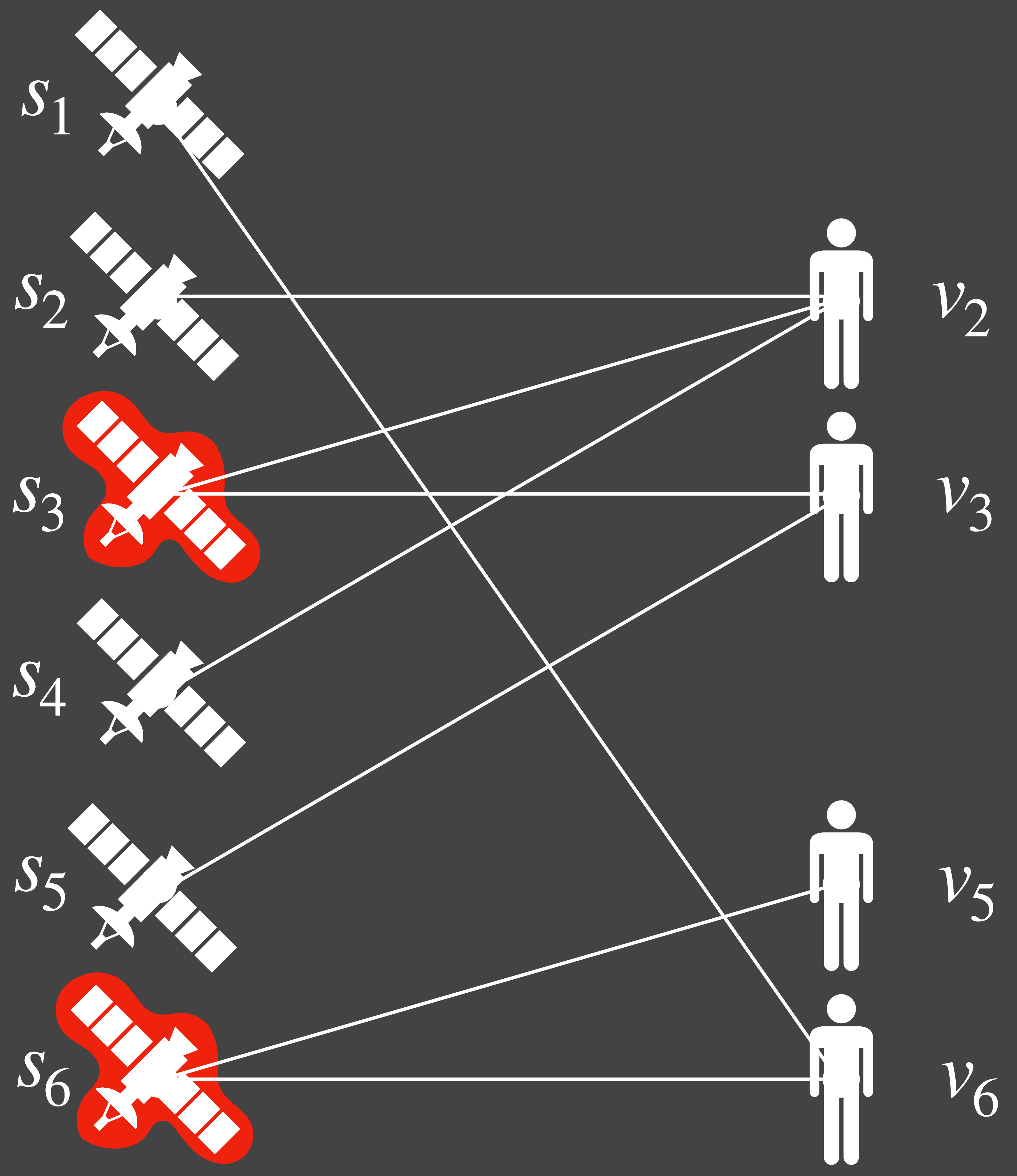
Motivating Problem



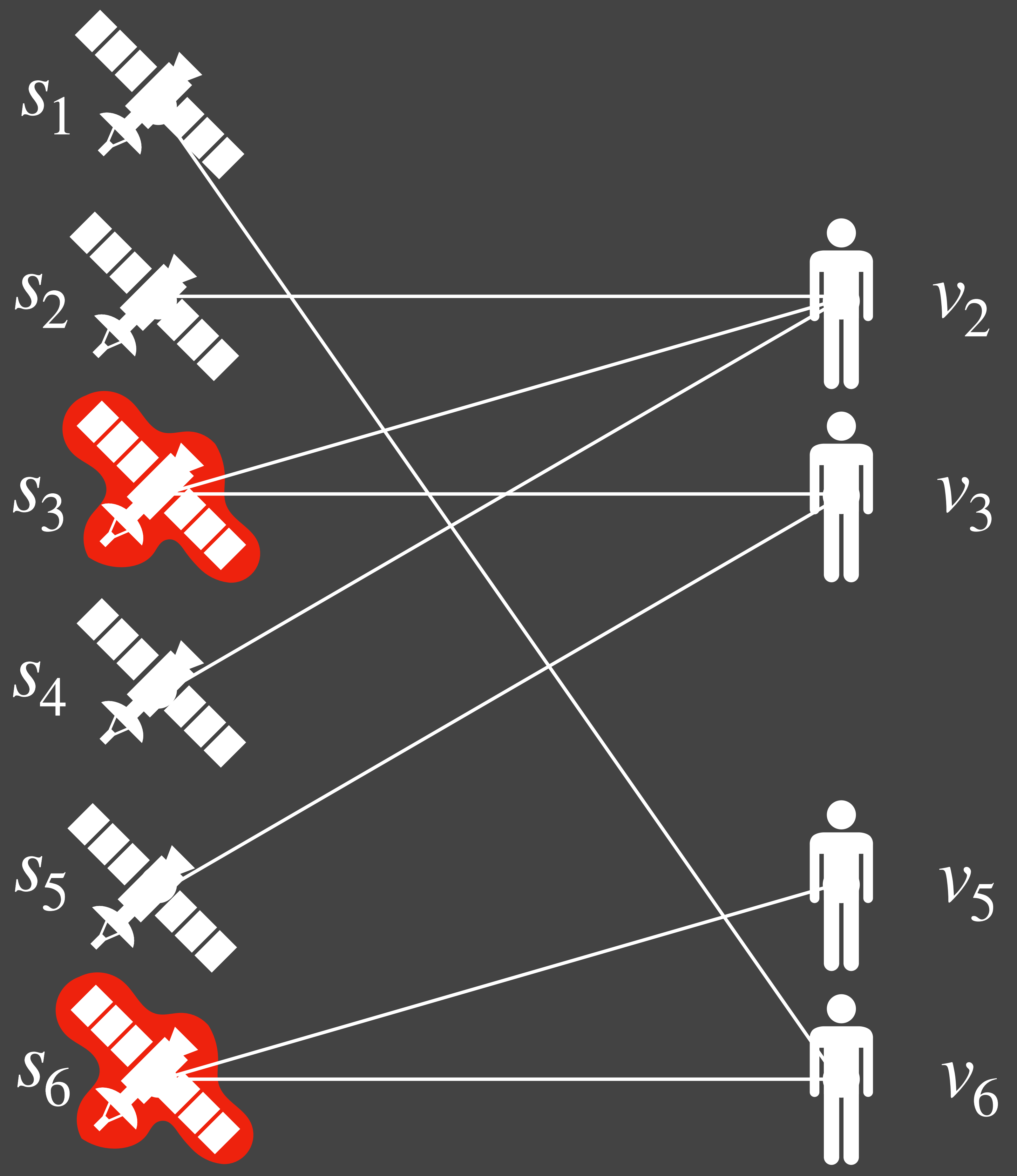
Motivating Problem



Motivating Problem

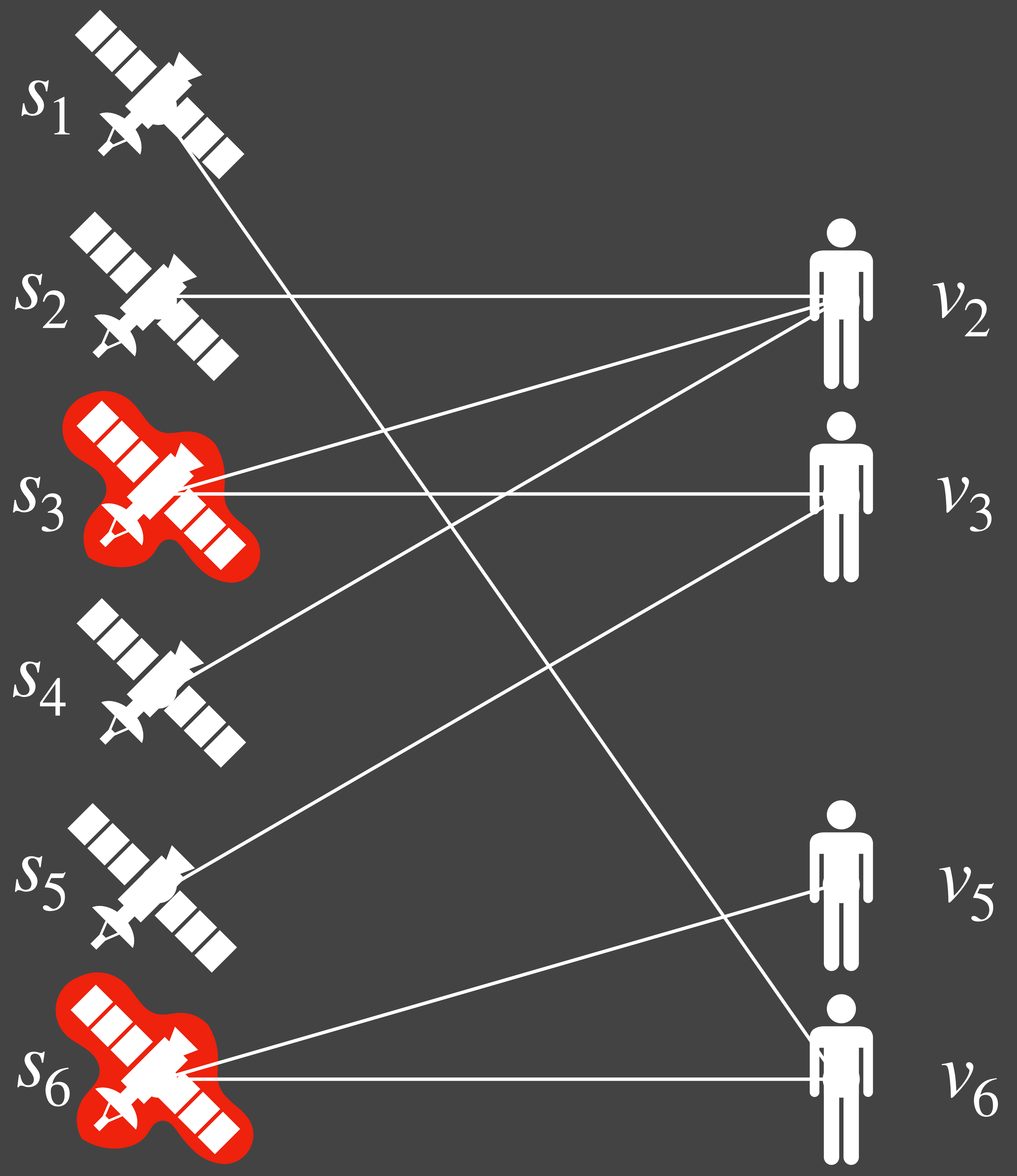


Motivating Problem



People come and go.

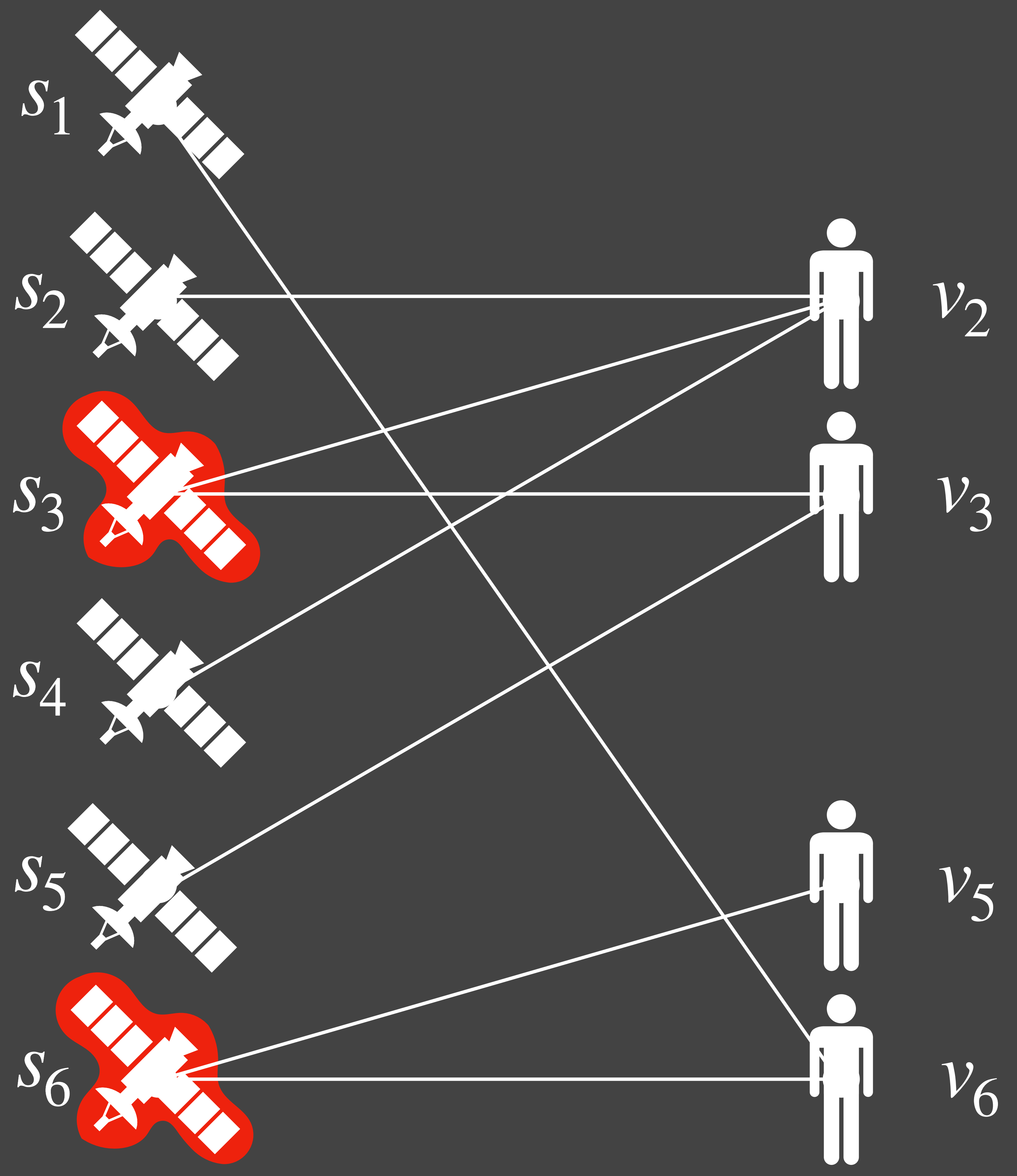
Motivating Problem



People come and go.

Want approximate minimum solution at every time step.

Motivating Problem

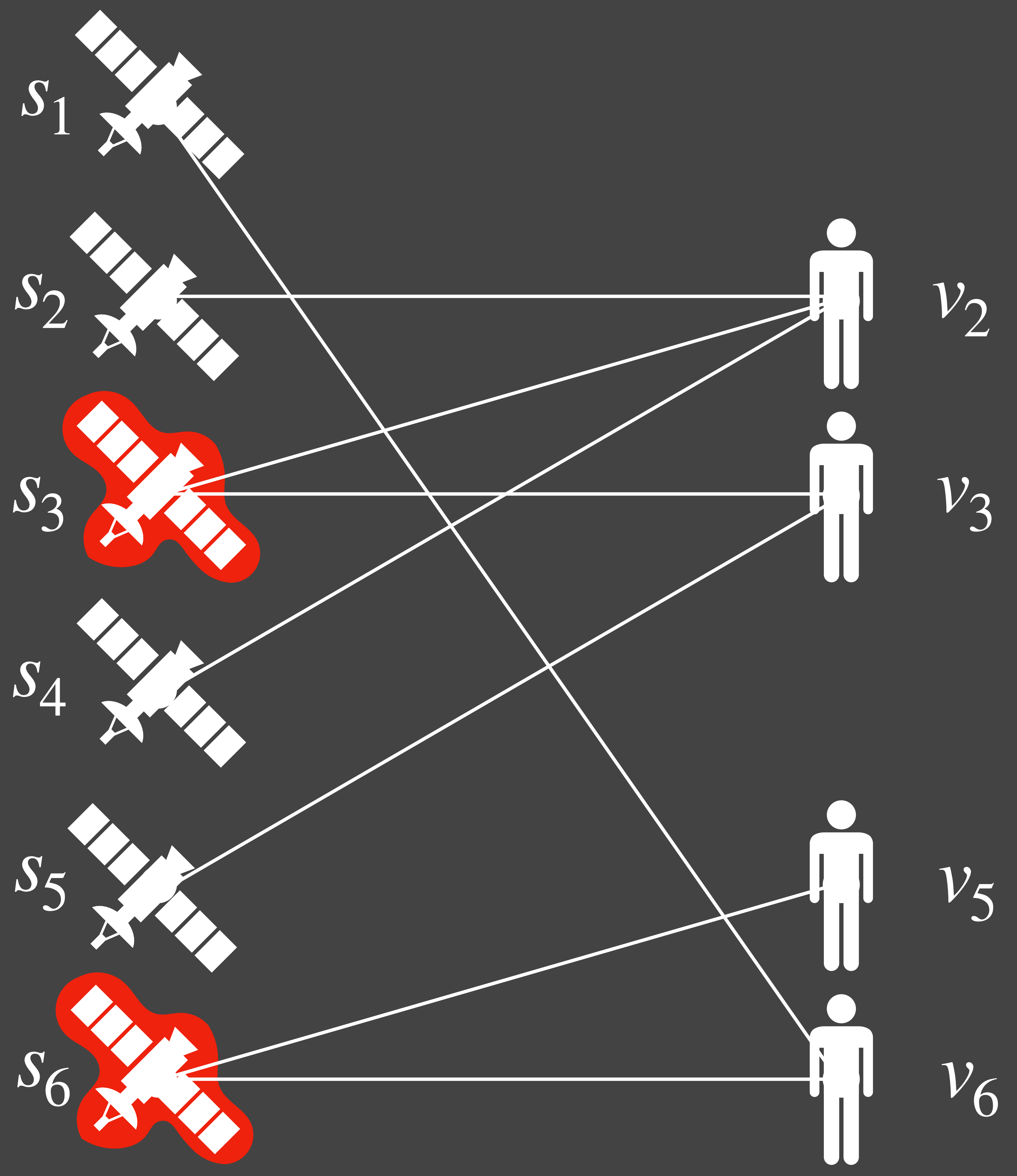


People come and go.

Want approximate minimum solution at every time step.

ALSO want minimum # edits, a.k.a. recourse.

Motivating Problem



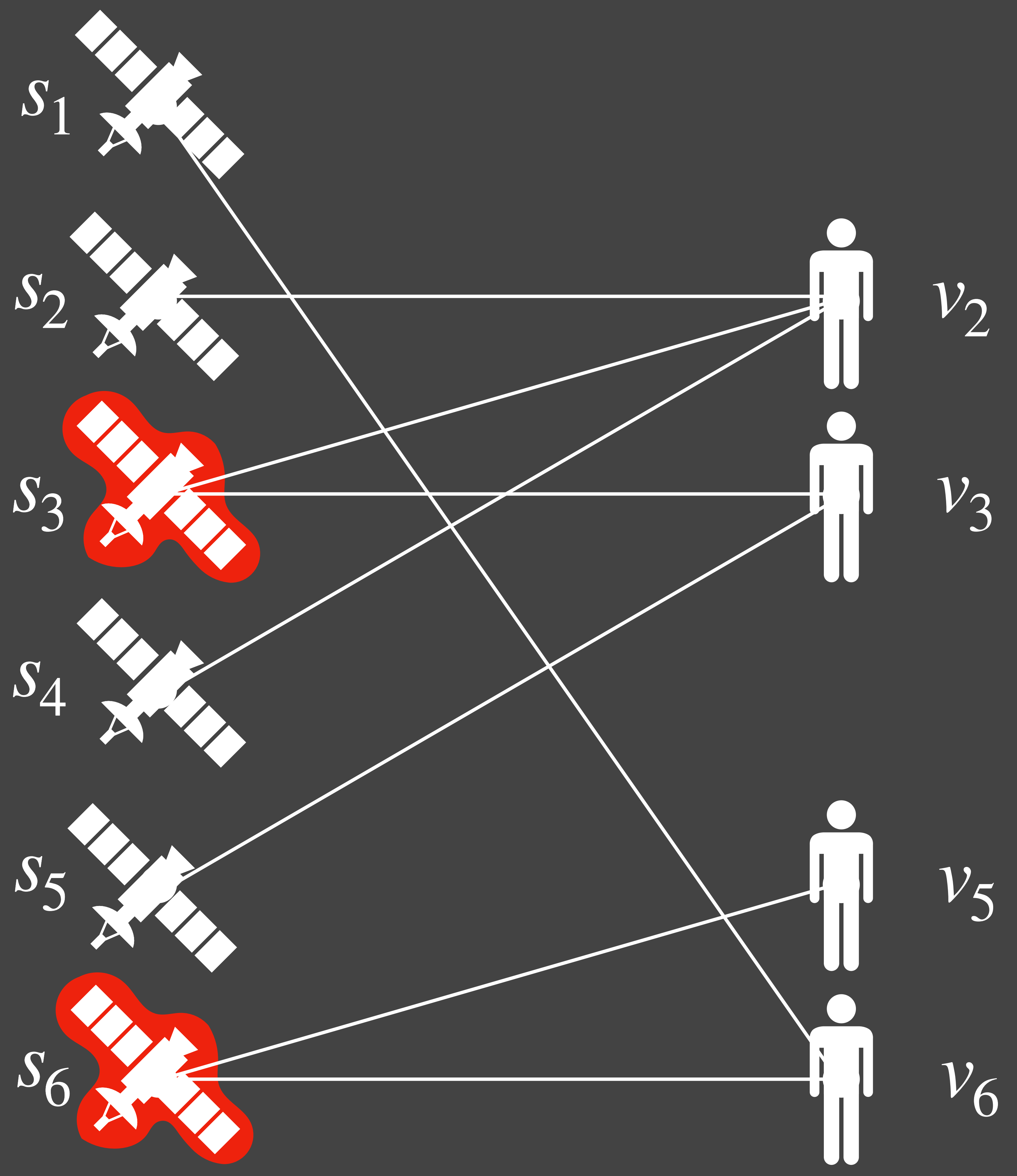
People come and go.

Want approximate minimum solution at every time step.

ALSO want minimum # edits, a.k.a. recourse.

a.k.a. Dynamic Set Cover!

Motivating Problem



People come and go.

Want approximate minimum solution at every time step.

ALSO want minimum # edits, a.k.a. recourse.

a.k.a. Dynamic Set Cover!

Q: What is recourse/ approximation tradeoff?

Low Recourse Dynamic Algorithms

Low Recourse Dynamic Algorithms

Want simultaneously:

Low Recourse Dynamic Algorithms

Want simultaneously:

1. Maintain competitive solution as input changes.

Low Recourse Dynamic Algorithms

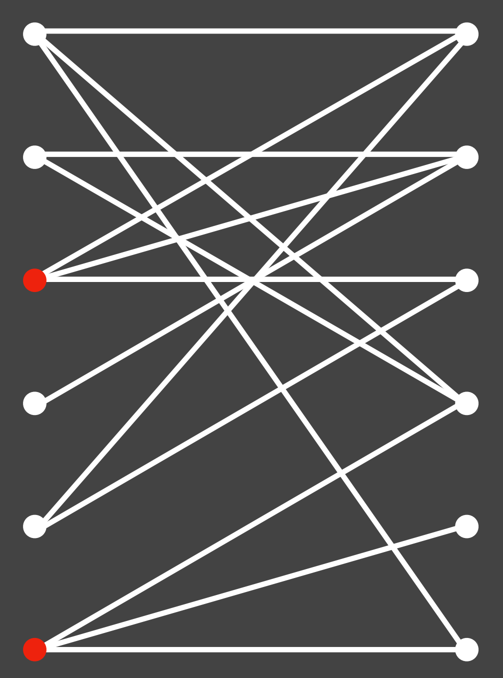
Want simultaneously:

1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).

Low Recourse Dynamic Algorithms

Want simultaneously:

1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).

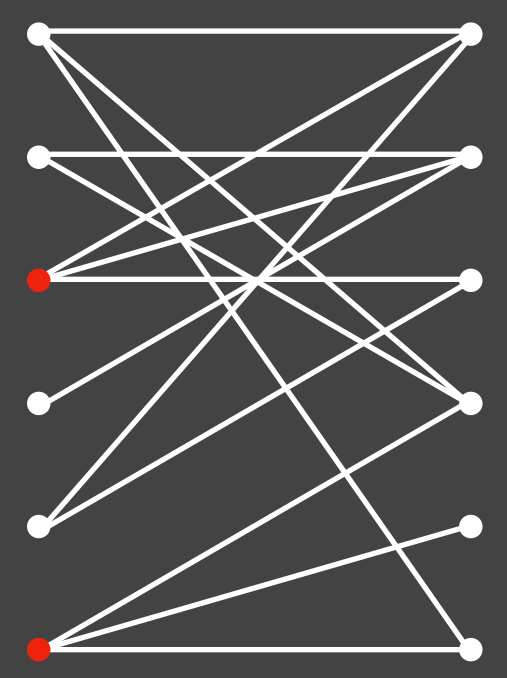


Set Cover

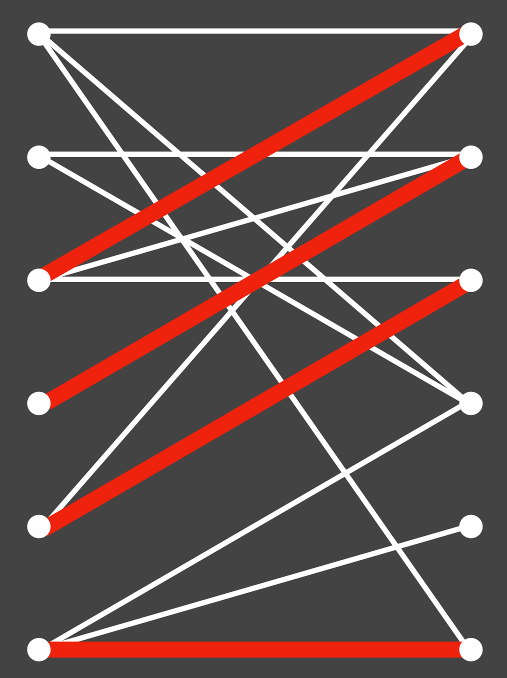
Low Recourse Dynamic Algorithms

Want simultaneously:

1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).



Set Cover

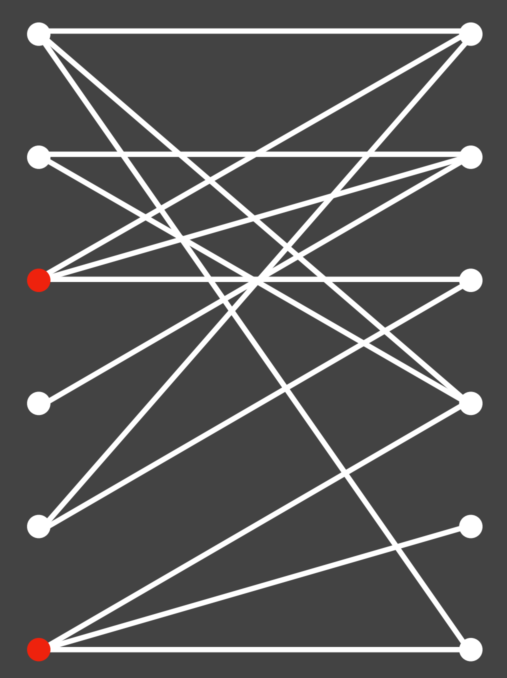


Matching

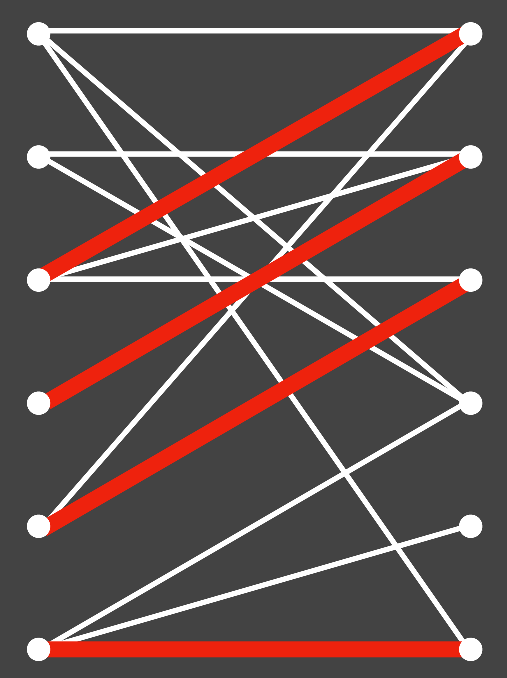
Low Recourse Dynamic Algorithms

Want simultaneously:

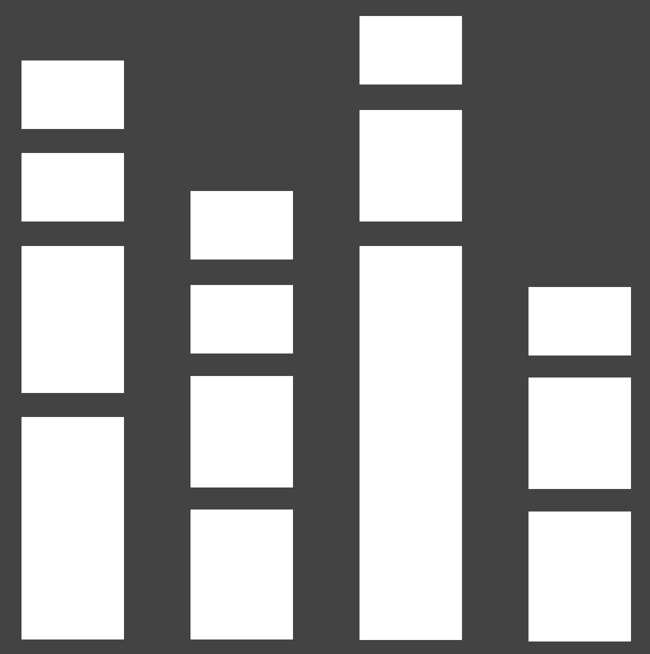
1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).



Set Cover



Matching

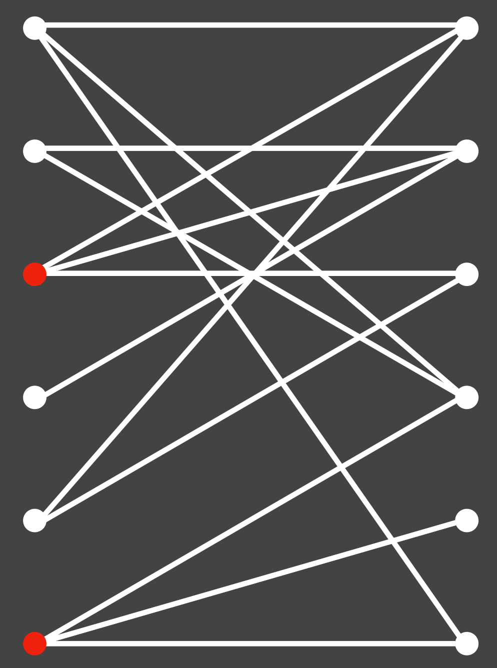


Load Balancing

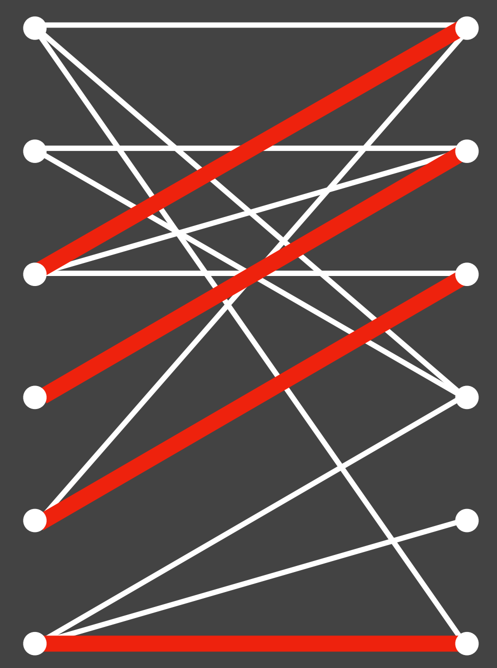
Low Recourse Dynamic Algorithms

Want simultaneously:

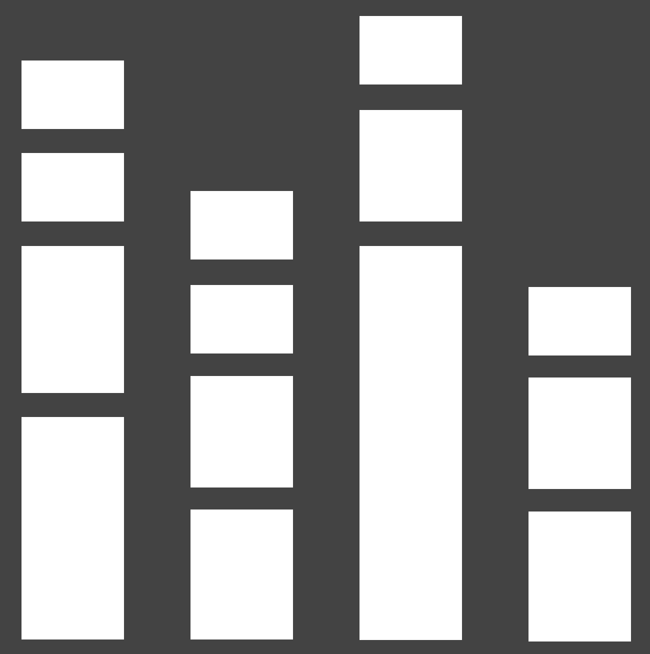
1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).



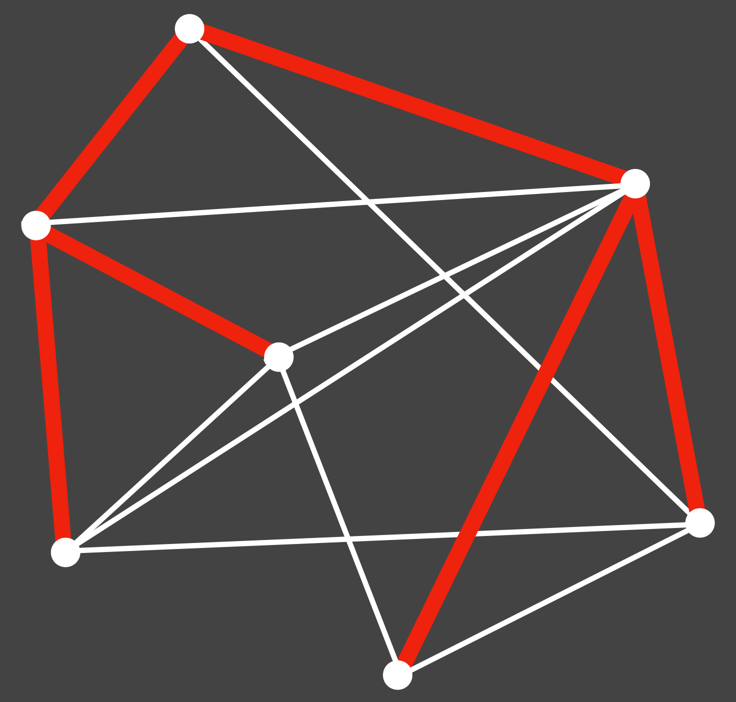
Set Cover



Matching



Load Balancing

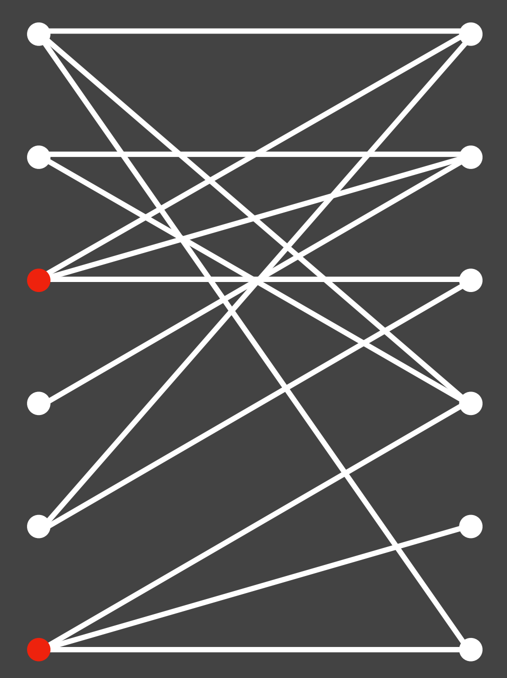


Minimum Spanning Tree

Low Recourse Dynamic Algorithms

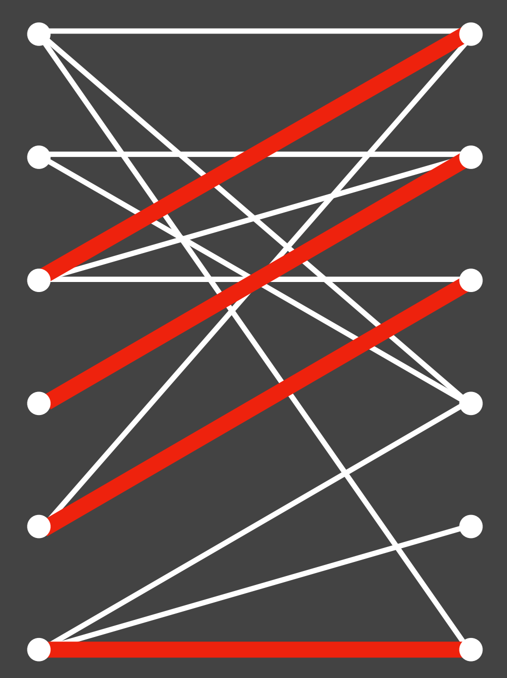
Want simultaneously:

1. Maintain competitive solution as input changes.
2. Minimize # edits over time (a.k.a. recourse).



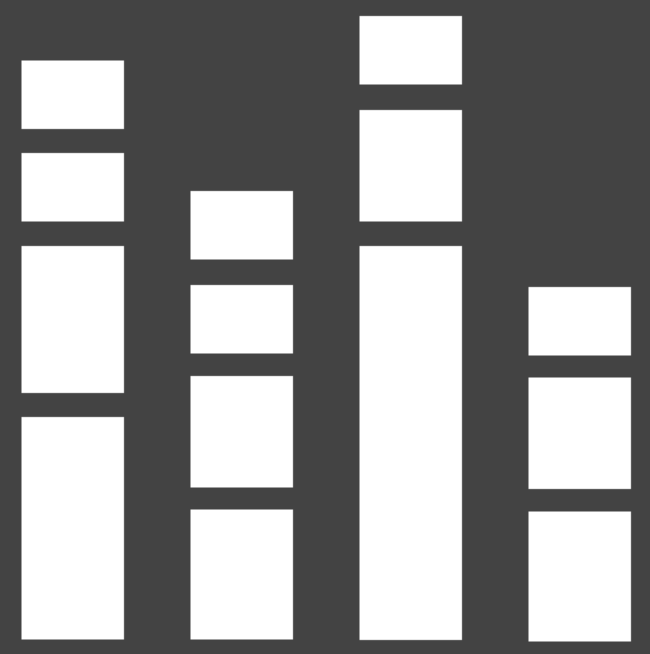
Set Cover

[Gupta Krishnaswamy Kumar Panigrahi 17] [Abboud+ 17] [Bhattacharya Henzinger Nanongkai 19] [Gupta **L.** 20] [Bhattacharya Henzinger Nanongkai Wu 21] [Assadi Solomon21]



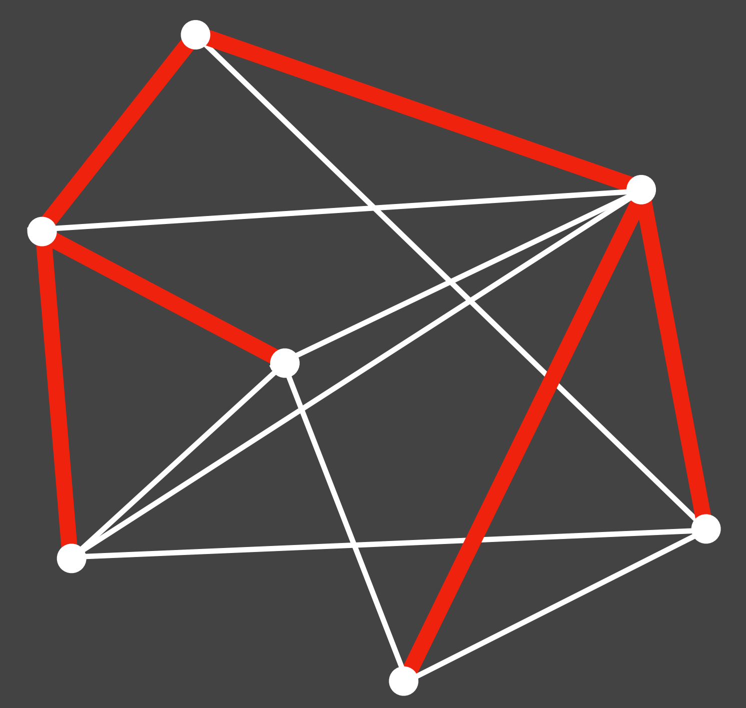
Matching

[Folklore] [Grove Kao Krishnan Vitter 95] [Chadhuri Daskalakis Kleinberg Lin 09] [Bosek Leniowski Sankowski Zych] [Bernstein Holm Rotenberg 18]



Load Balancing

[Awerbuch Azar Plotkin Warts 01] [Gupta Kumar Stein 14] [Krishnaswamy Li Suriyanarayana 23]



Minimum Spanning Tree

[Imase Waxman 91] [Gu Gupta Kumar 16] [Gupta Kumar 14] [Łącki+ 15] [Gupta **L.** 20]

Is There a Theory to Build?

Is There a Theory to Build?

Most work (mine included!) based on 1-off combinatorial insights.

Is There a Theory to Build?

Most work (mine included!) based on 1-off combinatorial insights.

- Difficult to come up with. 😰

Is There a Theory to Build?

Most work (mine included!) based on 1-off combinatorial insights.

- Difficult to come up with. 😰
- Difficult to generalize. 😰

Is There a Theory to Build?

Most work (mine included!) based on 1-off combinatorial insights.

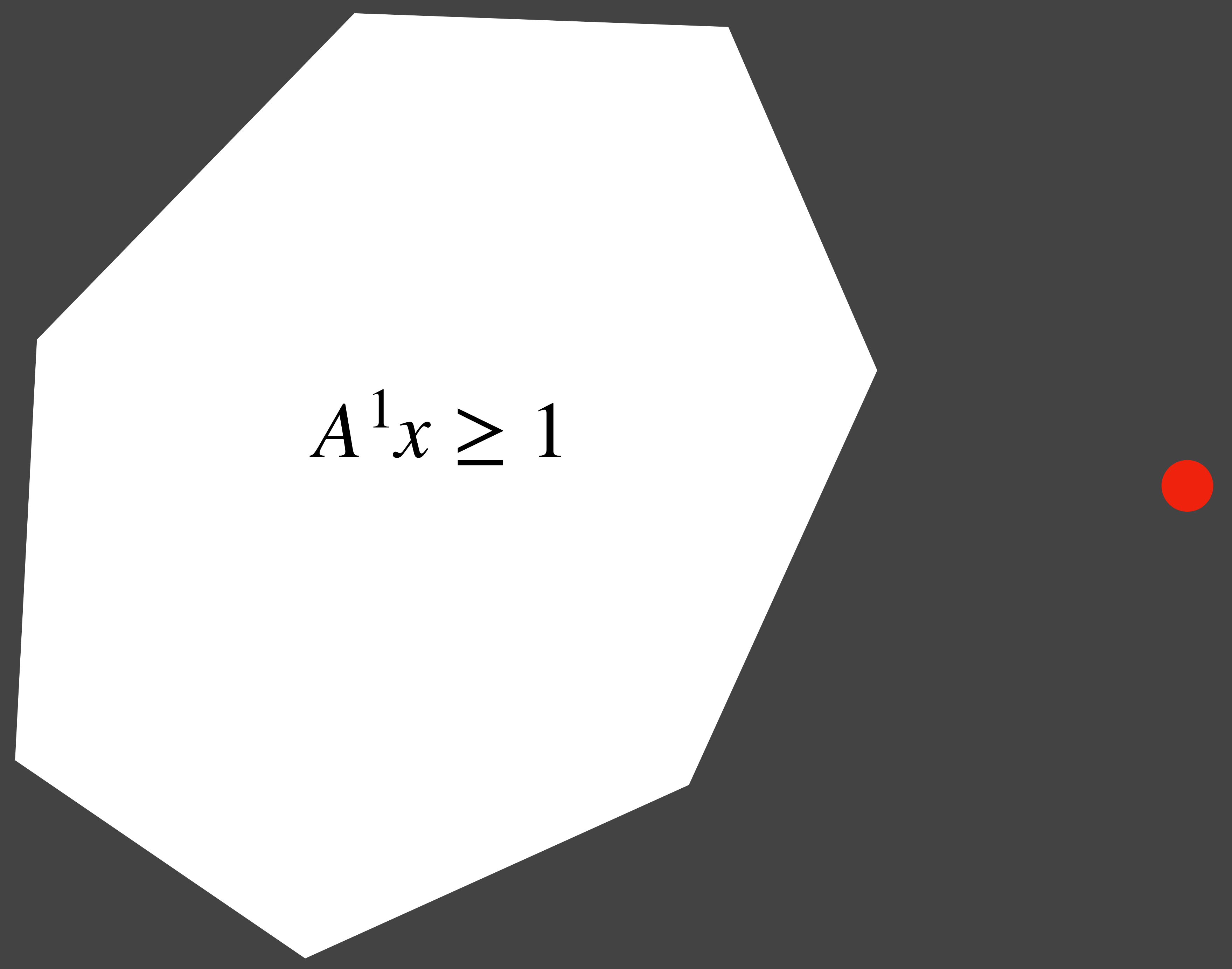
- Difficult to come up with. 😰
- Difficult to generalize. 😰

General recipe for designing low-recourse algorithms?

