Mathematical Program Networks

Forrest Laine

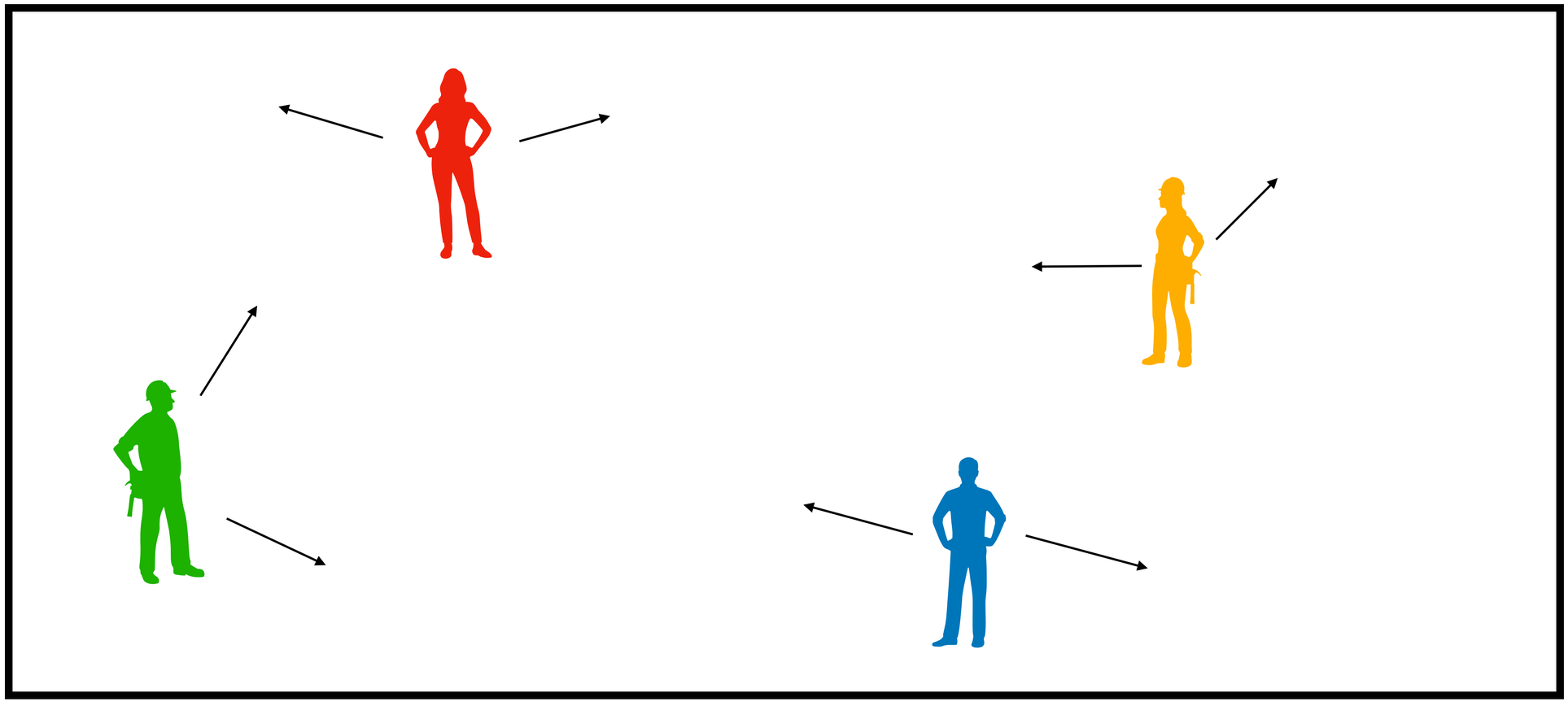
CEO, Eiro Inc
Assistant Professor of Computer Science, Vanderbilt University

Outline

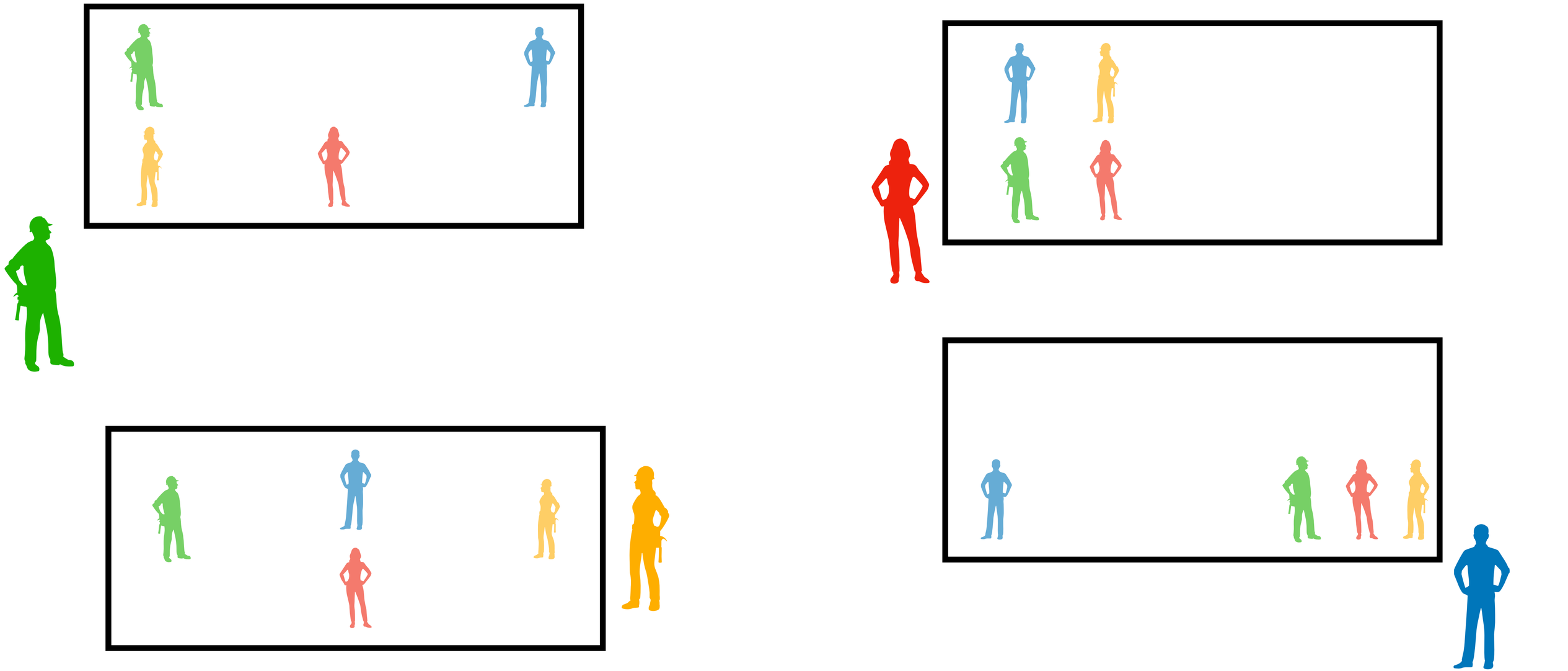
- Motivation
- Theory
- Computation
- Examples

When modeling real problems as games, the way players reason about each other is a modeling choice.