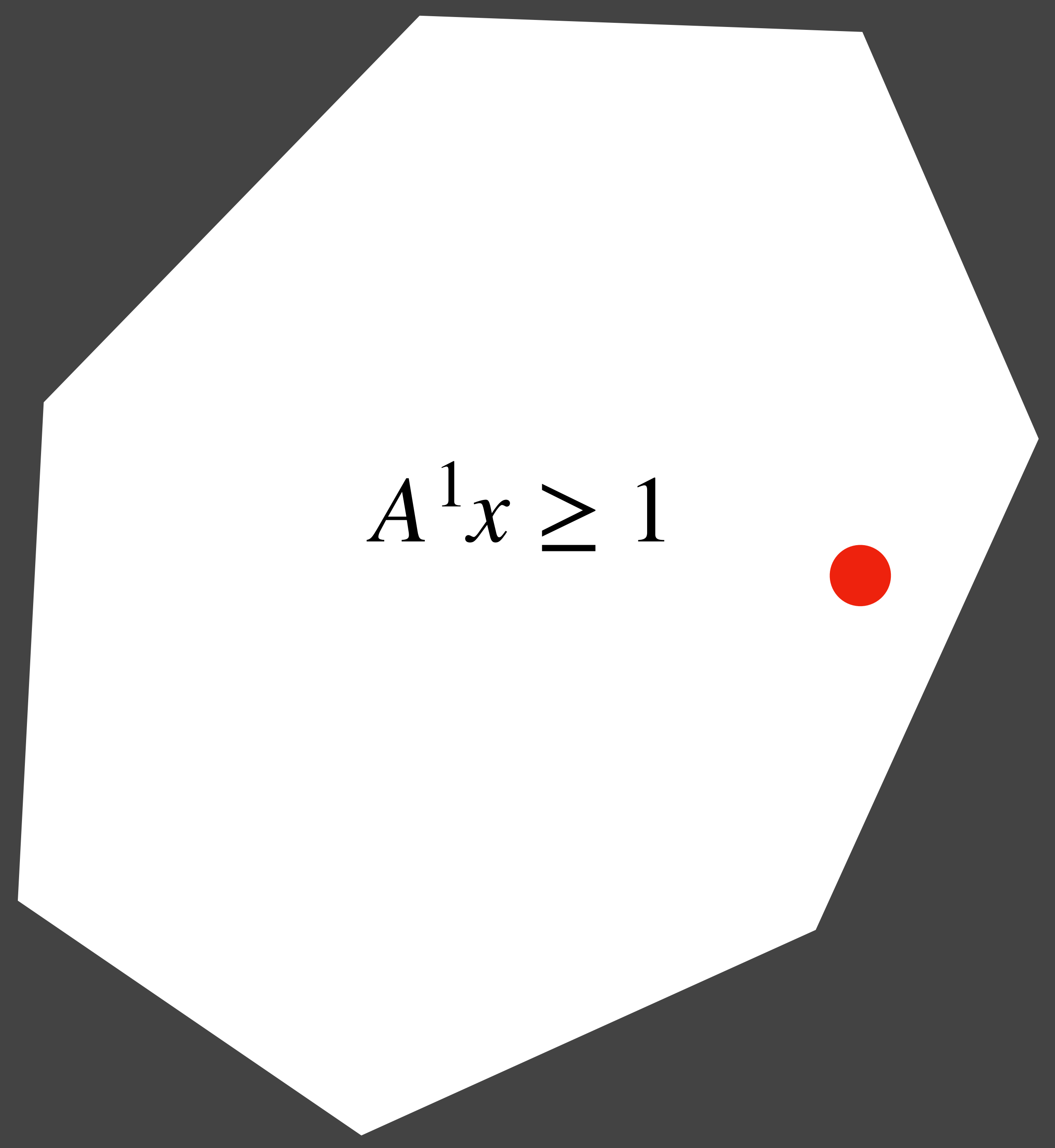
The Meta (Fractional) Problem

The Meta (Fractional) Problem

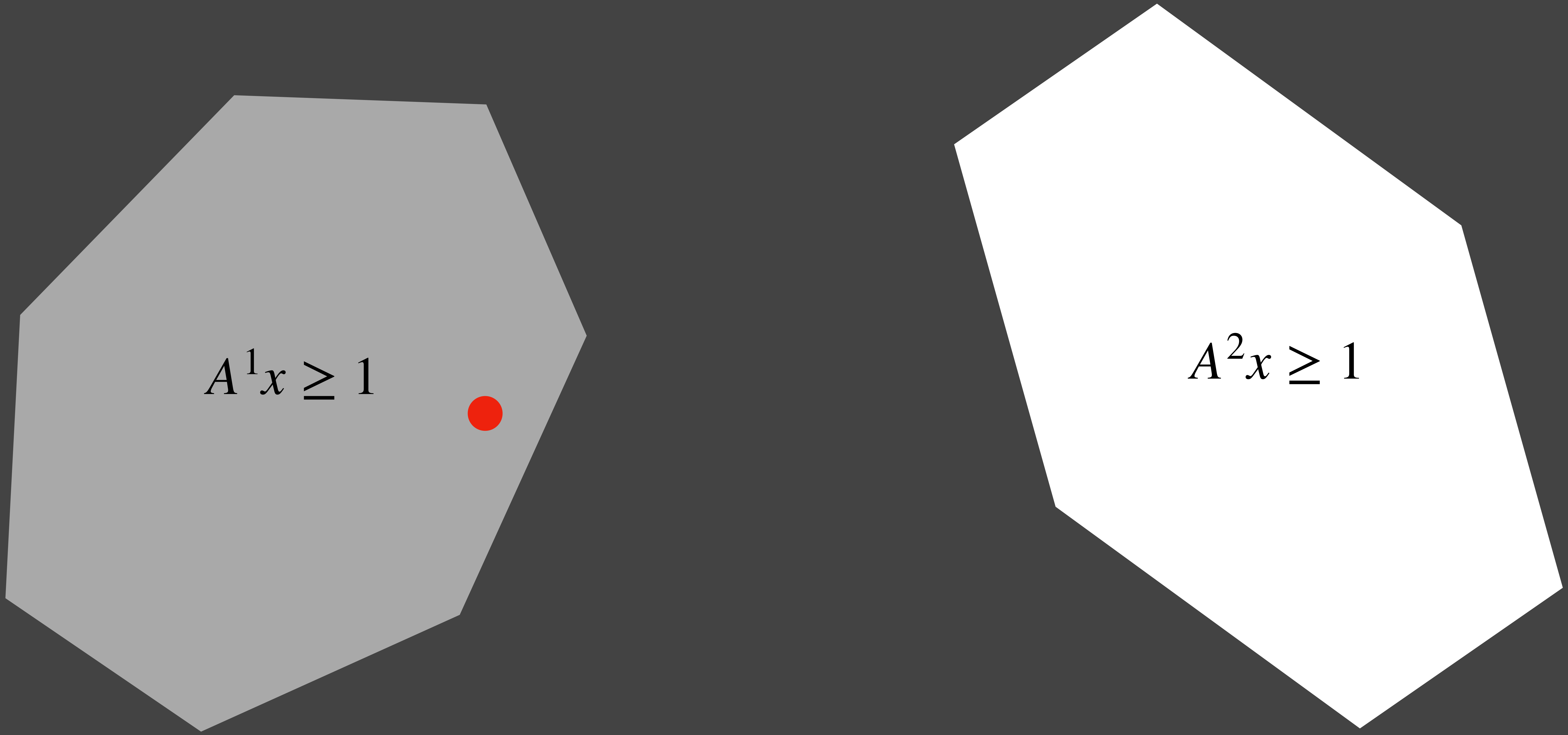
The Meta (Fractional) Problem



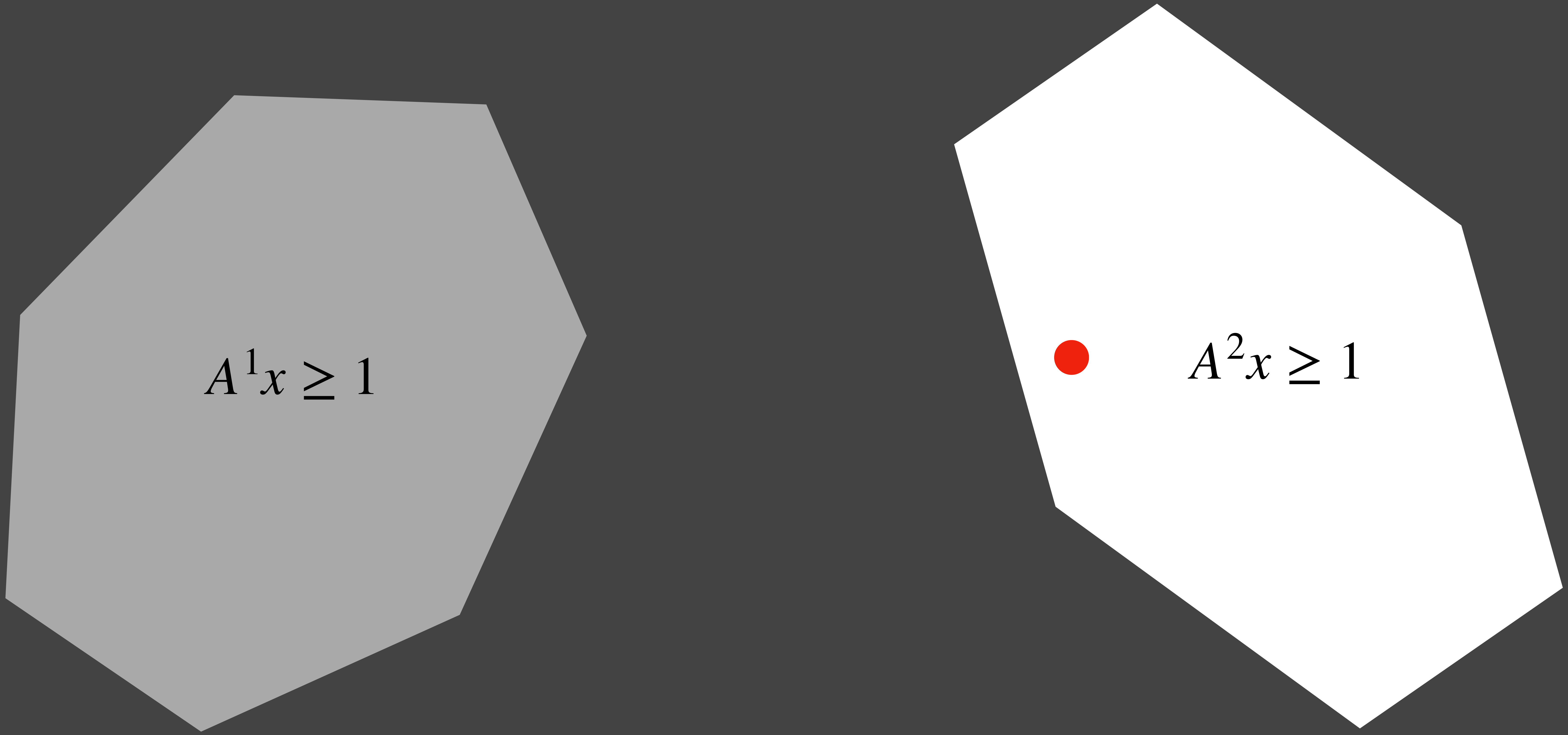
The Meta (Fractional) Problem



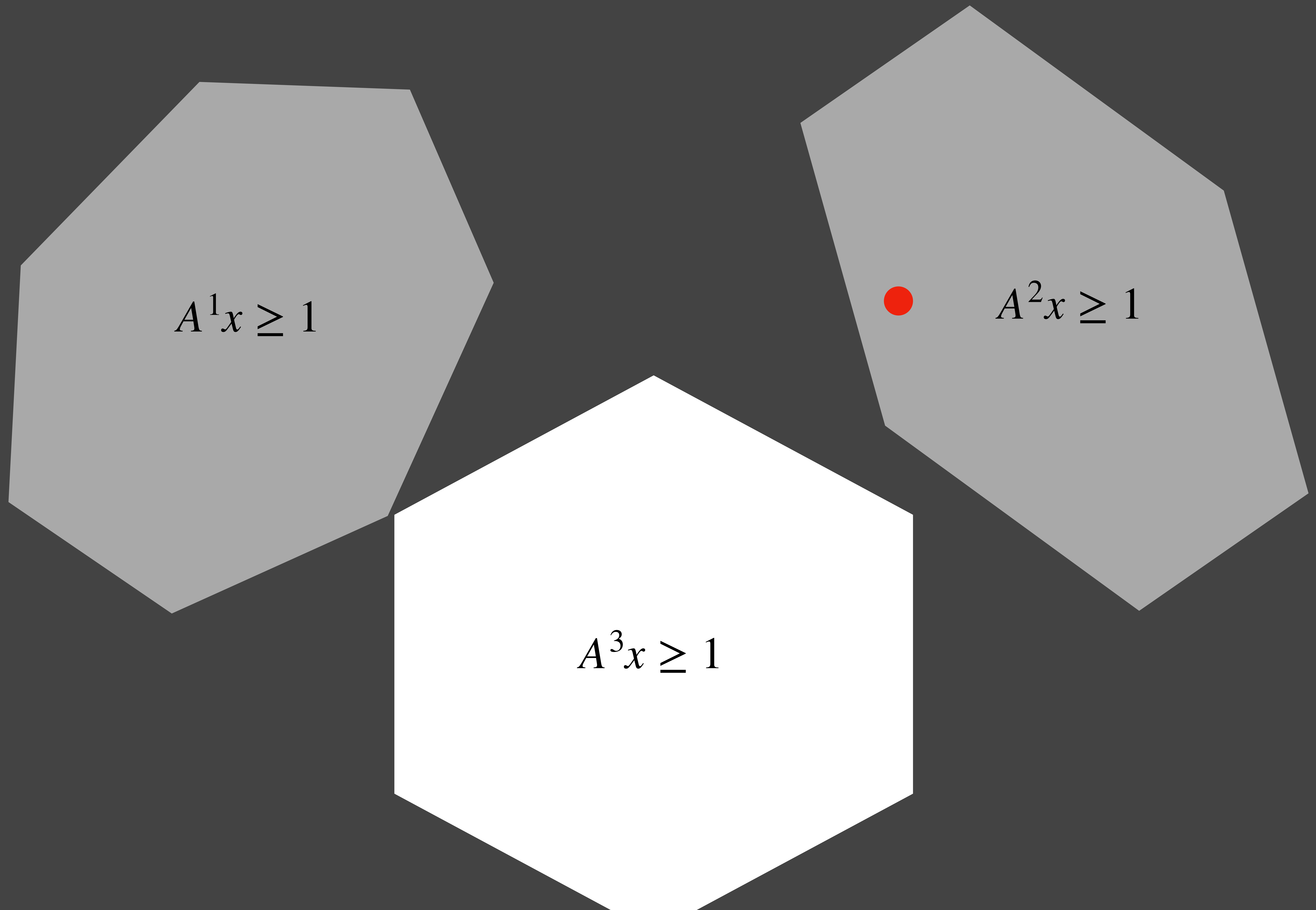
The Meta (Fractional) Problem



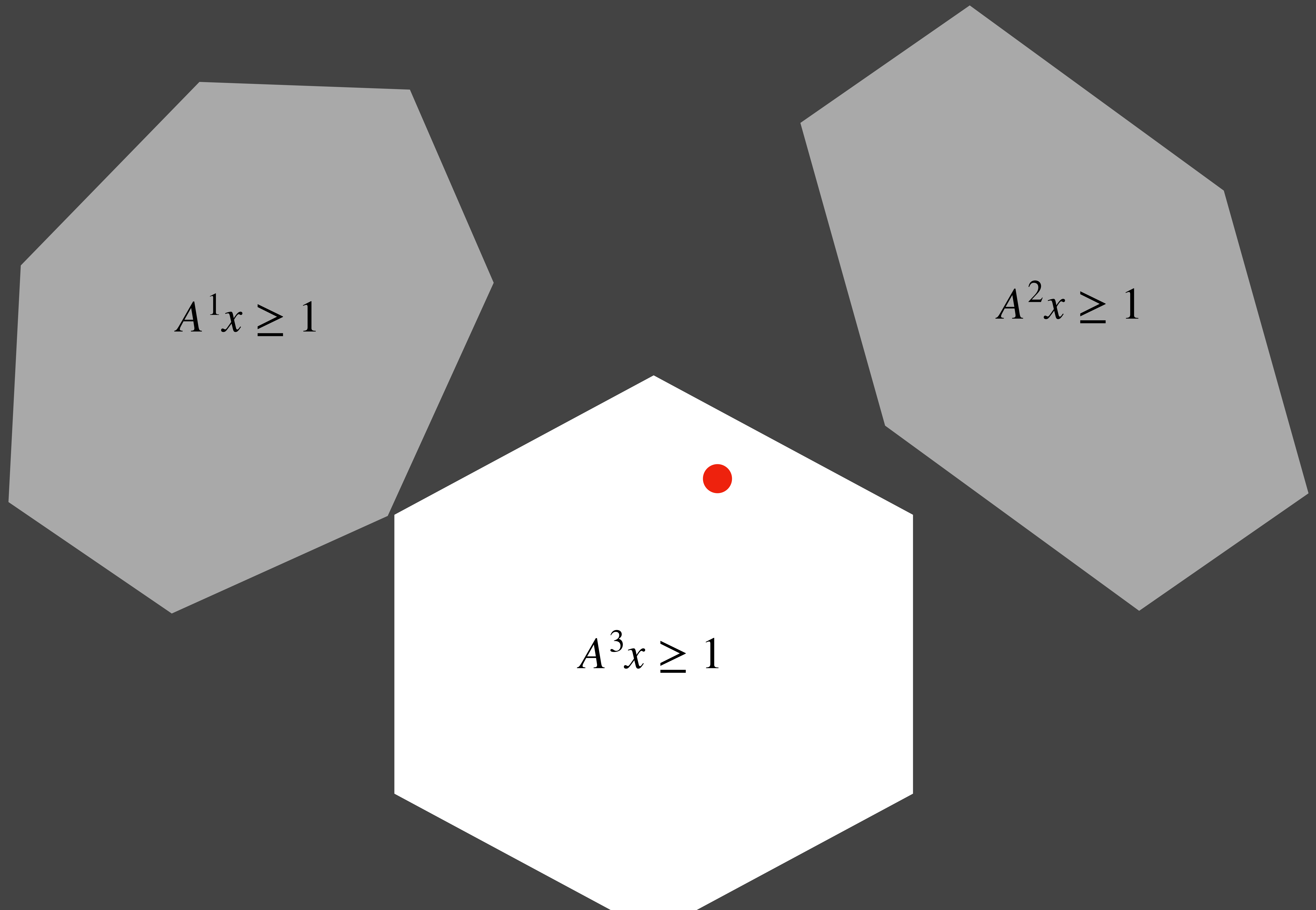
The Meta (Fractional) Problem



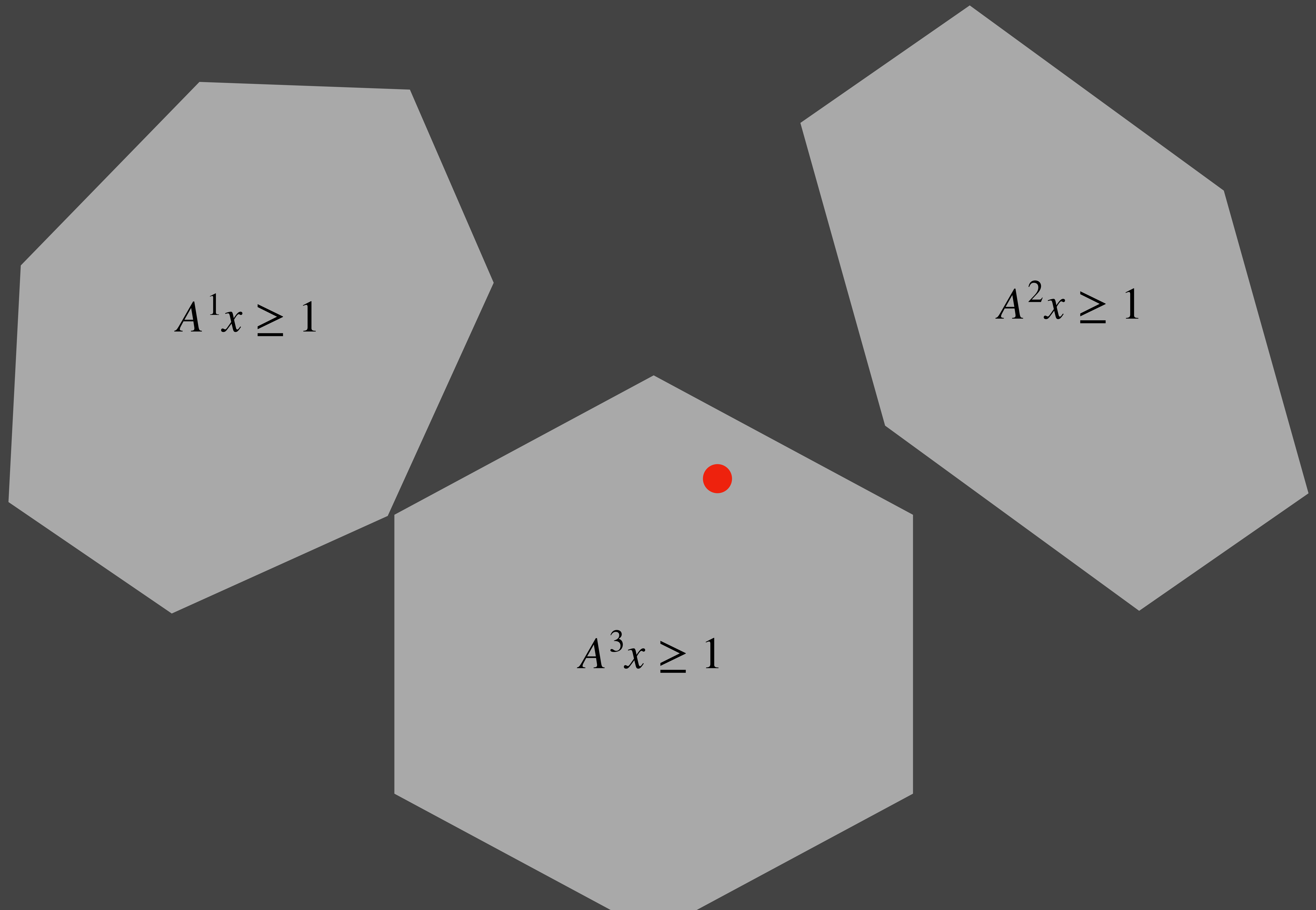
The Meta (Fractional) Problem



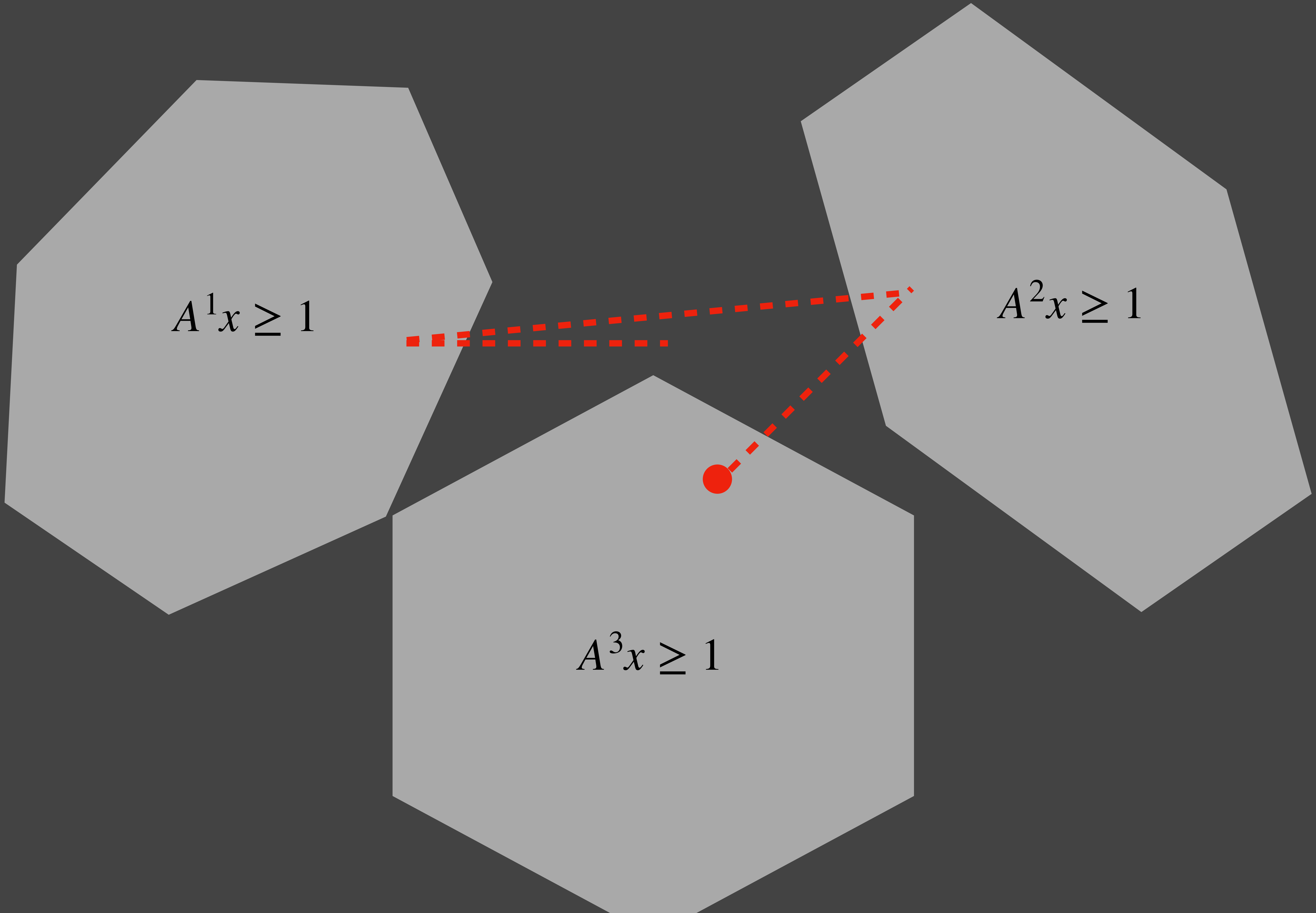
The Meta (Fractional) Problem



The Meta (Fractional) Problem



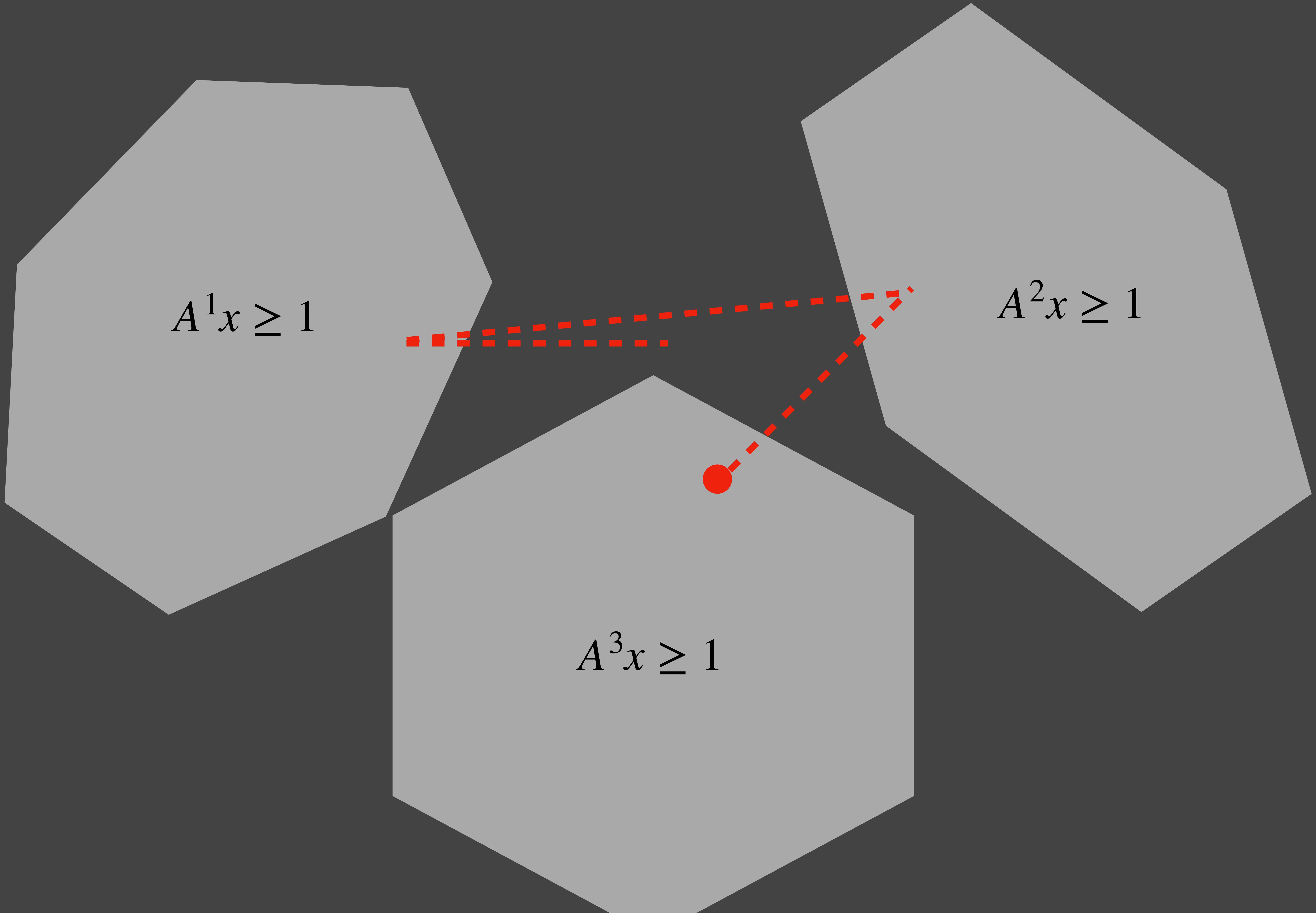
The Meta (Fractional) Problem



Goal: minimize distance traveled,

i.e. $\sum_t d(x^t, x^{t-1})$.

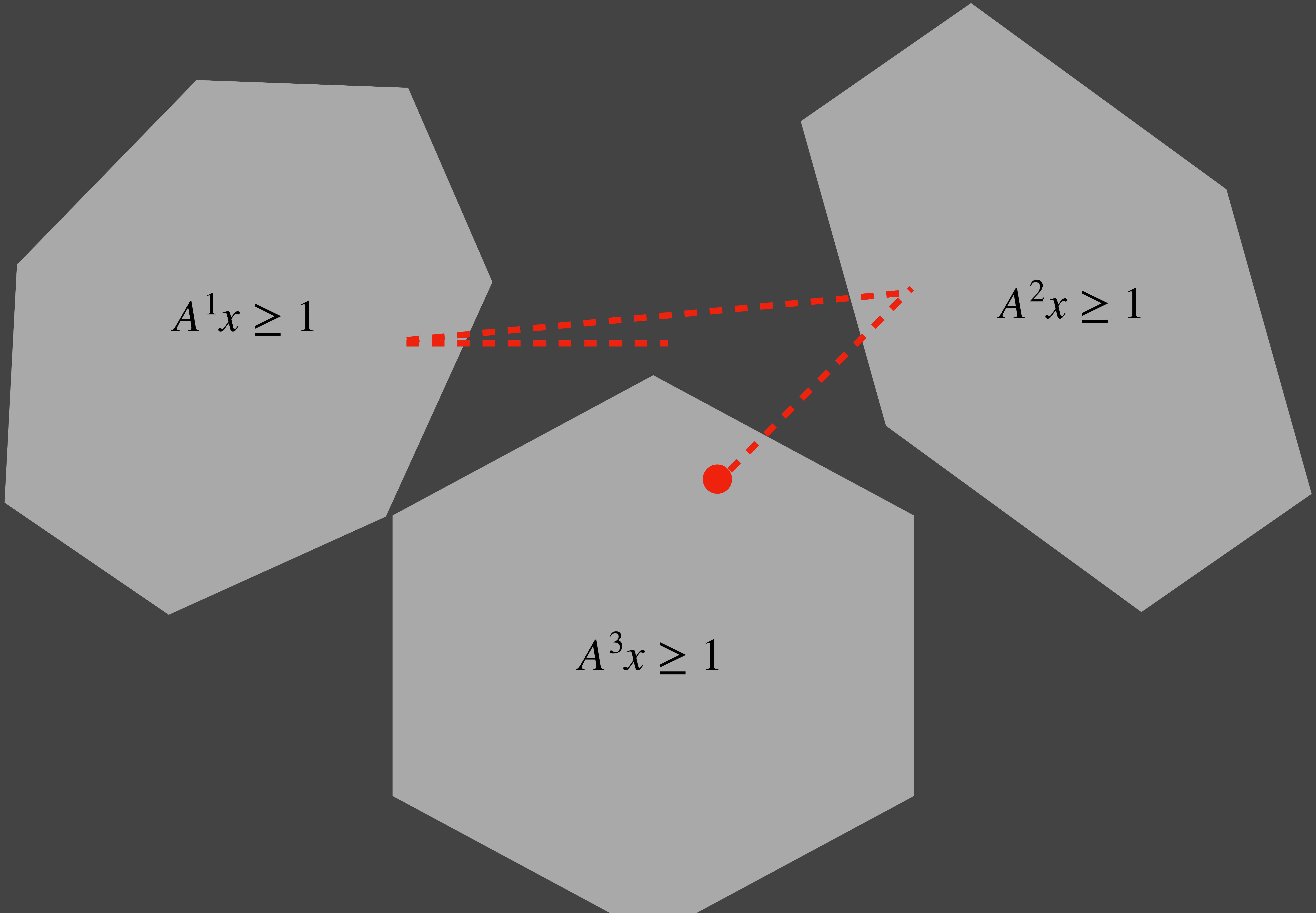
a.k.a. Convex Body Chasing!



Goal: minimize distance traveled,

i.e. $\sum_t d(x^t, x^{t-1})$.

a.k.a. Convex Body Chasing!

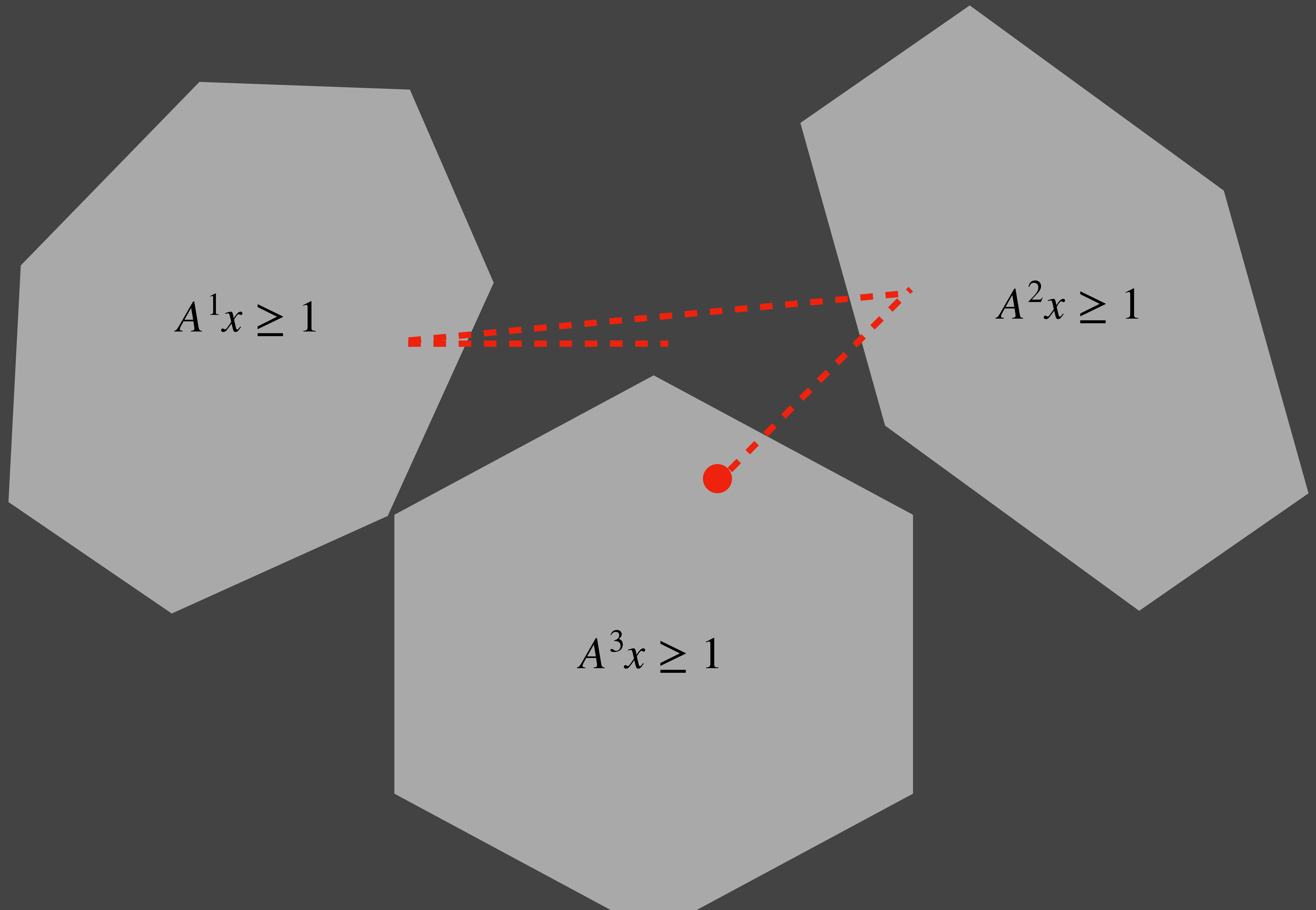


Goal: minimize distance traveled,

i.e. $\sum_t d(x^t, x^{t-1})$.

Introduced by [Friedman Linial 93] to abstract/unify online algorithms.

a.k.a. Convex Body Chasing!

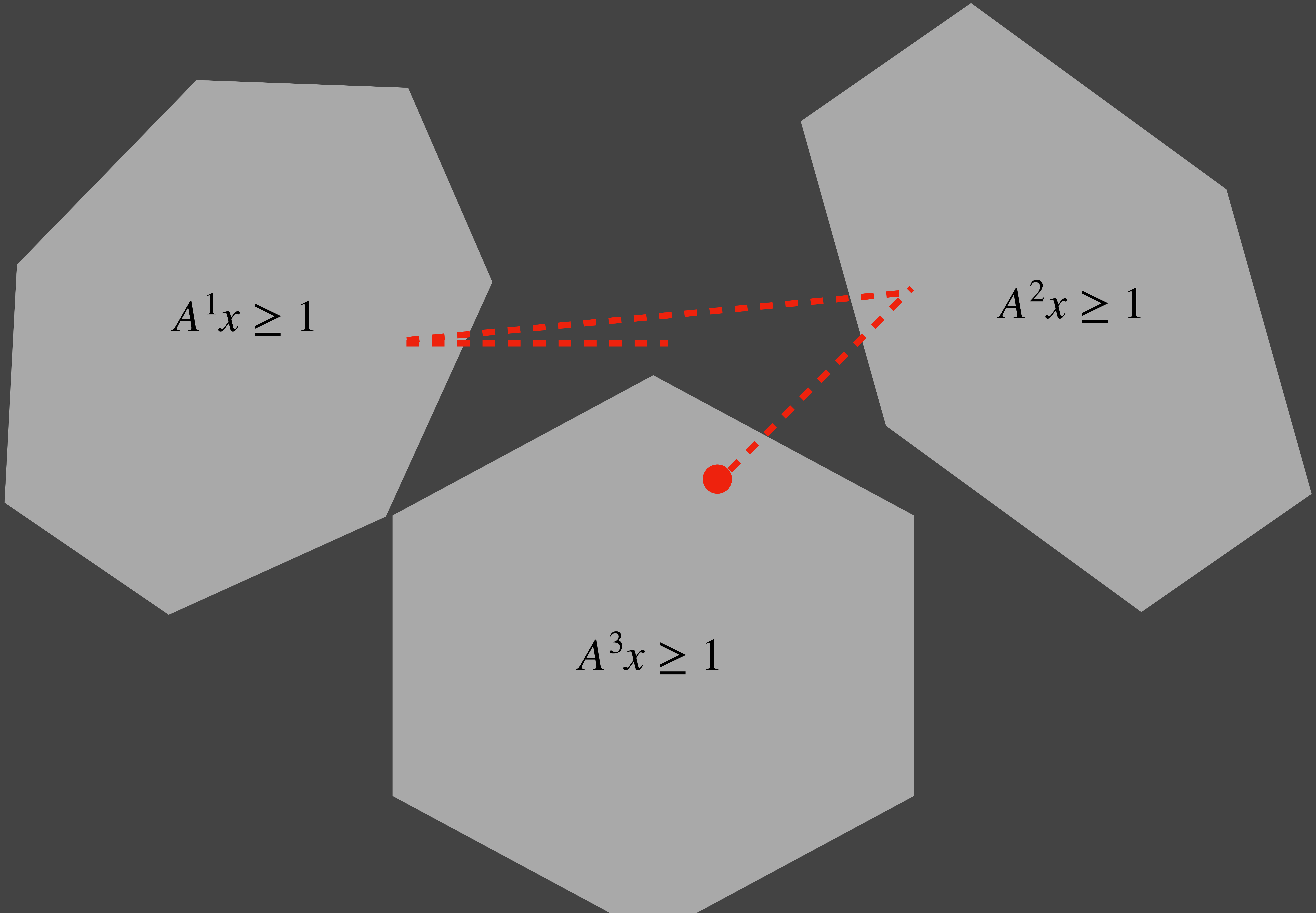


Goal: minimize distance traveled, i.e. $\sum_t d(x^t, x^{t-1})$.

Introduced by [Friedman Linial 93] to abstract/unify online algorithms.

[Argue, Gupta, Guruganesh, Tang 20] & [Selke 20] achieve $O(n)$ competitive ratio.

a.k.a. Convex Body Chasing!



Goal: minimize distance traveled,
i.e. $\sum_t d(x^t, x^{t-1})$.

Introduced by [Friedman Linial 93] to abstract/unify online algorithms.

[Argue, Gupta, Guruganesh, Tang 20] & [Selke 20] achieve $O(n)$ competitive ratio.

Super cool! Co-best papers @ SODA 2020!

But Convex Body Chasing is TOO general

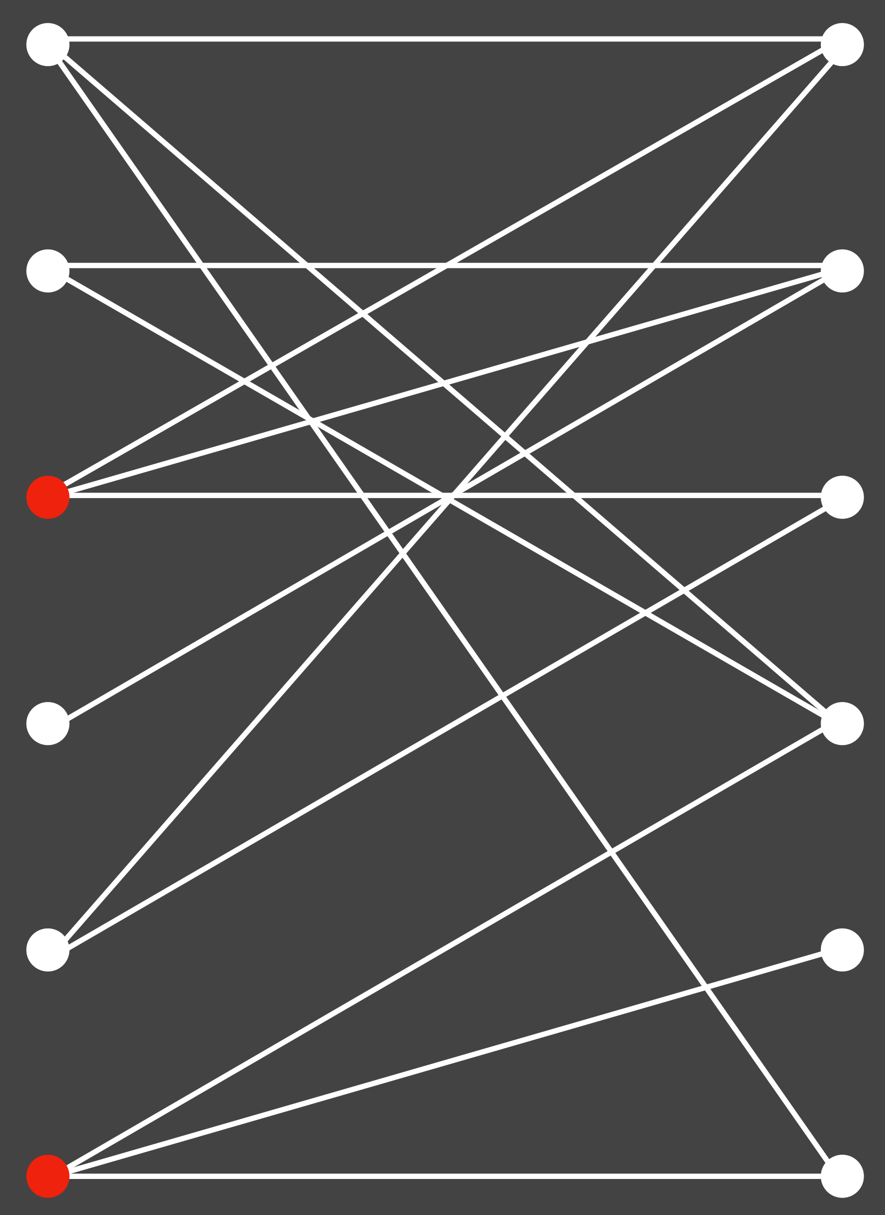
But Convex Body Chasing is TOO general

$\Omega(n)$ too weak in practice!

But Convex Body Chasing is TOO general

$\Omega(n)$ too weak in practice!

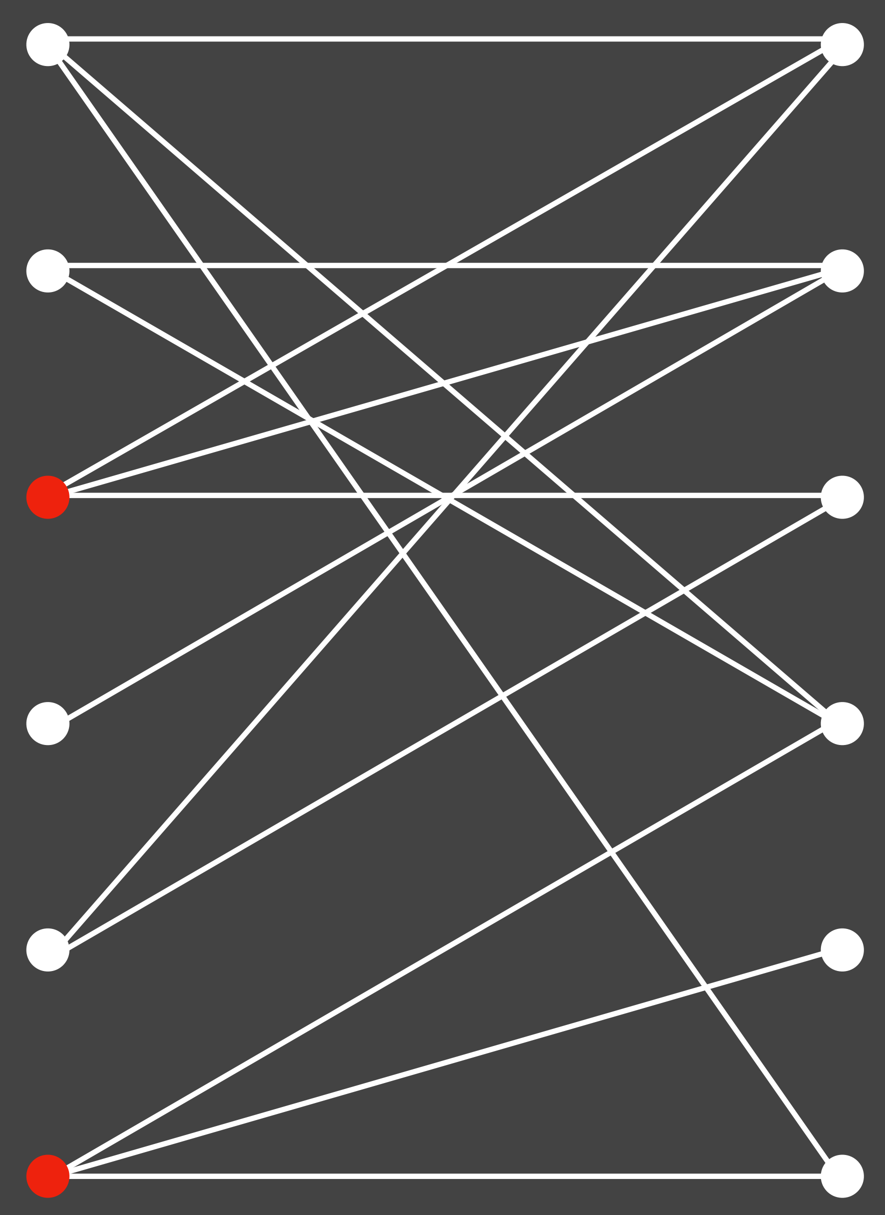
e.g. Set Cover.



But Convex Body Chasing is TOO general

$\Omega(n)$ too weak in practice!

e.g. Set Cover.

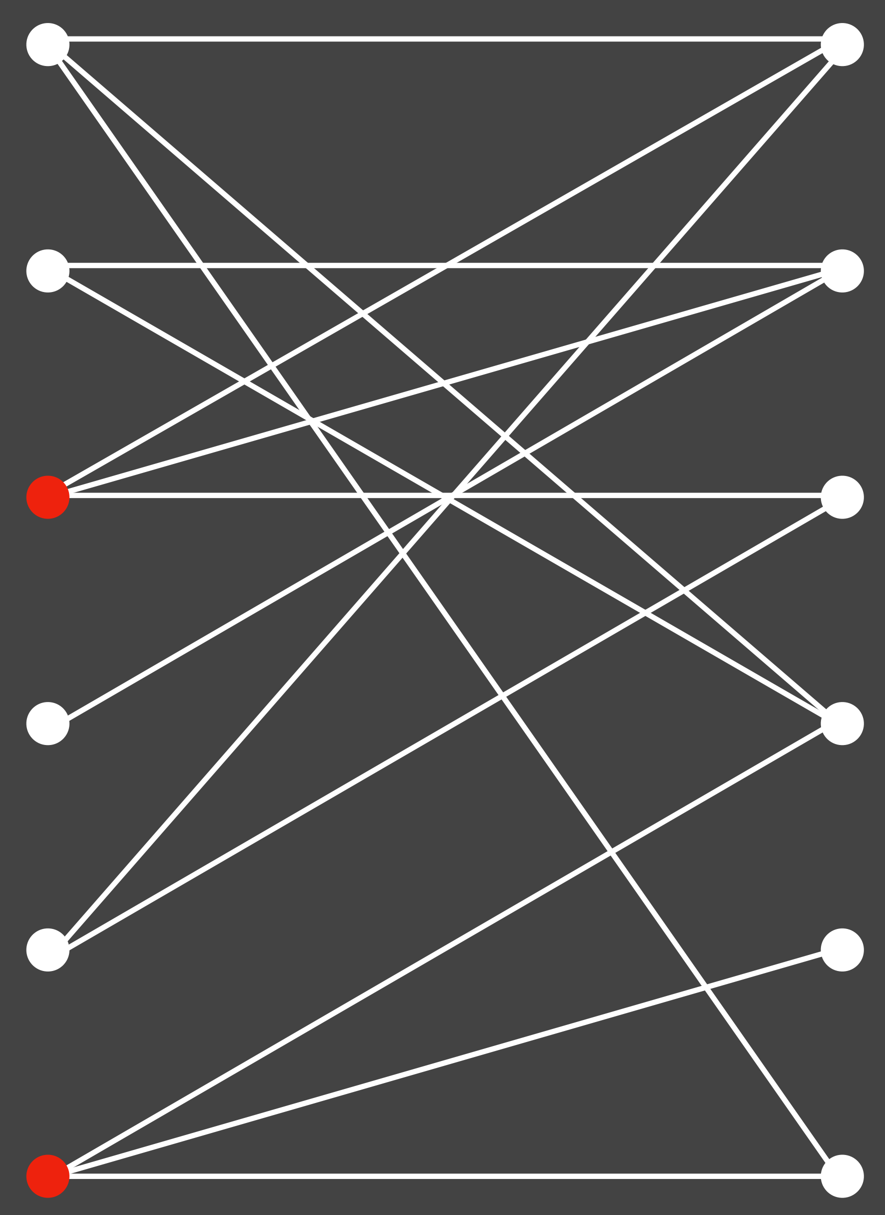


$\Omega(\sqrt{n})$ lower bound [Bubeck Klartag Lee Li Sellke 20] for all l_p metrics, $p \geq 1$.

But Convex Body Chasing is TOO general

$\Omega(n)$ too weak in practice!

e.g. Set Cover.



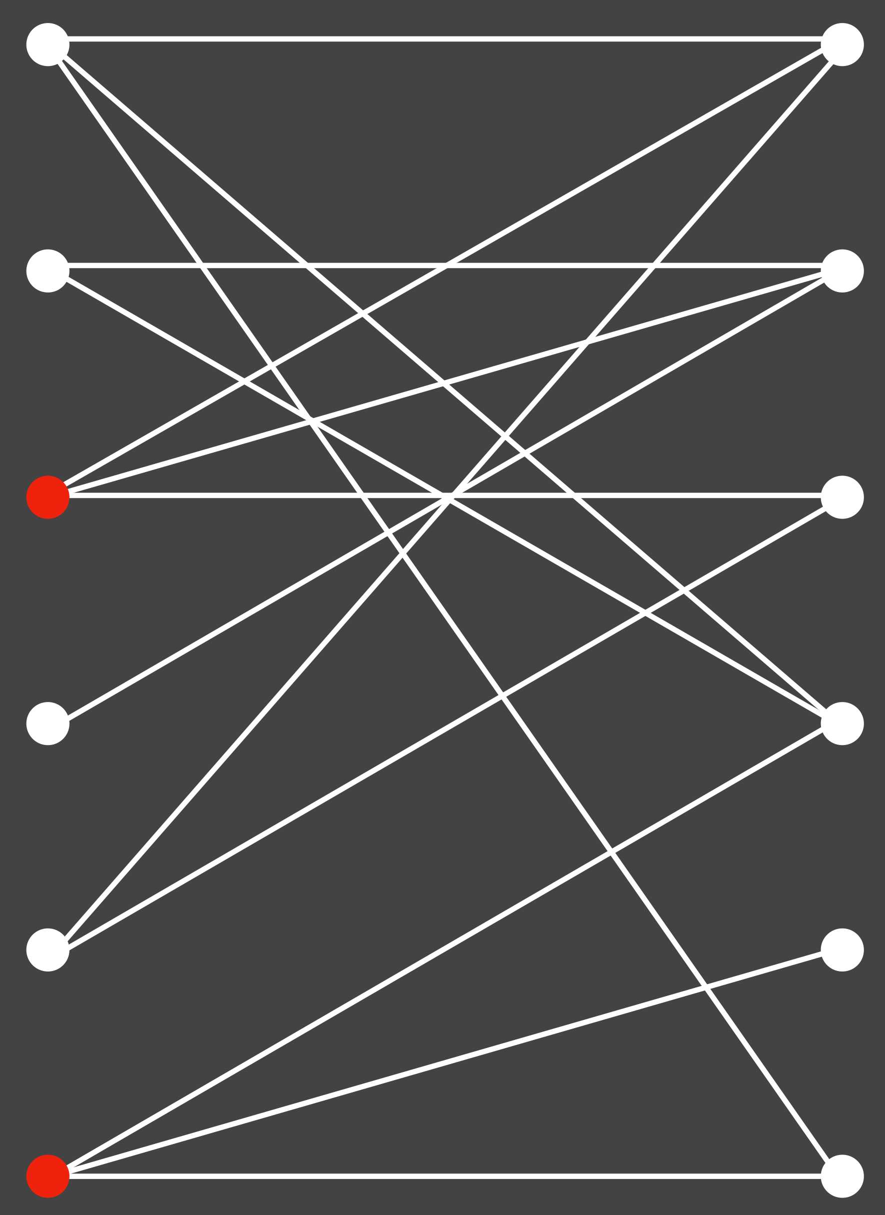
Convex Body chasing has yielded no concrete algorithmic applications.

$\Omega(\sqrt{n})$ lower bound [Bubeck Klartag Lee Li Sellke 20] for all l_p metrics, $p \geq 1$.

But Convex Body Chasing is TOO general

$\Omega(n)$ too weak in practice!

e.g. Set Cover.



Convex Body chasing has yielded no concrete algorithmic applications. 😰

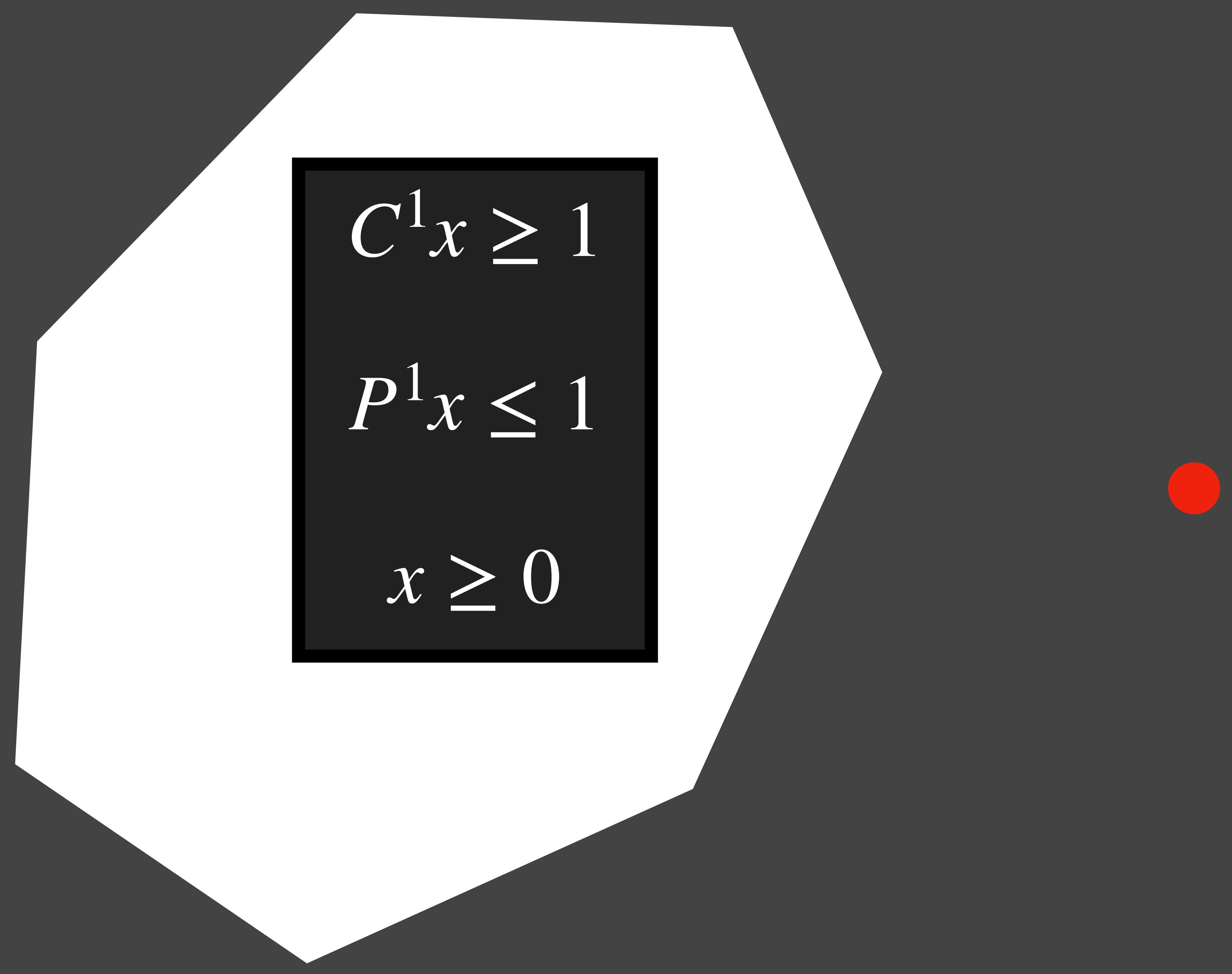
Q: Expressive AND tractable special families of Convex Body Chasing?

$\Omega(\sqrt{n})$ lower bound [Bubeck Klartag Lee Li Sellke 20] for all l_p metrics, $p \geq 1$.

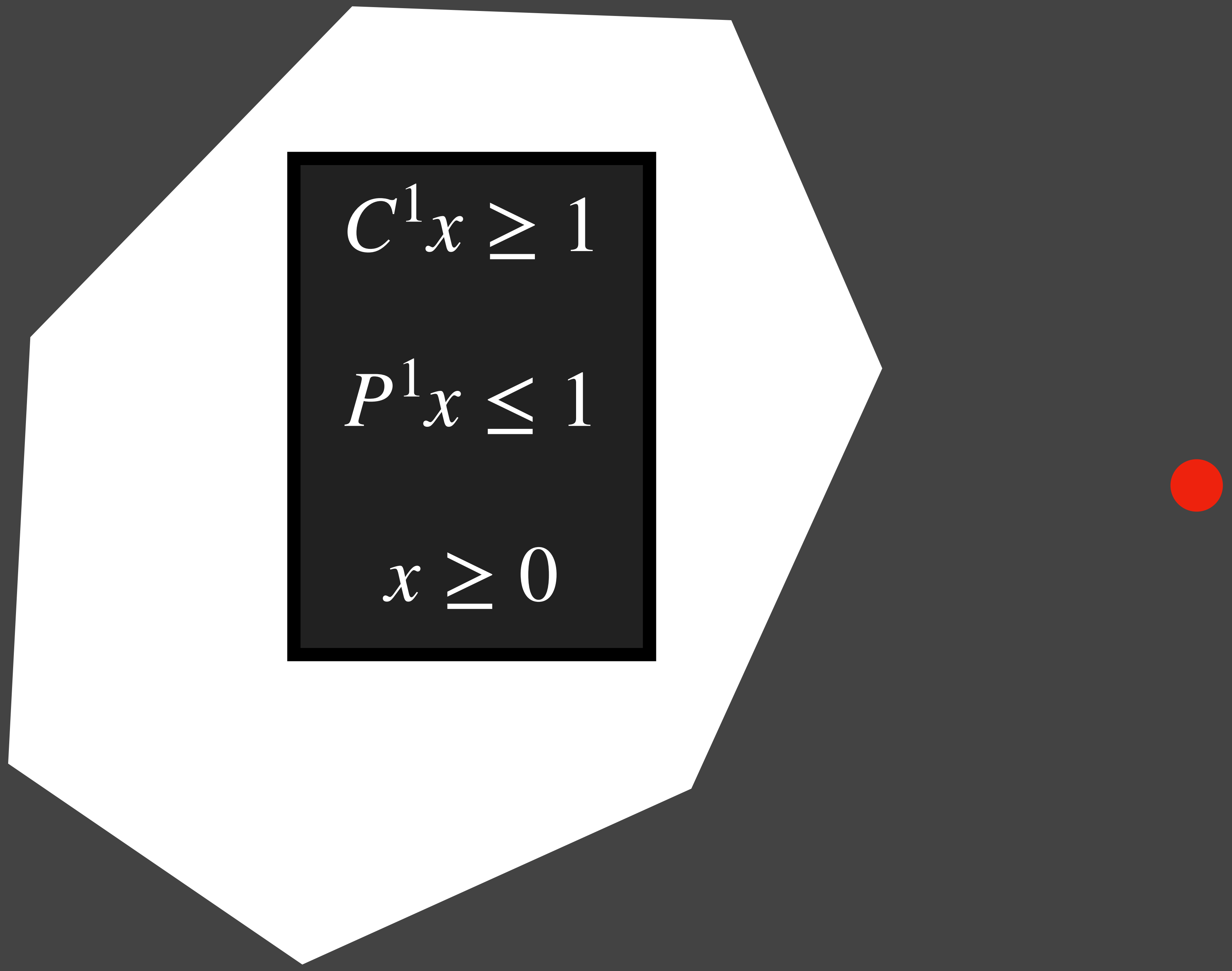
Yes! **Positive** Body Chasing in ℓ_1

Yes! Positive Body Chasing in ℓ_1

Yes! Positive Body Chasing in ℓ_1

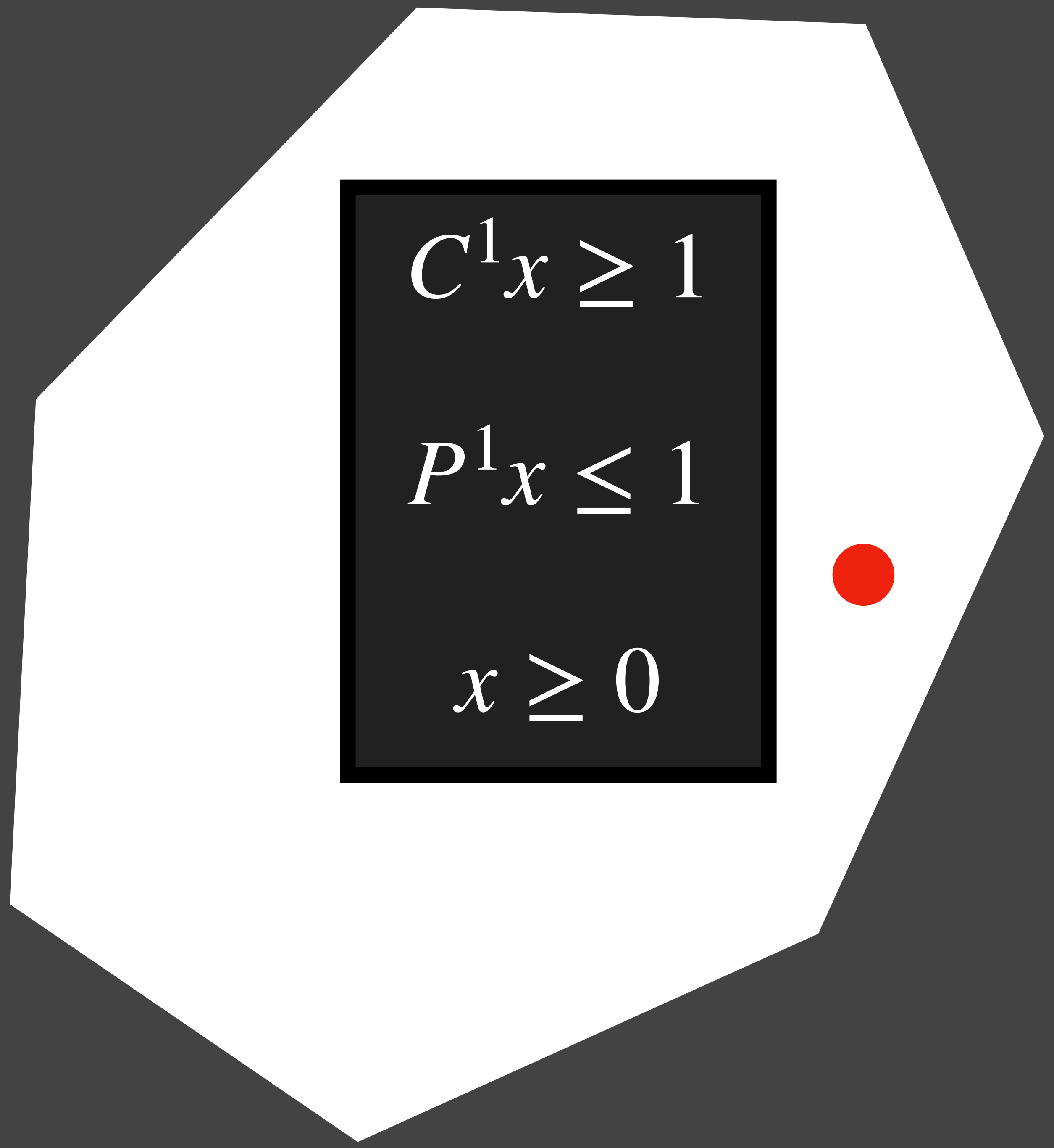


Yes! Positive Body Chasing in ℓ_1



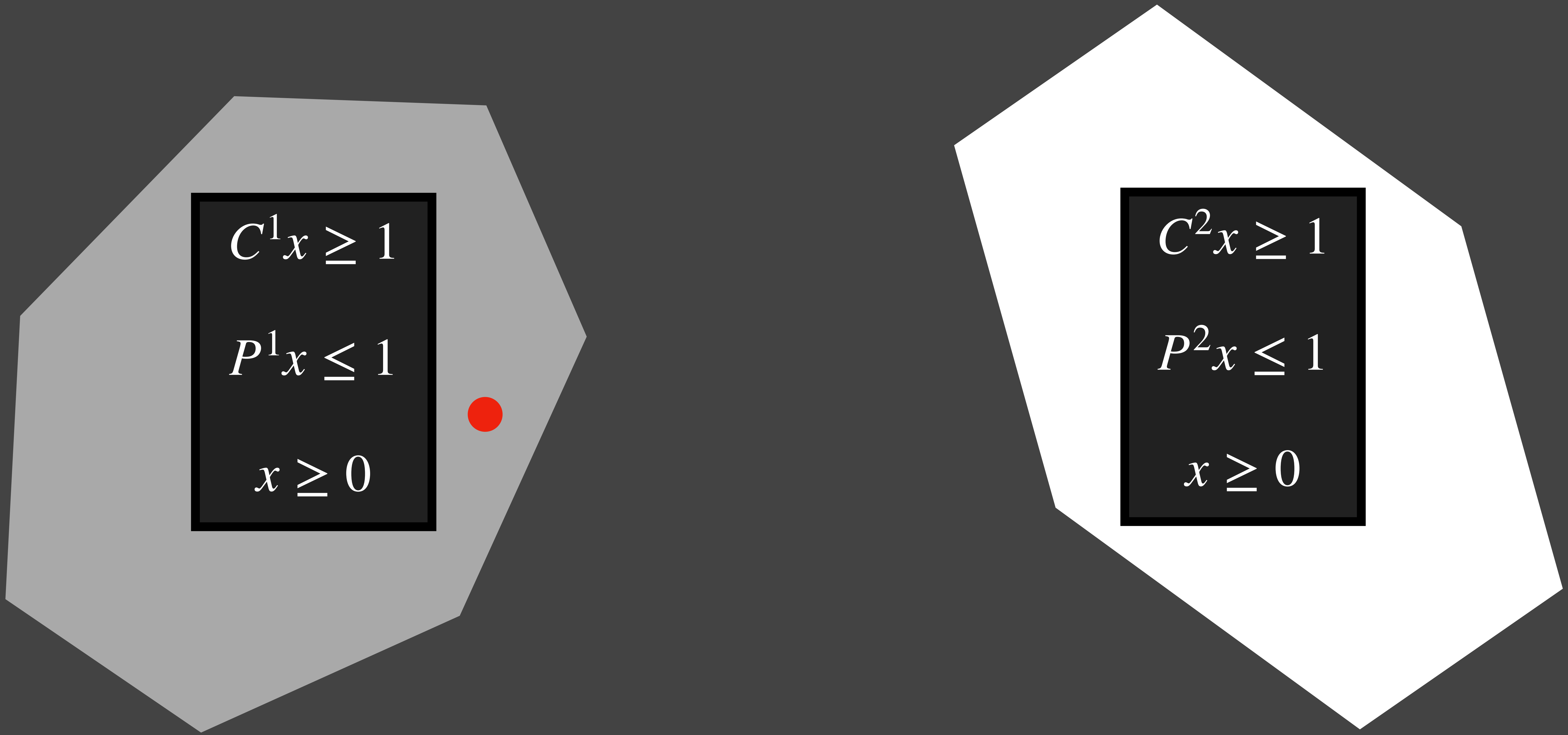
C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1



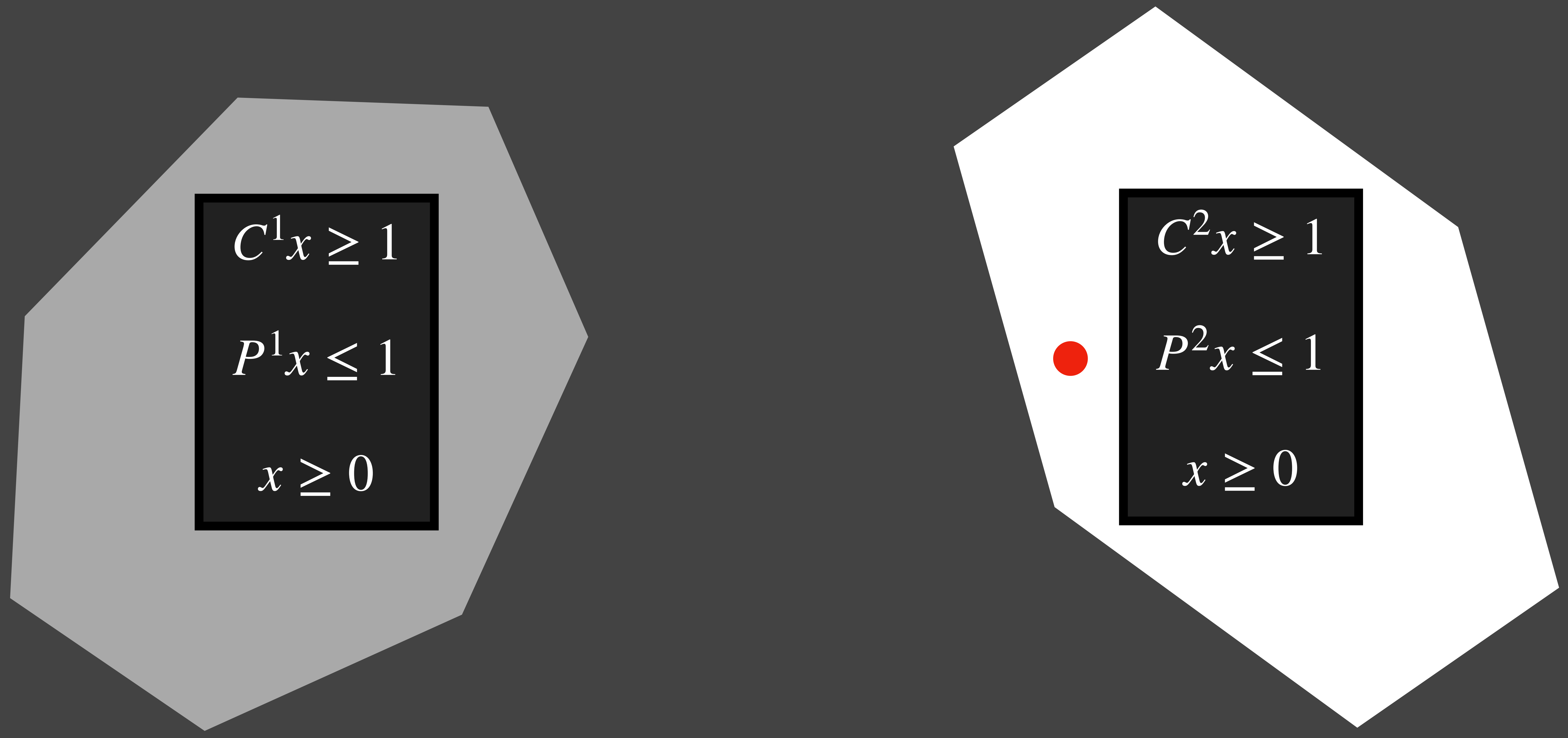
C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1



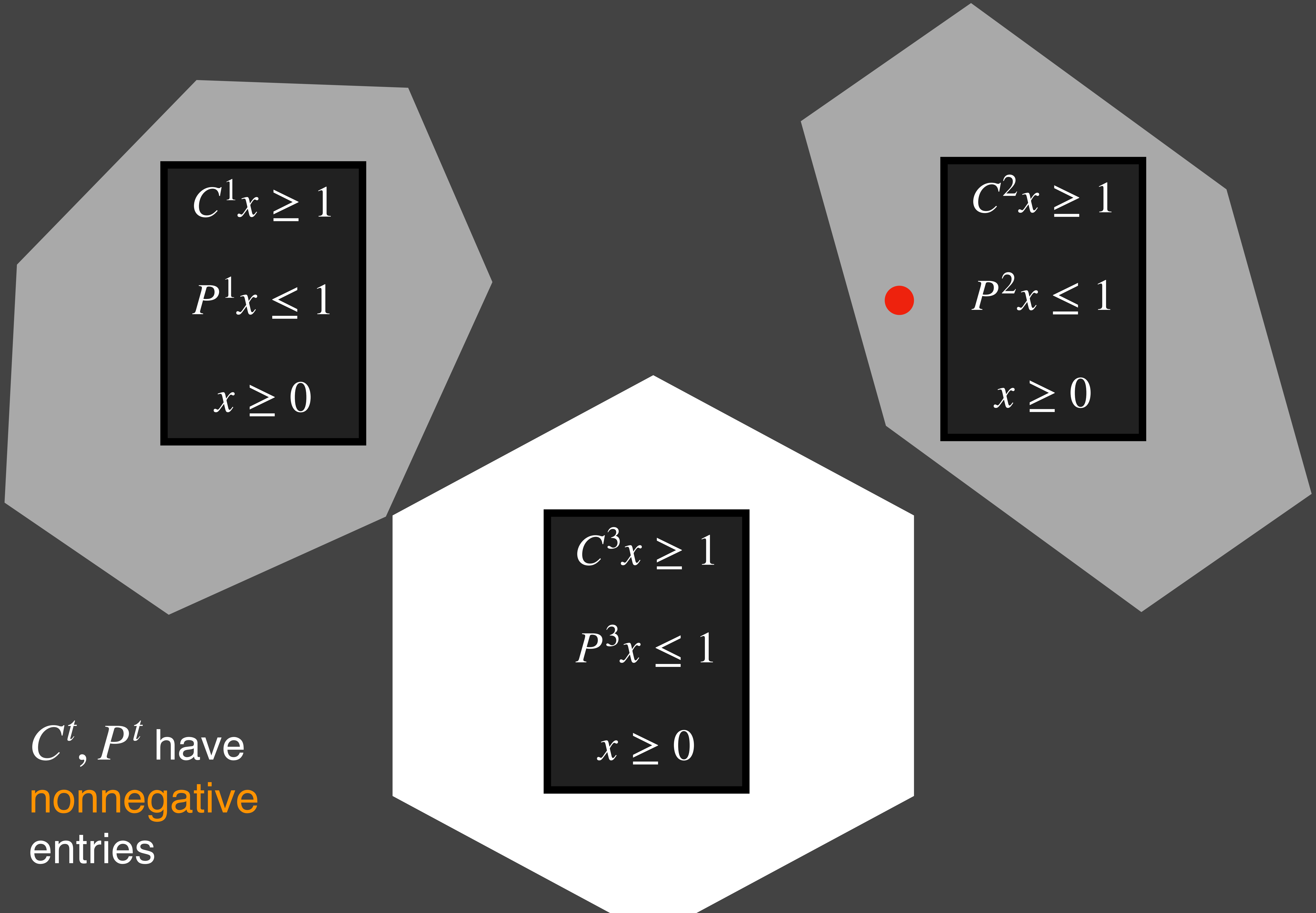
C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1



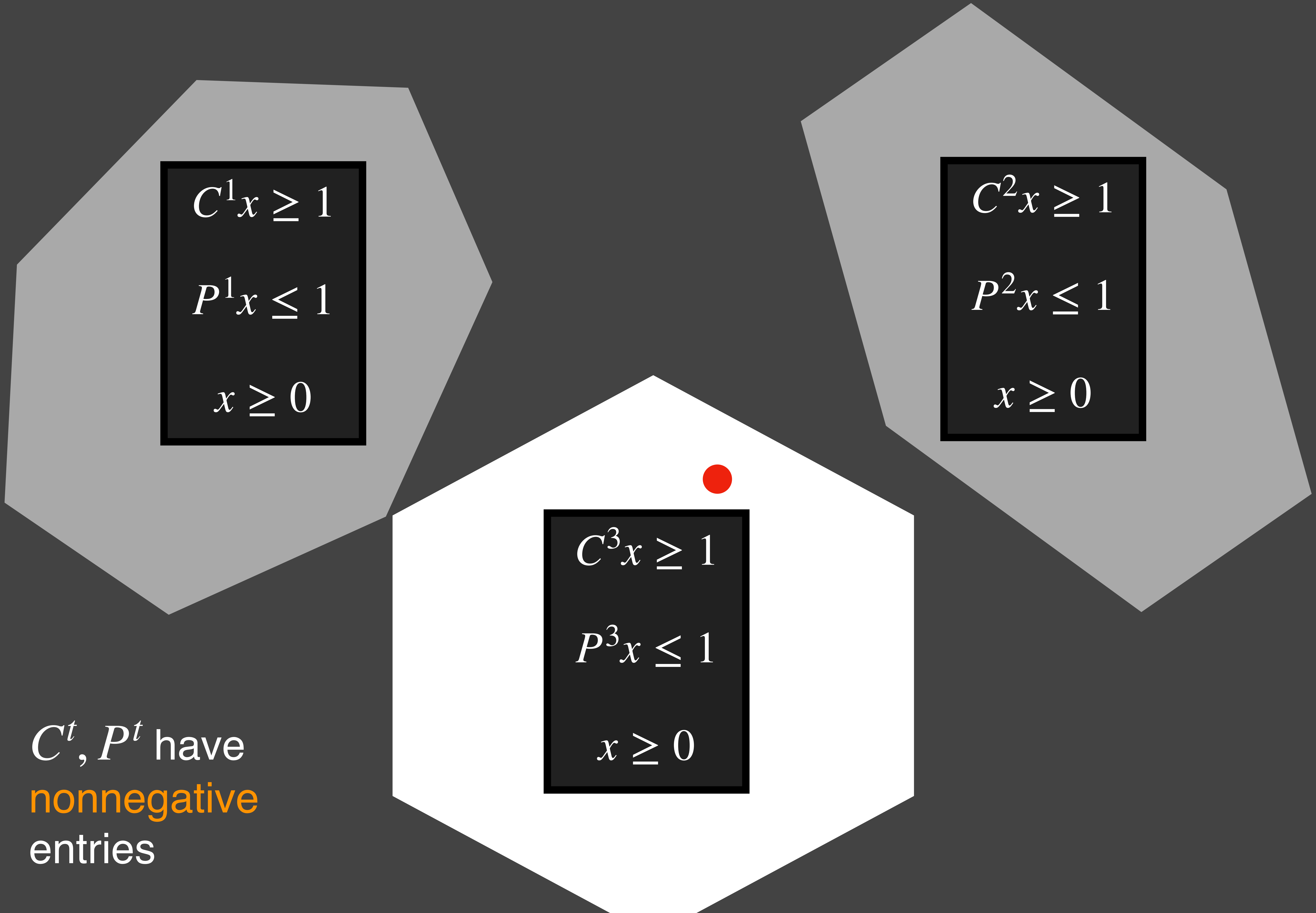
C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1



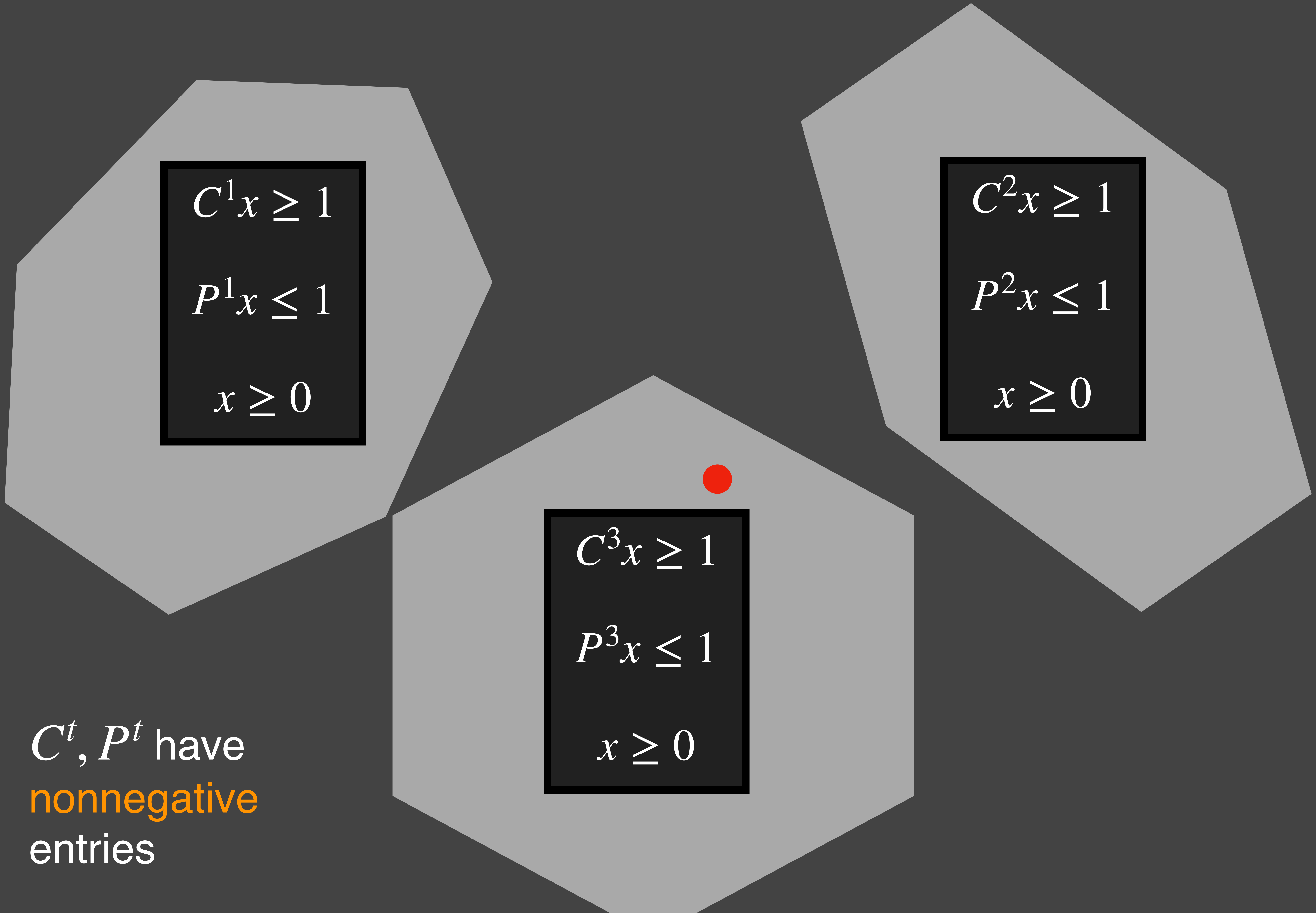
C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1



C^t, P^t have nonnegative entries

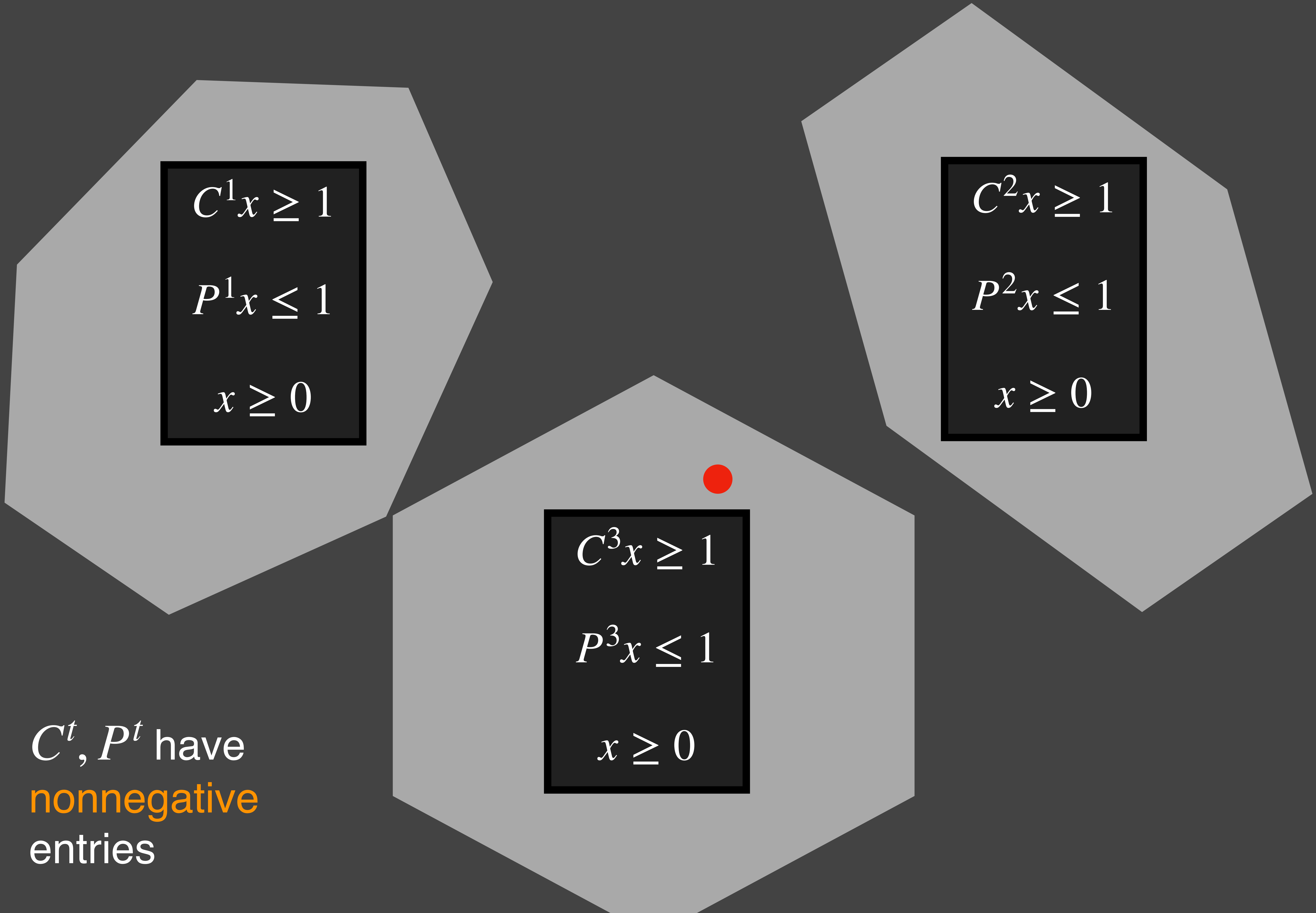
Yes! Positive Body Chasing in ℓ_1



C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1

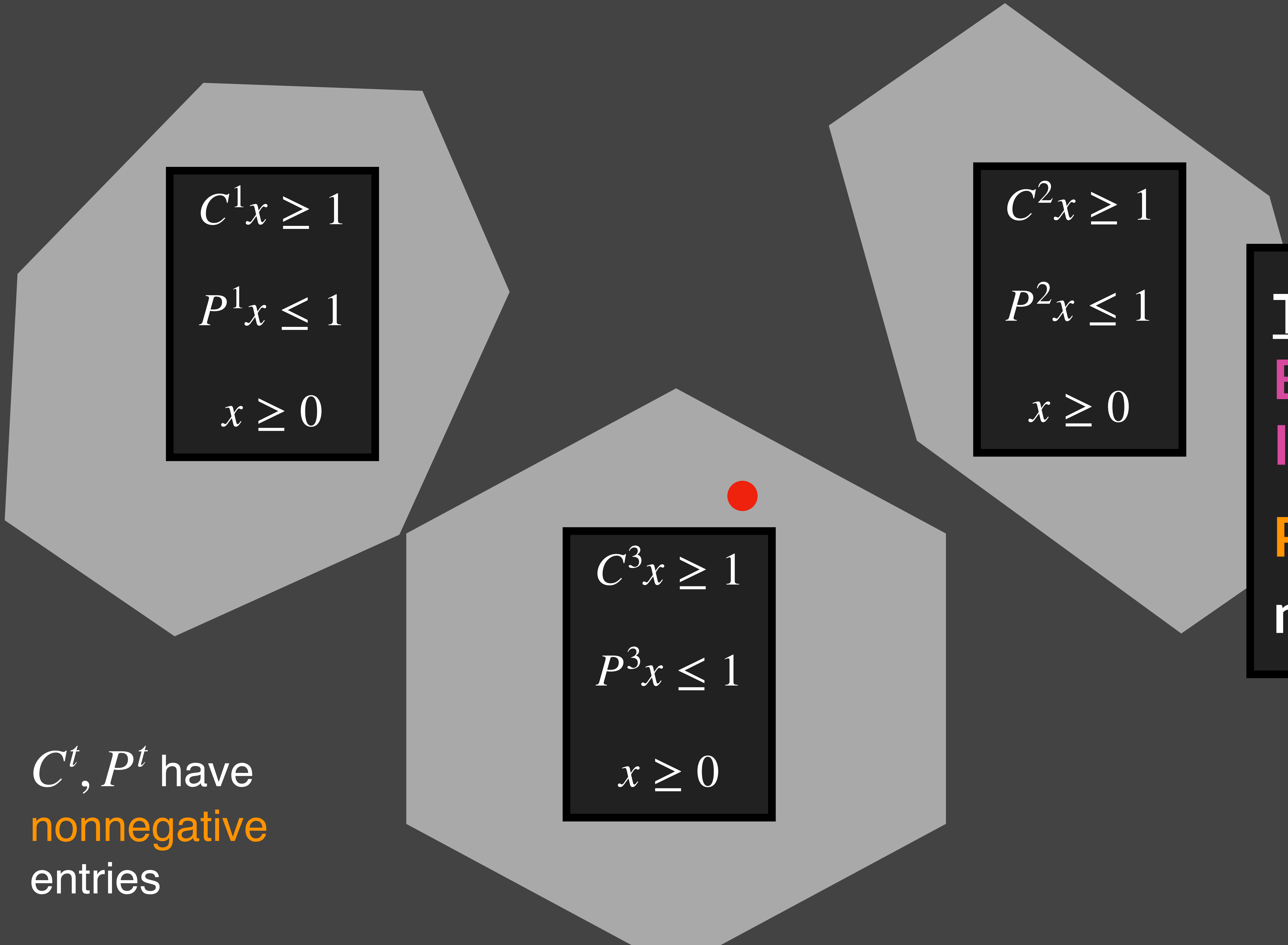
Captures dynamic Set Cover, Load Balancing, Matching, Minimum Spanning Tree!



C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1

Captures dynamic Set Cover, Load Balancing, Matching, Minimum Spanning Tree!



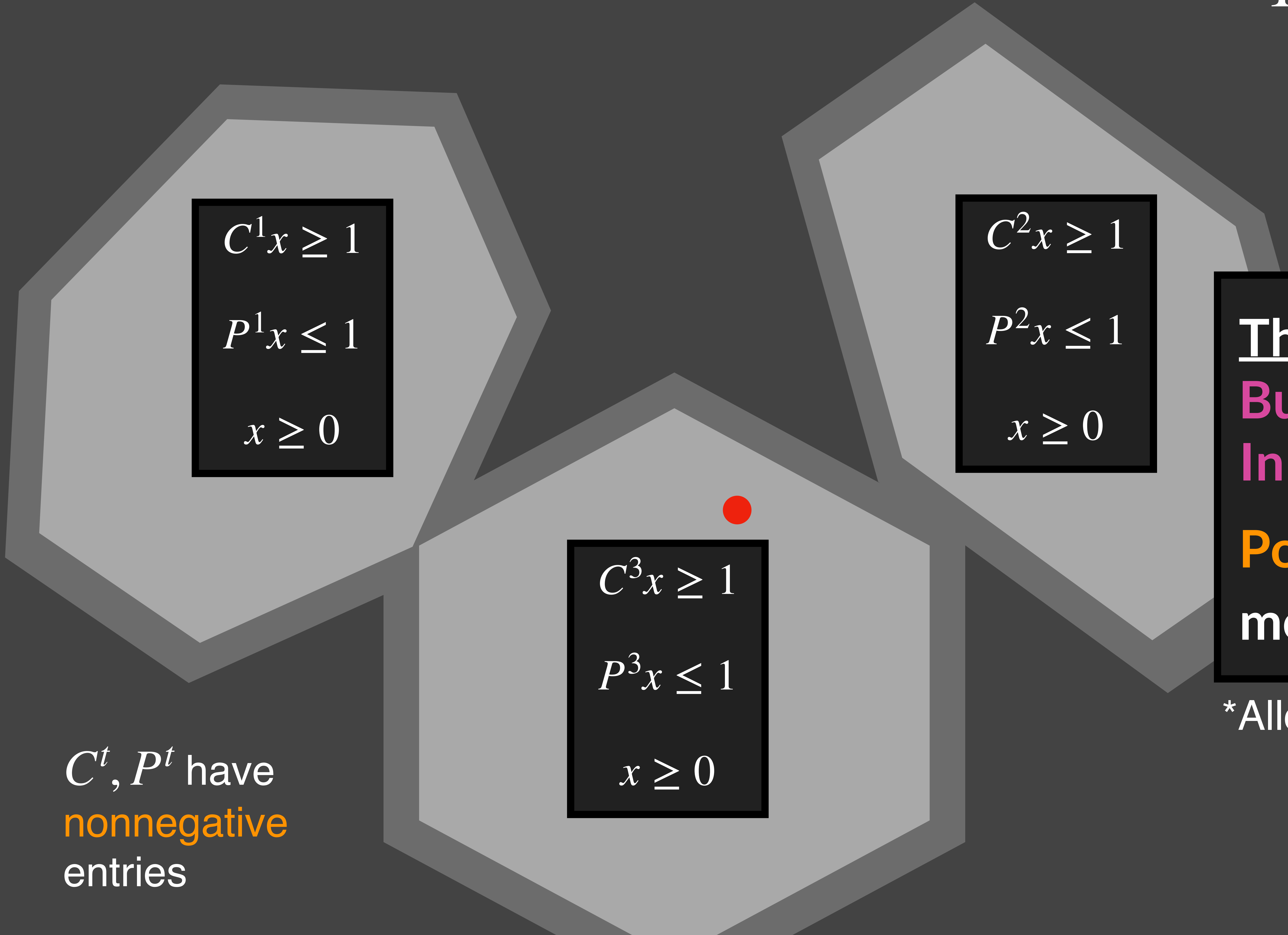
Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT. *

C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1

Captures dynamic Set Cover, Load Balancing, Matching, Minimum Spanning Tree!



Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

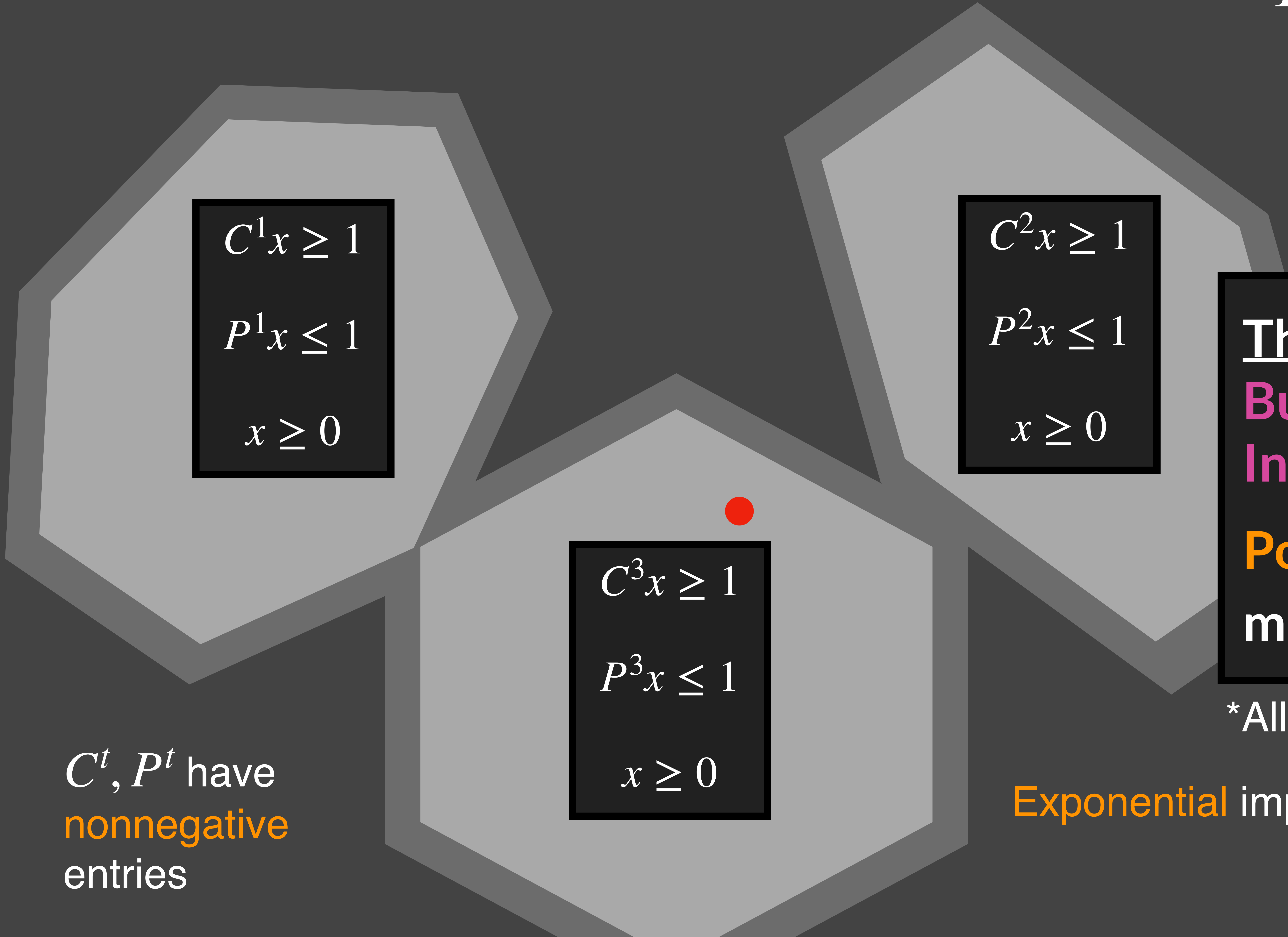
Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.*

*Allow $P^t x \leq 1 + \epsilon$.

C^t, P^t have nonnegative entries

Yes! Positive Body Chasing in ℓ_1

Captures dynamic Set Cover, Load Balancing, Matching, Minimum Spanning Tree!



Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.*

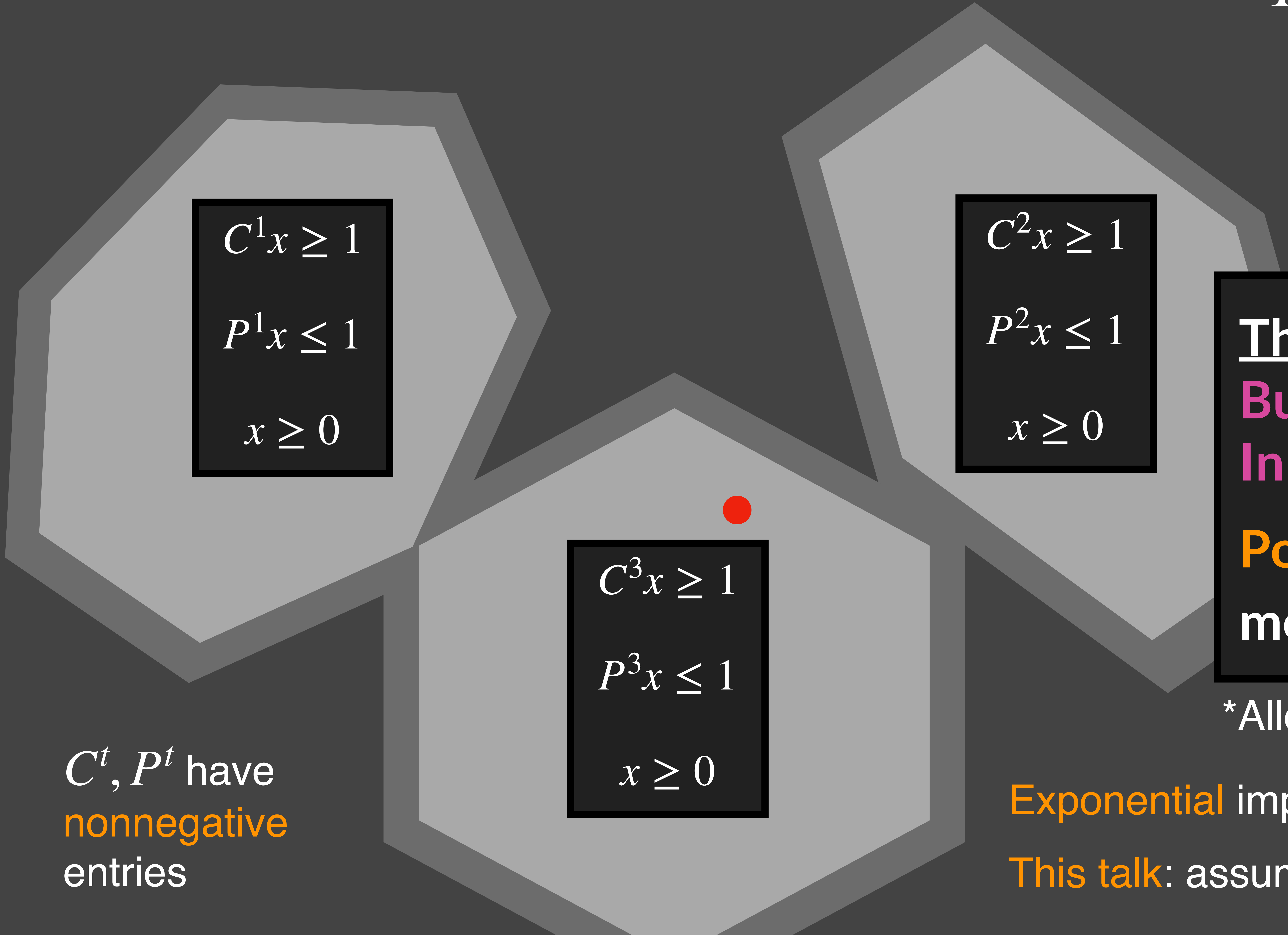
*Allow $P^t x \leq 1 + \epsilon$.

C^t, P^t have nonnegative entries

Exponential improvement general chasing!

Yes! Positive Body Chasing in ℓ_1

Captures dynamic Set Cover, Load Balancing, Matching, Minimum Spanning Tree!



Theorem **[Bhattacharya, Buchbinder, L., Saranurak, In submission]:**

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.*

*Allow $P^t x \leq 1 + \epsilon$.

C^t, P^t have nonnegative entries

Exponential improvement general chasing!

This talk: assume C^t, P^t are 0/1 matrices.

Applications via Rounding

Applications via Rounding

Theorem is dynamic analog of LP solver.

Applications via Rounding

Theorem is dynamic analog of LP solver.

We show how to round fractional solutions to give:

Applications via Rounding

Theorem is dynamic analog of LP solver.
We show how to round fractional solutions to give:

	Prior Work			Our Paper [BBLS]	
	Approx	**Recourse**	**Ref**	**Approx**	**Recourse**
Set Cover	O(log n)	O(T)	[GKKP 17]	O(log n)	O(log n log f) · OPT
	O(f)	O(T)		O(f)	O(f log f) · OPT
Load Balancing	2+ε	T · log n · poly(1/ε)	[KLS 23]	2+ε	poly(1/ε, log n) · OPT
Bipartite Matching	1+ε	O(T/ε)	[Folklore]	1+ε	poly(1/ε, log n) · OPT
Min. Spanning Tree	4	O(T)	[GK 14]	2+ε	poly(1/ε, log n) · OPT

Applications via Rounding

OPT := min recourse of any algorithm with same approx.

Theorem is dynamic analog of LP solver.
We show how to round fractional solutions to give:

	Prior Work			Our Paper [BBLS]	
	Approx	**Recourse**	**Ref**	**Approx**	**Recourse**
Set Cover	O(log n)	O(T)	[GKKP 17]	O(log n)	O(log n log f) · OPT
	O(f)	O(T)		O(f)	O(f log f) · OPT
Load Balancing	2+ε	T · log n · poly(1/ε)	[KLS 23]	2+ε	poly(1/ε, log n) · OPT
Bipartite Matching	1+ε	O(T/ε)	[Folklore]	1+ε	poly(1/ε, log n) · OPT
Min. Spanning Tree	4	O(T)	[GK 14]	2+ε	poly(1/ε, log n) · OPT

Applications via Rounding

OPT := min recourse of any algorithm with same approx.

Theorem is dynamic analog of LP solver.
We show how to round fractional solutions to give:

	Prior Work			Our Paper [BBLS]	
	Approx	**Recourse**	**Ref**	**Approx**	**Recourse**
Set Cover	O(log n)	O(T)	[GKKP 17]	O(log n)	O(log n log f) · OPT
	O(f)	O(T)		O(f)	O(f log f) · OPT
Load Balancing	2+ε	T · log n · poly(1/ε)	[KLS 23]	2+ε	poly(1/ε, log n) · OPT
Bipartite Matching	1+ε	O(T/ε)	[Folklore]	1+ε	poly(1/ε, log n) · OPT
Min. Spanning Tree	4	O(T)	[GK 14]	2+ε	poly(1/ε, log n) · OPT

Insert only.

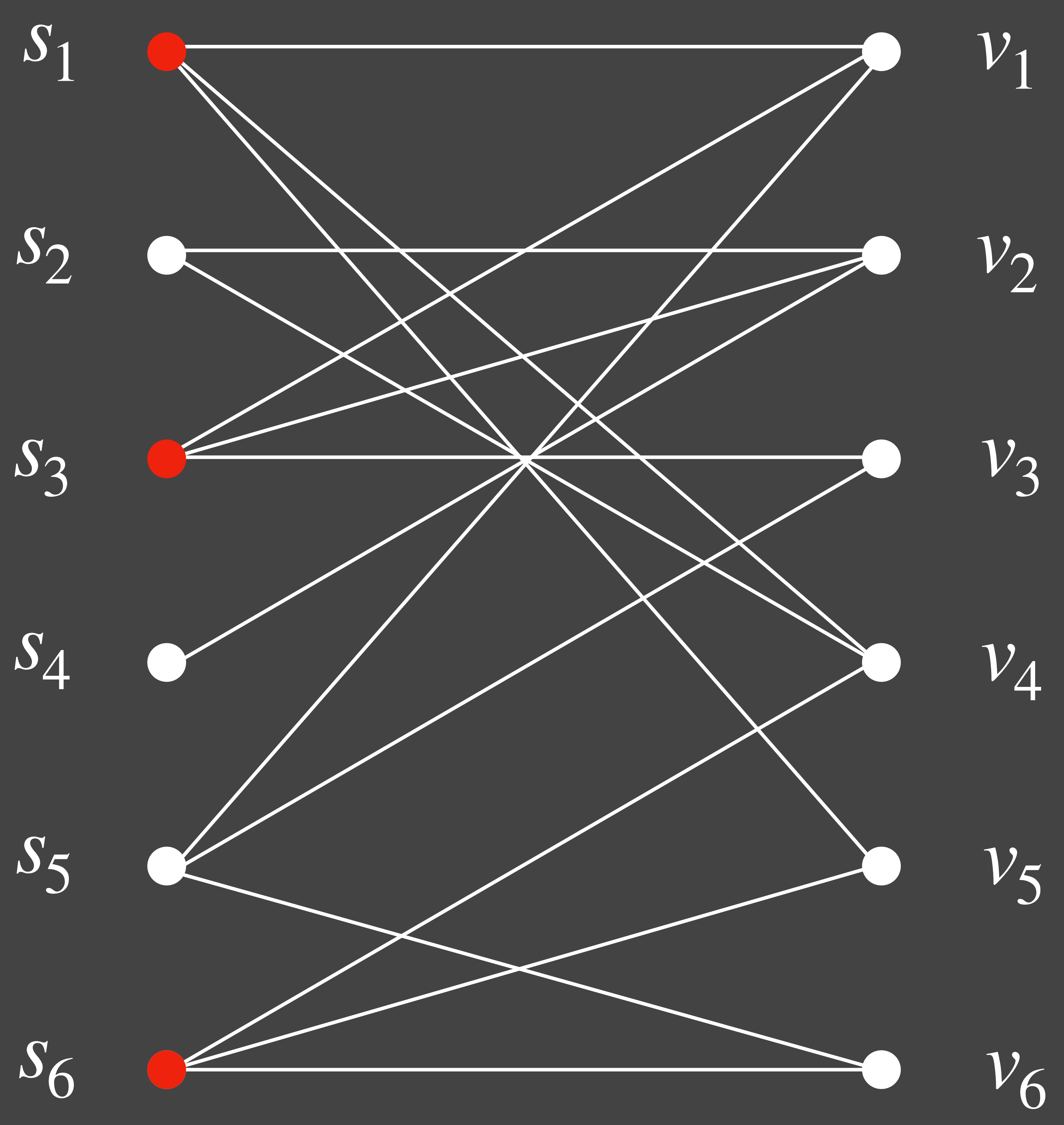
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\text{OPT})$

Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\text{OPT})$

1. Competitive recourse gives instance optimal guarantee.

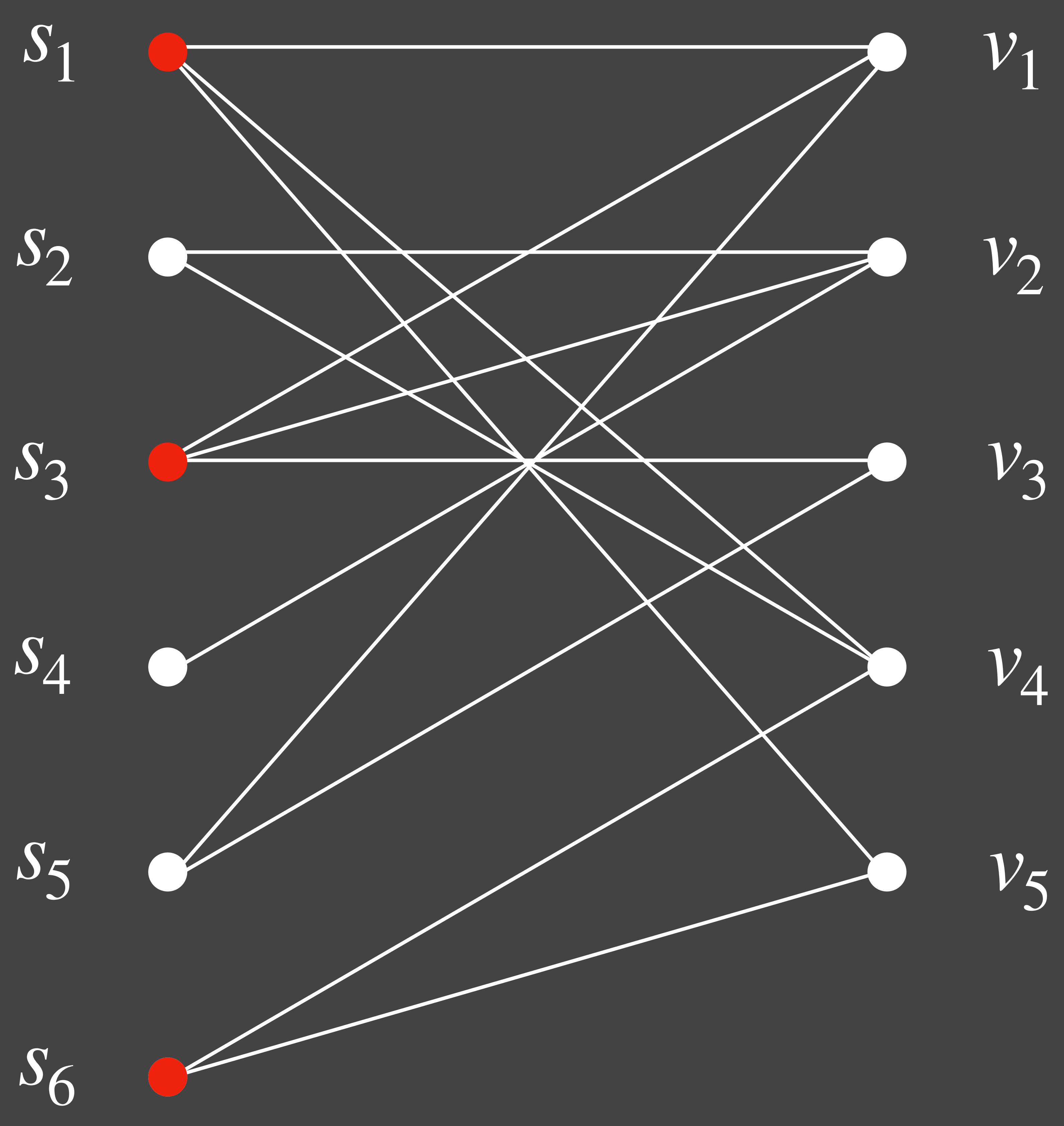
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathsf{OPT})$

1. Competitive recourse gives instance optimal guarantee.



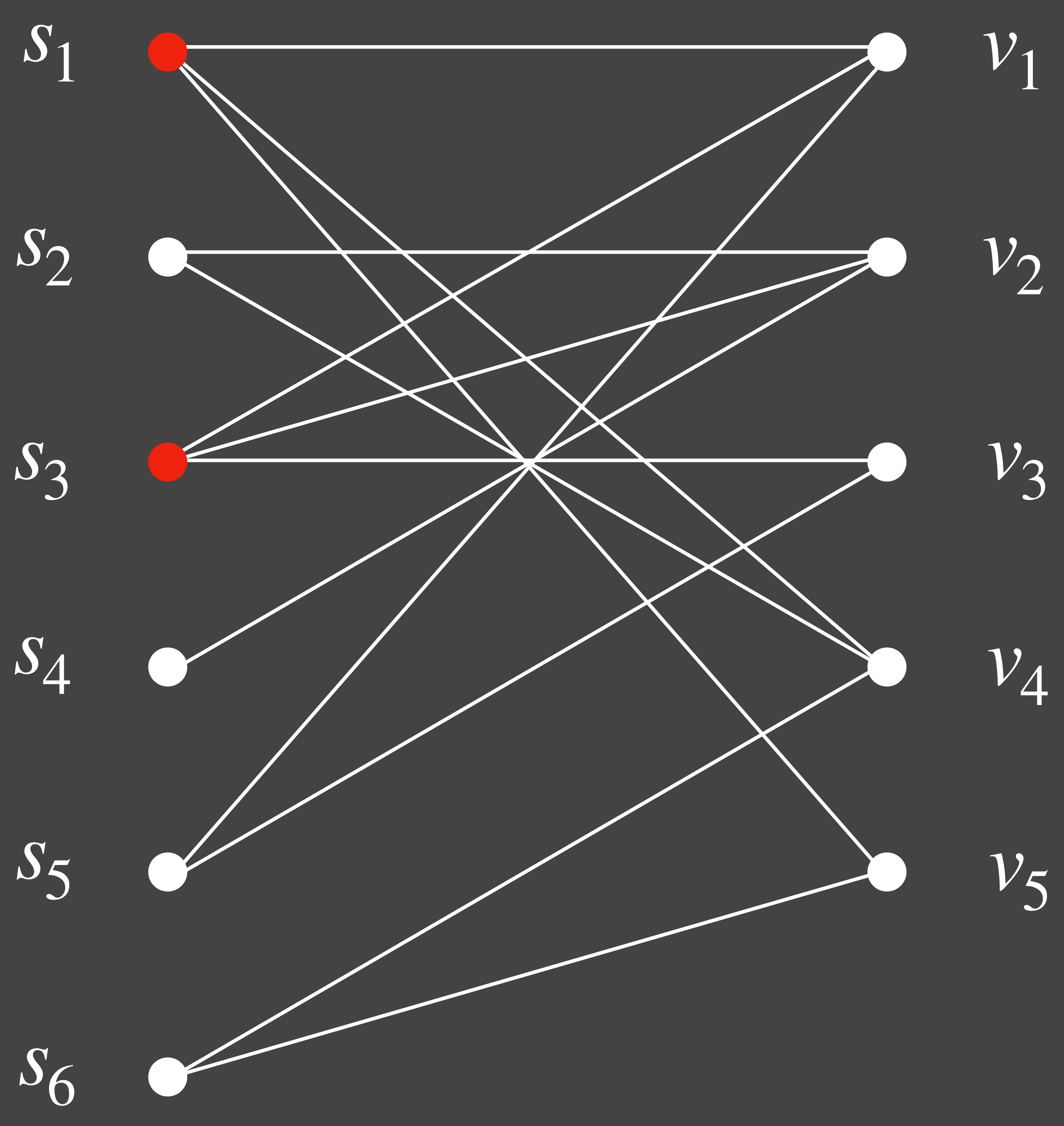
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



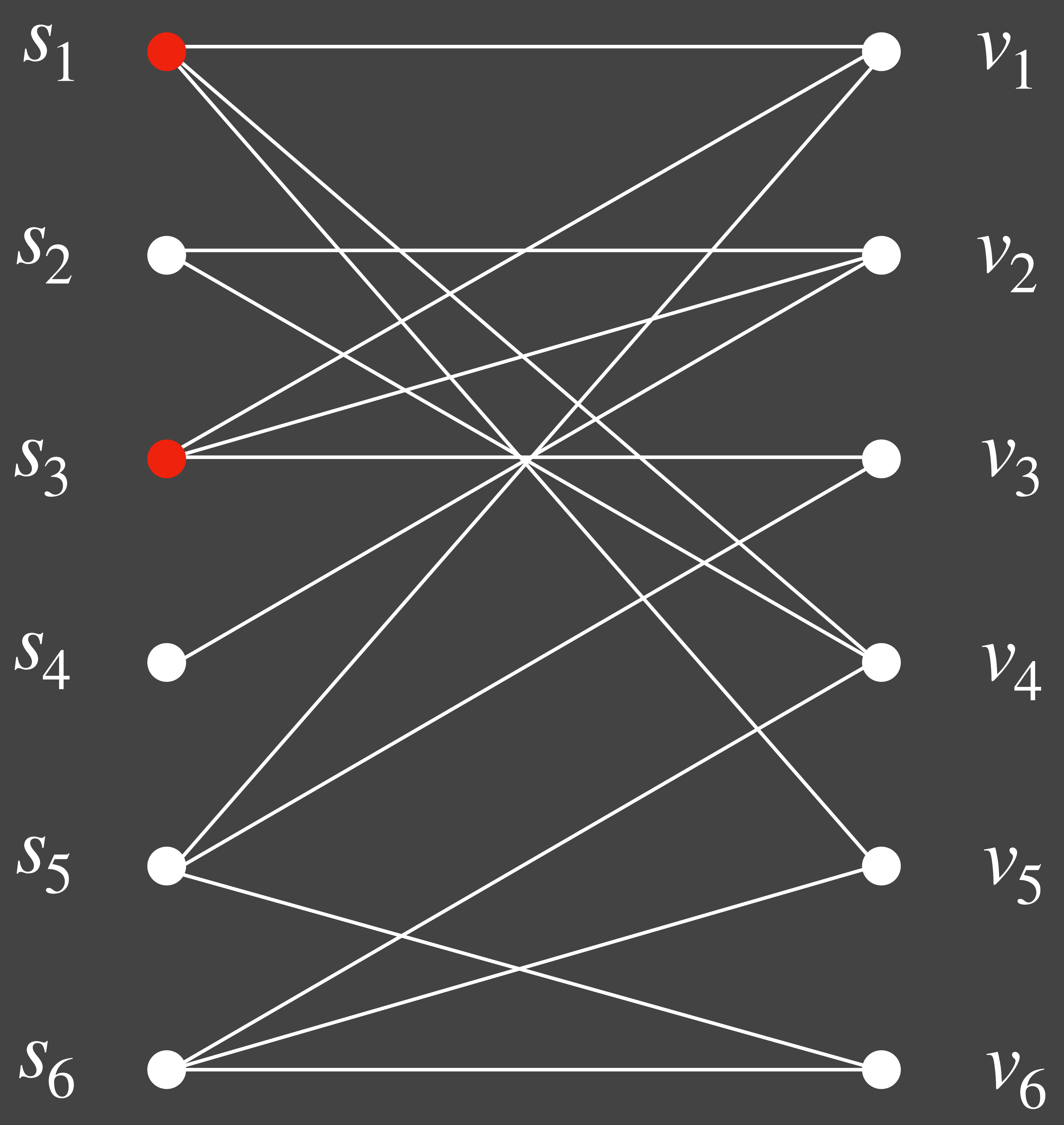
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



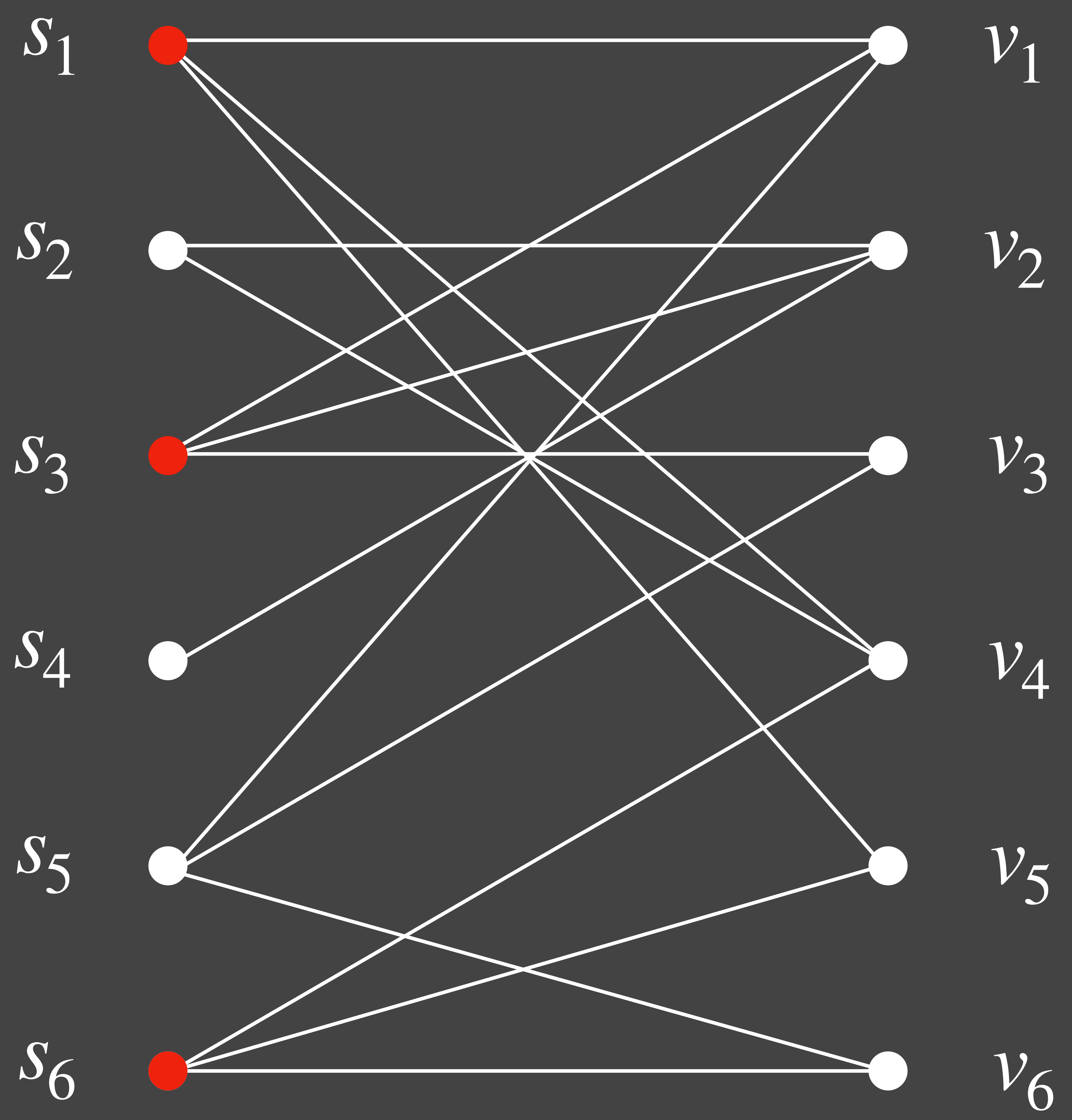
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\text{OPT})$

1. Competitive recourse gives instance optimal guarantee.



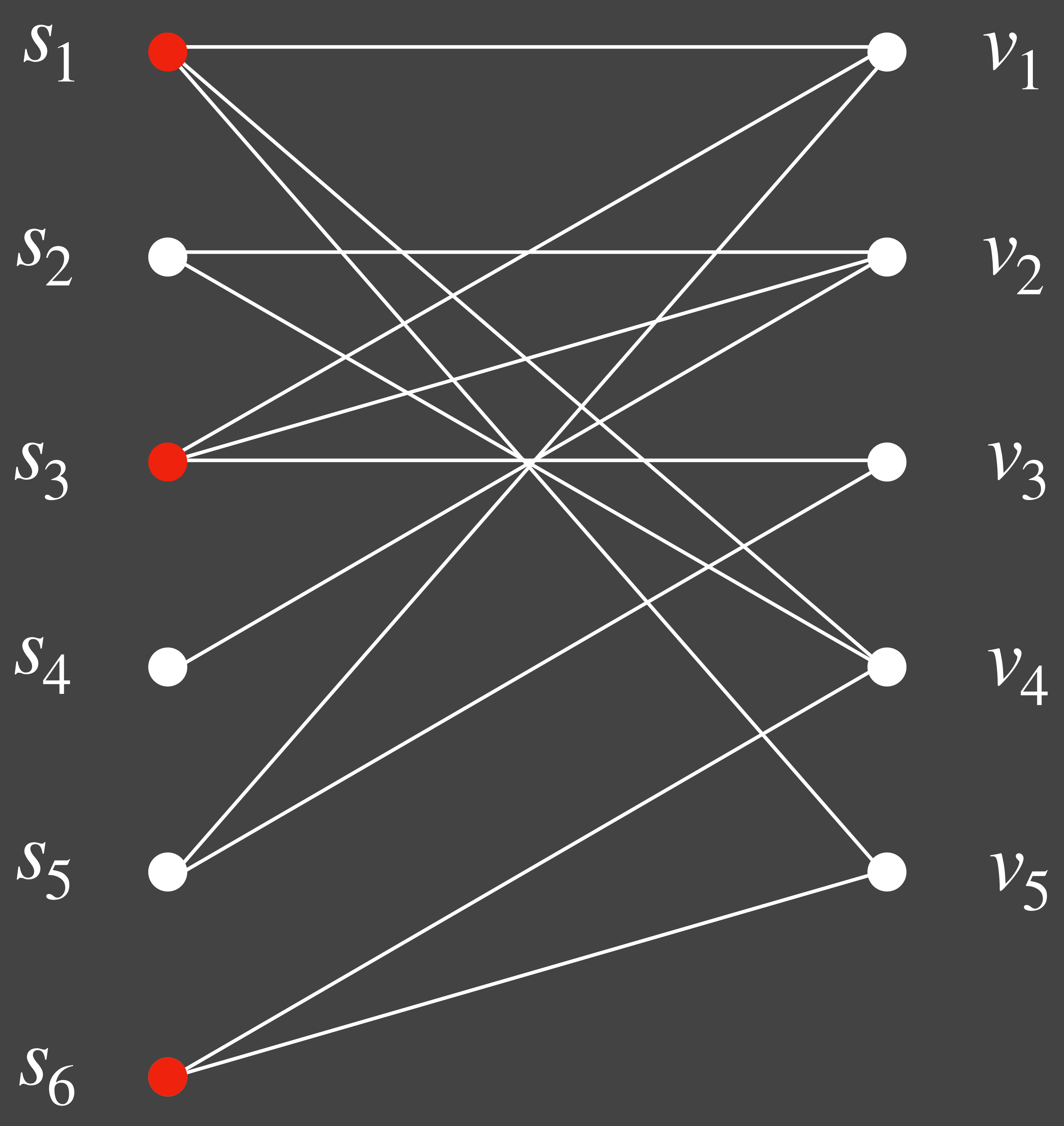
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathsf{OPT})$

1. Competitive recourse gives instance optimal guarantee.



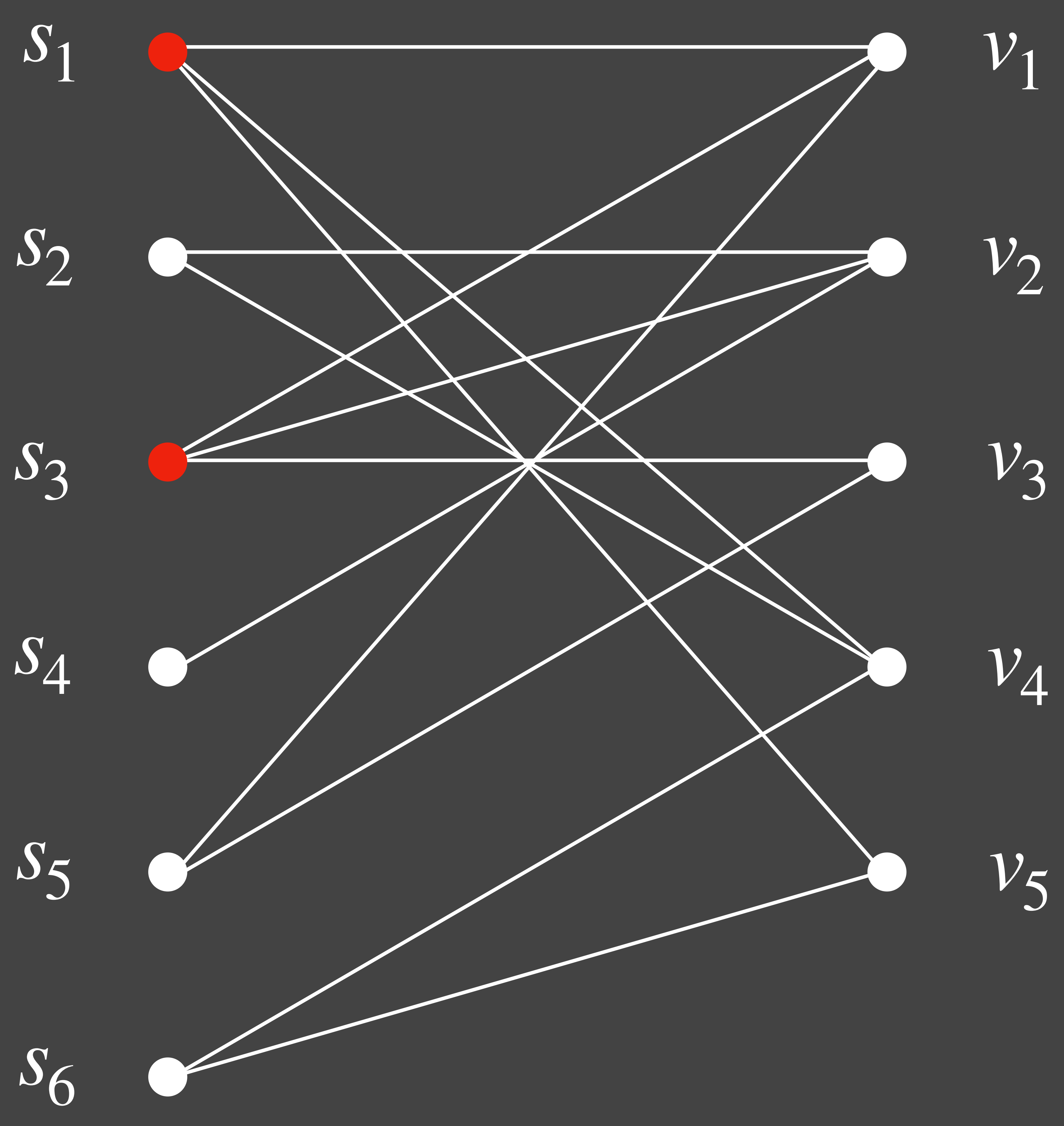
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



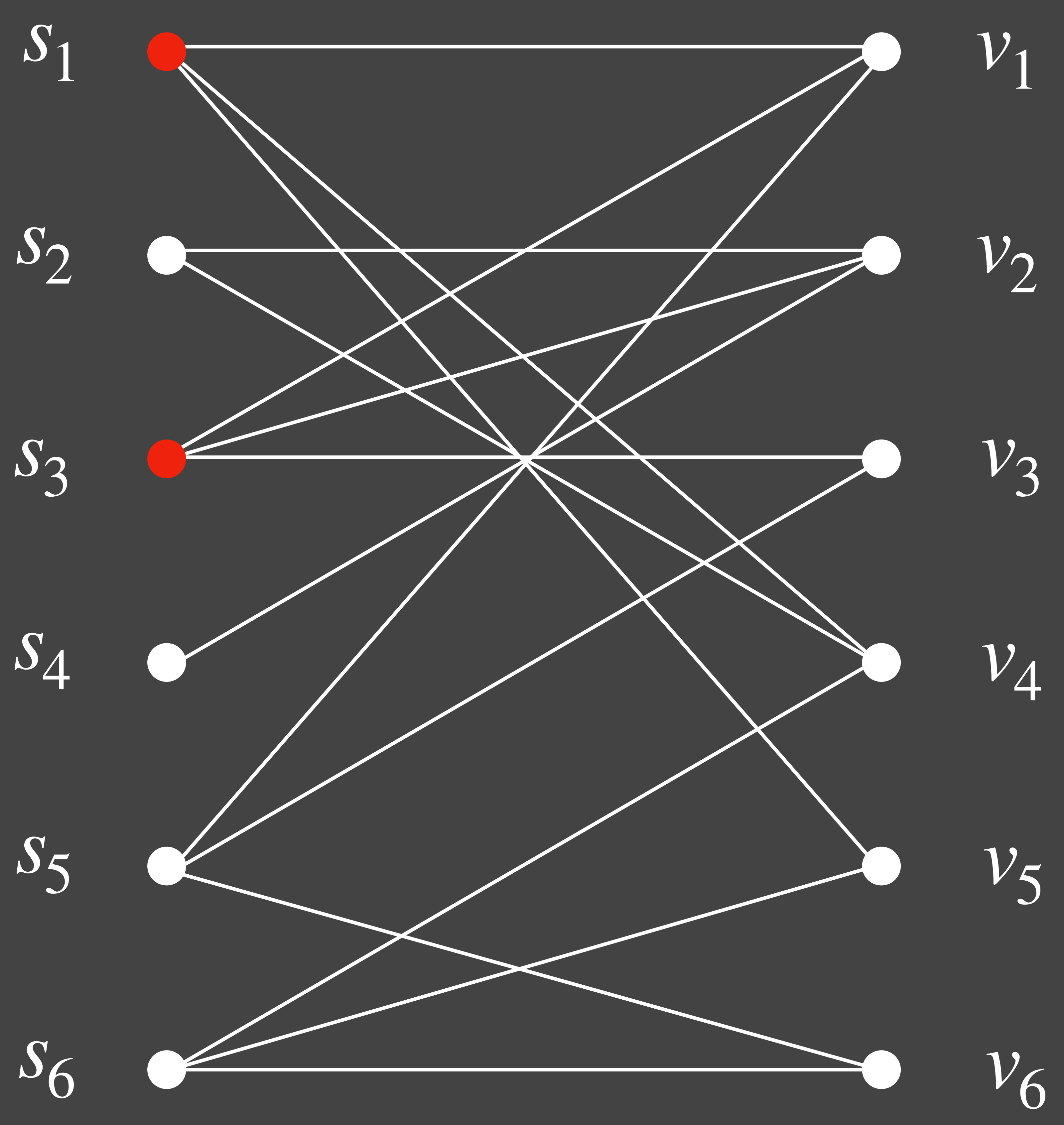
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathsf{OPT})$

1. Competitive recourse gives instance optimal guarantee.



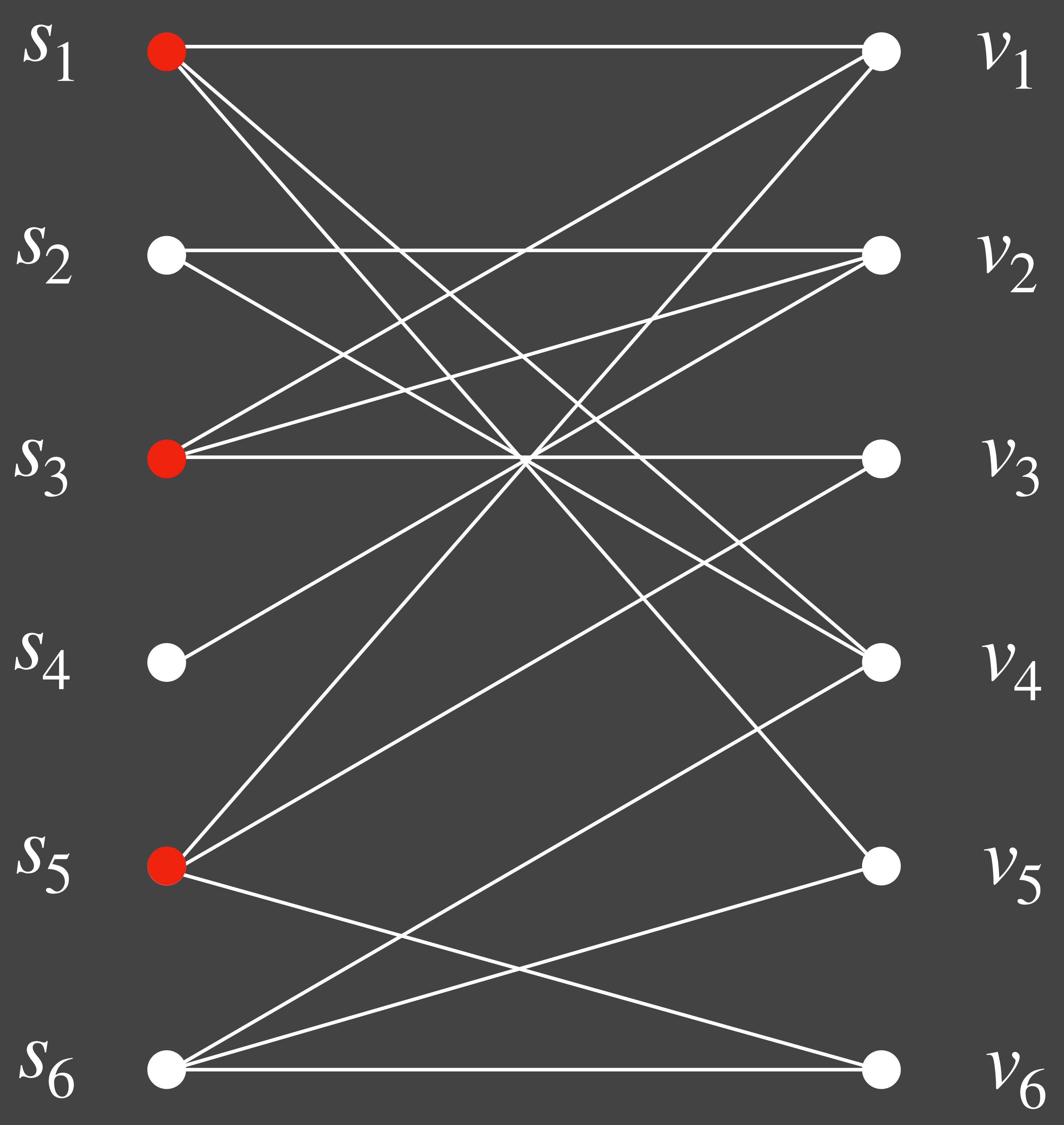
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



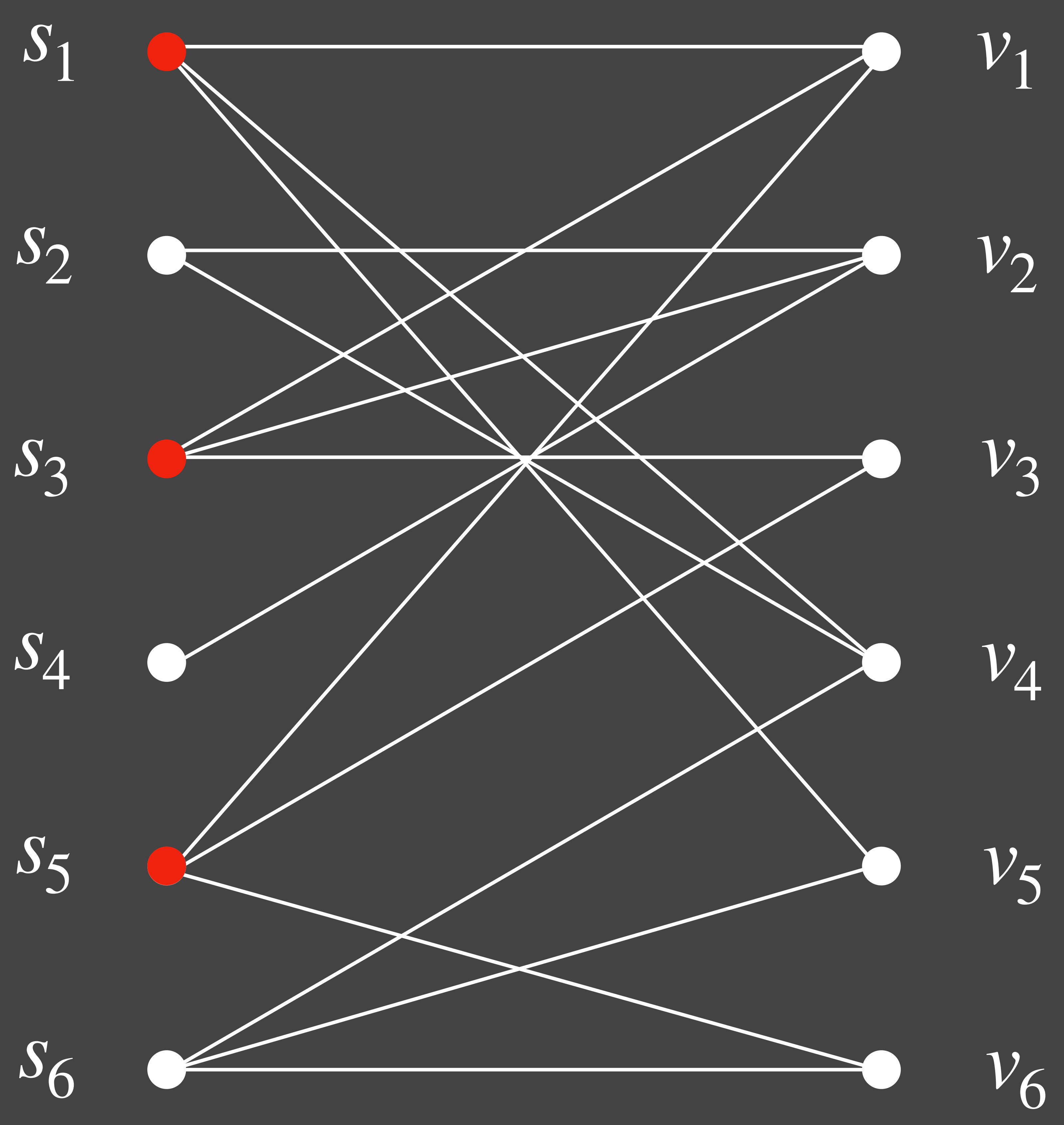
Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\text{OPT})$

1. Competitive recourse gives instance optimal guarantee.



Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

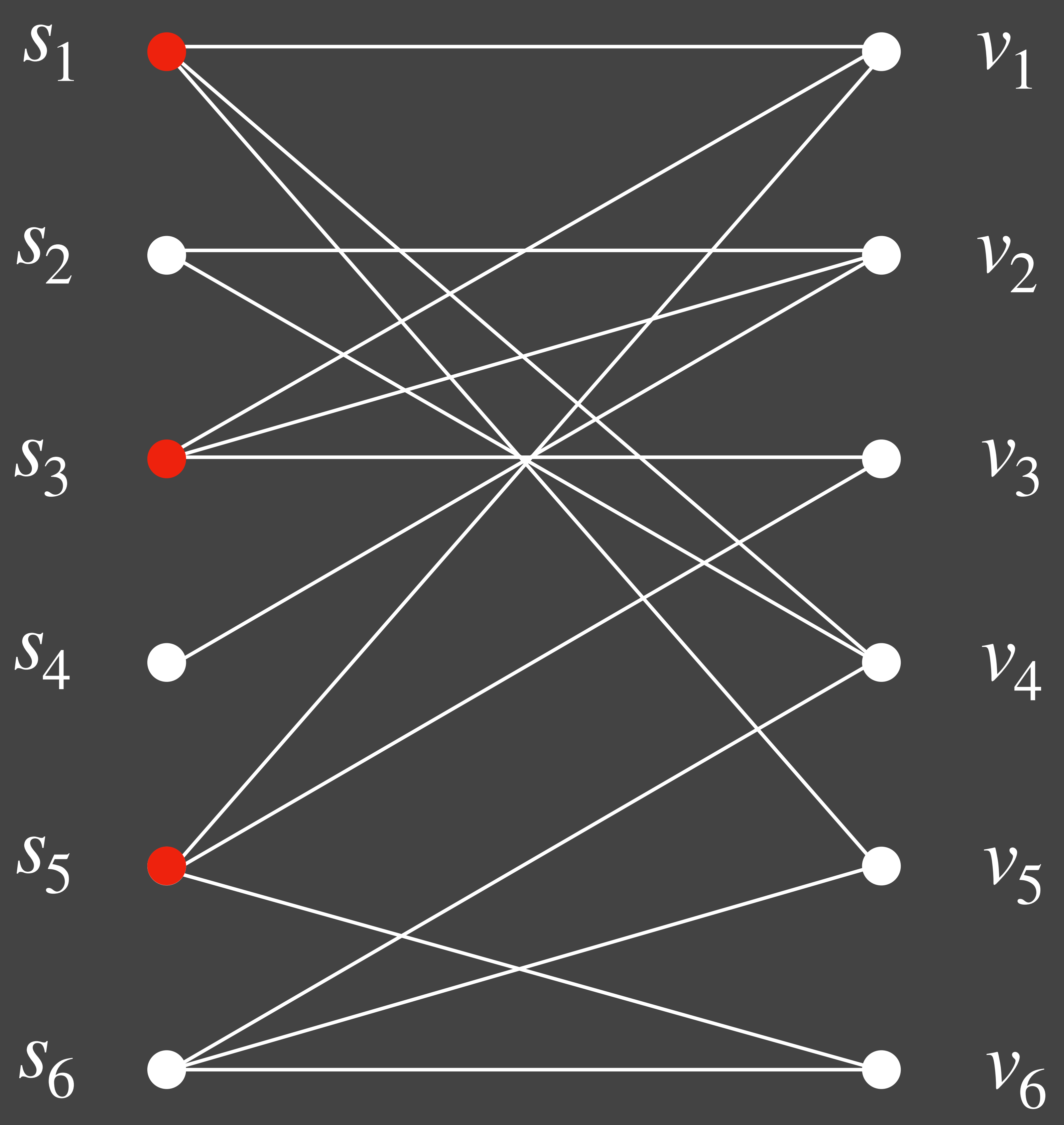
1. Competitive recourse gives instance optimal guarantee.