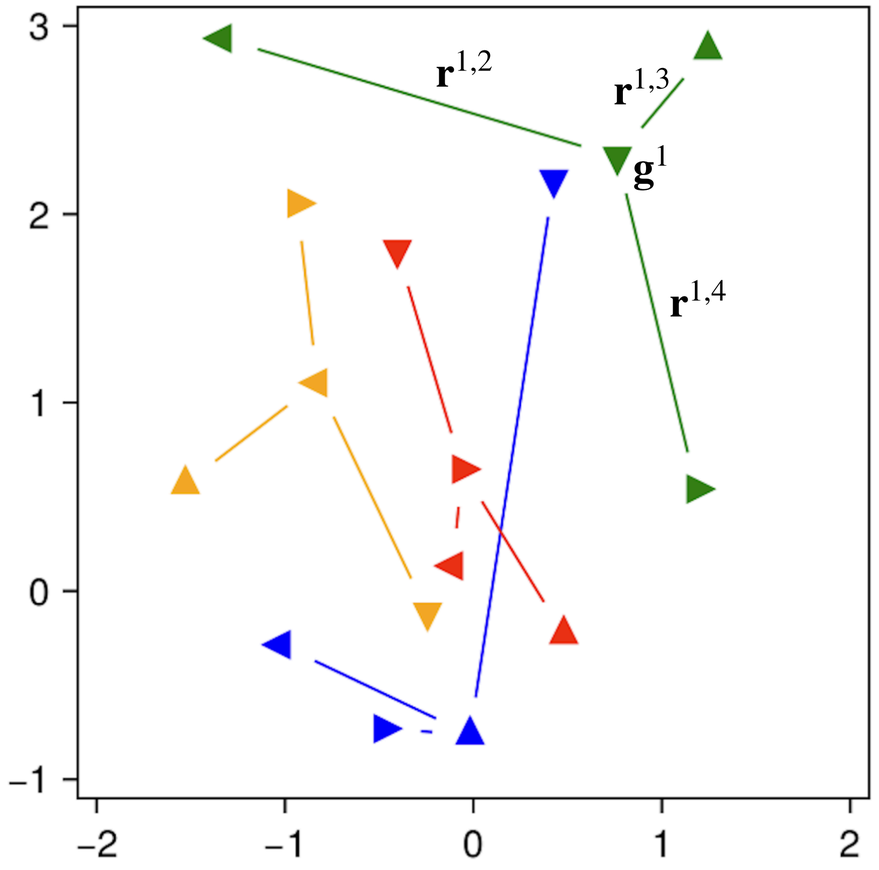
Consider a game played between four decision-makers.



Consider a game played between four decision-makers.



Consider a game played between four decision-makers.



Game played over $\mathbb{R}^8$. Decision variable index sets:

$$J^1 := \{1, 2\}, \quad J^2 := \{3, 4\}, \quad J^3 := \{5, 6\}, \quad J^4 := \{7, 8\},$$

Cost function for each player:

$$f^i(\mathbf{x}) := \|\mathbf{x}^i - \mathbf{g}^i\|_2^2 + \sum_{j=1, j\neq i}^{4} \|\mathbf{x}^j - \mathbf{x}^i - \mathbf{r}^{i,j}\|_2^2,$$

Feasible set for each player:

$$C^i := \{\mathbf{x} \in \mathbb{R}^8 : -5 \leq x_j \leq 5, \quad \forall j \in J^i\}.$$

How is equilibrium defined?

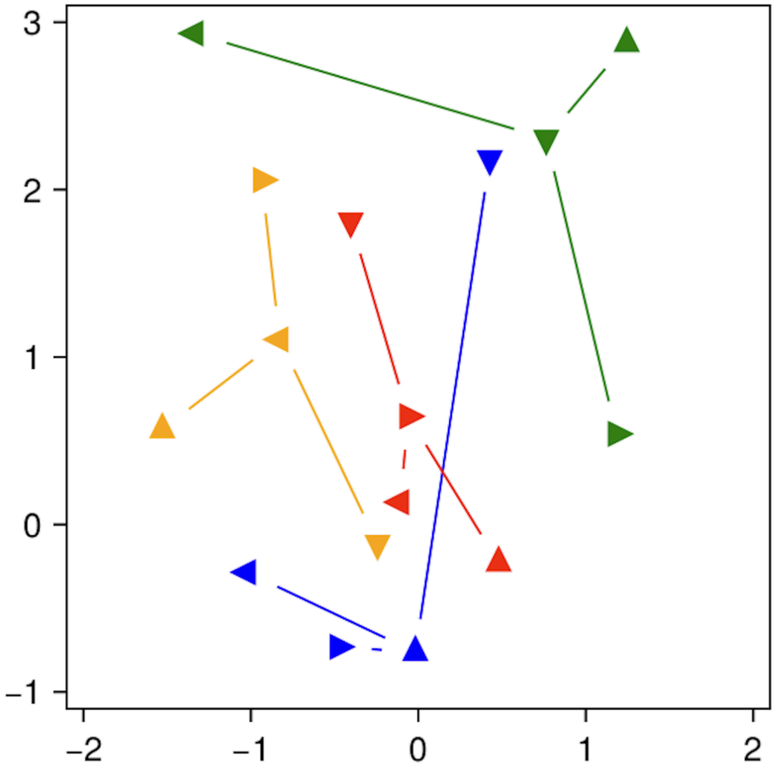
One definition: **Nash**

$$\mathbf{x}^* \in S^1 \cap S^2 \cap S^3 \cap S^4$$

$$S^i := \left\{ \begin{array}{l} \mathbf{x}^* \in \mathbb{R}^8 : (\mathbf{x}^*)^i \in \arg\min\limits_{\mathbf{x}^i} f^i(\mathbf{x}^i, (\mathbf{x}^*)^{\neg i}) \\ \qquad\qquad \text{s.t.} \quad (\mathbf{x}^i, (\mathbf{x}^*)^{\neg i}) \in C^i \end{array} \right\}$$

How to solve?

- Each problem is QP
- Solutions are projections of KKT points
- Find point simultaneously satisfying all KKT conditions (solve LCP), project into primal space



How is equilibrium defined?

What about other definitions?

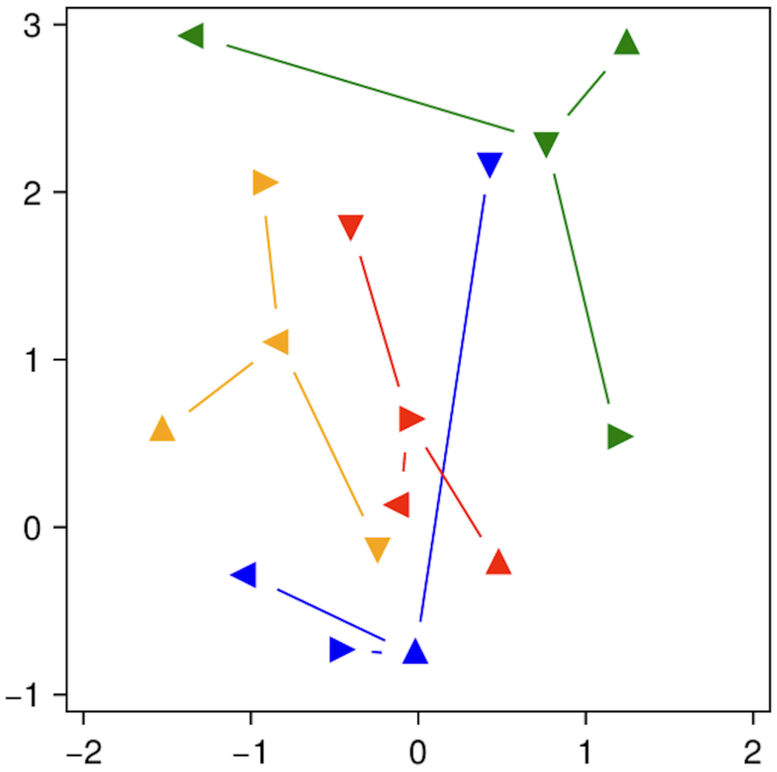
- Stackelberg?
- Multi-leader-multi-follower?
- Other?

What is constant among these formulations? what is different?

How to formalize?

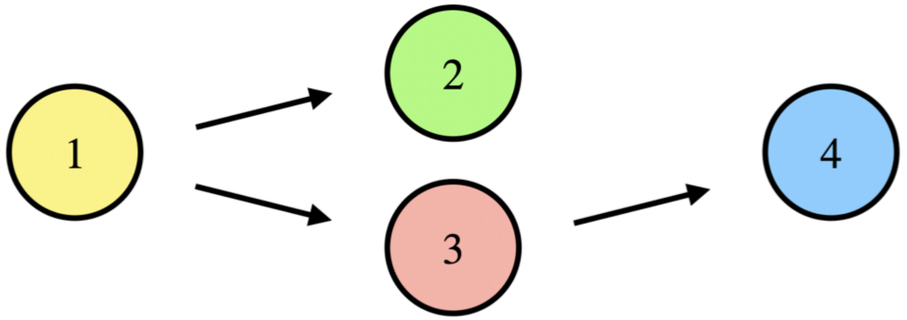
How to compute equilibria?

Mathematical Program Networks aims to provide a framework for answering these questions.



Games as Networks

- A Mathematical Program Network is defined as a directed acyclic graph, where nodes are mathematical programs (decision problems).
- N nodes, each defined by a tuple $\{f^i, C^i, J^i\}$.
  - $f^i : \mathbb{R}^n \to \mathbb{R}$, cost function
  - $C^i \subset \mathbb{R}^n$, feasible set
  - $J^i \subset [n]$, decision variable index set
- Network edges captured in set $E \subset [N] \times [N]$



Mathematical Program Networks

Technical definitions

- Private decision variables:
  $\mathbf{x}^i \in \mathbb{R}^{n^i} \quad := [x_j]_{j \in J^i}.$
- Reachable transitions:
  $R \subset [N] \times [N]$
- Influenceable decision variables:

$$D^i := \{i\} \cup \{j : (i,j) \in R\}$$

$$\mathbf{x}^{D^i} := [\mathbf{x}^j]_{j \in D^i}$$

- Solution graph:

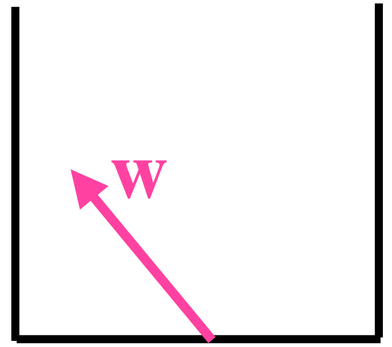
$$S^i := \left\{ \mathbf{x}^* \in \mathbb{R}^n \;:\; (\mathbf{x}^*)^{D^i} \in \underset{\mathbf{x}^{D^i}}{\operatorname{argmin}} \; f^i\left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \;\text{s.t.}\; \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in C^i \;\; \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in S^j,\; (i,j) \in E \right\}$$

- Equilibrium:
  $\mathbf{x}^* \in \bigcap_{i \in [N]} S^i$

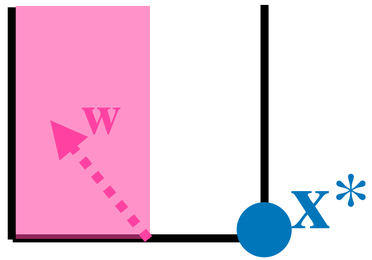
Note: the optimization problem at each node changes depending on network edge structure.

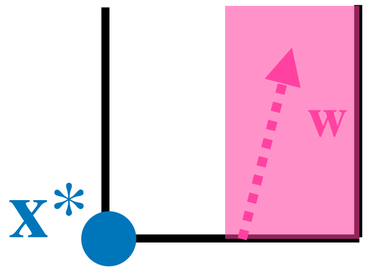
Solution Graph Example

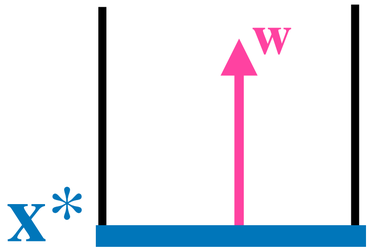
$$S := \left\{ \begin{array}{l} (\mathbf{x}^*, \mathbf{w}) : \mathbf{x}^* \in \arg\min\limits_{x \in \mathbb{R}^2} \ \mathbf{x}^\intercal \mathbf{w} \\ \qquad\qquad \text{s.t.} \ \ -1 \le x_1 \le 1 \\ \qquad\qquad\qquad\ \ 0 \le x_2 \end{array} \right\}$$



$$S := \left\{ \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{w} \right) : w_1 \le 0,\ w_2 \ge 0 \right\} \cup \left\{ \left(\begin{bmatrix} -1 \\ 0 \end{bmatrix}, \mathbf{w} \right) : w_1 \ge 0,\ w_2 \ge 0 \right\} \cup \left\{ \left(\mathbf{x}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) : -1 \le x_1 \le 1,\ x_2 = 0 \right\}$$

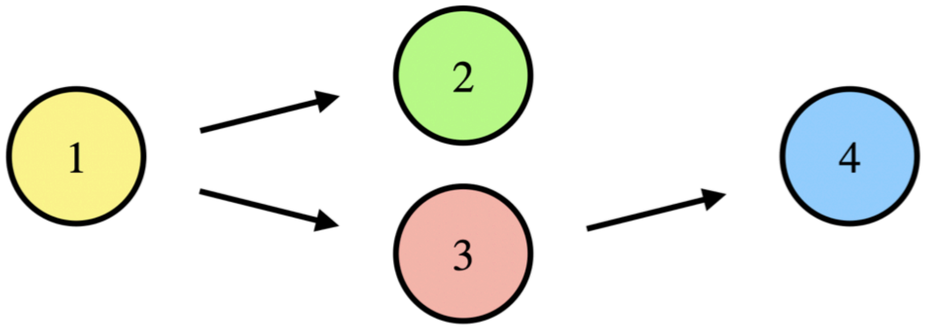






Mathematical Program Networks

Example



$$S^1 := \left\{ \mathbf{x}^* \in \mathbb{R}^n \;:\; \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \; f^i(\mathbf{x}) \;\; \text{s.t.} \;\; \mathbf{x} \in C^1,\; \mathbf{x} \in S^2,\; \mathbf{x} \in S^3 \right\}$$

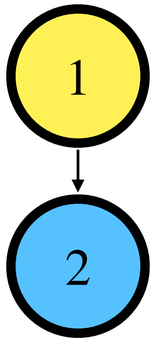
$$S^2 := \left\{ \mathbf{x}^* \in \mathbb{R}^n \;:\; (\mathbf{x}^*)^2 \in \underset{\mathbf{x}^2}{\operatorname{argmin}} \; f^2((\mathbf{x}^*)^1, \mathbf{x}^2, (\mathbf{x}^*)^3, (\mathbf{x}^*)^4) \;\; \text{s.t.} \;\; ((\mathbf{x}^*)^1, \mathbf{x}^2, (\mathbf{x}^*)^3, (\mathbf{x}^*)^4) \in C^2 \right\}$$

$$S^3 := \left\{ \mathbf{x}^* \in \mathbb{R}^n \;:\; ((\mathbf{x}^*)^3, (\mathbf{x}^*)^4) \in \underset{\mathbf{x}^3, \mathbf{x}^4}{\operatorname{argmin}} \; f^3((\mathbf{x}^*)^1, (\mathbf{x}^*)^2, \mathbf{x}^3, \mathbf{x}^4) \;\; \text{s.t.} \;\; ((\mathbf{x}^*)^1, (\mathbf{x}^*)^2, \mathbf{x}^3, \mathbf{x}^4) \in C^3,\; ((\mathbf{x}^*)^1, (\mathbf{x}^*)^2, \mathbf{x}^3, \mathbf{x}^4) \in S^4 \right\}$$