2. Sometimes absolute recourse impossible.

Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\text{OPT})$

1. Competitive recourse gives instance optimal guarantee.

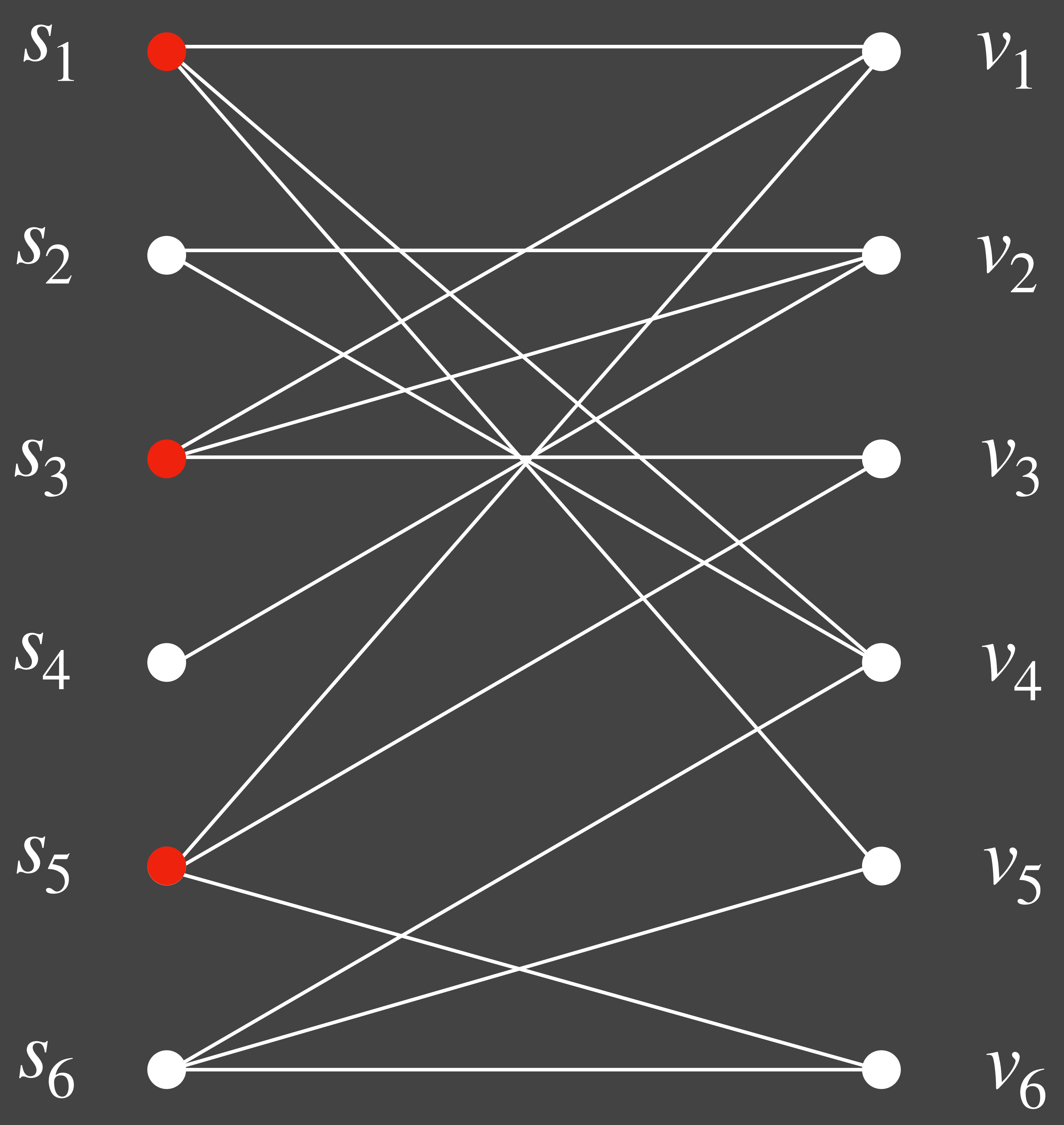


2. Sometimes absolute recourse impossible.

[KLS 23] show any $o(\log)$ approx. for fully-dynamic load balancing needs poly$(n) \cdot T$ recourse:

Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



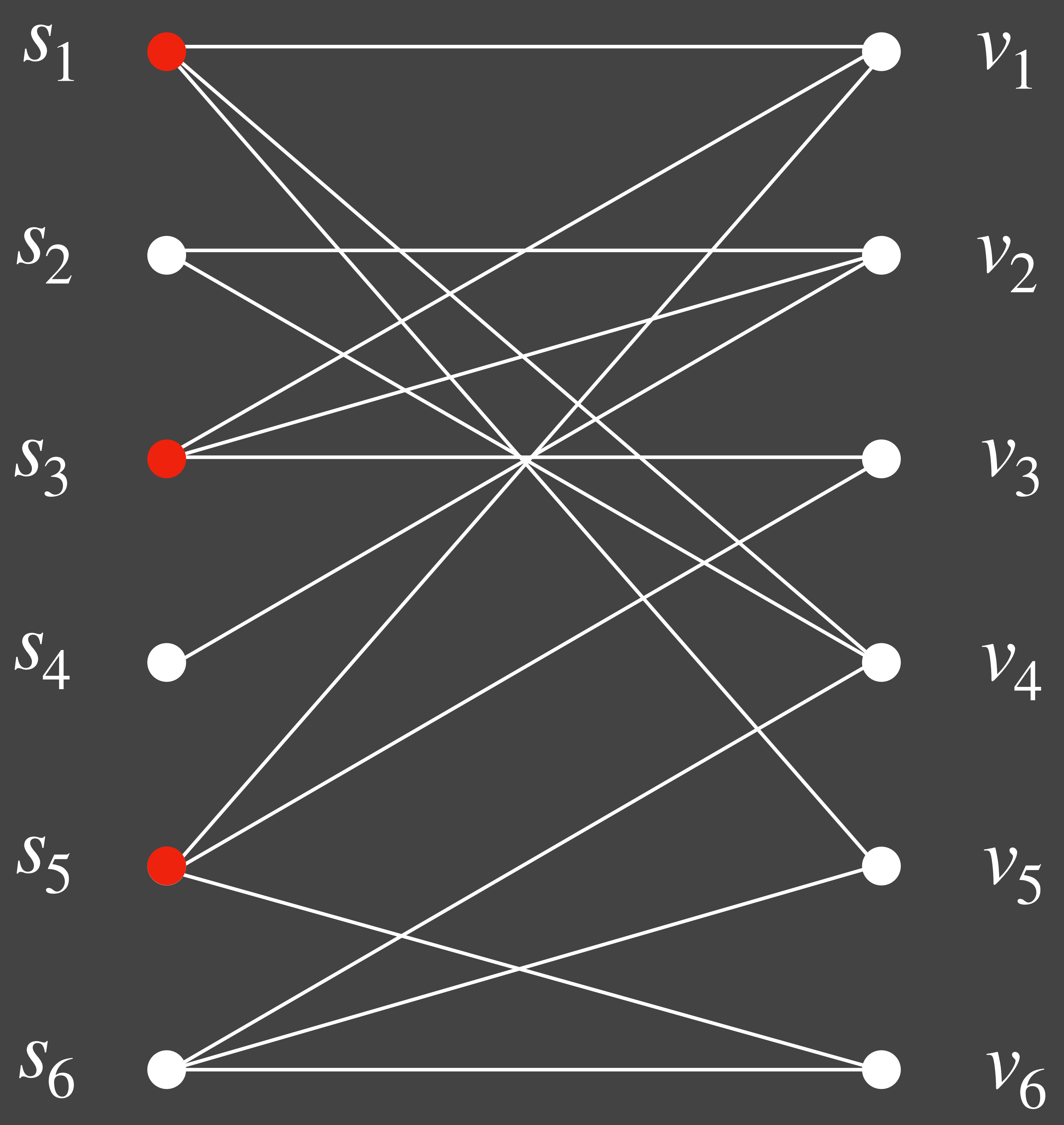
2. Sometimes absolute recourse impossible.

[KLS 23] show any $o(\log)$ approx. for fully-dynamic load balancing needs poly$(n) \cdot T$ recourse:

> *"To circumvent the negative result, one needs to consider a different measurement for recourse for the fully dynamic model."*

Absolute vs Competitive Recourse: $O(T)$ vs $\tilde{O}(\mathrm{OPT})$

1. Competitive recourse gives instance optimal guarantee.



2. Sometimes absolute recourse impossible.

[KLS 23] show any $o(\log)$ approx. for fully-dynamic load balancing needs poly$(n) \cdot T$ recourse:

> *"To circumvent the negative result, one needs to consider a different measurement for recourse for the fully dynamic model."*

Competitive recourse is the answer!

The Fractional Algorithm

Step 0: Reduction to Chasing Halfspaces

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Step 0: Reduction to Chasing Halfspaces

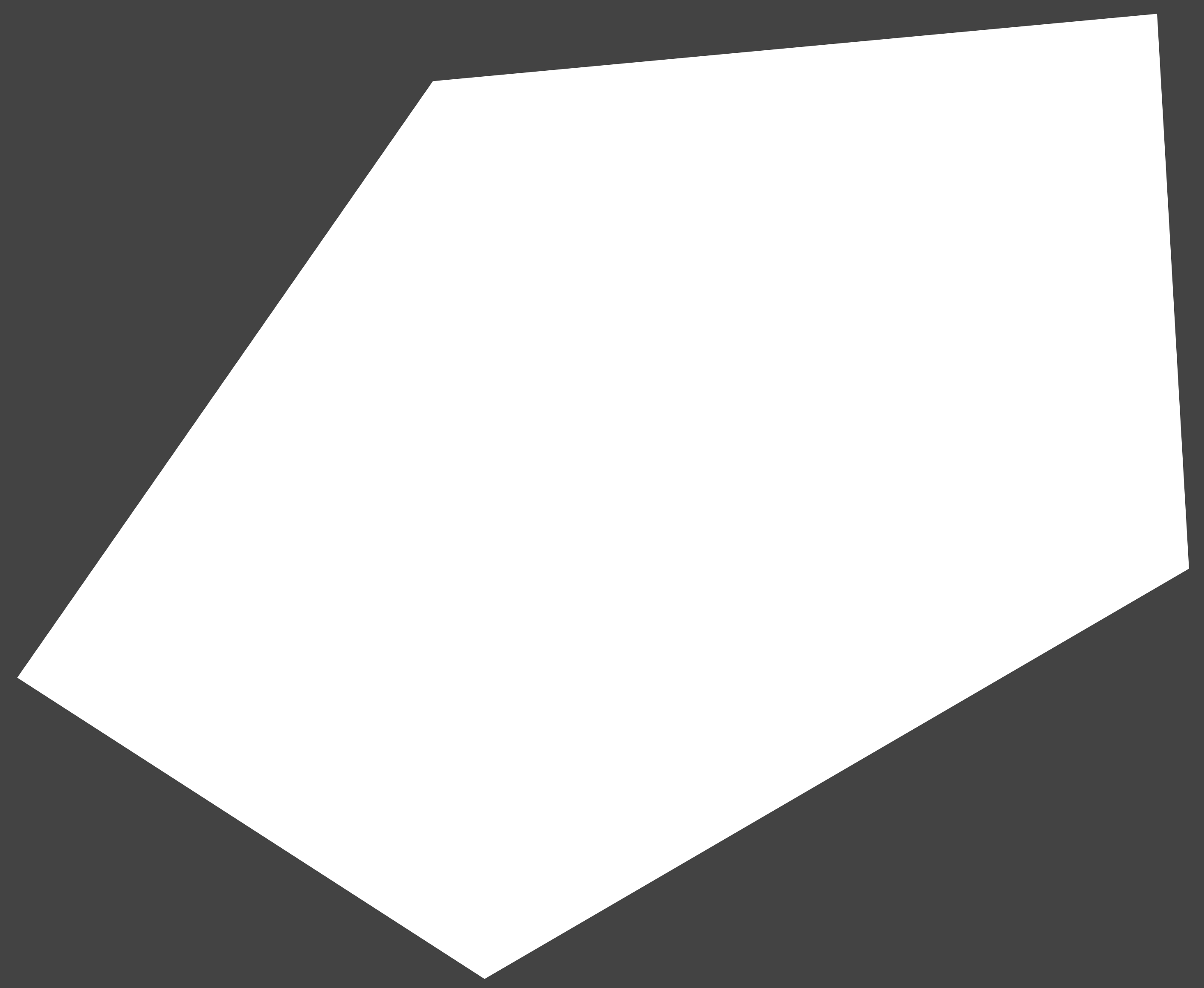
Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

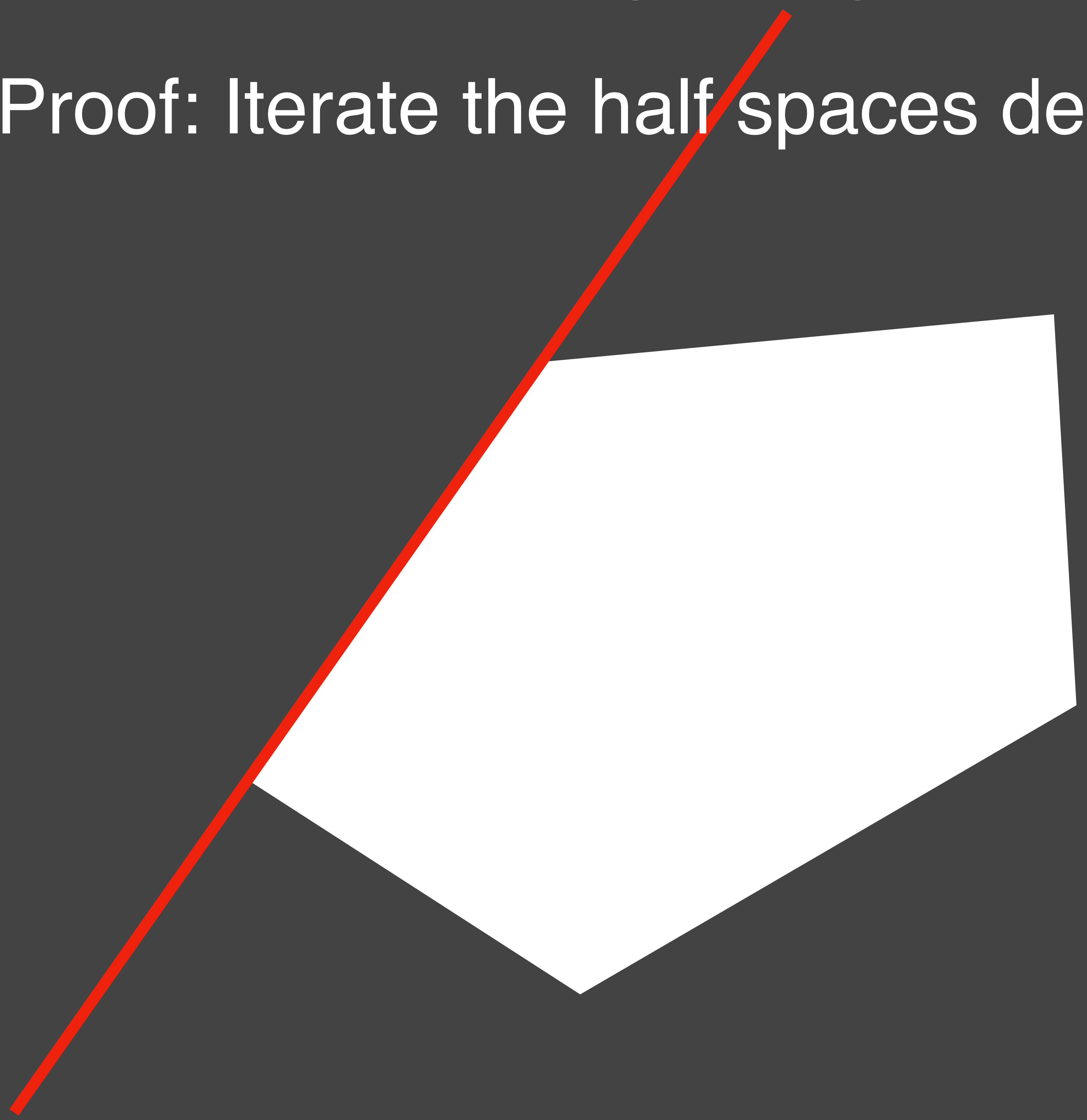
Proof: Iterate the half spaces defining the body $L \to \infty$ times.



Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

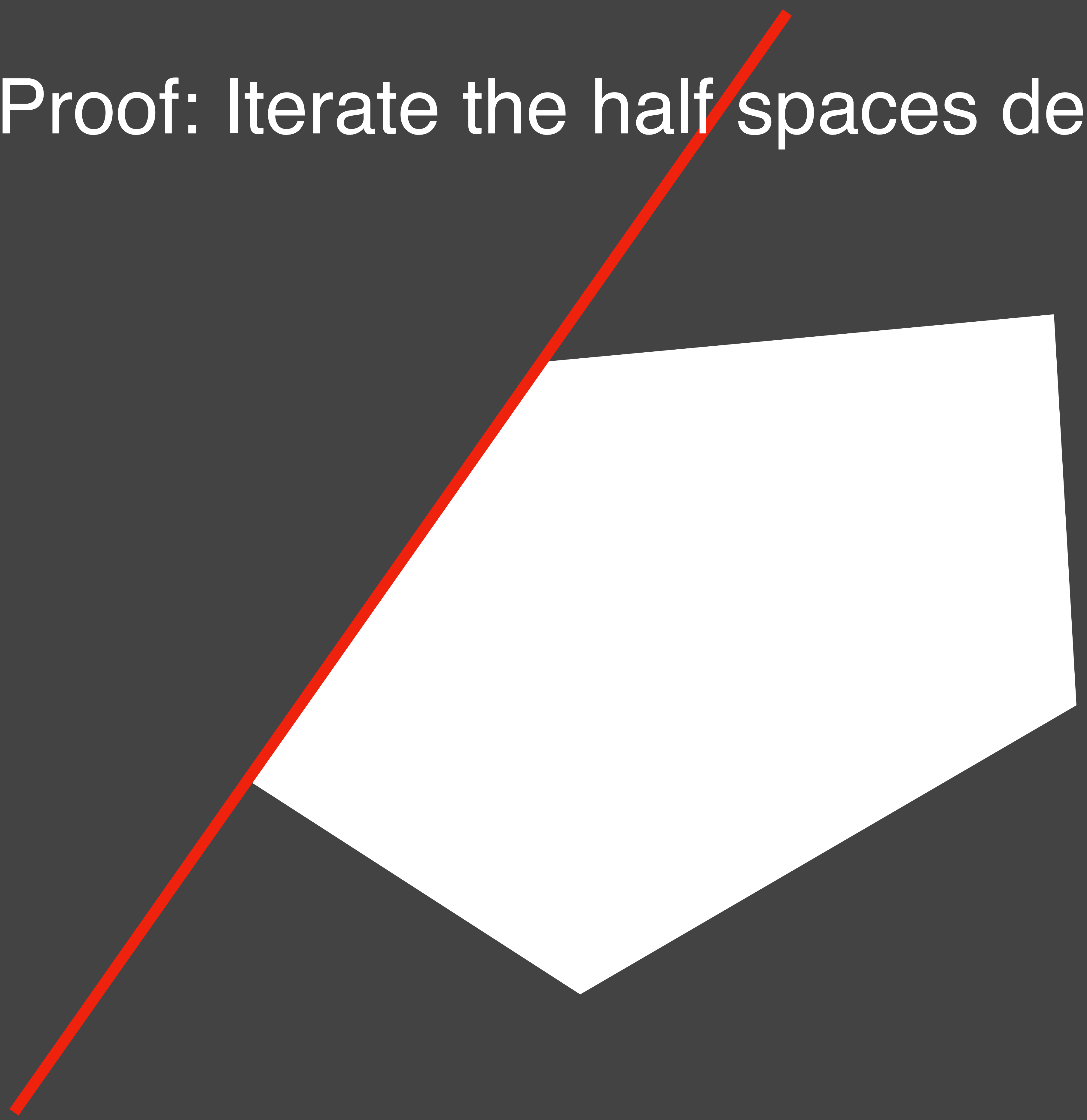
Proof: Iterate the half spaces defining the body $L \to \infty$ times.



Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.

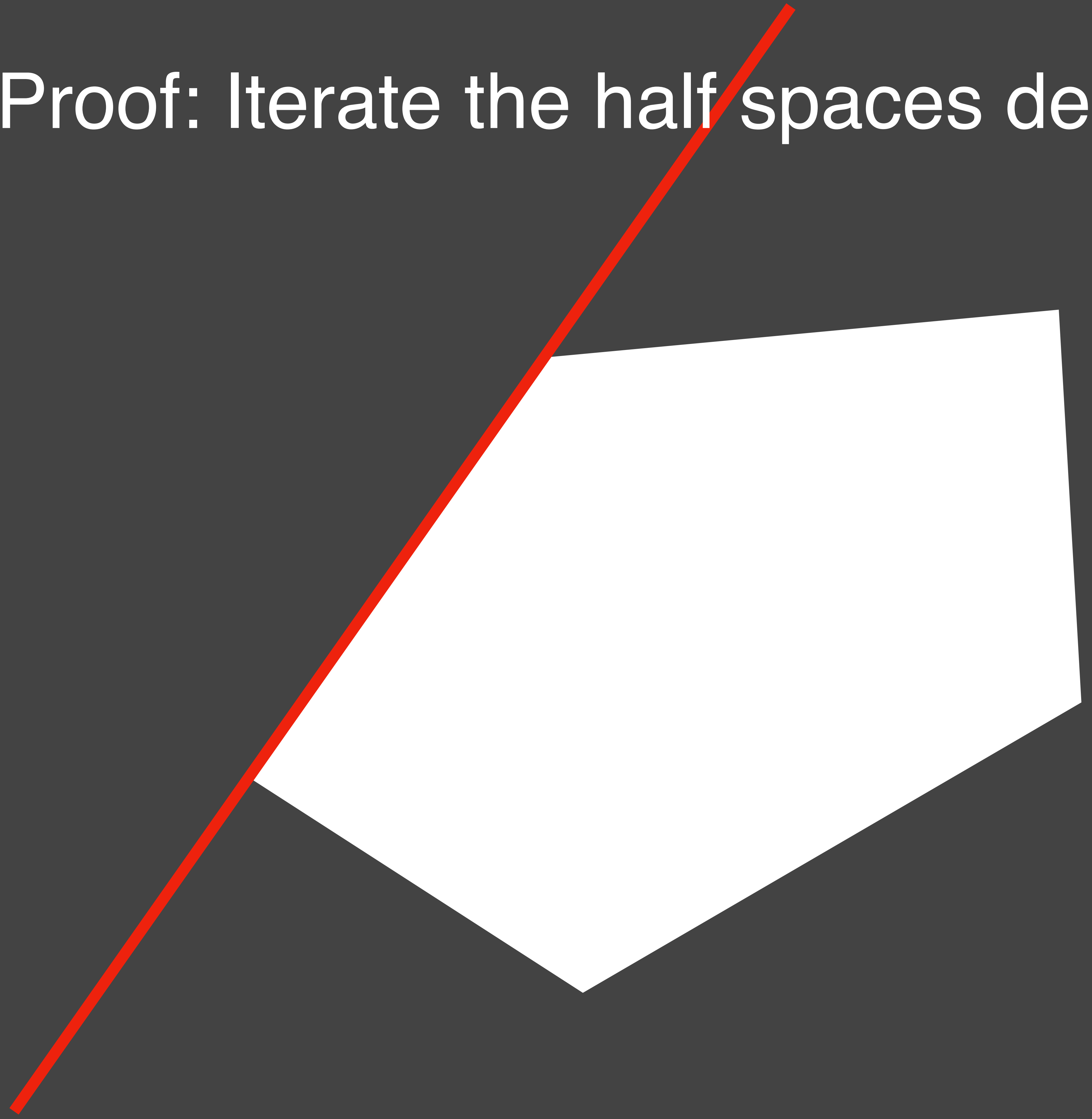


OPT feasible to halfspaces & finite.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.



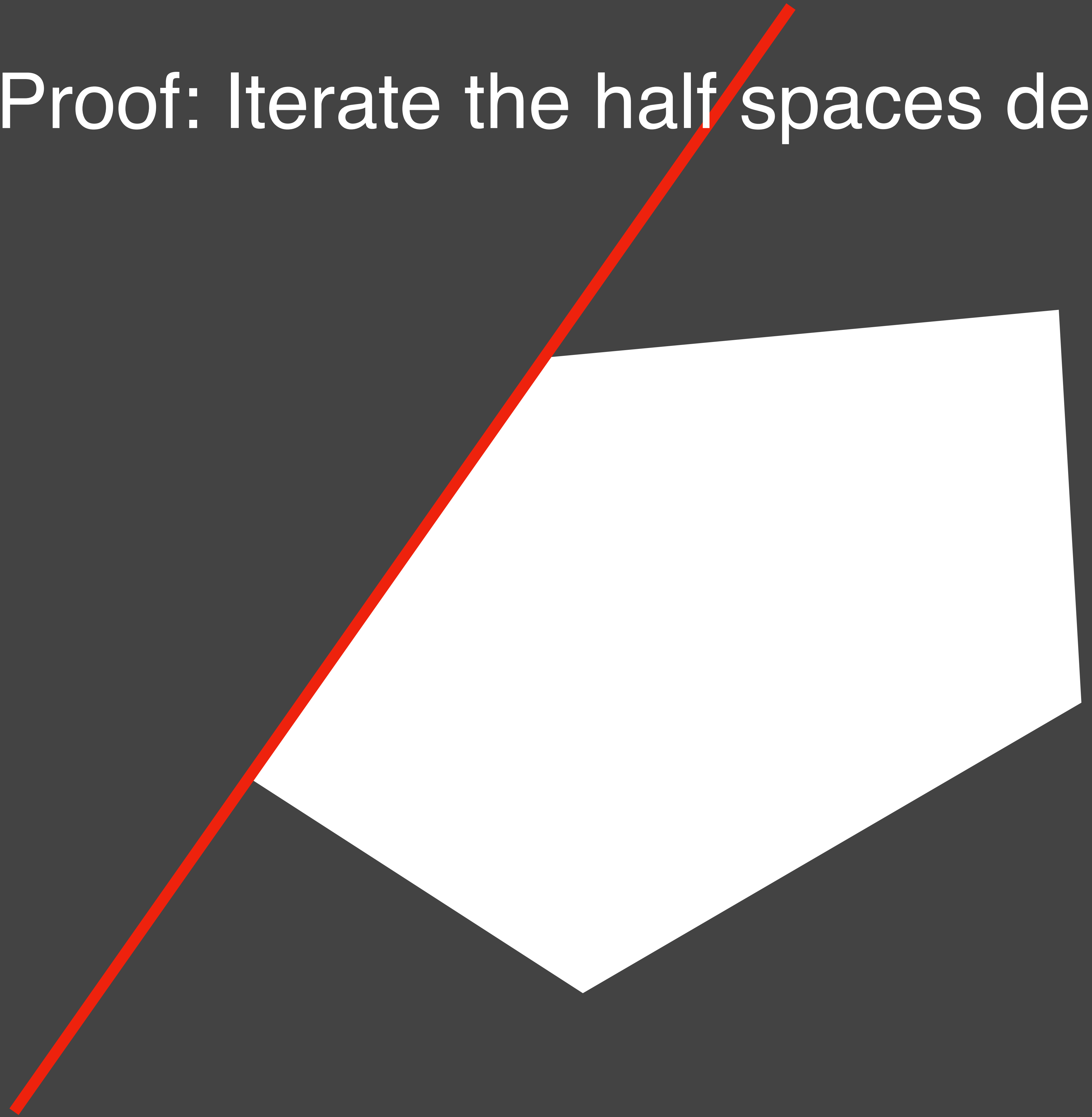
OPT feasible to halfspaces & finite.

If ALG stays δ far from polytope,
it pays $\delta \cdot L \to \infty$.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.



OPT feasible to halfspaces & finite.

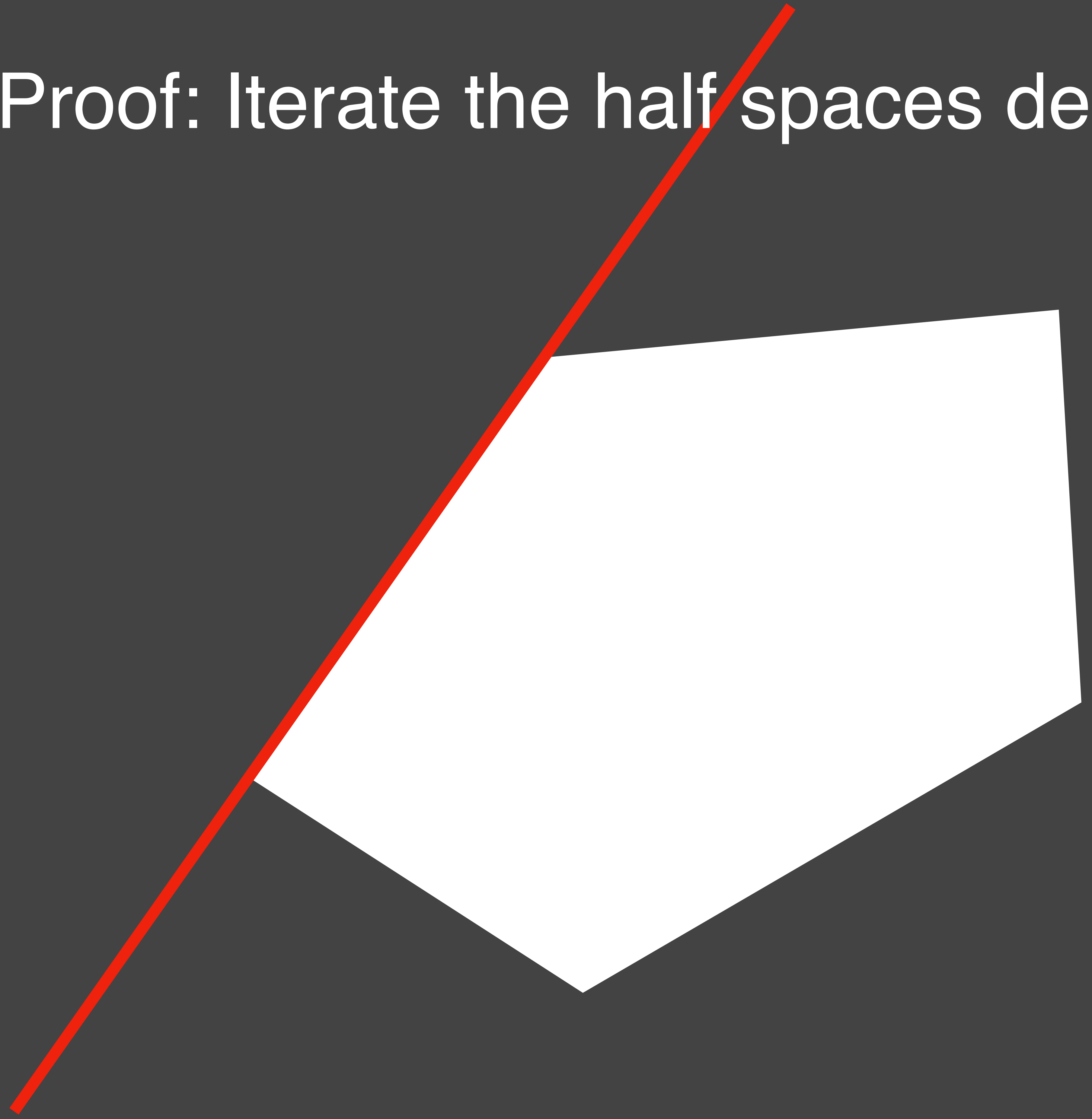
If ALG stays δ far from polytope,
it pays $\delta \cdot L \to \infty$.

If ALG competitive for halfspaces.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.



OPT feasible to halfspaces & finite.

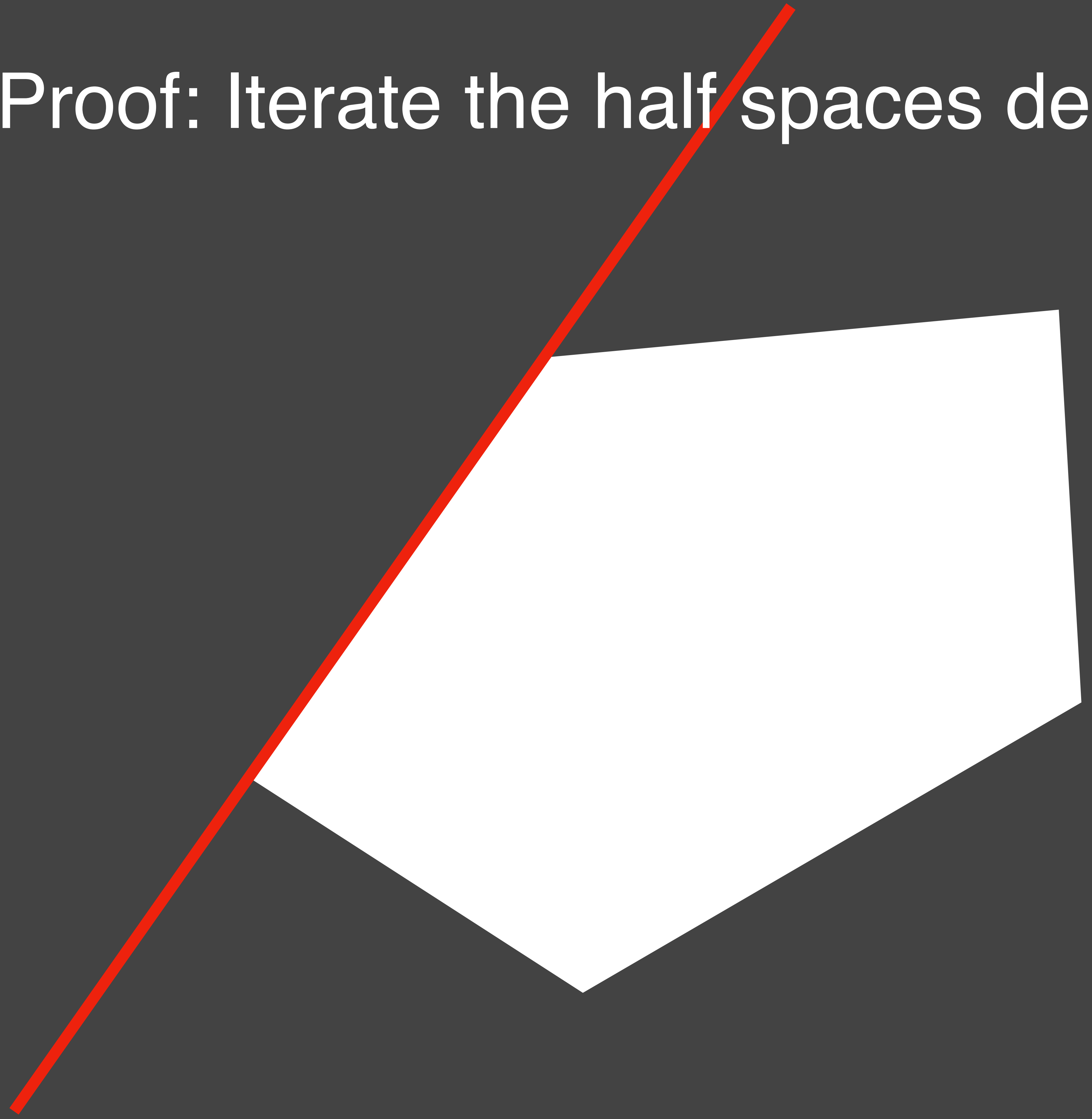
If ALG stays δ far from polytope,
it pays $\delta \cdot L \to \infty$.

If ALG competitive for halfspaces.
$\Rightarrow$ ALG converges to polytope.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.



OPT feasible to halfspaces & finite.

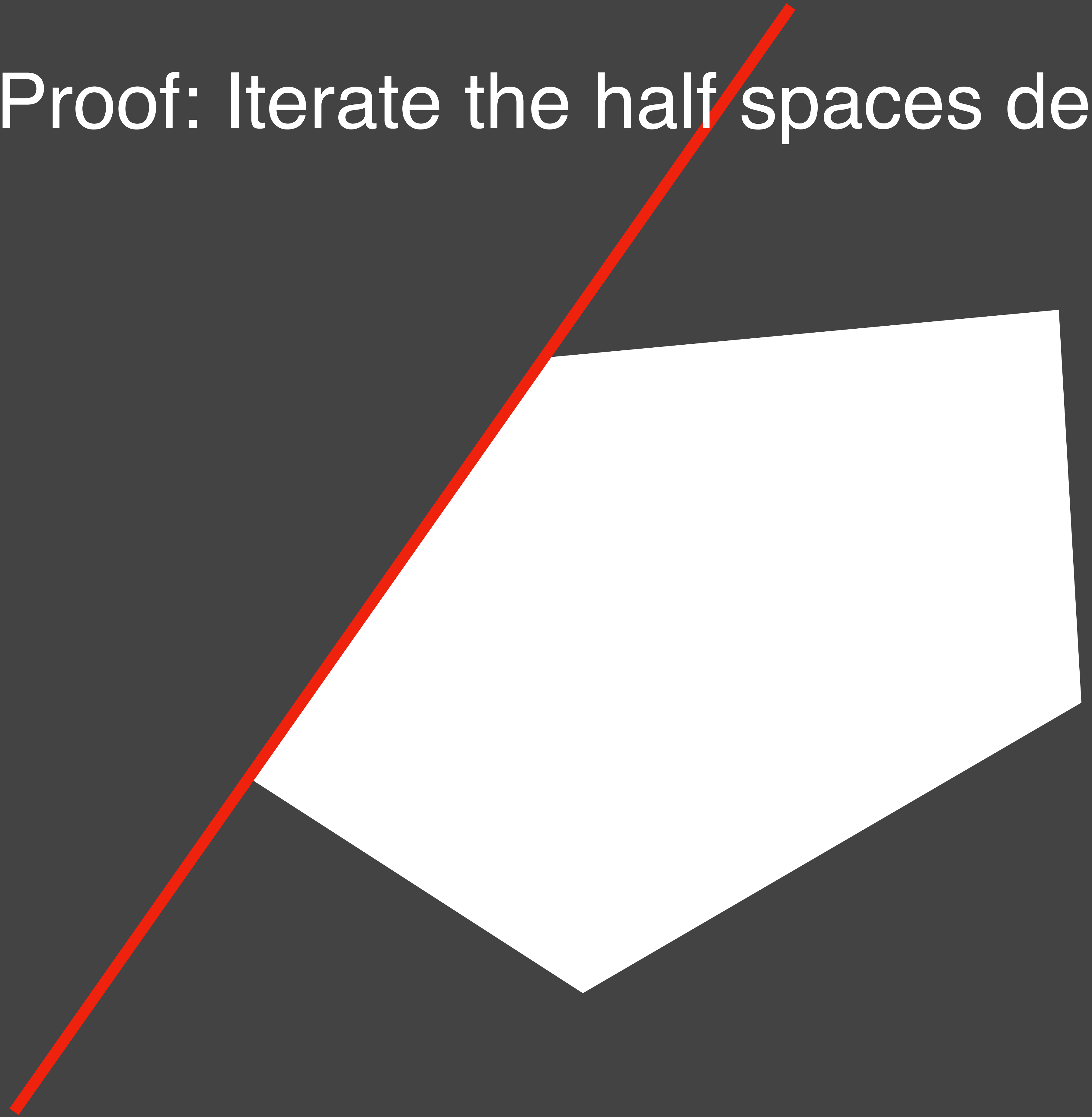
If ALG stays δ far from polytope,
it pays $\delta \cdot L \to \infty$.

If ALG competitive for halfspaces.
$\Rightarrow$ ALG converges to polytope.
$\Rightarrow$ ALG competitive for bodies.

Step 0: Reduction to Chasing Halfspaces

Claim: Suffices to give algorithm for chasing halfspaces.

Proof: Iterate the half spaces defining the body $L \to \infty$ times.



OPT feasible to halfspaces & finite.

If ALG stays δ far from polytope,
it pays $\delta \cdot L \to \infty$.

If ALG competitive for halfspaces.
$\Rightarrow$ ALG converges to polytope.
$\Rightarrow$ ALG competitive for bodies. $\blacksquare$

Step 0: Reduction to Chasing Halfspaces

Step 0: Reduction to Chasing Halfspaces

From now on, we assume every body is either covering halfspace

Step 0: Reduction to Chasing Halfspaces

From now on, we assume every body is either covering halfspace

$$x_1 + x_3 + x_7 \geq 1$$

Step 0: Reduction to Chasing Halfspaces

From now on, we assume every body is either covering halfspace

$$x_1 + x_3 + x_7 \geq 1$$

or packing halfspace

Step 0: Reduction to Chasing Halfspaces

From now on, we assume every body is either covering halfspace

$$x_1 + x_3 + x_7 \geq 1$$

or packing halfspace

$$x_2 + x_3 + x_4 \leq 1.$$

Step 0: Reduction to Chasing Halfspaces

From now on, we assume every body is either covering halfspace

$$x_1 + x_3 + x_7 \geq 1$$

or packing halfspace

$$x_2 + x_3 + x_4 \leq 1.$$

I.e. all coefficients are positive, variables on same side of $\leq$.

The Algorithm for Halfspaces

The Algorithm for Halfspaces

When covering constraint arrives

The Algorithm for Halfspaces

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \textbf{s.t.} \\[1em] \langle c^t, z\rangle \geq 1 \qquad (y^t) \\[1em] x \geq 0 \end{array}}$$

The Algorithm for Halfspaces

$$\left(\mathrm{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i \right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathrm{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.} \\ \langle c^t, z \rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

The Algorithm for Halfspaces

$$\left(\text{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i \right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min\limits_z \ \ \textbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \textbf{s.t.} \\ \\ \langle c^t, z \rangle \geq 1 \qquad (y^t) \\ \\ x \geq 0 \end{array}}$$

(for some small δ)

The Algorithm for Halfspaces

$\left(\text{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i \right)$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \textbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \textbf{s.t.} \\ \langle c^t, z \rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

(for some small δ)

When packing constraint arrives

The Algorithm for Halfspaces

$$\left(\mathsf{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i \right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{aligned} &\min_z \ \mathsf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \textsf{s.t.} \\ &\langle c^t, z \rangle \geq 1 \qquad (y^t) \\ &x \geq 0 \end{aligned}}$$

(for some small δ)

When packing constraint arrives

$$x^t = \boxed{\begin{aligned} &\min_z \ \mathsf{KL}(z \,||\, x^{t-1}) \quad \textsf{s.t.} \\ &\langle p^t, z \rangle \leq 1 + \epsilon \qquad (z^t) \\ &x \geq 0 \end{aligned}}$$

The Algorithm for Halfspaces

$$\left(\mathsf{KL}(a\,||\,b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i\right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathbf{KL}(z+\delta\,||\,x^{t-1}+\delta) \quad \textbf{s.t.} \\ \langle c^t, z\rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

(for some small δ)

By KKT:

When packing constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathbf{KL}(z\,||\,x^{t-1}) \quad \textbf{s.t.} \\ \langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t) \\ x \geq 0 \end{array}}$$

The Algorithm for Halfspaces

$$\left(\mathbf{KL}(a\,||\,b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i\right)$$

When covering constraint arrives

$$x^t = \begin{array}{|c|} \hline \min_z \ \mathbf{KL}(z+\delta\,||\,x^{t-1}+\delta) \quad \text{s.t.} \\ \langle c^t, z\rangle \geq 1 \qquad (y^t) \\ x \geq 0 \\ \hline \end{array}$$

(for some small δ)

By KKT:

$$\log \frac{x^t+\delta}{x^{t-1}+\delta} = c_i^t y^t$$

When packing constraint arrives

$$x^t = \begin{array}{|c|} \hline \min_z \ \mathbf{KL}(z\,||\,x^{t-1}) \quad \text{s.t.} \\ \langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t) \\ x \geq 0 \\ \hline \end{array}$$

The Algorithm for Halfspaces

$$\left(\mathsf{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i \right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathsf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \textsf{s.t.} \\ \langle c^t, z \rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

(for some small δ)

By KKT:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$x^t + \delta \leftarrow (x^{t-1} + \delta) \cdot e^{c_i^t y^t}$$

When packing constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathsf{KL}(z \,||\, x^{t-1}) \quad \textsf{s.t.} \\ \langle p^t, z \rangle \leq 1 + \epsilon \qquad (z^t) \\ x \geq 0 \end{array}}$$

The Algorithm for Halfspaces

$$\left(\text{KL}(a\,||\,b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i\right)$$

When covering constraint arrives

$$x^t = \begin{array}{l} \min_z \ \textbf{KL}(z + \delta\,||\,x^{t-1} + \delta) \quad \textbf{s.t.} \\ \qquad \langle c^t, z \rangle \geq 1 \qquad (y^t) \\ \qquad x \geq 0 \end{array}$$

(for some small δ)

By KKT:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$x^t + \delta \leftarrow (x^{t-1} + \delta) \cdot e^{c_i^t y^t}$$

When packing constraint arrives

$$x^t = \begin{array}{l} \min_z \ \textbf{KL}(z\,||\,x^{t-1}) \quad \textbf{s.t.} \\ \qquad \langle p^t, z \rangle \leq 1 + \epsilon \qquad (z^t) \\ \qquad x \geq 0 \end{array}$$

$$\log \frac{x^t}{x^{t-1}} = -\,p_i^t z^t$$

The Algorithm for Halfspaces

$$\left(\mathsf{KL}(a\,||\,b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i\right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathsf{KL}(z+\delta\,||\,x^{t-1}+\delta) \quad \text{s.t.} \\ \langle c^t, z\rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

(for some small δ)

By KKT:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$x^t + \delta \leftarrow (x^{t-1} + \delta) \cdot e^{c_i^t y^t}$$

When packing constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathsf{KL}(z\,||\,x^{t-1}) \quad \text{s.t.} \\ \langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t) \\ x \geq 0 \end{array}}$$

$$\log \frac{x^t}{x^{t-1}} = -\,p_i^t z^t$$

$$x^t \leftarrow x^{t-1} \cdot e^{-p_i^t z^t}$$

The Algorithm for Halfspaces

$$\left(\mathsf{KL}(a\,||\,b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i\right)$$

When covering constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathbf{KL}(z+\delta\,||\,x^{t-1}+\delta) \quad \text{s.t.} \\ \langle c^t, z\rangle \geq 1 \qquad (y^t) \\ x \geq 0 \end{array}}$$

(for some small δ)

When packing constraint arrives

$$x^t = \boxed{\begin{array}{c} \min_z \ \mathbf{KL}(z\,||\,x^{t-1}) \quad \text{s.t.} \\ \langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t) \\ x \geq 0 \end{array}}$$

By KKT:

$$\log \frac{x^t+\delta}{x^{t-1}+\delta} = c_i^t y^t \qquad\qquad \log \frac{x^t}{x^{t-1}} = -\,p_i^t z^t$$

$$x^t + \delta \leftarrow (x^{t-1}+\delta)\cdot e^{c_i^t y^t} \qquad\qquad x^t \leftarrow x^{t-1}\cdot e^{-p_i^t z^t}$$

Multiplicative weights update (almost)!

How to analyze? The Primal-Dual Method!

How to analyze? The Primal-Dual Method!

(Upwards) ℓ_1 movement can be written as an LP!

How to analyze? The Primal-Dual Method!

(Upwards) ℓ_1 movement can be written as an LP!

(P)

$$\min \sum_t \sum_i \ell_i^t \quad \text{s.t.}$$

$$\forall t \in C: \qquad \langle c^t, x^t \rangle \geq 1$$

$$\forall t \in P: \qquad \langle p^t, x^t \rangle \leq 1$$

$$\forall i, t: \qquad x_i^t - x_i^{t-1} \leq \ell_i^t$$

$$x \geq 0$$

How to analyze? The Primal-Dual Method!

(Upwards) ℓ_1 movement can be written as an LP!

(P)

$$\min \sum_t \sum_i \ell_i^t \quad \text{s.t.}$$

$$\forall t \in C: \qquad \langle c^t, x^t \rangle \geq 1$$

$$\forall t \in P: \qquad \langle p^t, x^t \rangle \leq 1$$

$$\forall i, t: \qquad x_i^t - x_i^{t-1} \leq \ell_i^t$$

$$x \geq 0$$

(D)

$$\max \sum_{t \in C} y^t - \sum_{t \in P} z^t \quad \text{s.t.}$$

$$\forall i,\ t \in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \qquad 0 \leq r_i^t \leq 1$$

$$y, z \geq 0$$

How to analyze? The Primal-Dual Method!

(Upwards) ℓ_1 movement can be written as an LP!

(P)

$$\min \sum_t \sum_i \ell_i^t \quad \text{s.t.}$$

$$\forall t \in C: \qquad \langle c^t, x^t \rangle \geq 1$$

$$\forall t \in P: \qquad \langle p^t, x^t \rangle \leq 1$$

$$\forall i, t: \qquad x_i^t - x_i^{t-1} \leq \ell_i^t$$

$$x \geq 0$$

(D)

$$\max \sum_{t \in C} y^t - \sum_{t \in P} z^t \quad \text{s.t.}$$

$$\forall i,\ t \in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \qquad 0 \leq r_i^t \leq 1$$

$$y, z \geq 0$$

We fit a dual to ALG's solution! How?

How to analyze? The Primal-Dual Method!

$$\min_z \ \textbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \textbf{s.t.}$$

$$\langle c^t, z \rangle \geq 1 \qquad (y^t)$$

$$x \geq 0$$

$$\min_z \ \textbf{KL}(z \,||\, x^{t-1}) \quad \textbf{s.t.}$$

$$\langle p^t, z \rangle \leq 1 + \epsilon \qquad (z^t)$$

$$x \geq 0$$

(D)

$$\max \sum_{t \in C} y^t - \sum_{t \in P} z^t \quad \textbf{s.t.}$$

$$\forall i,\, t \in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\, t \in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\, t \qquad 0 \leq r_i^t \leq 1$$

$$y, z \geq 0$$

How to analyze? The Primal-Dual Method!

$$\min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \textbf{s.t.}$$

$$\langle c^t, z\rangle \geq 1 \qquad (y^t)$$

$$x \geq 0$$

$$\min_z \ \mathbf{KL}(z \,||\, x^{t-1}) \quad \textbf{s.t.}$$

$$\langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t)$$

$$x \geq 0$$

(D)

$$\max \sum_{t\in C} y^t - \sum_{t\in P} z^t \quad \textbf{s.t.}$$

$$\forall i,\ t\in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t\in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \qquad 0 \leq r_i^t \leq 1$$

$$y, z \geq 0$$

Use the Lagrange multipliers from the convex program!

How to analyze? The Primal-Dual Method!

$$\min_z \; \mathbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.}$$
$$\langle c^t, z \rangle \geq 1 \qquad (y^t)$$
$$x \geq 0$$

$$\min_z \; \mathbf{KL}(z \,||\, x^{t-1}) \quad \text{s.t.}$$
$$\langle p^t, z \rangle \leq 1 + \epsilon \qquad (z^t)$$
$$x \geq 0$$

(D)

$$\max \sum_{t \in C} y^t - \sum_{t \in P} z^t \quad \text{s.t.}$$
$$\forall i, t \in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$
$$\forall i, t \in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$
$$\forall i, t \qquad 0 \leq r_i^t \leq 1$$
$$y, z \geq 0$$

Use the Lagrange multipliers from the convex program!

How to analyze? The Primal-Dual Method!

$$\min_z \ \textbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \textbf{s.t.}$$

$$\langle c^t, z\rangle \geq 1 \qquad (y^t)$$

$$x \geq 0$$

$$\min_z \ \textbf{KL}(z \,||\, x^{t-1}) \quad \textbf{s.t.}$$

$$\langle p^t, z\rangle \leq 1+\epsilon \qquad (z^t)$$

$$x \geq 0$$

(D)

$$\max \sum_{t\in C} y^t - \sum_{t\in P} z^t \quad \textbf{s.t.}$$

$$\forall i,\ t\in C: \qquad c_i^t y^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t\in P: \qquad -p_i^t z^t - r_i^t + r_i^{t+1} \leq 0$$

$$\forall i,\ t \qquad 0 \leq r_i^t \leq 1$$

$$y, z \geq 0$$

Use the Lagrange multipliers from the convex program!

Set $r = \log\left(\dfrac{1+4n/\epsilon}{1+4n\cdot x^{t-1}/\epsilon}\right)$.

The Proof

The Proof

Lemma 1: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D).

The Proof

Lemma 1: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D).

Proof: straightforward checking.

The Proof

Lemma 1: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D).

Proof: straightforward checking.

Lemma 2: **ALG** $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right]$.

The Proof

Lemma 1: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D).

Proof: straightforward checking.

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right]$.

(Dual objective)

The Proof

<u>Lemma 1</u>: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D)**.**

Proof: straightforward checking.

<u>Lemma 2</u>: **ALG** $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right]$. (Dual objective)

By weak duality:

The Proof

Lemma 1: (y, z, r) **scaled down by** $O(\log(n/\epsilon))$ **feasible to** (D).

Proof: straightforward checking.

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$ (Dual objective)

By weak duality:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(n/\epsilon)/\epsilon) \cdot$ **OPT.**

Relating ALG and Dual Objective

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right]$.

Relating ALG and Dual Objective

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

Relating ALG and Dual Objective

Lemma 2: $\text{ALG} \leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Relating ALG and Dual Objective

Lemma 2: $\text{ALG} \leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Relating ALG and Dual Objective

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Using KKT conditions + non-negativity of KL-divergence:

Relating ALG and Dual Objective

Lemma 2: ALG $\leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Using KKT conditions + non-negativity of KL-divergence:

Lemma 2a: ALG $\leq \left(1 + \frac{\epsilon}{4}\right)\sum_t y^t.$

Relating ALG and Dual Objective

Lemma 2: $\text{ALG} \leq O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Using KKT conditions + non-negativity of KL-divergence:

Lemma 2a: $\text{ALG} \leq \left(1 + \frac{\epsilon}{4}\right)\sum_t y^t.$

Lemma 2b: $0 \leq \left(1 + \frac{\epsilon}{4}\right)\sum_t y^t - (1+\epsilon)\sum_t z^t.$

Relating ALG and Dual Objective

$$\underline{\textbf{Lemma 2}}\text{:}\quad \text{ALG} \le O\left(\frac{1}{\epsilon}\right)\left[\sum y^t - \sum z^t\right].$$

This part is annoying, since we lose dual money.

New contribution of our work is to handle this!

Using KKT conditions + non-negativity of KL-divergence:

$$\underline{\textbf{Lemma 2a}}\text{:}\quad \text{ALG} \le \left(1 + \frac{\epsilon}{4}\right)\sum_t y^t.$$

Linear combination gives Lemma 2.

$$\underline{\textbf{Lemma 2b}}\text{:}\quad 0 \le \left(1 + \frac{\epsilon}{4}\right)\sum_t y^t - (1+\epsilon)\sum_t z^t.$$

Slack for this argument needs resource augmentation, i.e. violate packing by ϵ.

Lemma 2a Proof

<u>Lemma 2a</u>: ALG $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

Lemma 2a Proof

<u>Lemma 2a</u>: ALG $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -\, p_i^t z^t$$

Lemma 2a Proof

Lemma 2a: ALG $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

$$\sum_i (x_i^t - x_i^{t-1})_+ \leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta) \ \log\left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta}\right)$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -\, p_i^t z^t$$

Lemma 2a Proof

Lemma 2a: ALG $\leq \left(1+\frac{\epsilon}{4}\right)\sum_t y^t.$

$$\sum_i (x_i^t - x_i^{t-1})_+ \leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ \log\left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta}\right)$$

$$\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ c_i^t y^t$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -\, p_i^t z^t$$

Lemma 2a Proof

Lemma 2a: ALG $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

$$\sum_i (x_i^t - x_i^{t-1})_+ \leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta) \ \log \left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta} \right)$$

$$\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta) \ c_i^t y^t$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -\, p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}.$

Lemma 2a Proof

<u>**Lemma 2a:**</u> **ALG** $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

$$\sum_i (x_i^t - x_i^{t-1})_+ \leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ \log\left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta}\right)$$

$$\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ c_i^t y^t$$

$$\leq \frac{\epsilon}{4} y^t + y^t \sum_i c_i^t x_i^t$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}.$

Lemma 2a Proof

<u>**Lemma 2a:**</u> **ALG** $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

$$\sum_i (x_i^t - x_i^{t-1})_+ \leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ \log\left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta}\right)$$

$$\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ c_i^t y^t$$

$$\leq \frac{\epsilon}{4} y^t + y^t \sum_i c_i^t x_i^t$$

$$= \left(1 + \frac{\epsilon}{4}\right) y^t.$$

Set $\delta := \frac{\epsilon}{4n}.$

Lemma 2a Proof

Lemma 2a: ALG $\leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t.$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

$$\begin{aligned}
\sum_i (x_i^t - x_i^{t-1})_+ &\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ \log\left(\frac{x_i^t + \delta}{x_i^{t-1} + \delta}\right) \\
&\leq \sum_{i:\, x_i^t > x_i^{t-1}} (x_i^t + \delta)\ c_i^t y^t \\
&\leq \frac{\epsilon}{4} y^t + y^t \sum_i c_i^t x_i^t \\
&= \left(1 + \frac{\epsilon}{4}\right) y^t . \ \blacksquare
\end{aligned}$$

Set $\delta := \frac{\epsilon}{4n}$.

Lemma 2b Proof

Lemma 2b: $0 \leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1 + \epsilon) \sum_t z^t.$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}.$

Lemma 2b Proof

<u>**Lemma 2b**</u>: $0 \le \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1+\epsilon) \sum_t z^t.$

By positivity of KL divergence

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

Lemma 2b Proof

Lemma 2b: $0 \le \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1+\epsilon) \sum_t z^t.$

By positivity of KL divergence

$$0 \le \sum_{t \in C} \mathrm{KL}(x^t + \delta \,||\, x^{t-1} + \delta) + \sum_{t \in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

Lemma 2b Proof

Lemma 2b: $0 \leq \left(1+\frac{\epsilon}{4}\right)\sum_t y^t - (1+\epsilon)\sum_t z^t.$

By positivity of KL divergence

$$0 \leq \sum_{t\in C} \mathrm{KL}(x^t+\delta \,||\, x^{t-1}+\delta) + \sum_{t\in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

KKT conditions:

$$\log \frac{x^t+\delta}{x^{t-1}+\delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

$$\mathrm{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i$$

Lemma 2b Proof

Lemma 2b: $0 \leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1+\epsilon) \sum_t z^t.$

By positivity of KL divergence

$$0 \leq \sum_{t \in C} \mathrm{KL}(x^t + \delta \,||\, x^{t-1} + \delta) + \sum_{t \in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

$$\leq \sum_{t \in C} \sum_{i:\, c_i^t > 0} (x_i^t + \delta) \log \frac{x_i^t + \delta}{x_i^{t-1} + \delta} + \sum_{t \in P} \sum_{i:\, p_i^t > 0} x_i^t \log \frac{x_i^t}{x_i^{t-1}}$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

$$\mathrm{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i$$

Lemma 2b Proof

Lemma 2b: $0 \leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1+\epsilon) \sum_t z^t.$

By positivity of KL divergence

$$0 \leq \sum_{t \in C} \mathrm{KL}(x^t + \delta \,||\, x^{t-1} + \delta) + \sum_{t \in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

$$\leq \sum_{t \in C} \sum_{i: c_i^t > 0} (x_i^t + \delta) \log \frac{x_i^t + \delta}{x_i^{t-1} + \delta} + \sum_{t \in P} \sum_{i: p_i^t > 0} x_i^t \log \frac{x_i^t}{x_i^{t-1}}$$

$$\leq \sum_{t \in C} y^t \sum_{i: c_i^t > 0} (x_i^t + \delta) - \sum_{t \in P} z^t \sum_{i: p_i^t > 0} x_i^t$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

$$\mathrm{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i$$

Lemma 2b Proof

Lemma 2b: $0 \le \left(1+\frac{\epsilon}{4}\right)\sum_t y^t - (1+\epsilon)\sum_t z^t.$

By positivity of KL divergence

$$0 \le \sum_{t\in C} \mathrm{KL}(x^t+\delta \,||\, x^{t-1}+\delta) + \sum_{t\in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

$$\le \sum_{t\in C}\sum_{i:\, c_i^t>0} (x_i^t+\delta)\ \log\frac{x_i^t+\delta}{x_i^{t-1}+\delta} + \sum_{t\in P}\sum_{i:\, p_i^t>0} x_i^t \log\frac{x_i^t}{x_i^{t-1}}$$

$$\le \sum_{t\in C} y^t \sum_{i:\, c_i^t>0} (x_i^t+\delta) - \sum_{t\in P} z^t \sum_{i:\, p_i^t>0} x_i^t$$

$$\le \left(1+\frac{\epsilon}{4}\right)\sum_{t\in C} y^t - (1+\epsilon)\sum_{t\in P} z^t .$$

KKT conditions:

$$\log\frac{x^t+\delta}{x^{t-1}+\delta} = c_i^t y^t$$

$$\log\frac{x^t}{x^{t-1}} = -\,p_i^t z^t$$

Set $\delta := \dfrac{\epsilon}{4n}$.

$$\mathrm{KL}(a\,||\,b) = \sum_i a_i \log\frac{a_i}{b_i} - a_i + b_i$$

Lemma 2b Proof

Lemma 2b: $0 \leq \left(1 + \frac{\epsilon}{4}\right) \sum_t y^t - (1+\epsilon) \sum_t z^t.$

By positivity of KL divergence

$$0 \leq \sum_{t \in C} \mathrm{KL}(x^t + \delta \,||\, x^{t-1} + \delta) + \sum_{t \in P} \mathrm{KL}(x^t \,||\, x^{t-1})$$

$$\leq \sum_{t \in C} \sum_{i:\, c_i^t > 0} (x_i^t + \delta) \log \frac{x_i^t + \delta}{x_i^{t-1} + \delta} + \sum_{t \in P} \sum_{i:\, p_i^t > 0} x_i^t \log \frac{x_i^t}{x_i^{t-1}}$$

$$\leq \sum_{t \in C} y^t \sum_{i:\, c_i^t > 0} (x_i^t + \delta) - \sum_{t \in P} z^t \sum_{i:\, p_i^t > 0} x_i^t$$

$$\leq \left(1 + \frac{\epsilon}{4}\right) \sum_{t \in C} y^t - (1+\epsilon) \sum_{t \in P} z^t. \quad \blacksquare$$

KKT conditions:

$$\log \frac{x^t + \delta}{x^{t-1} + \delta} = c_i^t y^t$$

$$\log \frac{x^t}{x^{t-1}} = -p_i^t z^t$$

Set $\delta := \frac{\epsilon}{4n}$.

$$\mathrm{KL}(a \,||\, b) = \sum_i a_i \log \frac{a_i}{b_i} - a_i + b_i$$

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

Our KL Projection Algorithm

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

$$\min_z \ \mathsf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \text{s.t.}$$

$$\langle c^t, z\rangle \geq 1$$

$$x \geq 0$$

Our KL Projection Algorithm

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

$$\min_z \ \textbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.}$$
$$\langle c^t, z \rangle \geq 1$$
$$x \geq 0$$

Our KL Projection Algorithm

$$\min_z \ \textbf{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.}$$
$$\langle c^t, z \rangle \geq 1$$
$$x \geq 0$$

$$\min_z \ \textbf{KL}(z \,||\, x^{t-1}) \quad \text{s.t.}$$
$$\langle p^t, z \rangle \leq 1 + \epsilon$$
$$x \geq 0$$

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

$$\min_z \ \mathrm{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.}$$
$$\langle c^t, z \rangle \geq 1$$
$$x \geq 0$$

Our KL Projection Algorithm

$$\min_z \ \mathrm{KL}(z + \delta \,||\, x^{t-1} + \delta) \quad \text{s.t.}$$
$$\langle c^t, z \rangle \geq 1$$
$$x \geq 0$$

$$\min_z \ \mathrm{KL}(z \,||\, x^{t-1}) \quad \text{s.t.}$$
$$\langle p^t, z \rangle \leq 1 + \epsilon$$
$$x \geq 0$$

Our algorithm reduces to [Buchbinder Naor 09] for pure covering.
Guarantees are tight in that case!

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

$$\min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \text{s.t.}$$
$$\langle c^t, z\rangle \geq 1$$
$$x \geq 0$$

Our KL Projection Algorithm

$$\min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \text{s.t.}$$
$$\langle c^t, z\rangle \geq 1$$
$$x \geq 0$$

$$\min_z \ \mathbf{KL}(z \,||\, x^{t-1}) \quad \text{s.t.}$$
$$\langle p^t, z\rangle \leq 1+\epsilon$$
$$x \geq 0$$

Our algorithm reduces to [Buchbinder Naor 09] for pure covering.
Guarantees are tight in that case!

Covering/Packing asymmetry is crucial for our analysis.

Comparison to Online Covering

Online Covering [Buchbinder Naor 09]

$$\min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \text{s.t.}$$
$$\langle c^t, z\rangle \geq 1$$
$$x \geq 0$$

Our KL Projection Algorithm

$$\min_z \ \mathbf{KL}(z+\delta \,||\, x^{t-1}+\delta) \quad \text{s.t.}$$
$$\langle c^t, z\rangle \geq 1$$
$$x \geq 0$$

$$\min_z \ \mathbf{KL}(z \,||\, x^{t-1}) \quad \text{s.t.}$$
$$\langle p^t, z\rangle \leq 1+\epsilon$$
$$x \geq 0$$

Our algorithm reduces to [Buchbinder Naor 09] for pure covering.
Guarantees are tight in that case!

Covering/Packing asymmetry is crucial for our analysis.

Was a barrier to prior work.

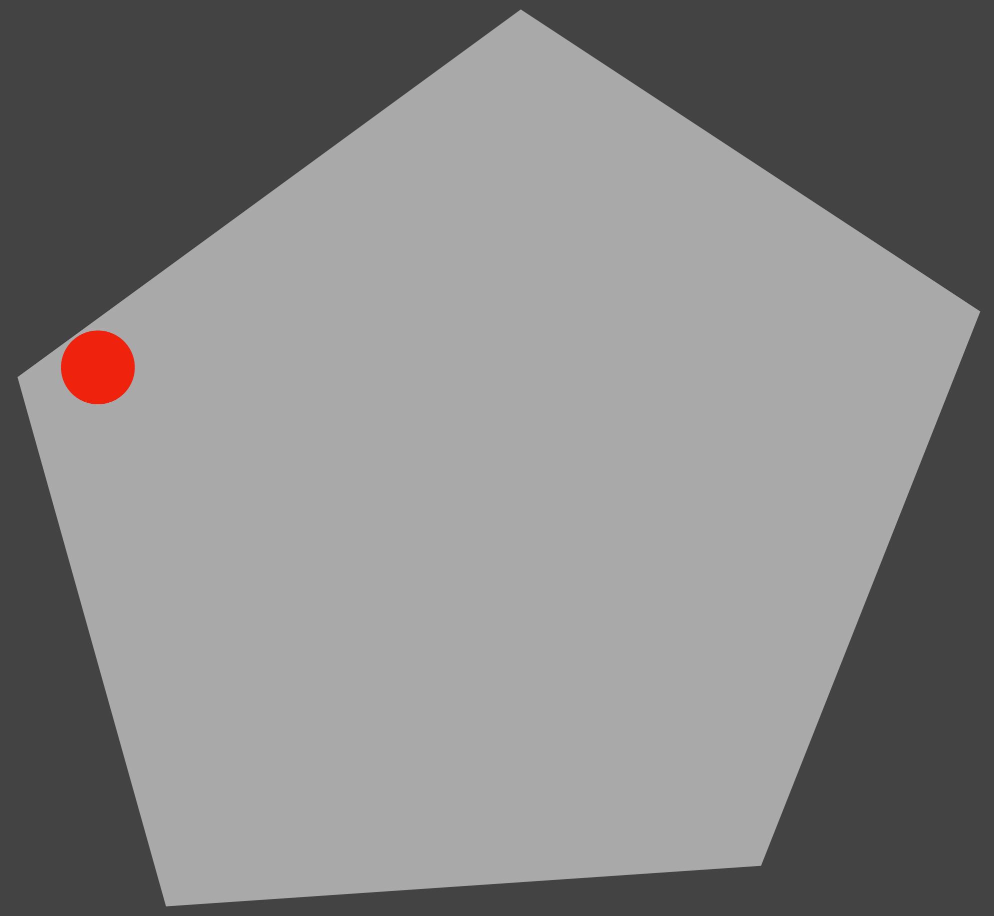
Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]

Our KL Projection Algorithm

Comparison to General Body Chasing

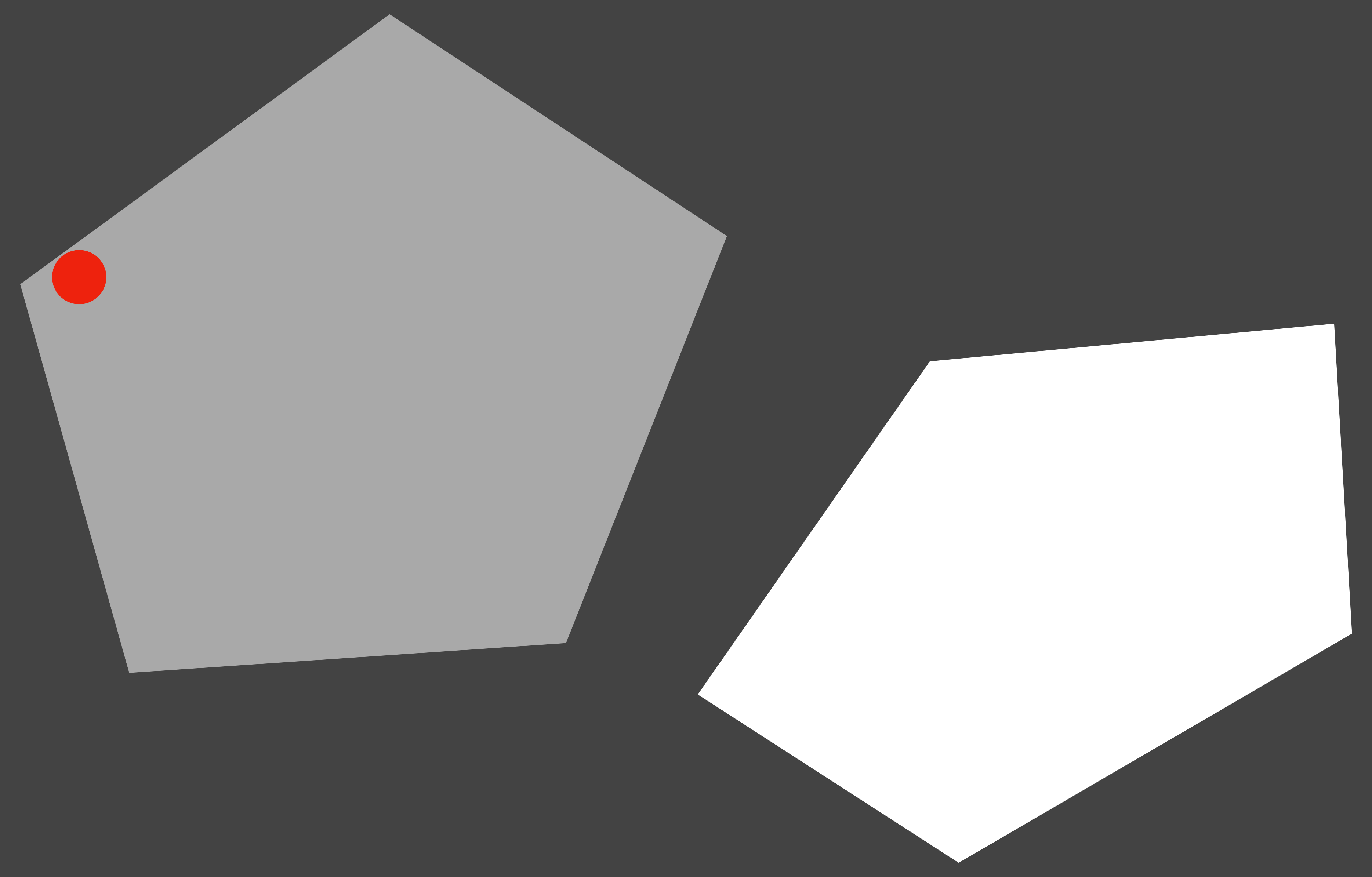
Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



Our KL Projection Algorithm

Comparison to General Body Chasing

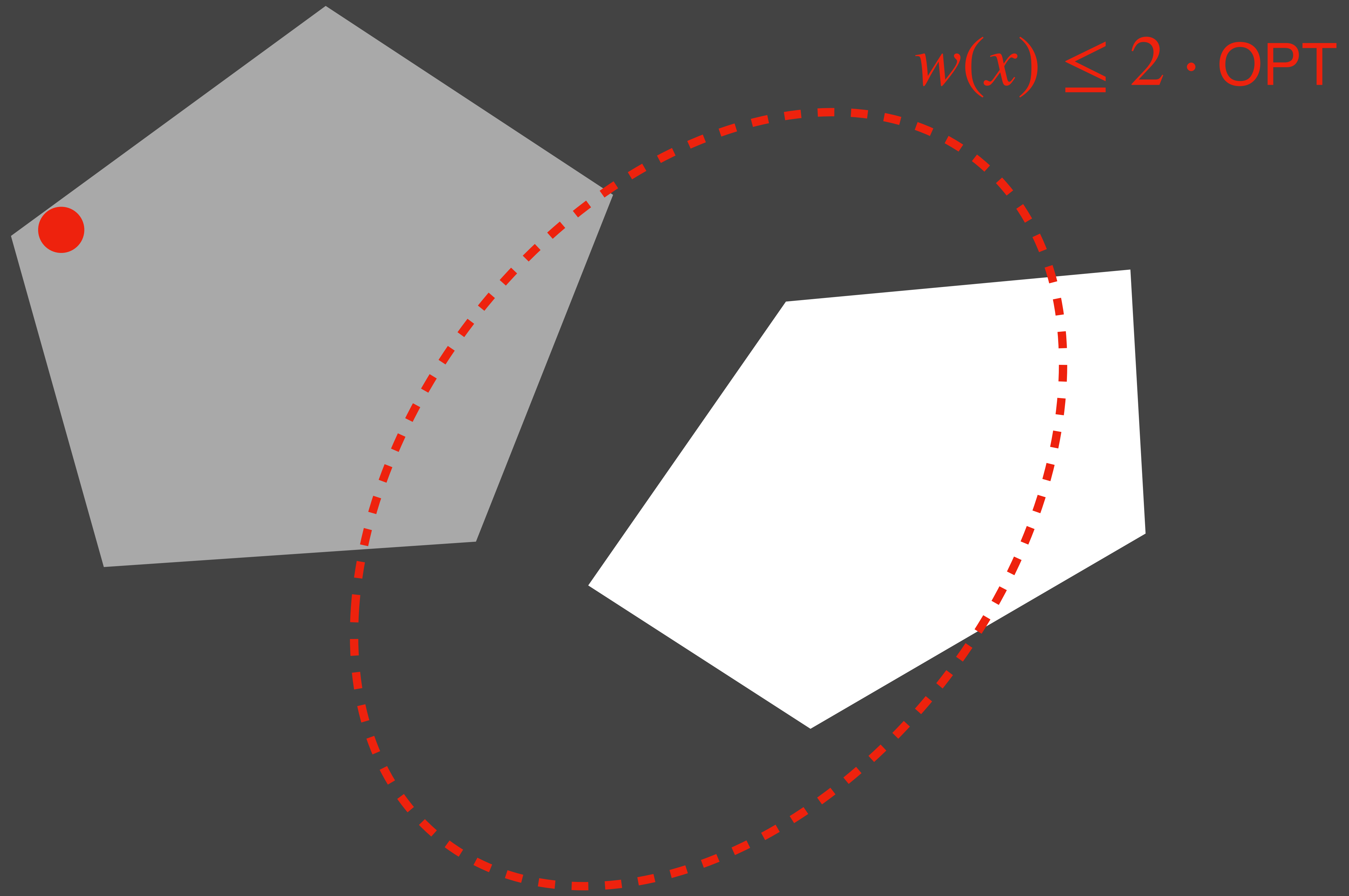
Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



Our KL Projection Algorithm

Comparison to General Body Chasing

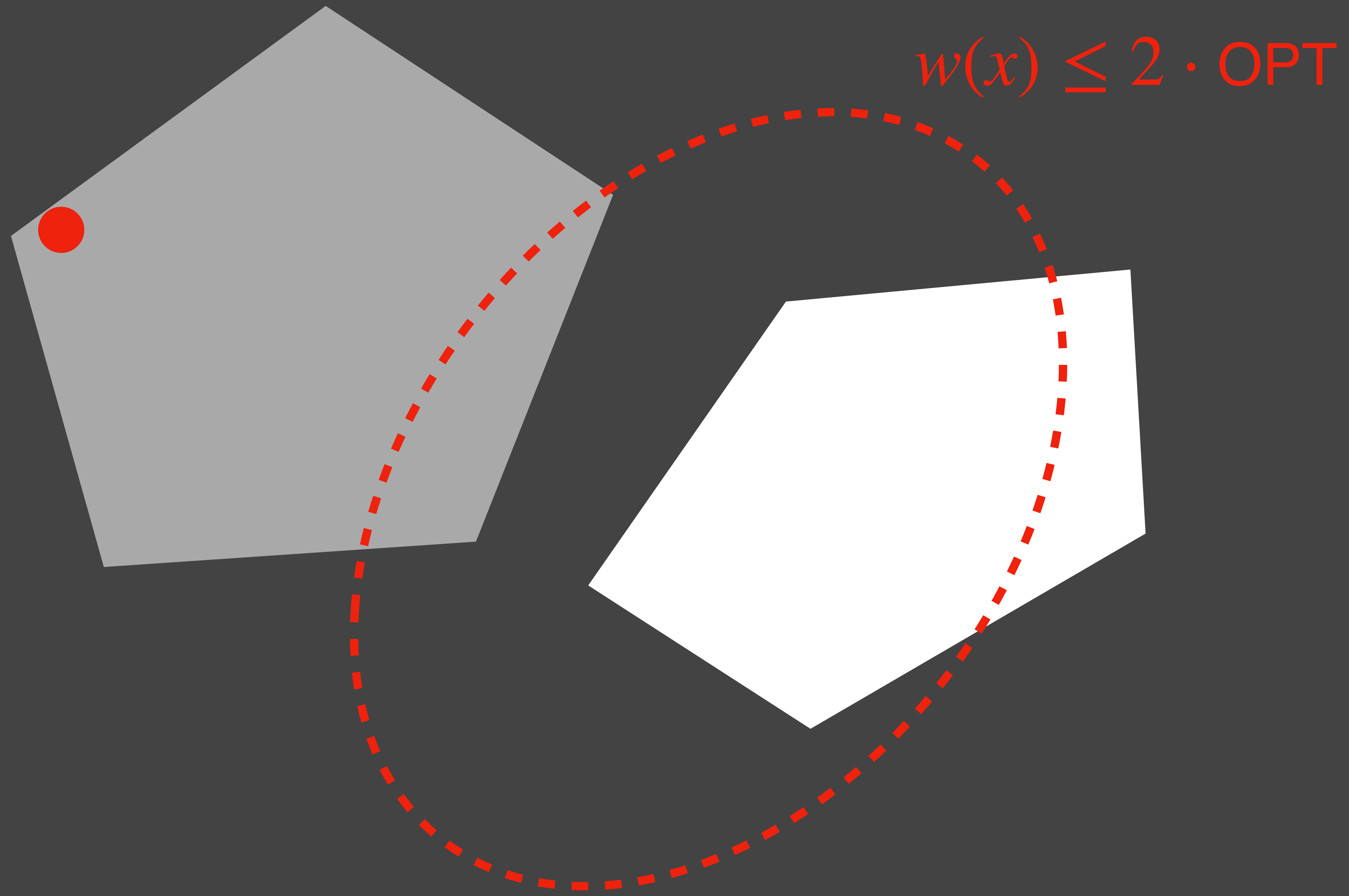
Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



Our KL Projection Algorithm

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]

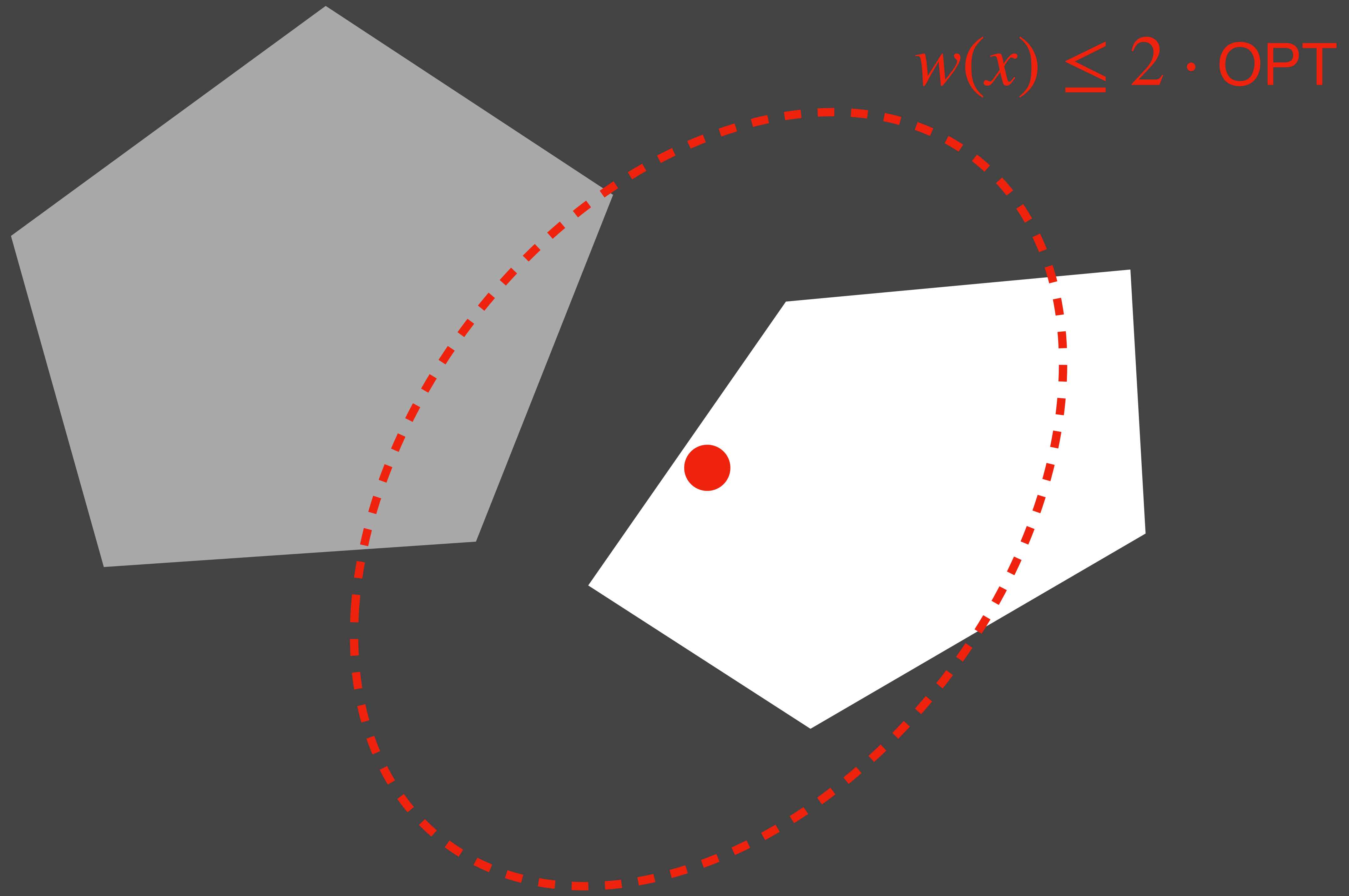


Our KL Projection Algorithm

$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]

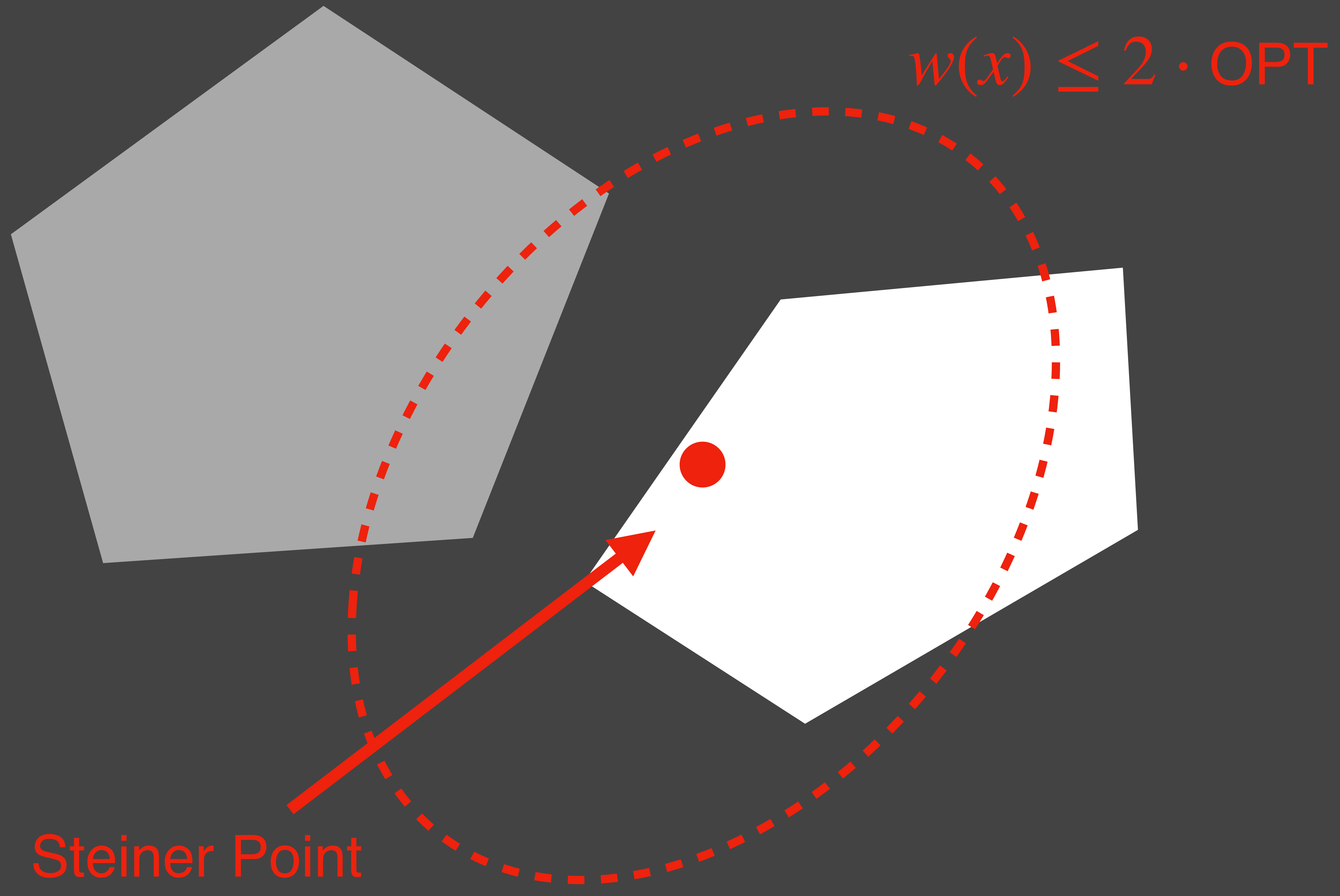


$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Our KL Projection Algorithm

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]

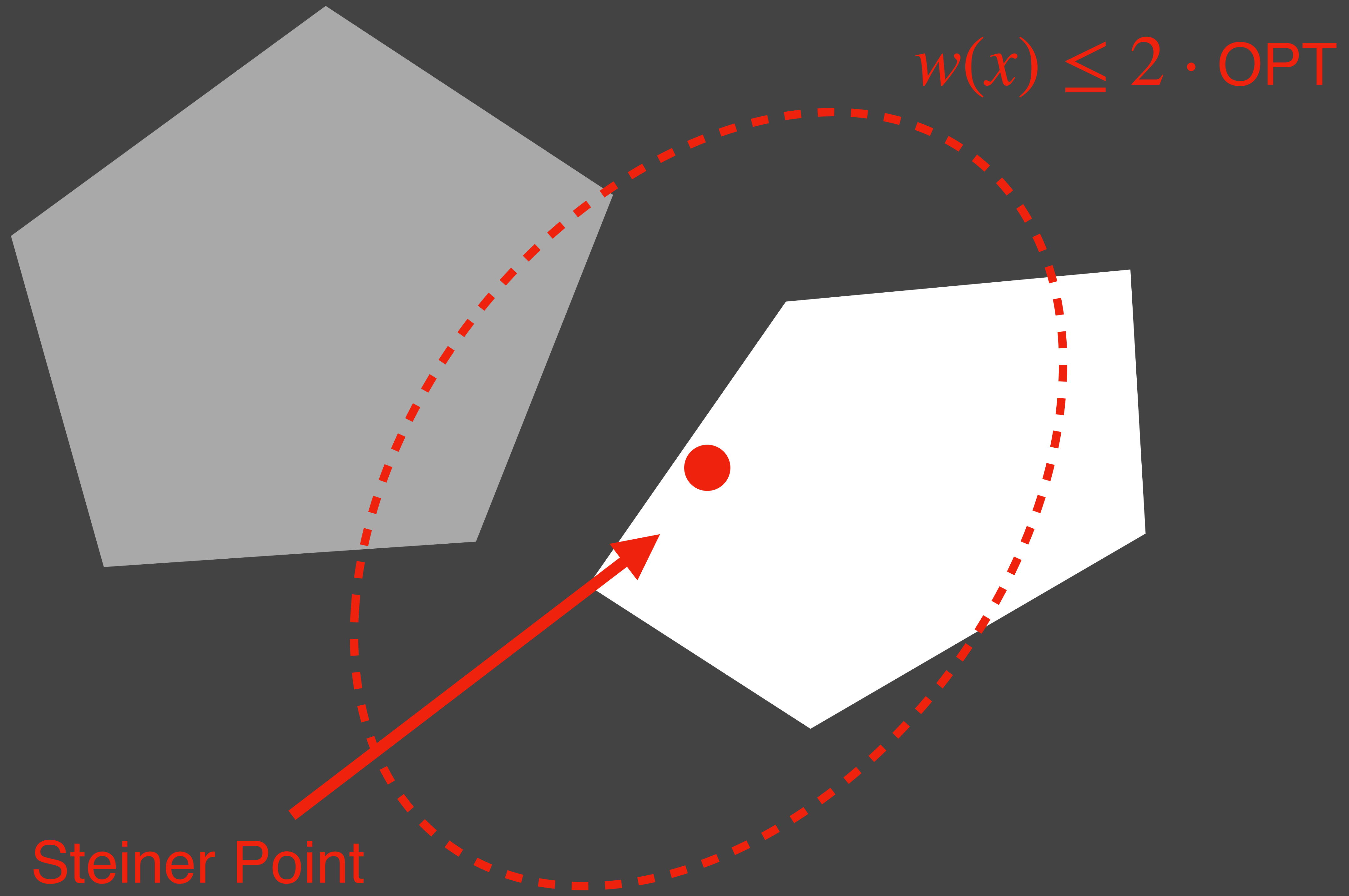


Our KL Projection Algorithm

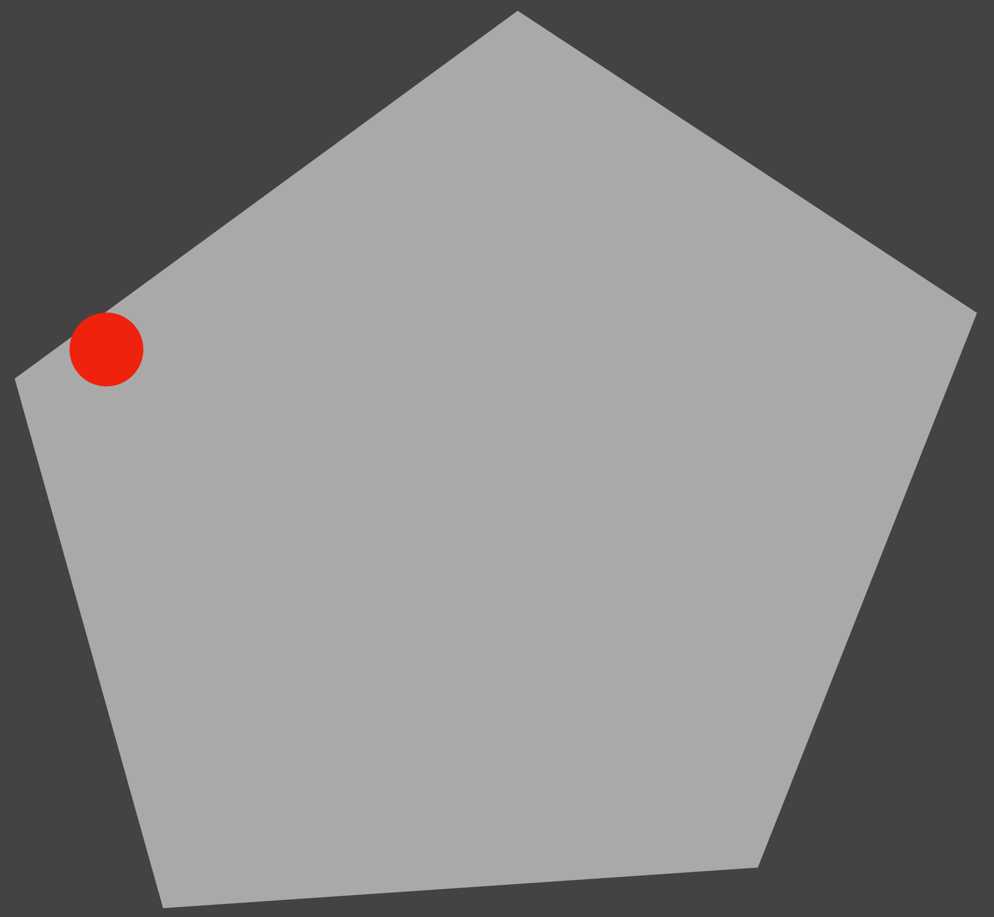
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