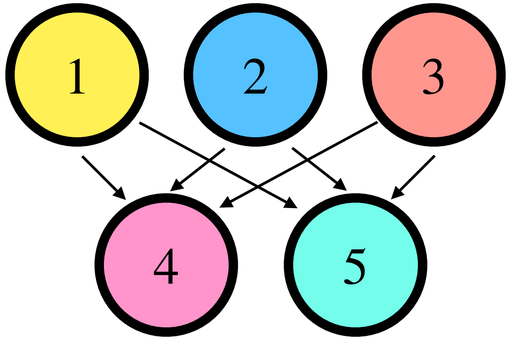
$$S^4 := \left\{ \mathbf{x}^* \in \mathbb{R}^n \;:\; (\mathbf{x}^*)^4 \in \underset{\mathbf{x}^4}{\operatorname{argmin}} \; f^4((\mathbf{x}^*)^1, (\mathbf{x}^*)^2, (\mathbf{x}^*)^3, \mathbf{x}^4) \;\; \text{s.t.} \;\; ((\mathbf{x}^*)^1, (\mathbf{x}^*)^2, (\mathbf{x}^*)^3, \mathbf{x}^4) \in C^4 \right\}$$

One framework, many equilibrium concepts.

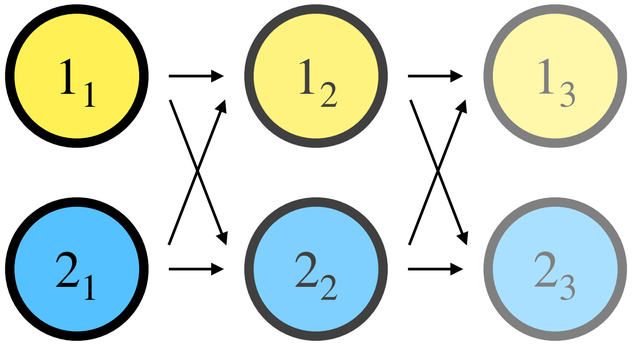
Nash



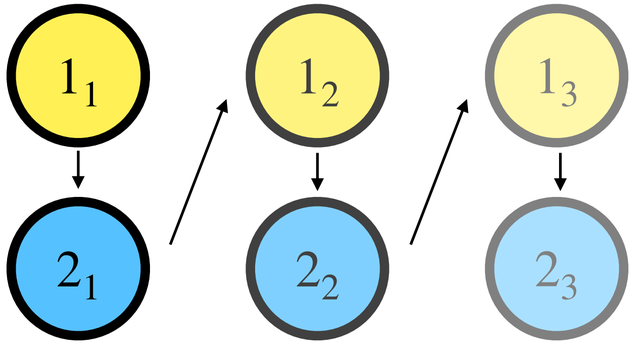
Stackelberg / Bilevel



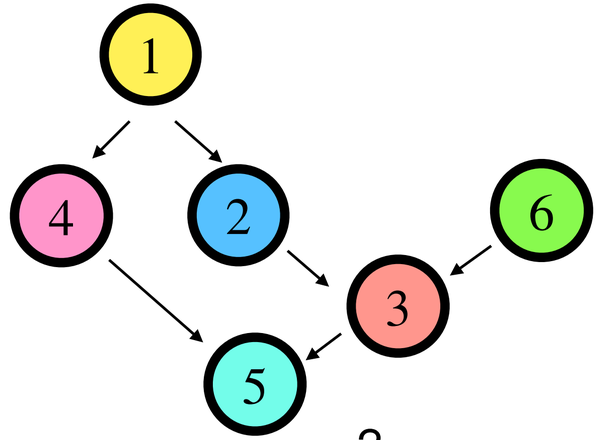
Multi-leader-multi-follower



Feedback Nash

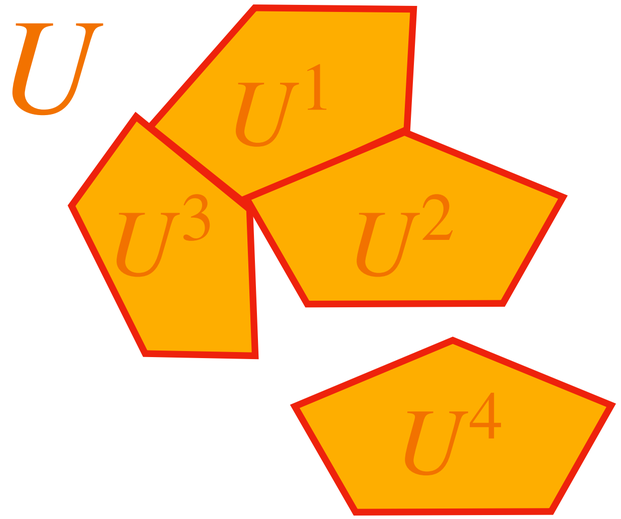


Feedback Stackelberg



?

What about computation?



- Formulating problems is one thing, but can they actually be solved?

Consider some union of sets: $U := \bigcup_{j \in J} U^j,$

Define the local index set: $\gamma_U(\mathbf{x}, \mathbf{w}) := \{j \in J : (\mathbf{x}, \mathbf{w}) \in \overline{U^j}\}.$

Indices of of all sets for which $(\mathbf{x}, \mathbf{w})$ belongs

And the two associated solution graphs for a problem defined by the objective f and feasible set U:

$$R := \left\{ (\mathbf{x}^*, \mathbf{w}) : \ \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \quad f(\mathbf{x}, \mathbf{w}) \right.$$
$$\left. \text{s.t.} \quad (\mathbf{x}, \mathbf{w}) \in U \right\}$$

Solution graph of interest

$$R^j := \left\{ (\mathbf{x}^*, \mathbf{w}) : \ \mathbf{x}^* \in \underset{\mathbf{x}}{\operatorname{argmin}} \quad f(\mathbf{x}, \mathbf{w}) \right.$$
$$\left. \text{s.t.} \quad (\mathbf{x}, \mathbf{w}) \in \overline{U^j} \right\}.$$

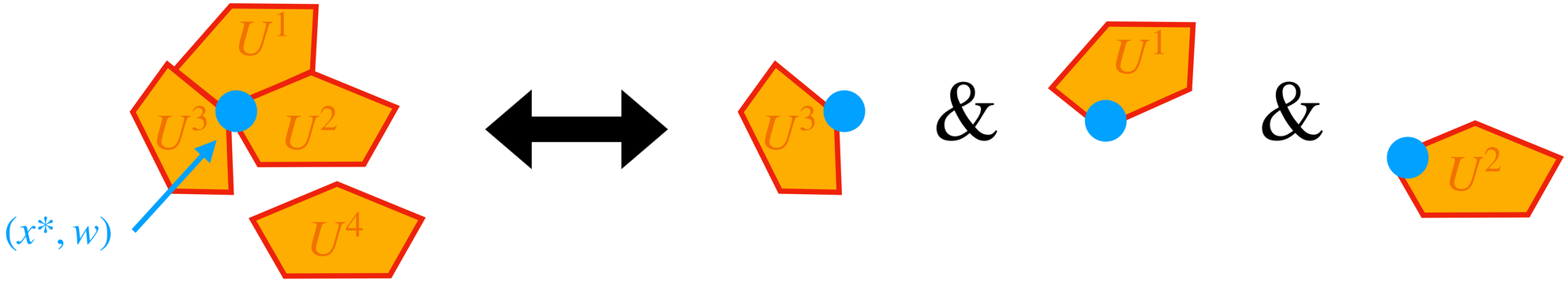
Solution graph of local approximate problem j