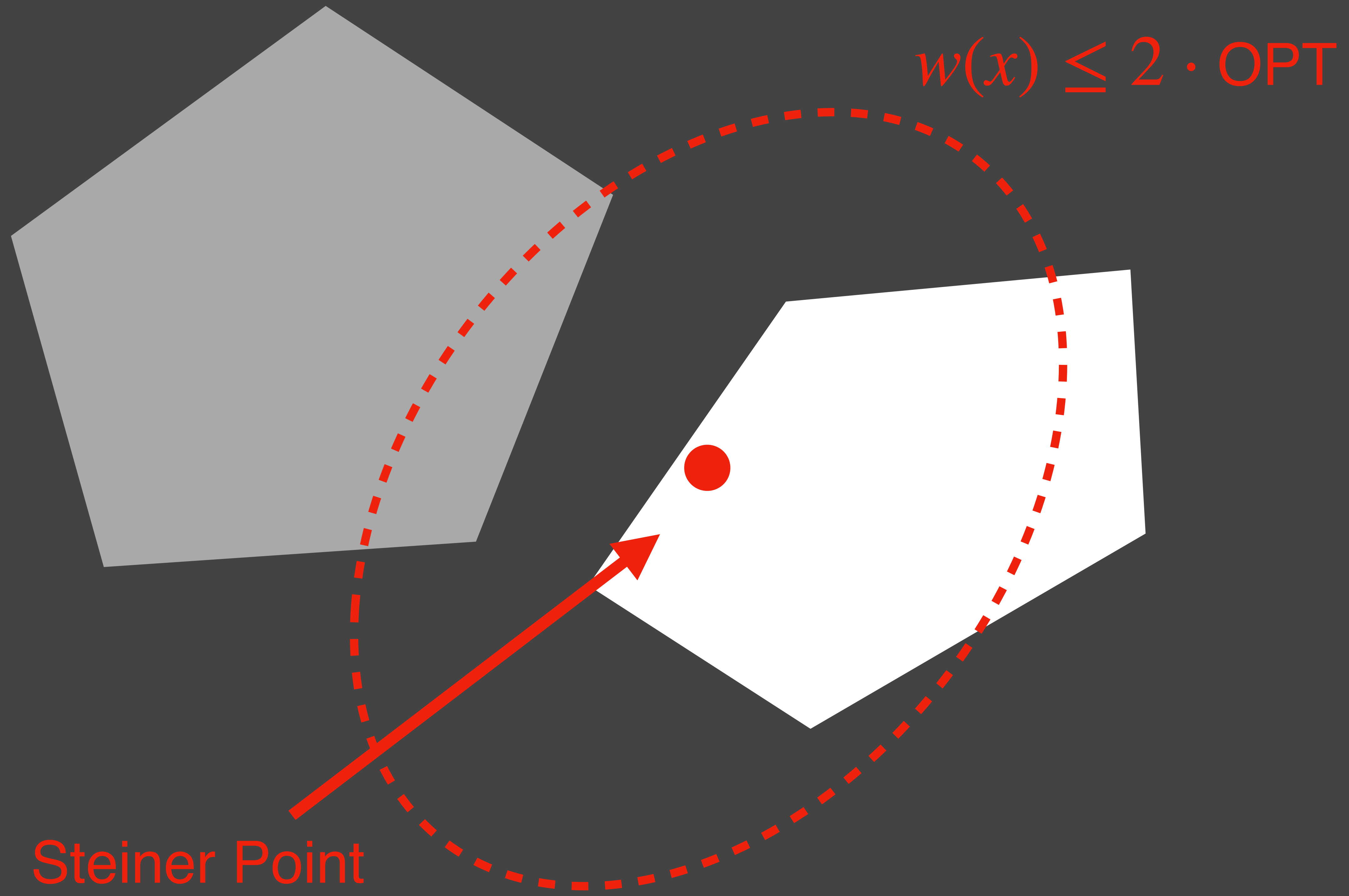
Our KL Projection Algorithm



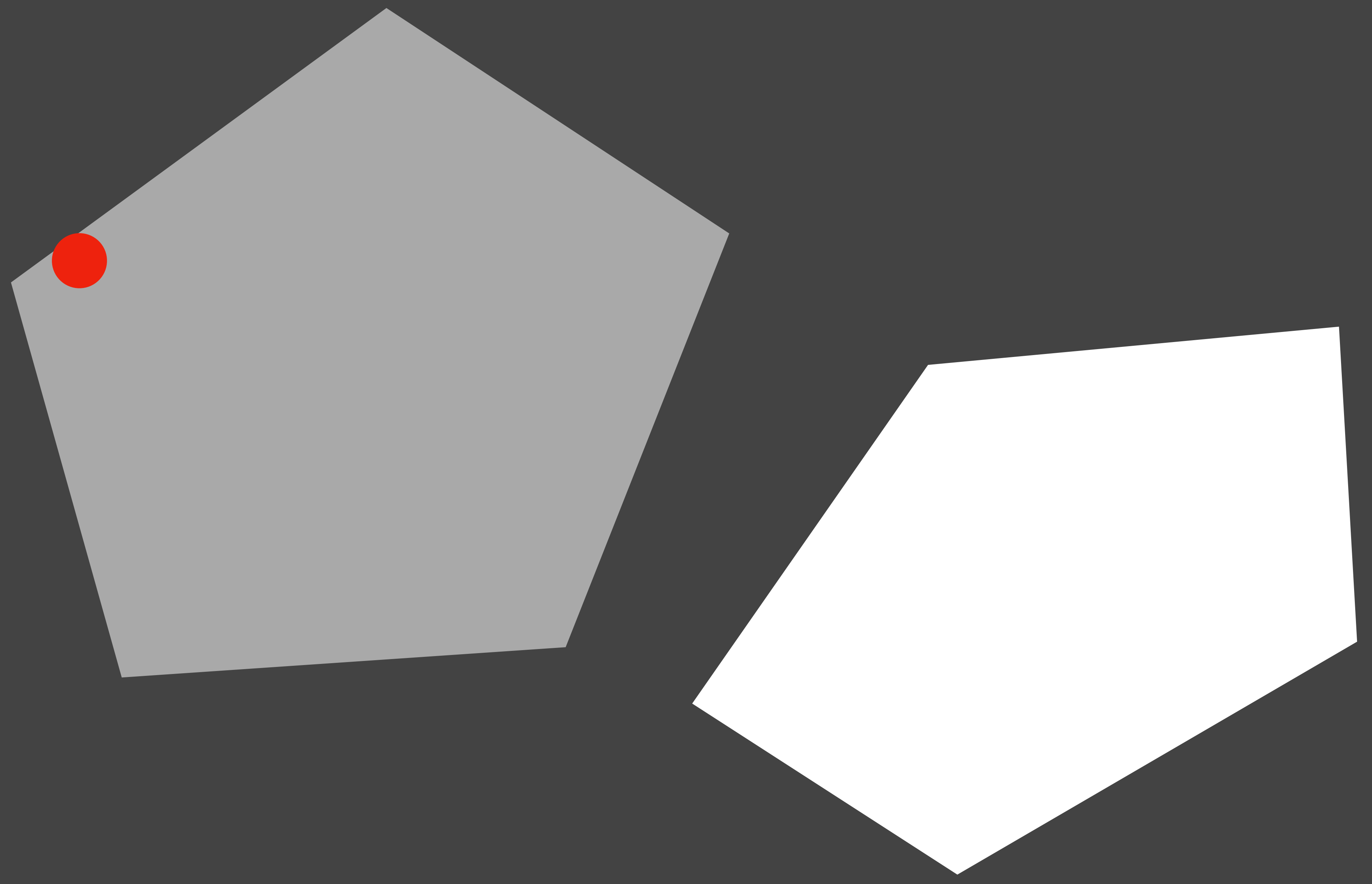
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



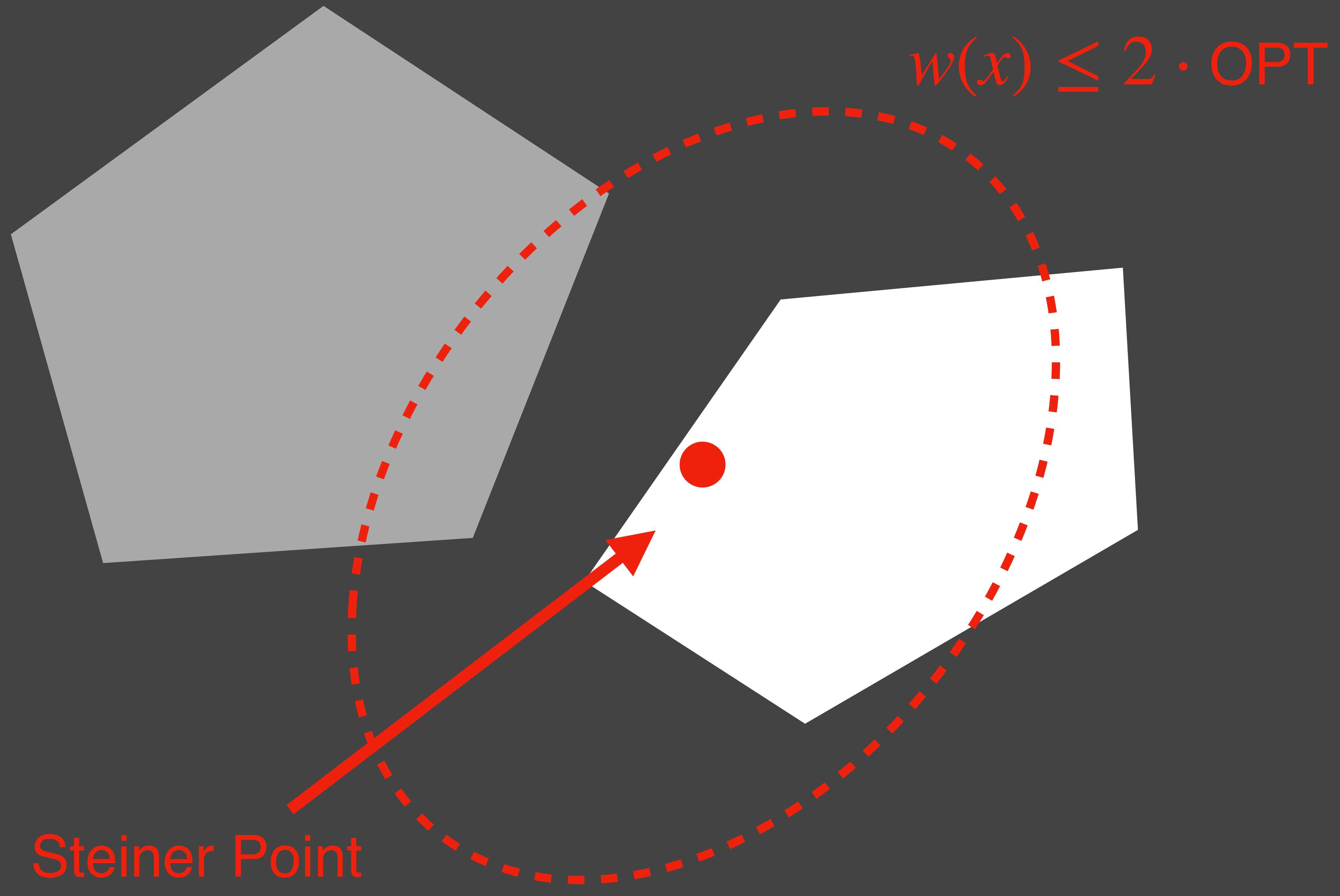
Our KL Projection Algorithm



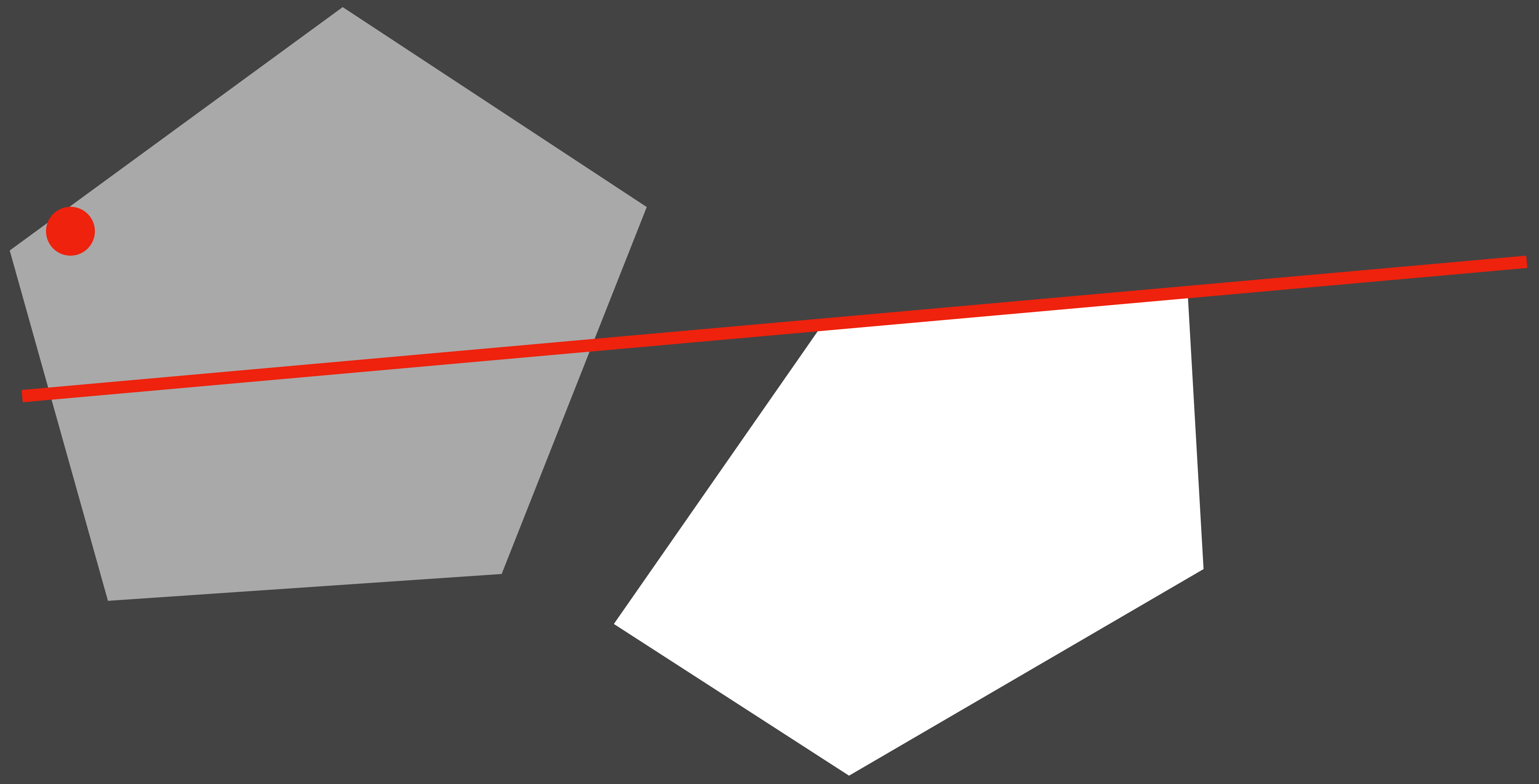
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



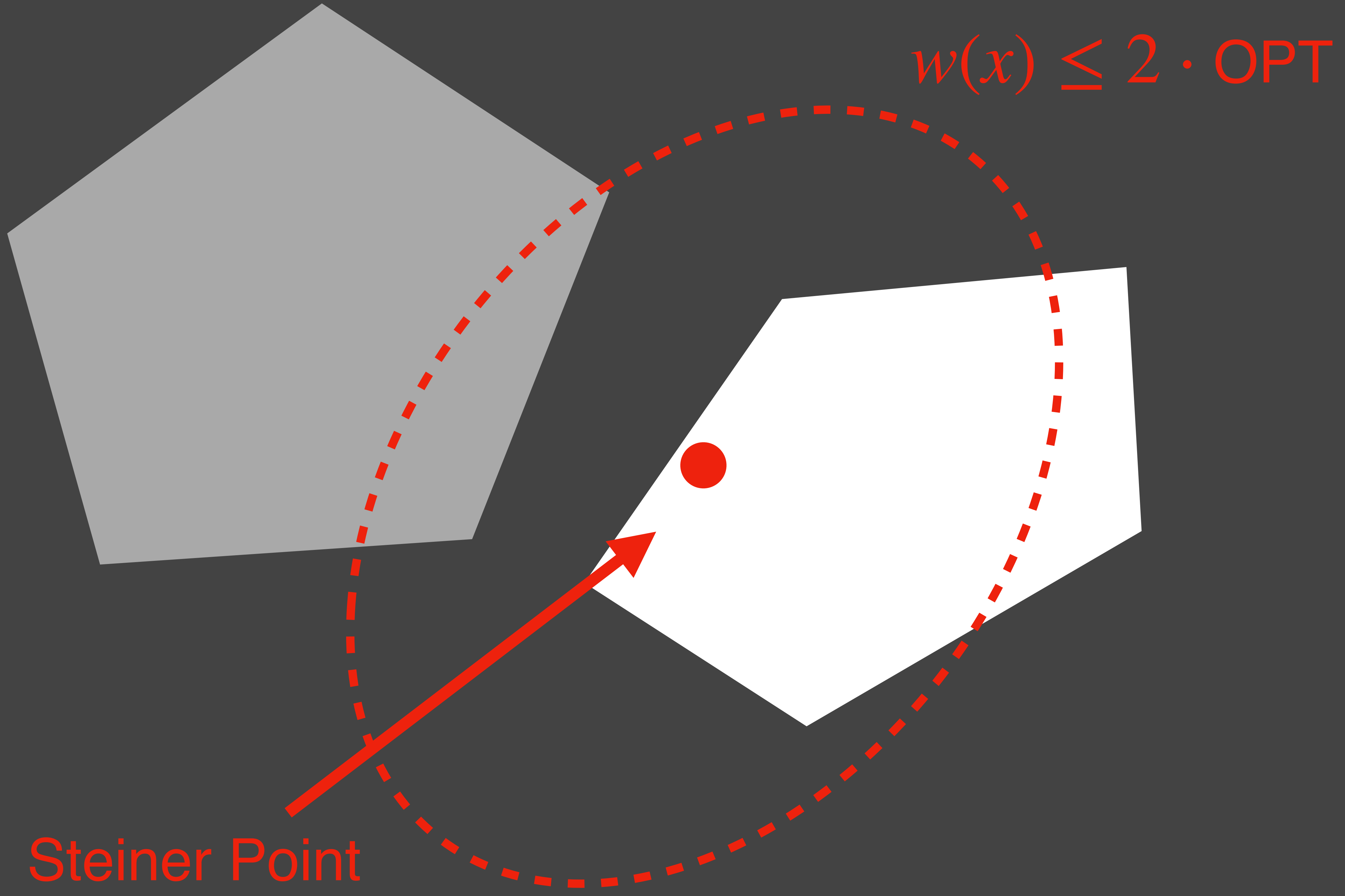
Our KL Projection Algorithm



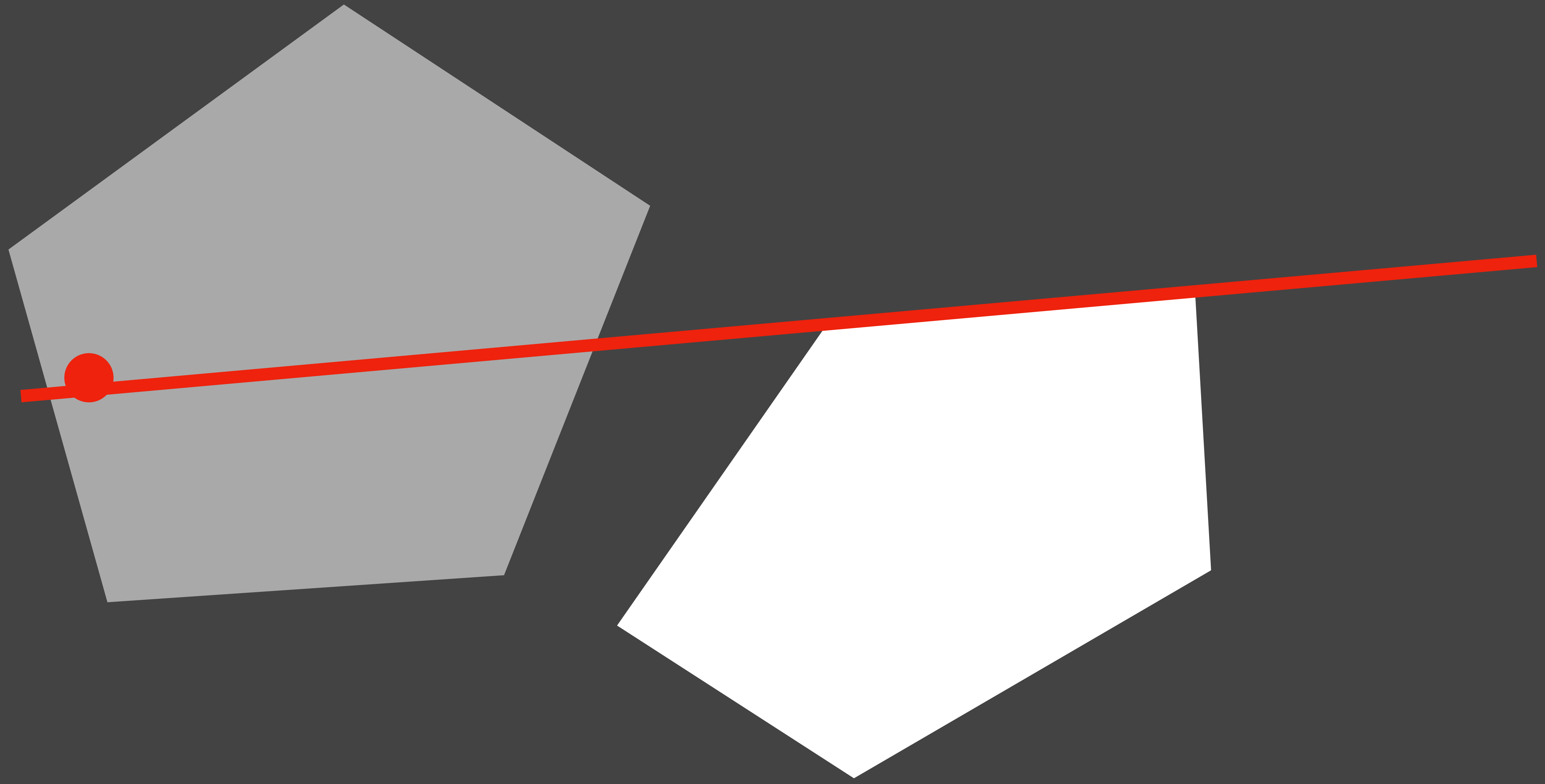
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



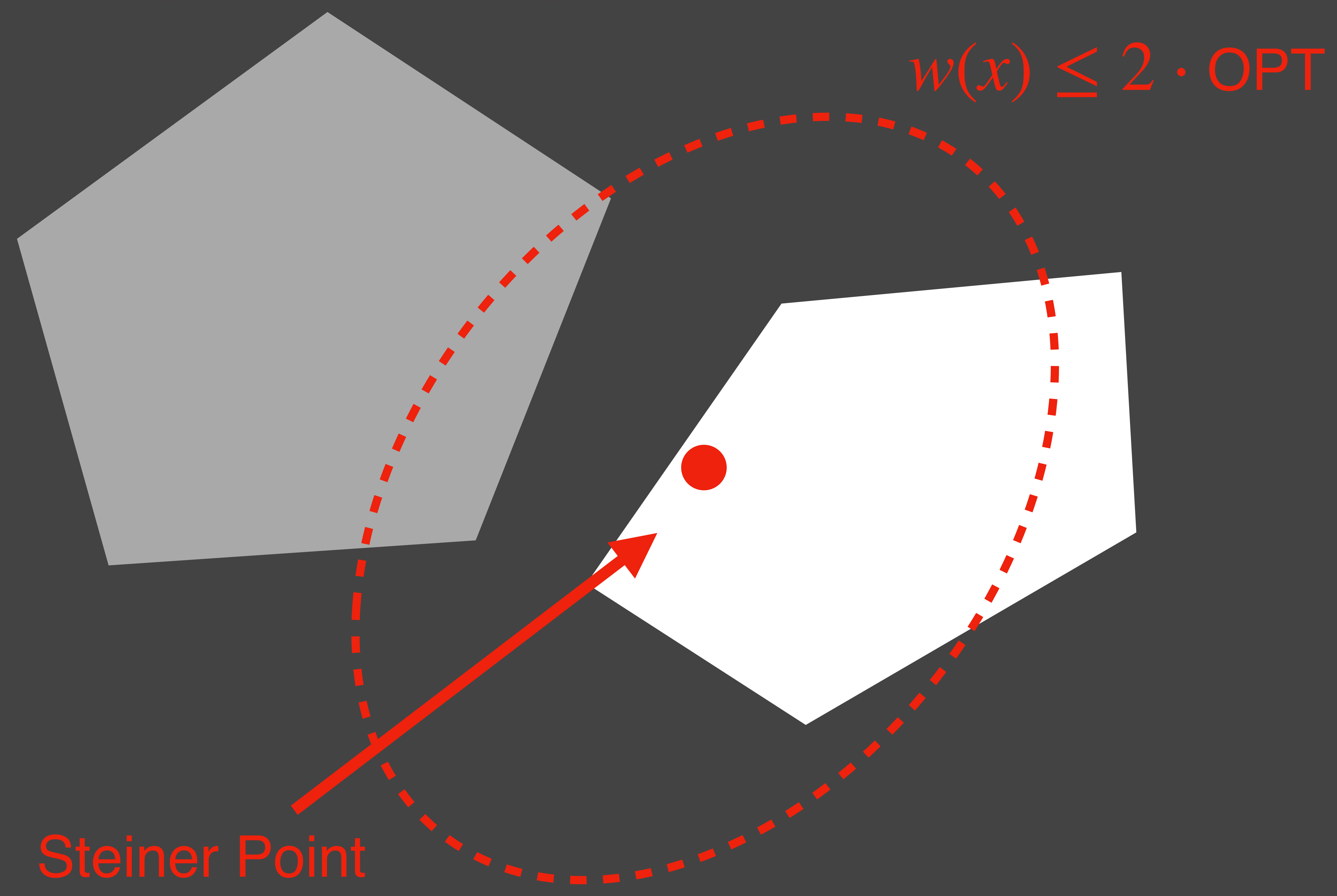
Our KL Projection Algorithm



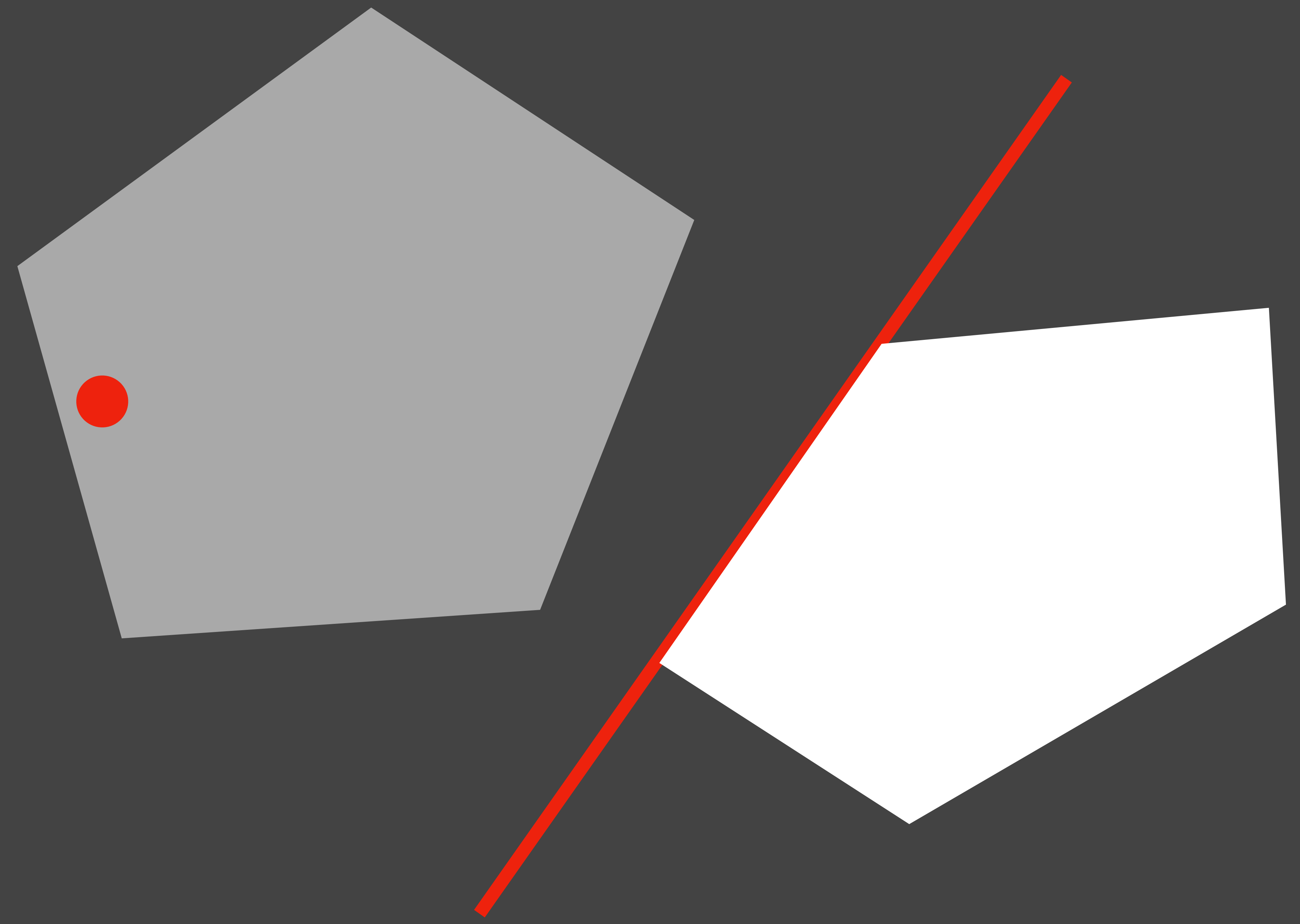
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



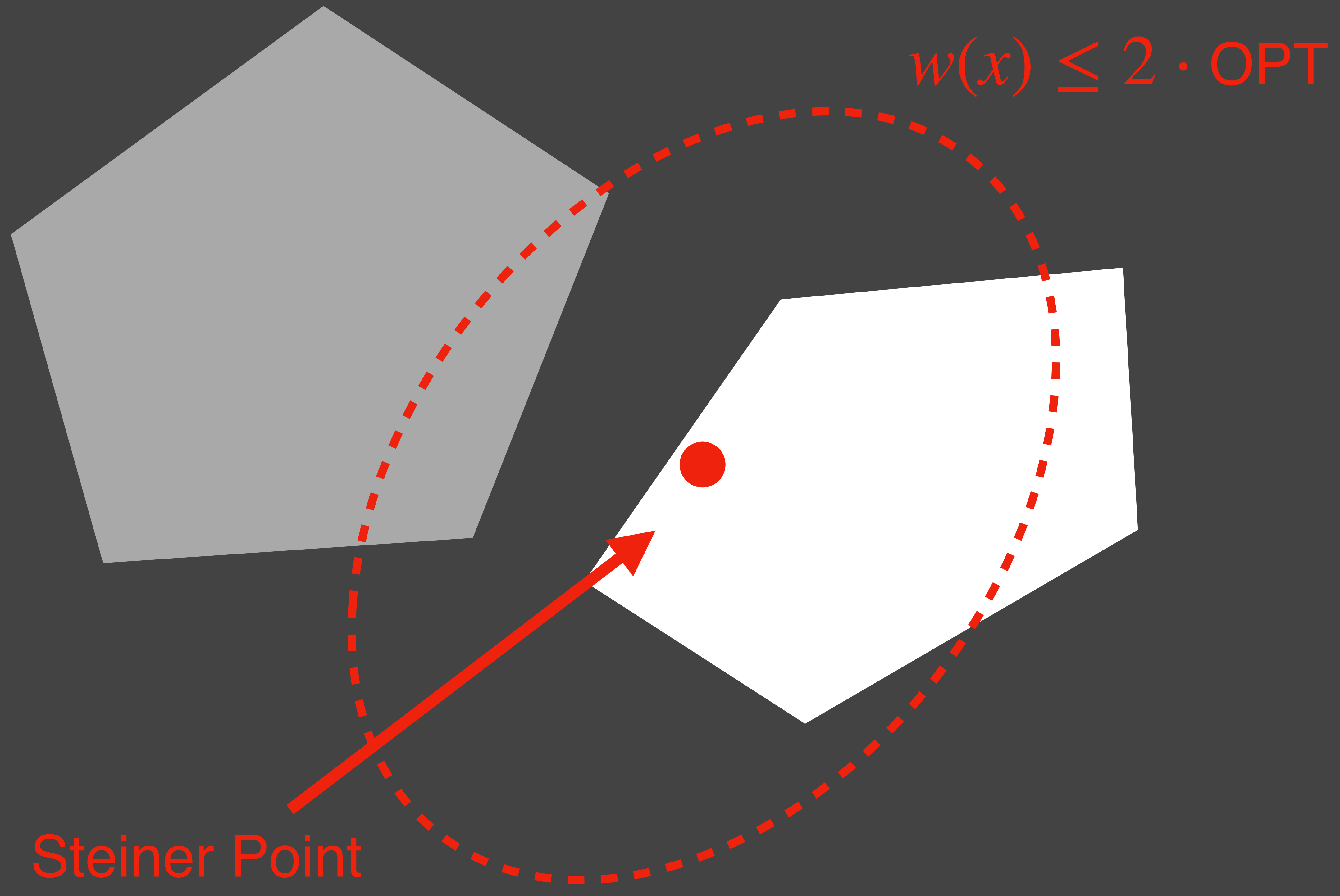
Our KL Projection Algorithm



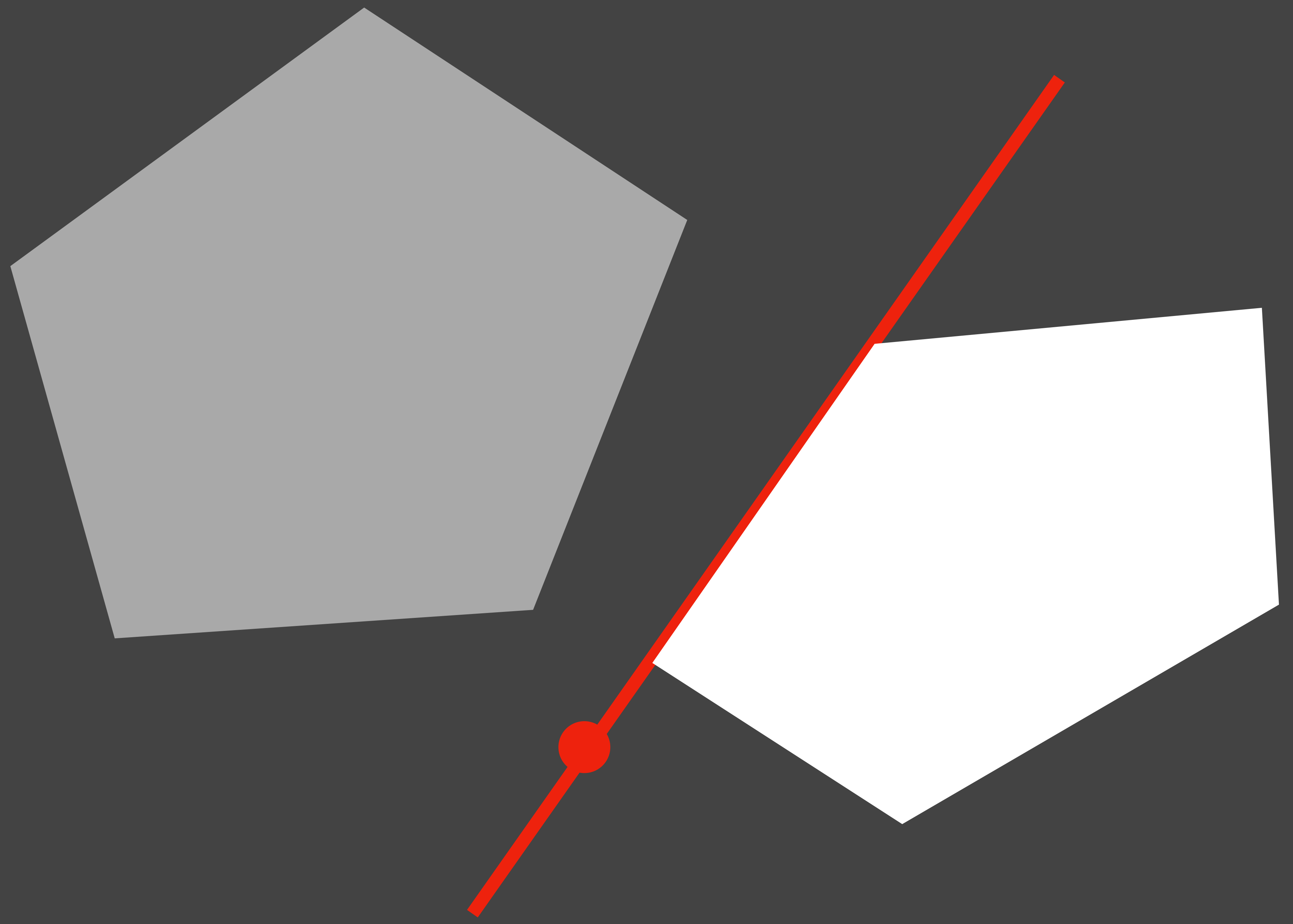
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



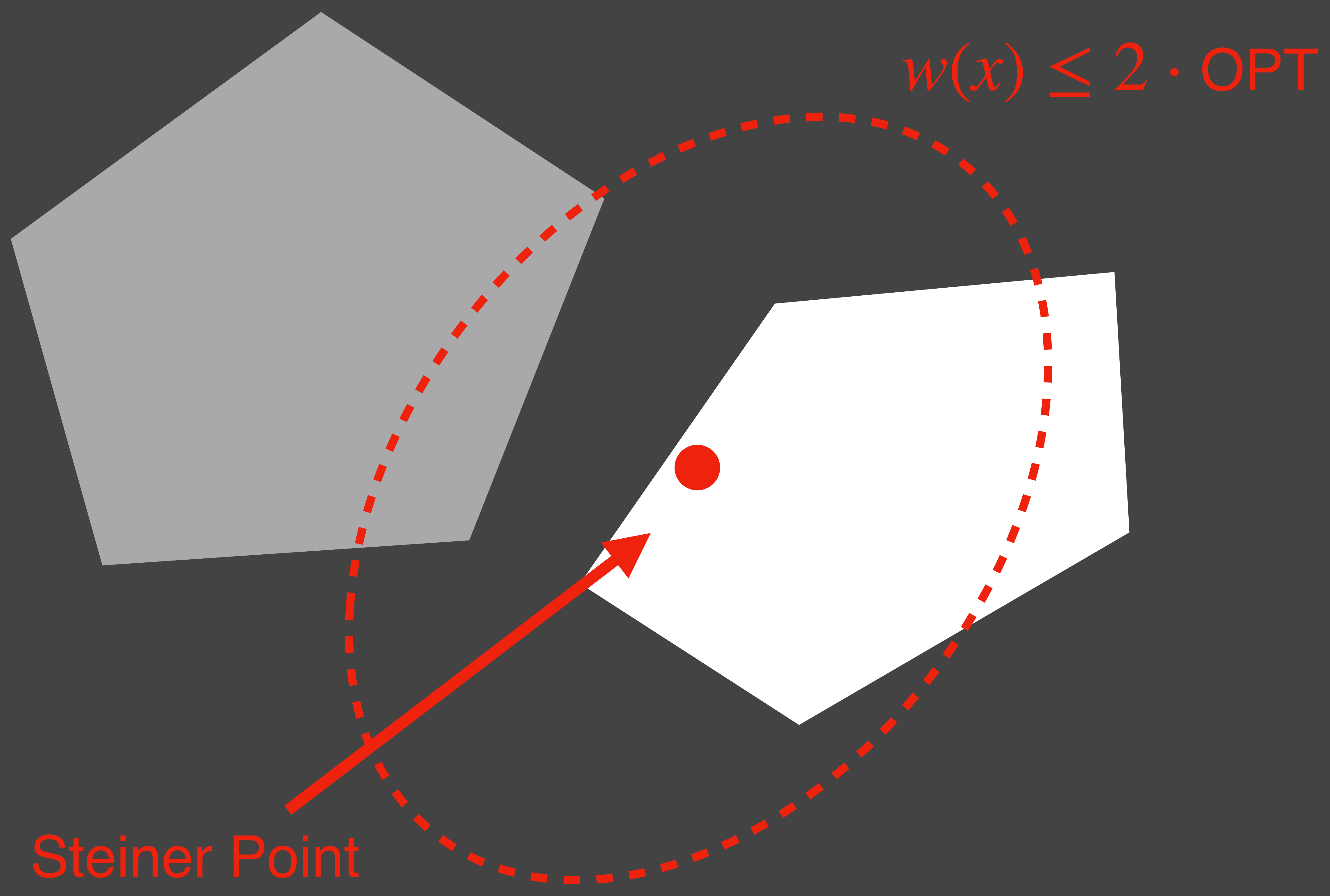
Our KL Projection Algorithm



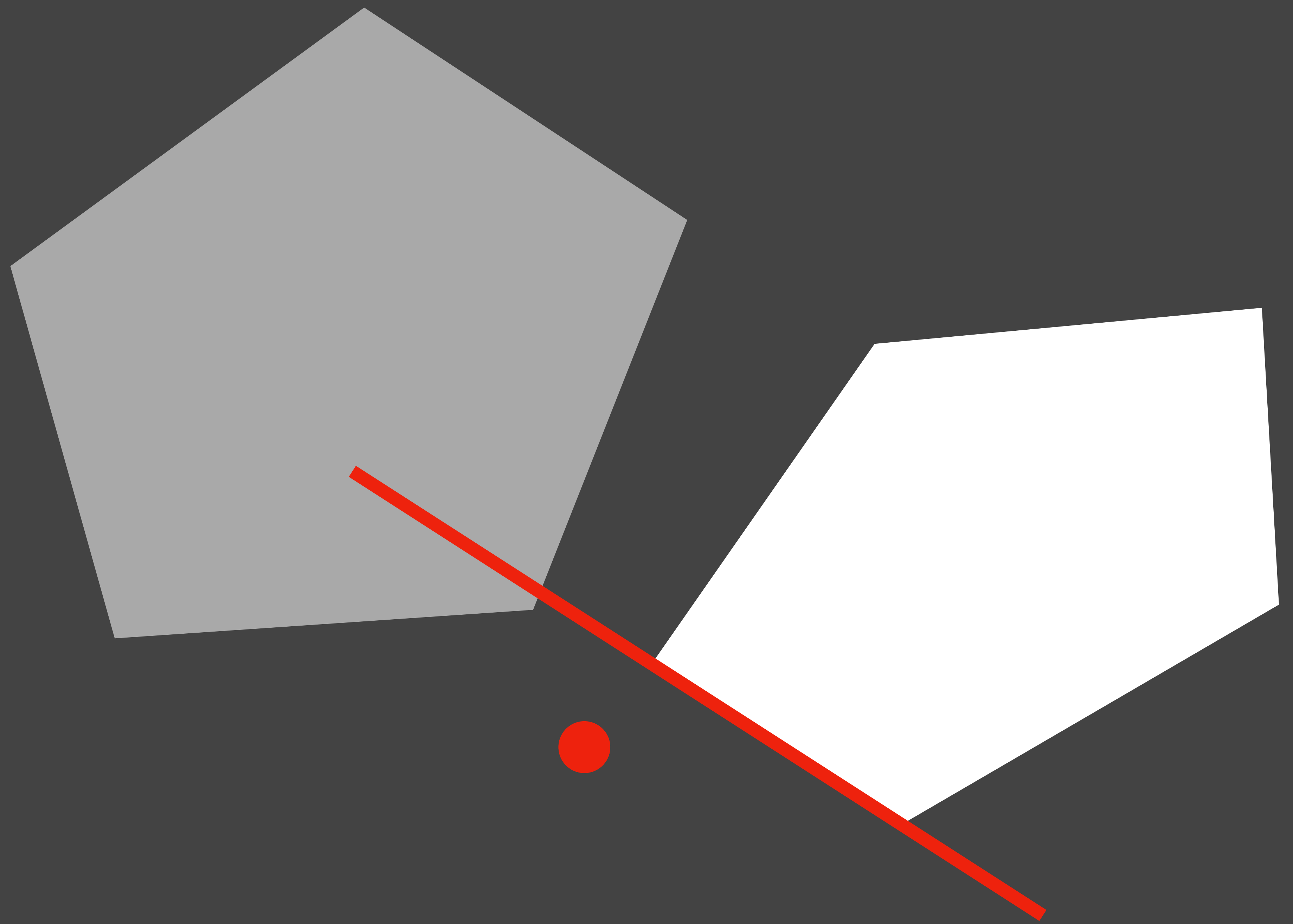
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



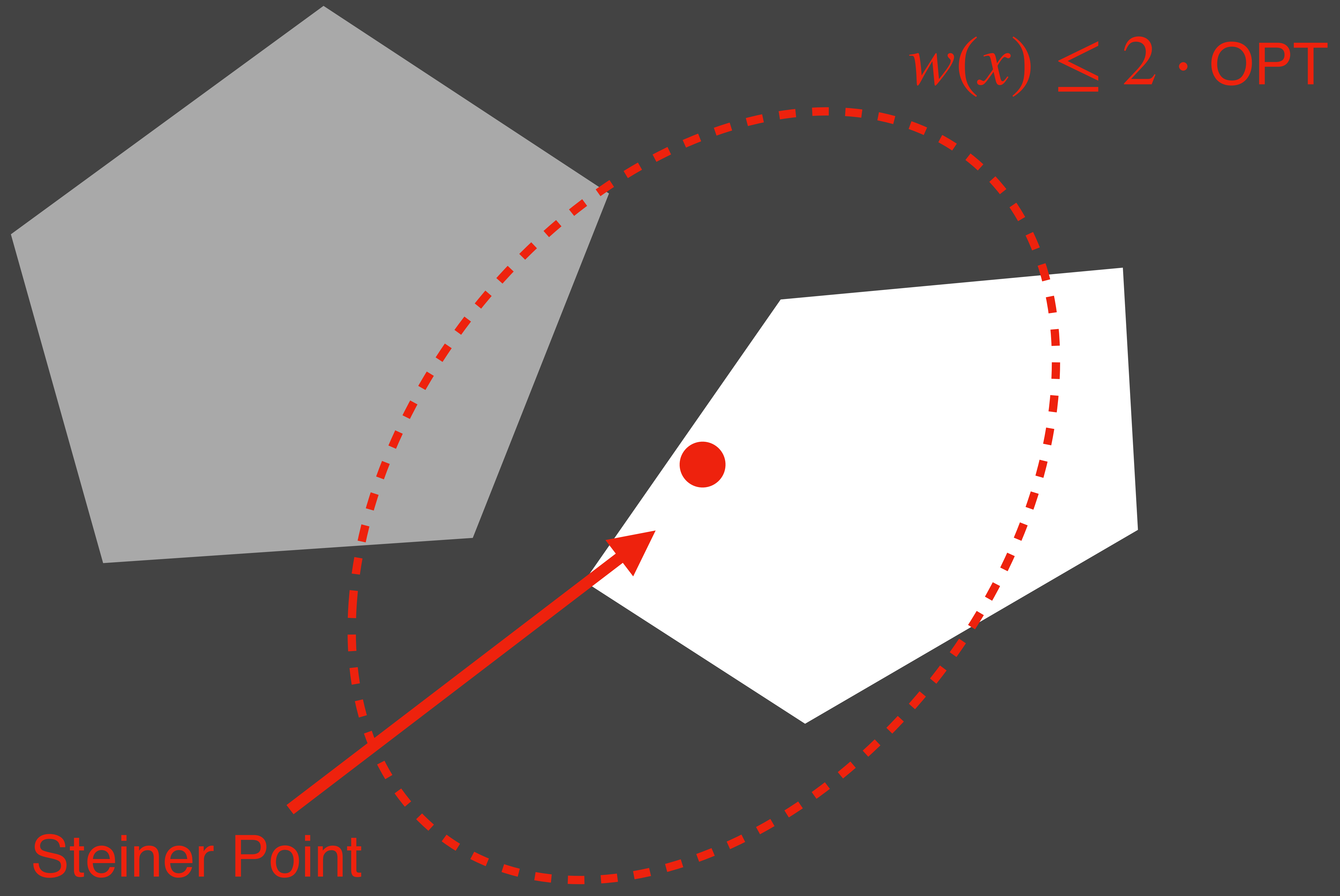
Our KL Projection Algorithm



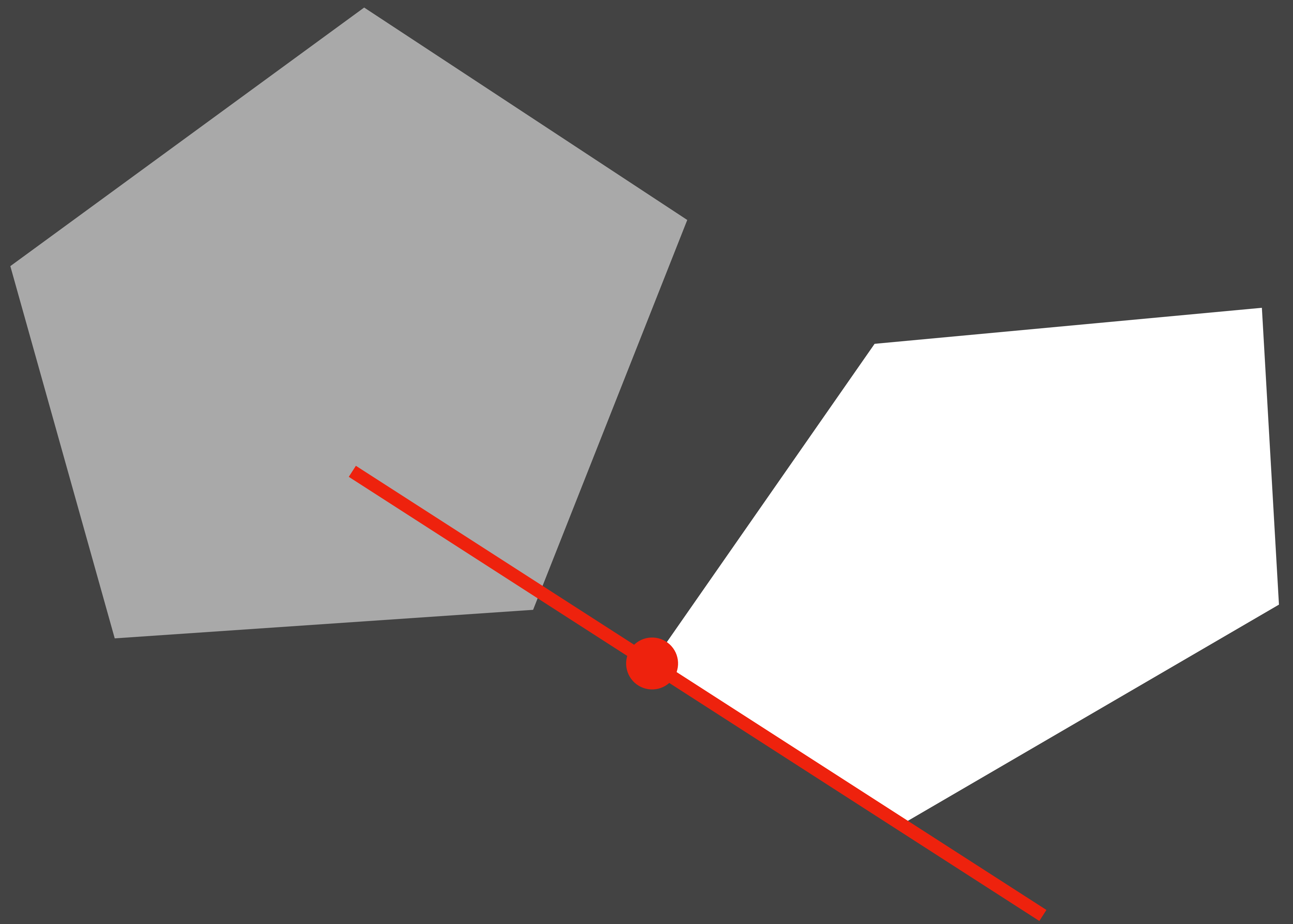
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



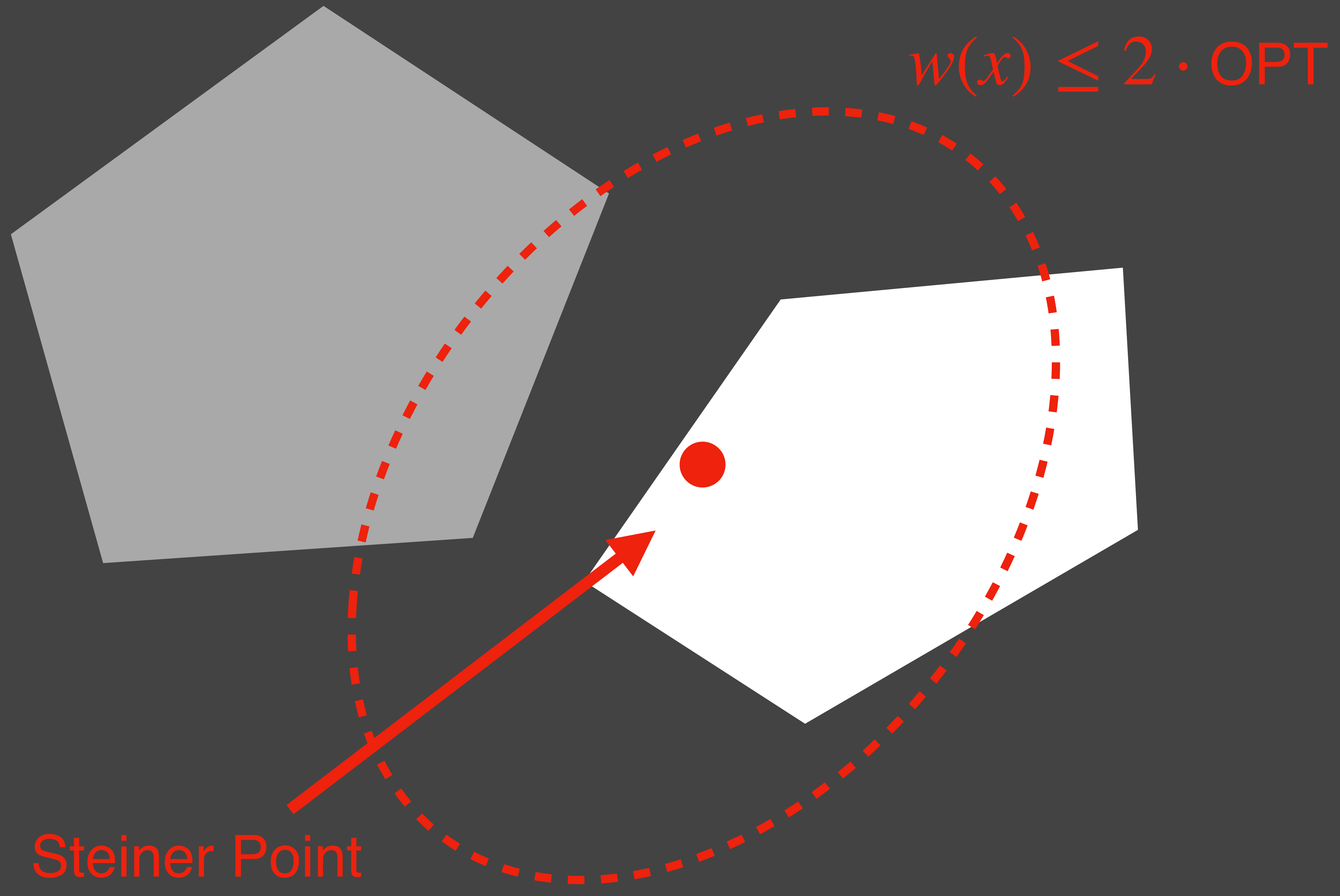
Our KL Projection Algorithm



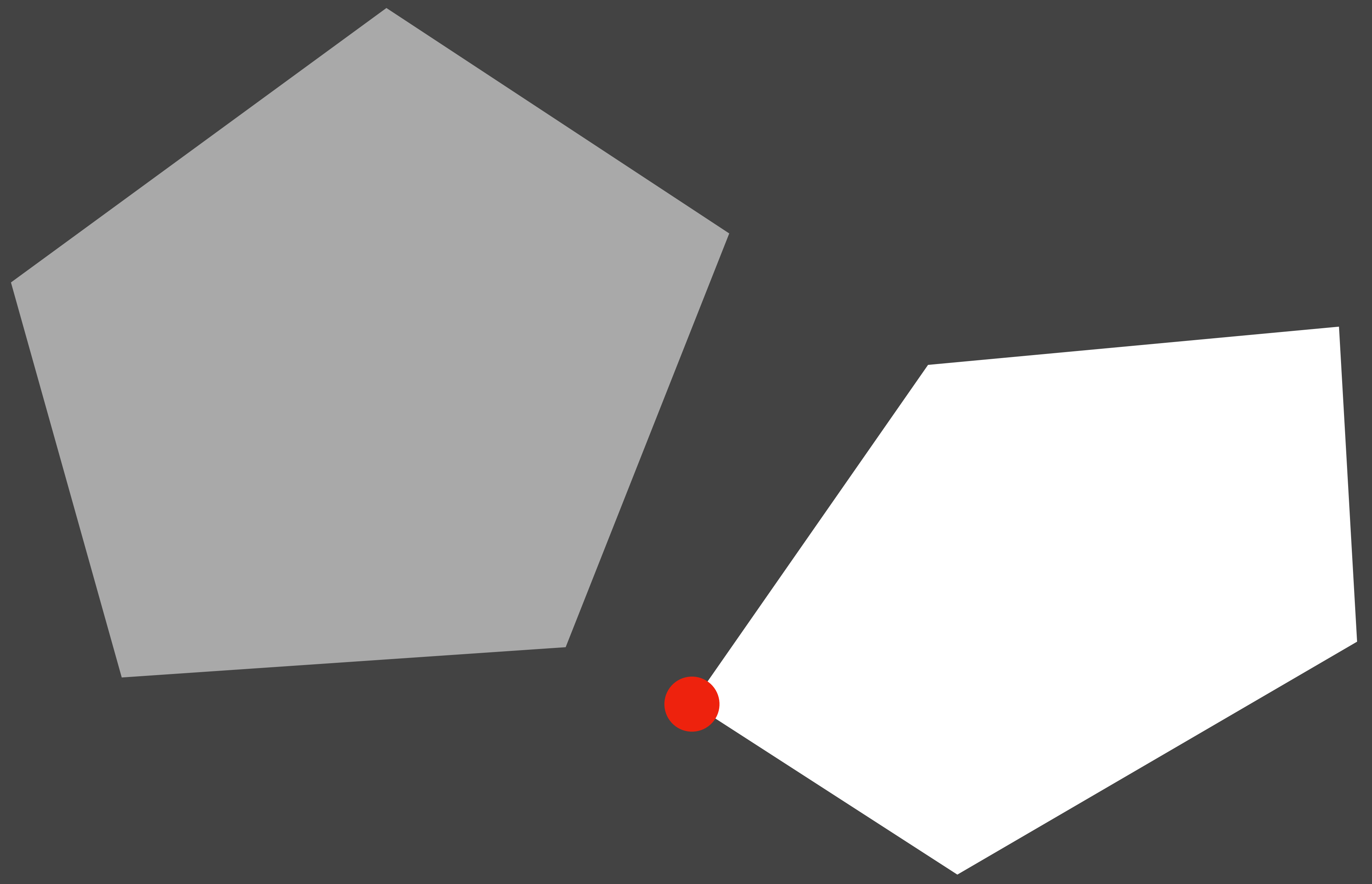
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



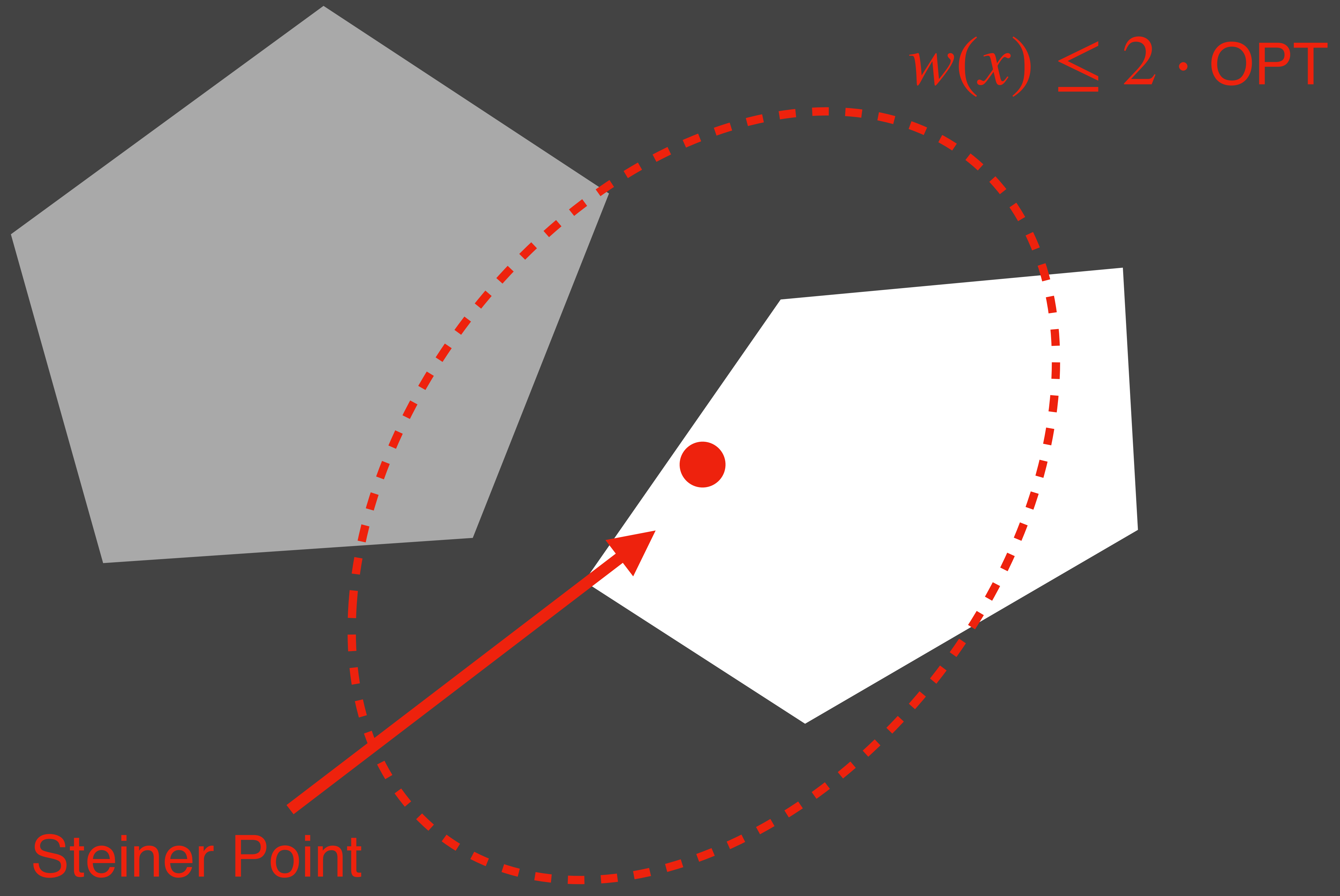
Our KL Projection Algorithm



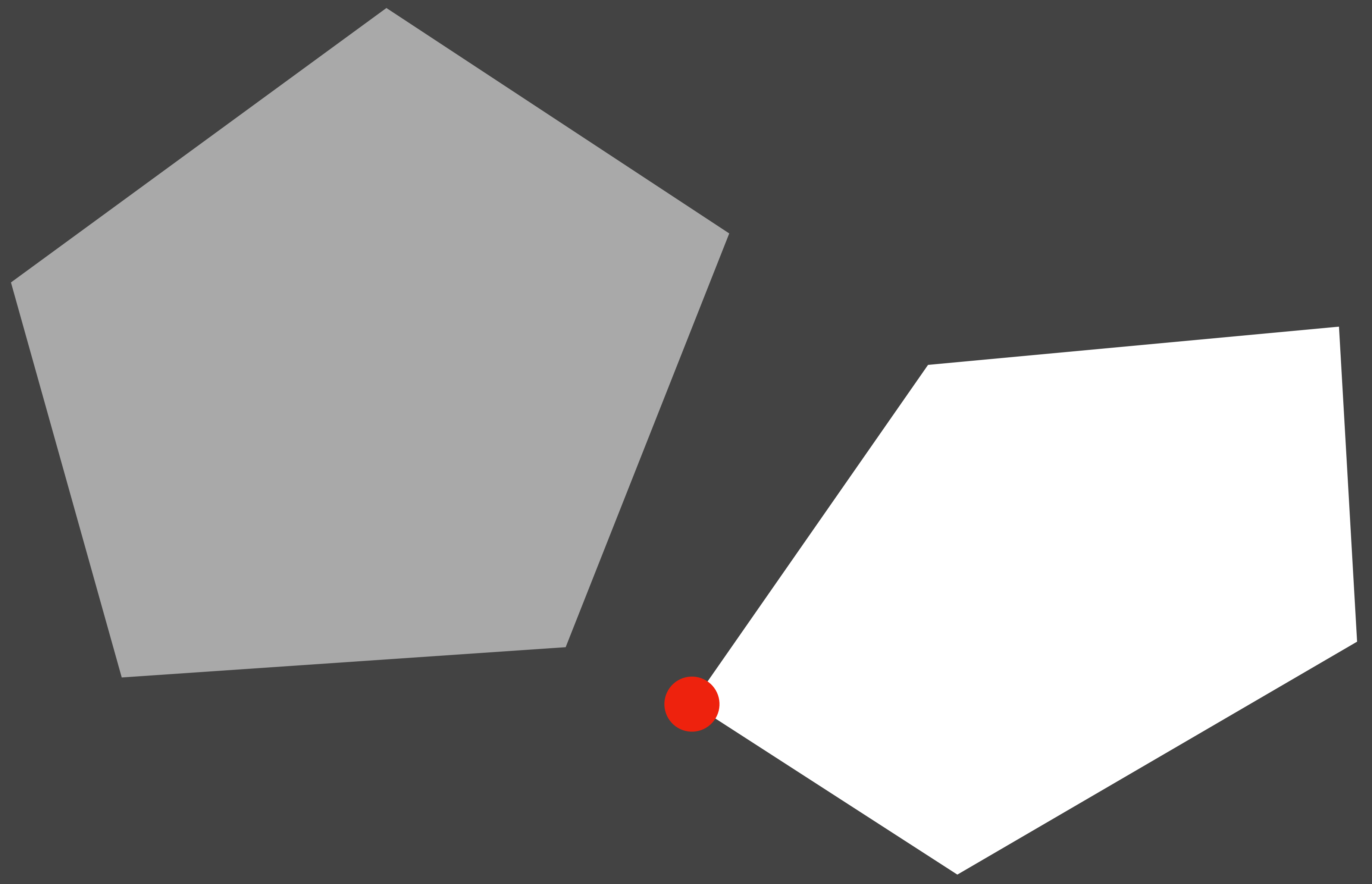
$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



Our KL Projection Algorithm

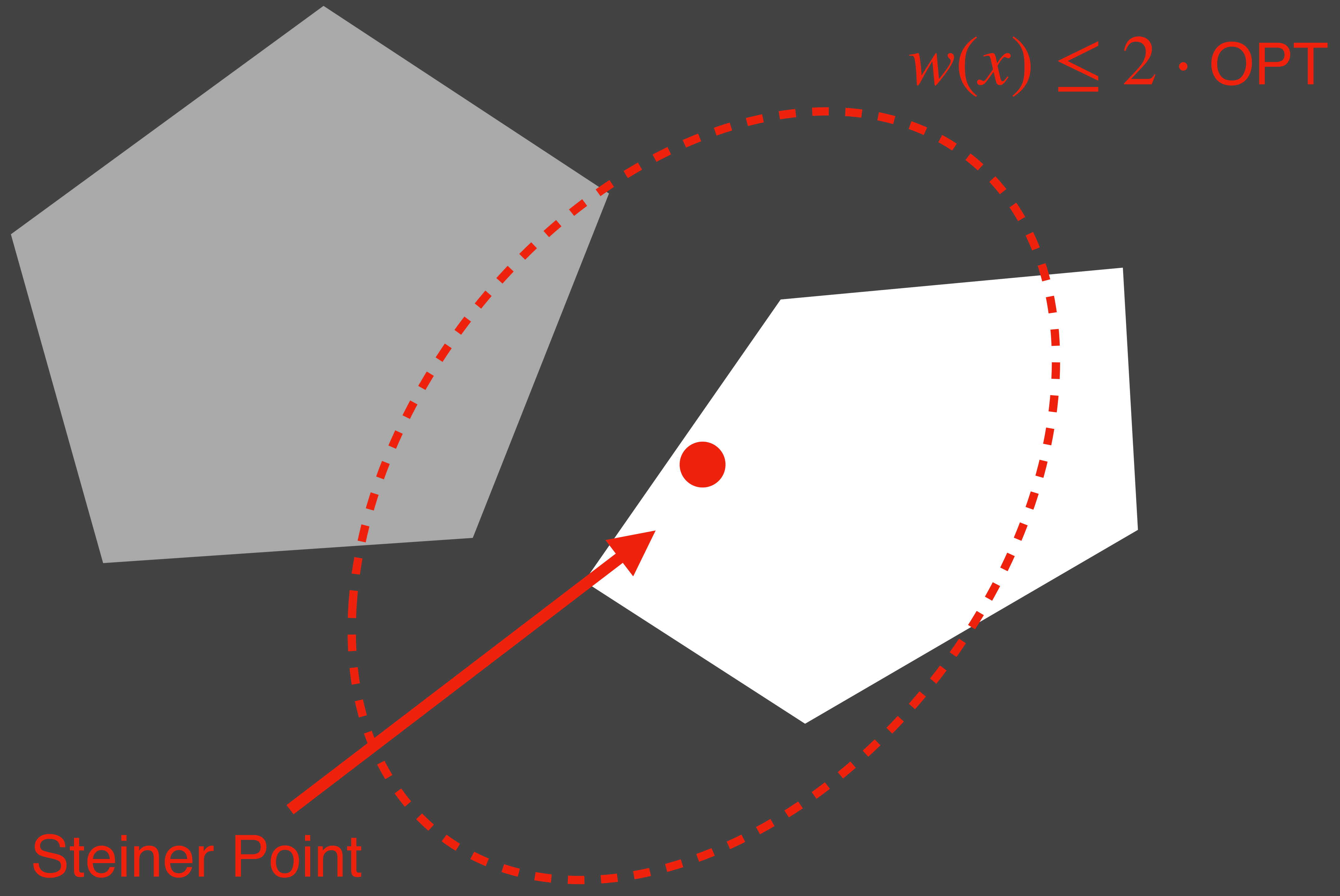


Memoryless! Feature or bug?

$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

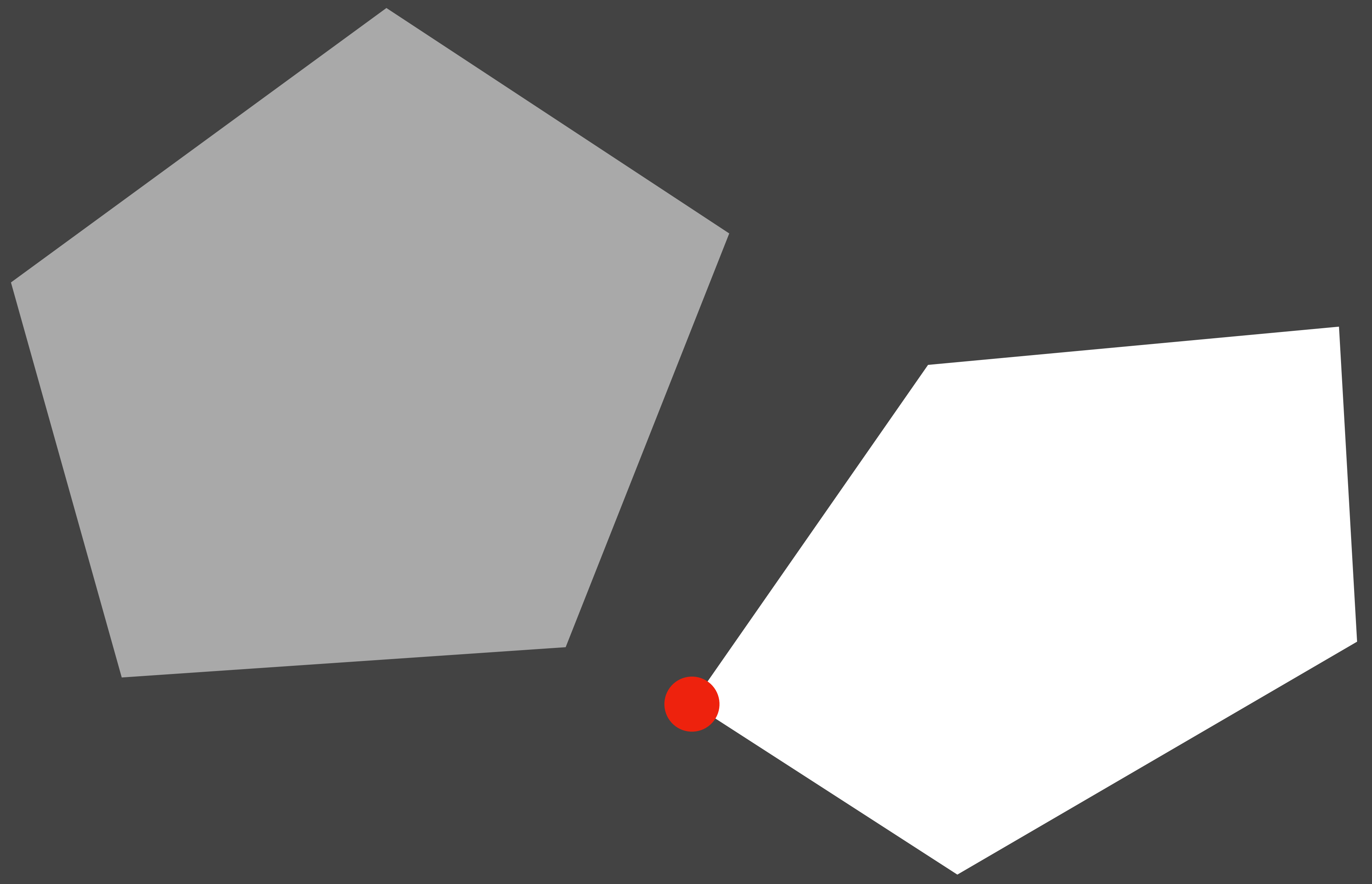
Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Our KL Projection Algorithm

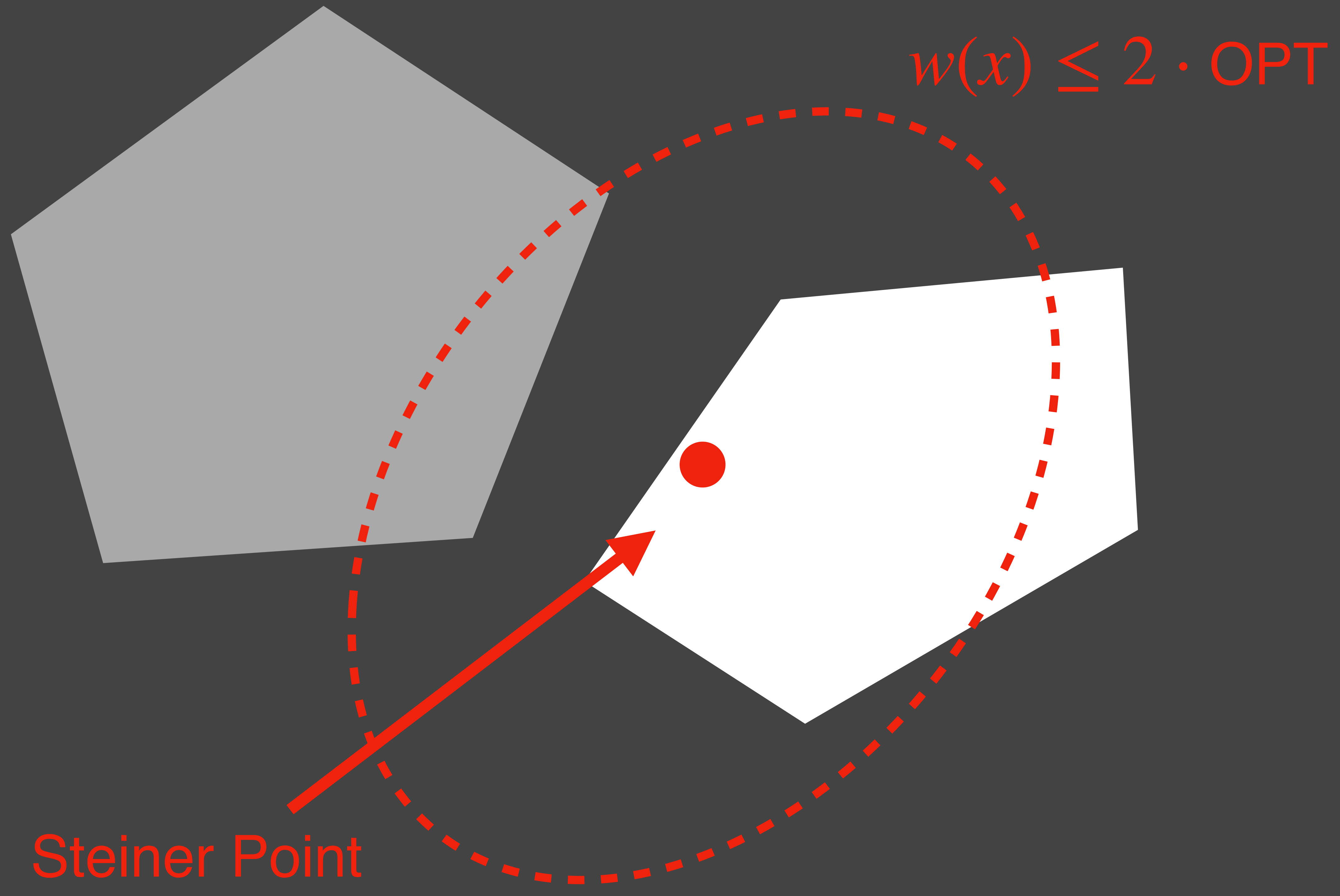


Memoryless! Feature or bug?

Simple, but leads to $1/\epsilon$ dependence in recourse…

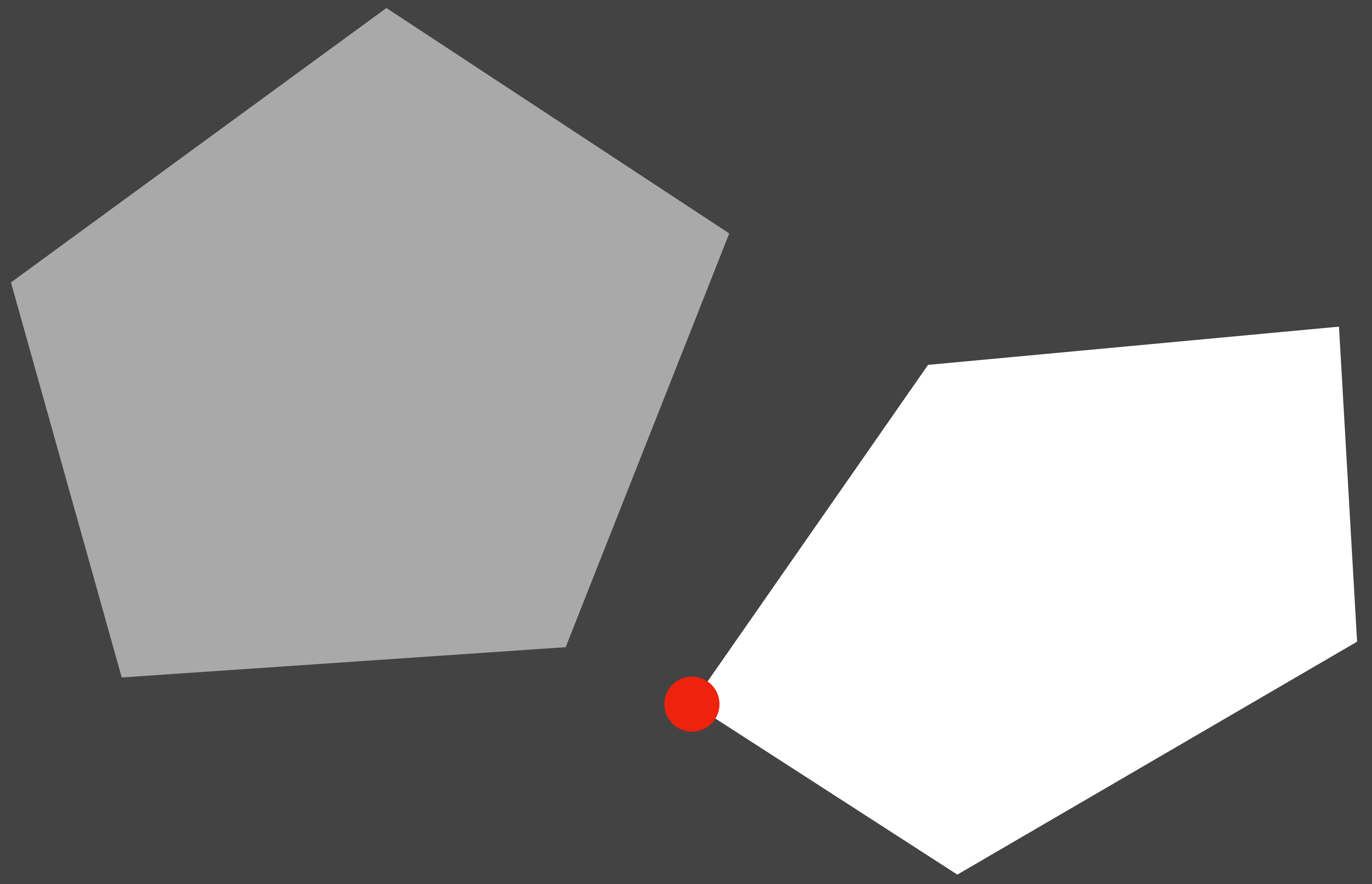
Comparison to General Body Chasing

Steiner Point Algorithm for Convex Body Chasing
[AGGT 20] & [Sellke 20]



$w(x)$:= optimal cost of chasing bodies so far, then moving to x.

Our KL Projection Algorithm



Memoryless! Feature or bug?

Simple, but leads to $1/\epsilon$ dependence in recourse…

Removing resource augmentation seems to need memory.

Improved Guarantee

Improved Guarantee

Beyond 0/1 matrices:

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(n\Delta/\epsilon)/\epsilon) \cdot$ OPT.

Improved Guarantee

Beyond 0/1 matrices:

<u>**Theorem**</u> **[BBLS]:**

Positive Body Chasing with movement $O(\log(n\Delta/\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

Improved Guarantee

Beyond 0/1 matrices:

<u>Theorem</u> [BBLS]:

Positive Body Chasing with movement $O(\log(n\Delta/\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

We improve!

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(\ f\ /\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

We improve!

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(f /\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

We improve!
f := max sparsity of any covering constraint c^t.

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(f/\epsilon)/\epsilon) \cdot$ OPT.

$\Delta :=$ Aspect ratio of constraint matrices.

We improve!
$f :=$ max sparsity of any covering constraint c^t.

Improvement crucial for some applications!
e.g. Hyperedge orientation.

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(f /\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

Most technical part of the paper.

We improve!

$f :=$ max sparsity of any covering constraint c^t.

Improvement crucial for some applications!

e.g. Hyperedge orientation.

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(f/\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

Most technical part of the paper.

We improve!
f := max sparsity of any covering constraint c^t.

Can show that improved guarantee impossible with online duals.

Improvement crucial for some applications!
e.g. Hyperedge orientation.

Improved Guarantee

Beyond 0/1 matrices:

Theorem [BBLS]:

Positive Body Chasing with movement $O(\log(\ f\ /\epsilon)/\epsilon) \cdot$ OPT.

Δ := Aspect ratio of constraint matrices.

We improve!
$f :=$ max sparsity of any covering constraint c^t.

Improvement crucial for some applications!
e.g. Hyperedge orientation.

Most technical part of the paper.

Can show that improved guarantee impossible with online duals.

To overcome, we go back in time and modify old duals. (ALG still online!)

Rounding

Sample Application 1 : Set Cover

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \ \ \forall e \in U^t, \ \sum_{S} x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Sample Application 1 : Set Cover

$\mathrm{OPT}^t :=$ cost of best cover @ time t.

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \mathrm{OPT}^t \right\}$$

Sample Application 1 : Set Cover

$\mathrm{OPT}^t :=$ cost of best cover @ time t.

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \quad \forall e \in U^t, \; \sum_{S} x_S \leq \beta \cdot \mathrm{OPT}^t \right\}$$

<u>Theorem</u> [BBLS]:

Dynamic Set Cover with:

(1) Approx $O(\log n) \cdot \beta \cdot \mathrm{OPT}^t$.

(1) Recourse $O(\log^2 n) \cdot \mathrm{OPT}_{\mathrm{recourse}}(\beta)$.

Sample Application 1 : Set Cover

OPT^t := cost of best cover @ time t.

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \; \sum_S x_S \leq \beta \cdot \text{OPT}^t \right\}$$

Theorem [BBLS]:

Dynamic Set Cover with:

(1) Approx $O(\log n) \cdot \beta \cdot \text{OPT}^t$.

(1) Recourse $O(\log^2 n) \cdot \text{OPT}_{\text{recourse}}(\beta)$.

$\text{OPT}_{\text{recourse}}(\beta)$:= best recourse if required to maintain β approx.

Sample Application 1 : Set Cover

$\mathsf{OPT}^t :=$ cost of best cover @ time t.

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Theorem [BBLS]:

Dynamic Set Cover with:

(1) Approx $O(\log n) \cdot \mathsf{OPT}^t$.

(1) Recourse $O(\log^2 n) \cdot \mathsf{OPT}_{\mathsf{recourse}}(O(1))$.

$\mathsf{OPT}_{\mathsf{recourse}}(\beta) :=$ best recourse if required to maintain β approx.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \text{OPT}^t \right\}$$

Rounding algo ($O(\log n)$ version):

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \ \ \forall e \in U^t, \ \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo ($O(\log n)$ version):

1. Give each set $\lambda_S \sim \mathrm{Exp}(\log n)$ before game begins.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \mathrm{OPT}^t \right\}$$

Rounding algo ($O(\log n)$ version):

1. Give each set $\lambda_S \sim \mathrm{Exp}(\log n)$ before game begins.
2. Hold set S in solution iff $x_S \geq \lambda_S$.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \; \sum_S x_S \leq \beta \cdot \text{OPT}^t \right\}$$

Rounding algo ($O(\log n)$ version):

1. Give each set $\lambda_S \sim \text{Exp}(\log n)$ before game begins.
2. Hold set S in solution iff $x_S \geq \lambda_S$.

Why good idea? Exp is memoryless, so $E[S^t \Delta S^{t-1}] \approx \log n \cdot \Delta x$.
$\sum_t \Delta x$ bounded by fractional algorithm!

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \ \ \forall e \in U^t, \ \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo ($O(\log n)$ version):

1. Give each set $\lambda_S \sim \mathsf{Exp}(\log n)$ before game begins.
2. Hold set S in solution iff $x_S \geq \lambda_S$.

Why good idea? Exp is memoryless, so $E[S^t \Delta S^{t-1}] \approx \log n \cdot \Delta x$.
$\sum_t \Delta x$ bounded by fractional algorithm!

Cost of solution is $O(\log n) \cdot \beta \cdot \mathsf{OPT}^t$, feasible w.h.p. @ every time t.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \quad \forall e \in U^t, \; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Theorem [BBLS]:

Dynamic Set Cover with:

(1) Approx $O(f) \cdot \beta \cdot \mathsf{OPT}^t$.

(1) Recourse $O(f \log f) \cdot \mathsf{OPT}_{\mathsf{recourse}}(\beta)$.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Theorem [BBLS]:

Dynamic Set Cover with:

(1) Approx $O(f) \cdot \beta \cdot \mathsf{OPT}^t$.

(1) Recourse $O(f \log f) \cdot \mathsf{OPT}_{\mathsf{recourse}}(\beta)$.

Covering constraint sparsity
$f :=$ max # sets any element is in

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Covering constraint sparsity
$f :=$ max # sets any element is in

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \text{OPT}^t \right\}$$

Rounding algo ($O(f)$ version):

Covering constraint sparsity
$f :=$ max # sets any element is in

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo ($O(f)$ version):

1. If x_S goes above $1/f$, buy S.

Covering constraint sparsity
$f :=$ max # sets any element is in

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \quad \forall e \in U^t, \;\; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo ($O(f)$ version):

1. If x_S goes above $1/f$, buy S.
2. If x_S drops below $1/2f$, remove S.

Covering constraint sparsity
$f :=$ max # sets any element is in

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \;\; \forall e \in U^t, \;\; \sum_S x_S \leq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo ($O(f)$ version):

1. If x_S goes above $1/f$, buy S.
2. If x_S drops below $1/2f$, remove S.

Covering constraint sparsity
$f :=$ max # sets any element is in

Cost of solution is $O(f) \cdot \beta \cdot \mathsf{OPT}^t$ and feasible.

Sample Application 1 : Set Cover

$$K_t = \left\{ x \;\middle|\; \sum_{S \ni e} x_S \geq 1 \quad \forall e \in U^t, \;\; \sum_{S} x_S \leq \beta \cdot \text{OPT}^t \right\}$$

Rounding algo ($O(f)$ version):

1. If x_S goes above $1/f$, buy S.
2. If x_S drops below $1/2f$, remove S.

Covering constraint sparsity
$f :=$ max # sets any element is in

Cost of solution is $O(f) \cdot \beta \cdot \text{OPT}^t$ and feasible.

Can charge every set purchase to $1/2f$ movement of fractional solution.

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \leq 1 \;\; \forall v \in V, \; \sum_{e} x_e \geq \beta \cdot \mathsf{OPT}^t \right\}$$

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \leq 1 \;\; \forall v \in V, \; \sum_e x_e \geq \beta \cdot \mathsf{OPT}^t \right\}$$

Theorem [BBLS]:

Dynamic Bipartite Matching with:

(1) Approx $(1-\epsilon)\cdot\beta\cdot\mathsf{OPT}^t$.

(1) Recourse $O(\log n)\cdot\mathsf{OPT}_{\mathsf{recourse}}(\beta)$.

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \le 1 \;\; \forall v \in V, \; \sum_e x_e \ge \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo:

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \le 1 \;\; \forall v \in V, \; \sum_e x_e \ge \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo:

1. Maintain subgraph H containing $(1-\epsilon)$ matching with $\sum_t |H^t \Delta H^{t-1}| = O(\log n) \sum_t \Delta x$.

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \le 1 \;\; \forall v \in V,\; \sum_e x_e \ge \beta \cdot \mathrm{OPT}^t \right\}$$

Rounding algo:

1. Maintain subgraph H containing $(1-\epsilon)$ matching with $\sum_t |H^t \Delta H^{t-1}| = O(\log n) \sum_t \Delta x.$

(We call this a stabilizer!)

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \leq 1 \quad \forall v \in V, \; \sum_e x_e \geq \beta \cdot \mathsf{OPT}^t \right\}$$

Rounding algo:

1. Maintain subgraph H containing $(1-\epsilon)$ matching with $\sum_t |H^t \Delta H^{t-1}| = O(\log n) \sum_t \Delta x$.

(We call this a stabilizer!)

2. Run absolute recourse algorithm on H: $(1+\epsilon)$ approx with $O(\epsilon^{-1})$ recourse.

Sample Application 2 : Bipartite Matching

$$K_t = \left\{ x \;\middle|\; \sum_{e \in \partial(v)} x_e \le 1 \;\; \forall v \in V, \; \sum_e x_e \ge \beta \cdot \mathrm{OPT}^t \right\}$$

Rounding algo:

1. Maintain subgraph H containing $(1-\epsilon)$ matching with
$\sum_t |H^t \Delta H^{t-1}| = O(\log n) \sum_t \Delta x$.

(We call this a stabilizer!)

2. Run absolute recourse algorithm on H:
$(1+\epsilon)$ approx with $O(\epsilon^{-1})$ recourse.

Matching recourse $= O(\epsilon^{-1}) \cdot$ [# edges updates to H] $= O(\epsilon^{-1}) \cdot [O(\log n) \cdot \mathrm{OPT}_{\text{recourse}}(\beta)]$.

Conclusion

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

<u>Theorem</u> [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

Open Questions:

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

Open Questions:

Rounding for other problems?

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

Open Questions:

Rounding for other problems?

Other important + tractable families of Convex Body Chasing?

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

Theorem [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

Open Questions:

Rounding for other problems?

Other important + tractable families of Convex Body Chasing?

E.g. Can we capture clustering problems?

Conclusion

Introduce fundamental primitive (mixed packing cover LP) to dynamic algos.

<u>Theorem</u> [Bhattacharya, Buchbinder, L., Saranurak, In submission]:

Positive Body Chasing with movement $O_\epsilon(\log n) \cdot$ OPT.

Use to give SotA algorithms for Set Cover, Matching, Load Balancing, MST.

<u>Open Questions:</u>

Rounding for other problems?

Other important + tractable families of Convex Body Chasing?

E.g. Can we capture clustering problems?

Remove resource augmentation?

Thanks!