Union constraints can be broken up and considered independently

Lemma 5.4 *Given the definitions of* R *and* R^j *in* (26) *and* (27),

$$(\mathbf{x}^*, \mathbf{w}) \in R \iff \left((\mathbf{x}^*, \mathbf{w}) \in U, \text{ and } (\mathbf{x}^*, \mathbf{w}) \in R^j \ \forall j \in \gamma_U(\mathbf{x}^*, \mathbf{w})\right).$$

For parameter $\mathbf{w}$, the value $\mathbf{x}^*$ is a solution iff it is a solution for all local restrictions to the problem.



Union constraints can be broken up and considered independently

Corollary 5.2 *Given the definitions of R and R^j in (26) and (27), there exists $\epsilon > 0$ such that*

$$R \cap \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) = \bigcup_{\substack{J_1 \in \mathcal{P}(\gamma_U(\mathbf{x},\mathbf{w})) \\ J_2 = \gamma_U(\mathbf{x},\mathbf{w}) \setminus J_1}} \left(U \bigcap_{j_1 \in J_1} R^{j_1} \bigcap_{j_2 \in J_2} \left(\overline{U^{j_2}} \right)' \bigcap \mathcal{N}_\epsilon(\mathbf{x}, \mathbf{w}) \right). \quad (39)$$

This result is constructive — it implies a method for building a local representation of a solution graph.

Quadratic Program Networks

- The previous results are especially germane to *Quadratic Program Networks (QPNs)*, which have desirable computational properties
- QPNs: Mathematical Program Networks in which, for each decision node, the cost function is convex quadratic, and the feasible set is polyhedral

$$S^i := \left\{ \mathbf{x}^* \in \mathbb{R}^n \ : \ (\mathbf{x}^*)^{D^i} \in \underset{\mathbf{x}^{D^i}}{\operatorname{argmin}} \quad f^i\left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \right.$$
$$\left. \text{s.t.} \quad \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in C^i \right.$$
$$\left. \left(\mathbf{x}^{D^i}, (\mathbf{x}^*)^{D^{-i}}\right) \in S^j, \ (i,j) \in E \right\}$$

Quadratic, convex in $\mathbf{x}^{D^i}$

Polyhedral
(finite intersection of halfspaces)

Solution graphs are nice for QPNs

Lemma 5.7 *The solution graph of any node i in an acyclic QPN can be expressed as a union of not-necessarily closed polyhedral regions, i.e.*

$$S^i = \bigcup_{j \in J_i} P^j, \tag{50}$$

This means previous results apply — and each "local" problem is a QP! Solution graphs can be constructed.

Solution graph for QP:

$$S = \bigcup_{L \in \mathcal{P}([l])} \left\{ (\mathbf{x}, \mathbf{w}) \in C : \begin{array}{l} (\mathbf{x}, \mathbf{w}) \in \cap_{i \in L} \partial H^i \\ \nabla_x f(\mathbf{x}, \mathbf{w}) \in \operatorname{coni}(\{\mathbf{a}^i\}_{i \in L}) \end{array} \right\}.$$

Existence of equilibrium is very hard to guarantee, even for QPNs.

Nevertheless, searching for equilibrium remains useful and interesting.

Checking for equilibrium:

Algorithm 1 QPN Equilibrium Check

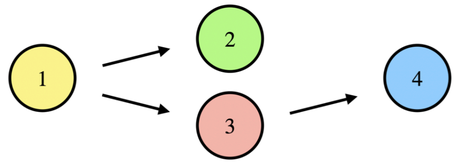
Require: QPN
Require: Candidate $\mathbf{x} \in \mathbb{R}^n$
Require: Reverse topological ordering T over the nodes in the QPN.

for $i \in T$ **do**
  $U \leftarrow C^i \bigcap_{j:(i,j)\in E} \tilde{S}^j_\epsilon(\mathbf{x})$
  for $U^j \in U$ **do**
    Compute $R^j_\epsilon(\mathbf{x}) = R^j \cap \mathcal{N}_\epsilon(\mathbf{x})$ (27)
    if $\mathbf{x} \notin R^j_\epsilon(\mathbf{x})$ **then**
      Return false
    end if
  end for
  Compute $\tilde{S}^i_\epsilon(\mathbf{x}) = R \cap N_\epsilon(\mathbf{x})$ (39)
end for
Return true

Gather local polyhedral pieces of child nodes

Using one piece at a time, compute individual solution graphs

Combine individual solution graphs into graph for node, move on to parent



Searching for equilibrium:

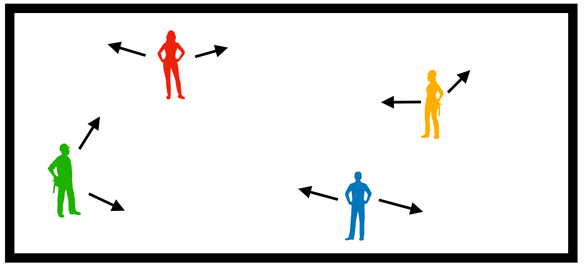
Algorithm 2 QPN Equilibrium Search

Require: QPN
Require: Candidate $\mathbf{x} \in \mathbb{R}^n$
Require: Reverse topological ordering T over the nodes in the QPN.

1: **for** $i \in T$ **do**
2: $\quad U \leftarrow C^i \bigcap_{j:(i,j)\in E} \tilde{S}^j_\epsilon(\mathbf{x})$
3: $\quad$ **for** $U^j \in U$ **do**
4: $\quad\quad$ Compute $R^j_\epsilon(\mathbf{x}) = R^j \cap \mathcal{N}_\epsilon(\mathbf{x})$ (27)
5: $\quad\quad$ **if** $\mathbf{x} \notin R^j_\epsilon(\mathbf{x})$ **then**
6: $\quad\quad\quad$ Choose $\mathbf{x} \leftarrow\in R^j_\epsilon(\mathbf{x})$
7: $\quad\quad\quad$ **go to line 1**
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: $\quad$ Compute $\tilde{S}^i_\epsilon(\mathbf{x}) = R \cap N_\epsilon(\mathbf{x})$ (39)
11: **end for**
12: **Return true**

If $\mathbf{x}$ doesn't solve node, find a value that does and restart algo.

Return to our example.



Take the perspective of ego player in randomly generated games.

$$\mathbf{g}^i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \mathbf{r}^{i,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

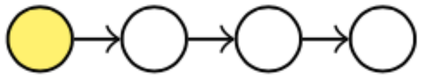
$$f^i(\mathbf{x}) := \|\mathbf{x}^i - \mathbf{g}^i\|_2^2 + \sum_{j=1, j\neq i}^{4} \|\mathbf{x}^j - \mathbf{x}^i - \mathbf{r}^{i,j}\|_2^2,$$

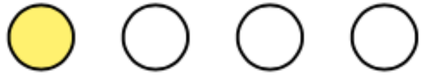
Other players are interchangeable from perspective of ego player

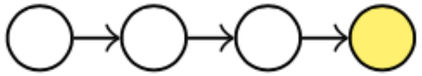
Exploiting interchangeability results in **47** unique networks from perspective of ego player.

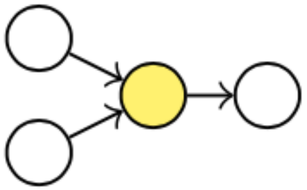
Over large scale empirical analysis, each network configuration can be compared in terms of average cost.

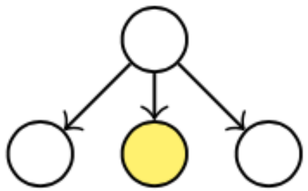
Empirical comparison of network architectures:

A 

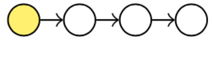
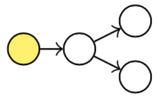
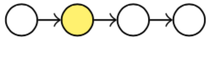
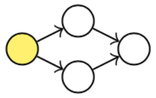
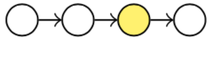
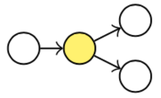
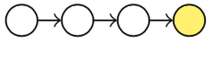
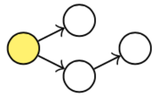
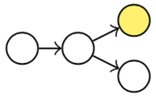
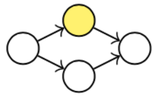
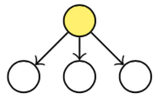
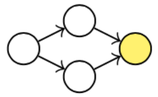
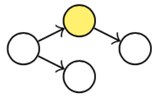
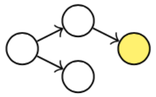
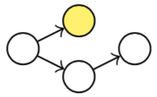
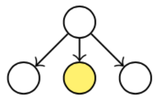
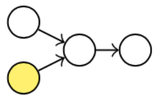
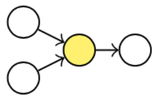
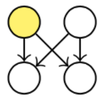
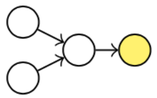
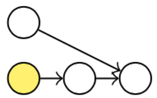
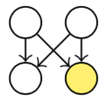
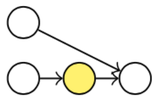
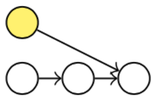
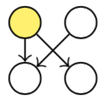
B 

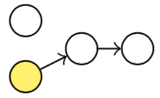
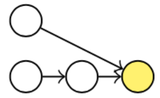
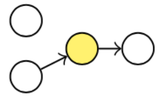
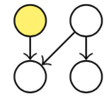
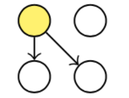
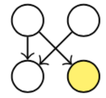
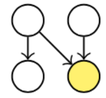
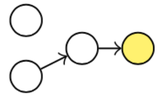
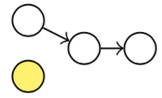
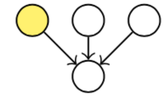
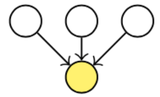
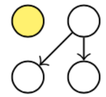
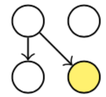
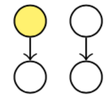
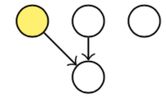
C 

D 

E 

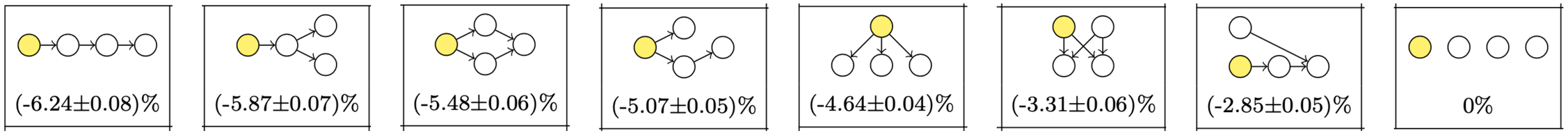
Empirical comparison of network architectures:

 (-6.24±0.08)%	 (-5.87±0.07)%	 (-5.72±0.10)%	 (-5.48±0.06)%	 (-5.40±0.11)%
 (-5.37±0.10)%	 (-5.19±0.11)%	 (-5.07±0.05)%	 (-4.83±0.11)%	 (-4.72±0.10)%
 (-4.64±0.04)%	 (-4.45±0.10)%	 (-4.34±0.09)%	 (-4.10±0.09)%	 (-4.08±0.09)%
 (-3.70±0.09)%	 (-3.69±0.07)%	 (-3.34±0.08)%	 (-3.31±0.06)%	 (-3.12±0.08)%
 (-2.85±0.05)%	 (-2.73±0.07)%	 (-2.53±0.06)%	 (-2.52±0.06)%	 (-2.43±0.04)%

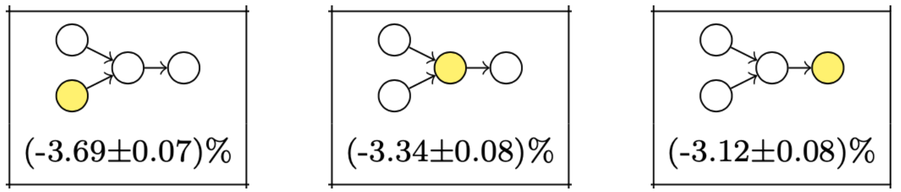
 (-2.40±0.04)%	 (-2.29±0.07)%	 (-2.07±0.05)%	 (-2.07±0.05)%	 (-1.95±0.03)%
 (-1.88±0.06)%	 (-1.86±0.06)%	 (-1.86±0.06)%	 (-1.84±0.06)%	 (-1.63±0.05)%
 (-1.42±0.05)%	 (-1.42±0.05)%	 (-1.41±0.05)%	 (-1.19±0.03)%	 (-1.17±0.03)%
(-0.98±0.04)%	(-0.98±0.04)%	(-0.98±0.04)%	(-0.70±0.02)%	(-0.49±0.03)%
(-0.48±0.03)%	0%			

Empirical comparison of network architectures:

1. Strong hierarchy > weak hierarchy

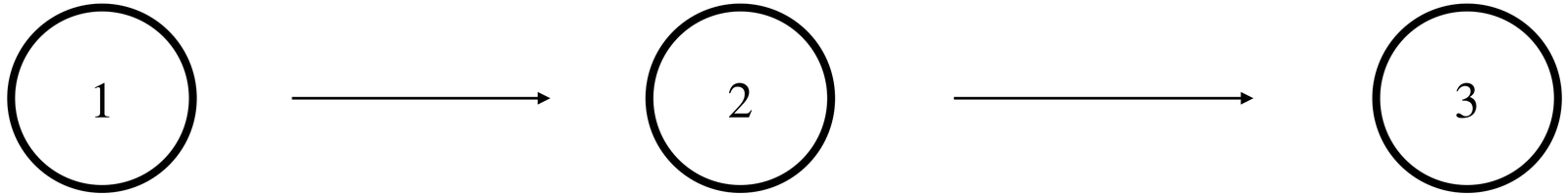


2. Leading > following



Examples

"Ridge" Problem

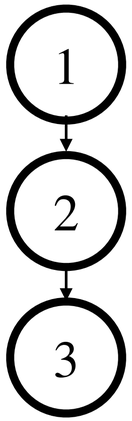


$$J^1 = \{1,2\} \qquad J^2 = \{3\} \qquad J^3 = \{4\}$$

$$f^1(\mathbf{x}) = \ldots \qquad f^2(\mathbf{x}) = \left\| \begin{matrix} x_3 - x_1 \\ x_4 - x_2 \end{matrix} \right\|_2^2 \qquad f^3(\mathbf{x}) = \frac{1}{2}(x_4 - x_3)^2$$

$$C^1 = \ldots \qquad C^2 = \mathbb{R}^4 \qquad C^3 = \{\mathbf{x} : x_4 \geq 0\}$$

"Ridge" Problem



$J^1 = \{1,2\}$ $J^2 = \{3\}$ $J^3 = \{4\}$

$f^1(\mathbf{x}) = \ldots$ $f^2(\mathbf{x}) = \left\| \begin{matrix} x_3 - x_1 \\ x_4 - x_2 \end{matrix} \right\|_2^2$ $f^3(\mathbf{x}) = \frac{1}{2}(x_4 - x_3)^2$

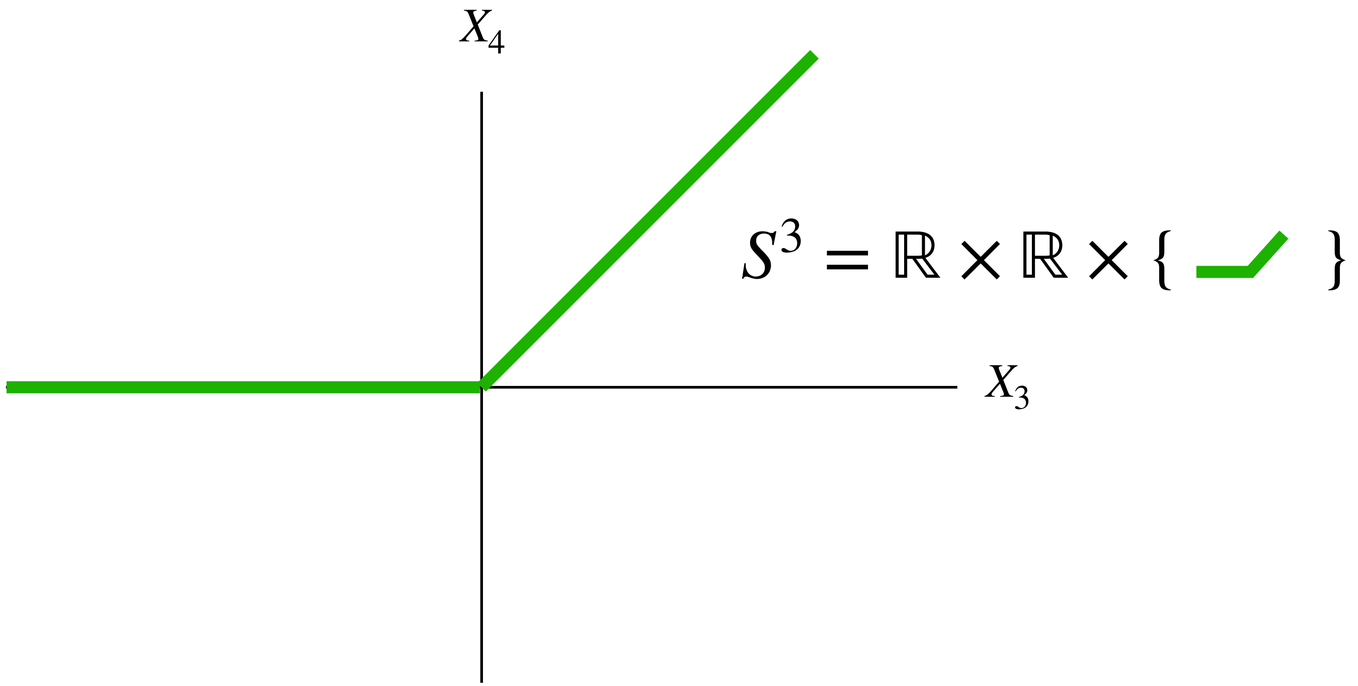
$C^1 = \ldots$ $C^2 = \mathbb{R}^4$ $C^3 = \{\mathbf{x} : x_4 \geq 0\}$

Is $\mathbf{x} = [0,0,0,0]$ in S^3?

How many "pieces" in solution graph S^3?

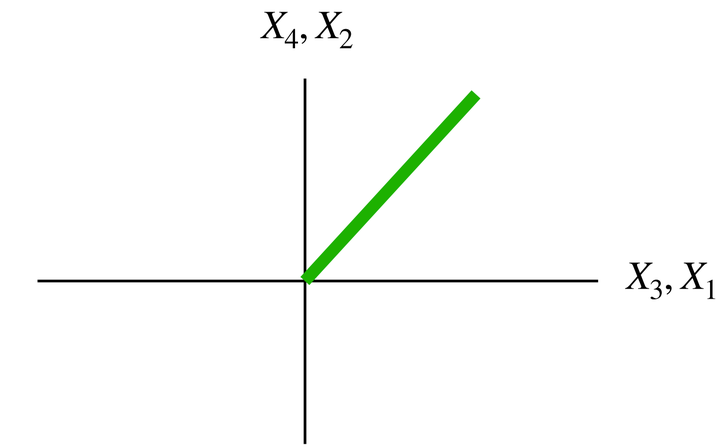
Is $\mathbf{x} = [0,0,0,0]$ in S^2?

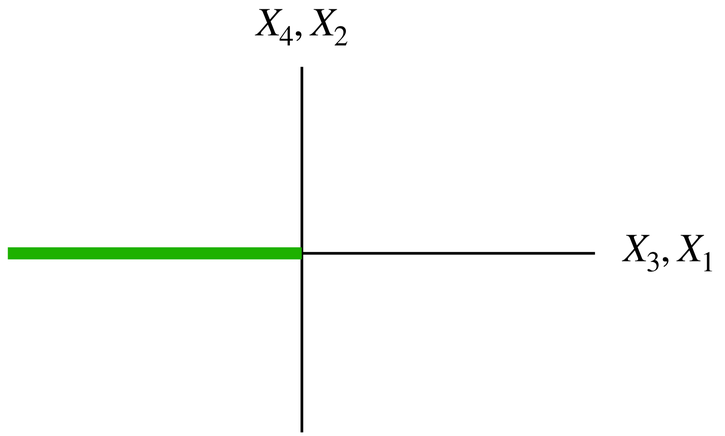
How many "pieces" in solution graph S^2?



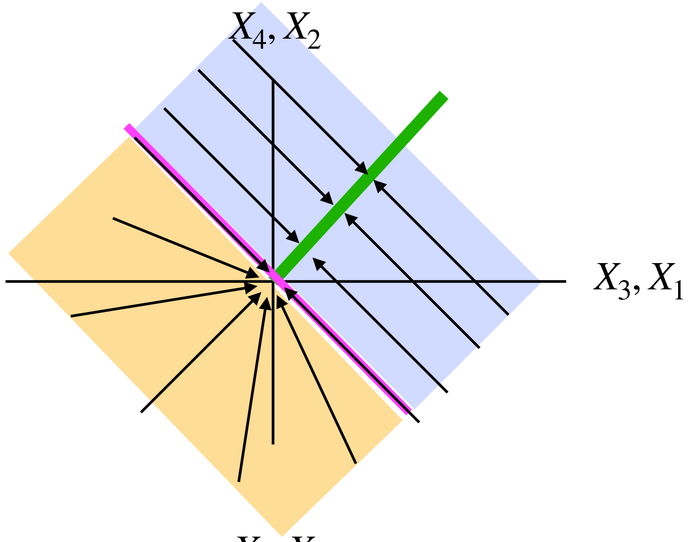
"Ridge" Problem

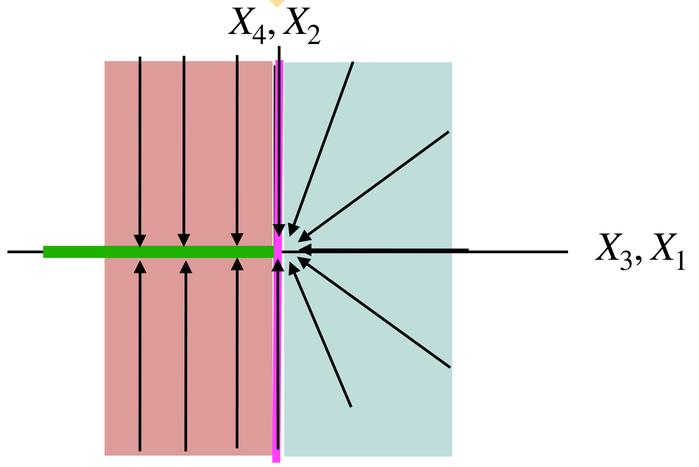
Decompose S^3 into constituent pieces.

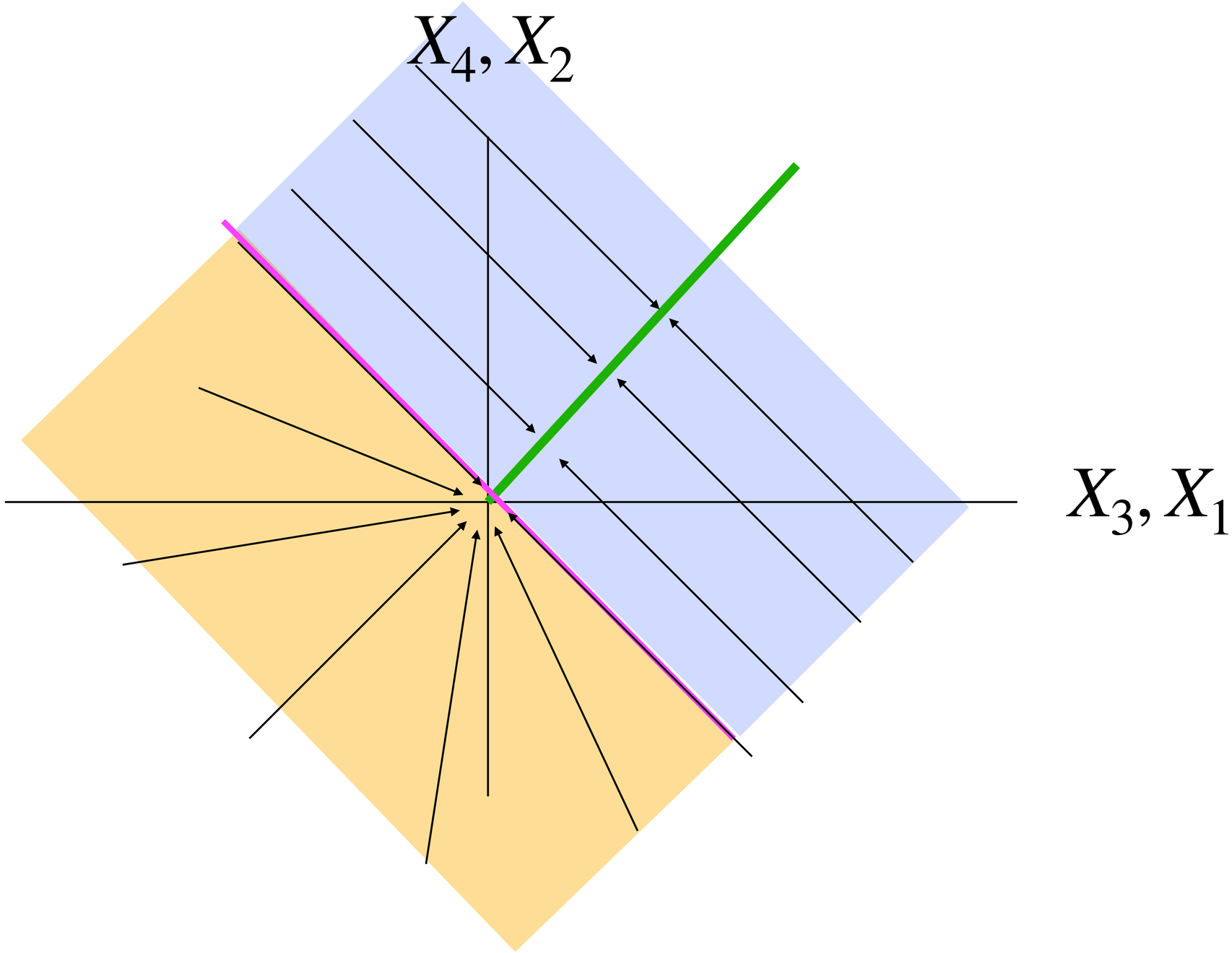




Form constituent solution graphs.

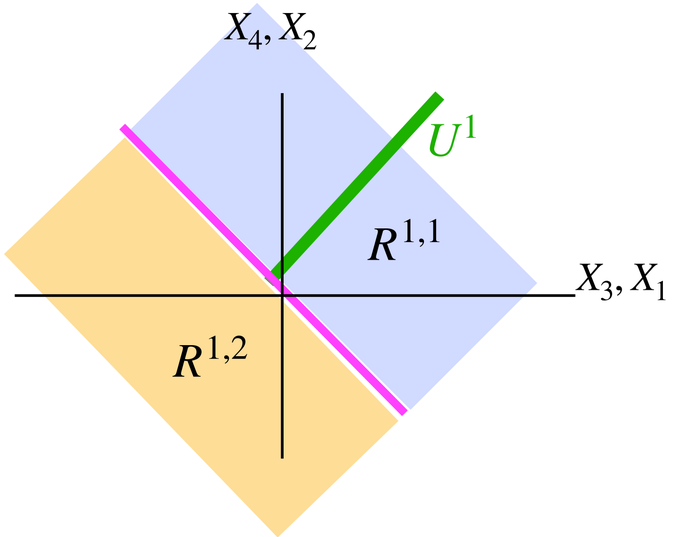


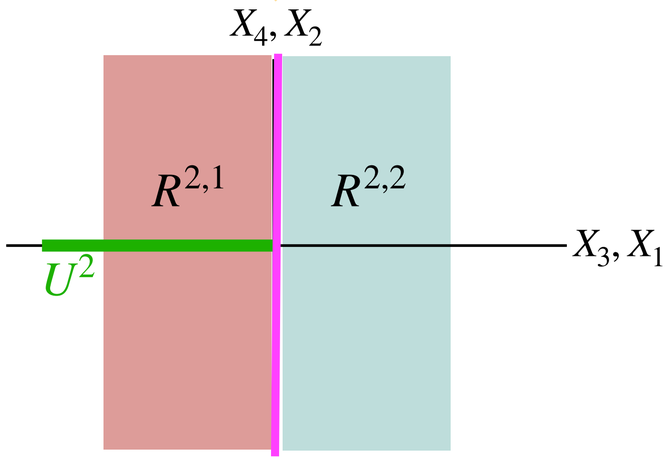



X_4, X_2
X_3, X_1

"Ridge" Problem

Form constituent solution graphs.

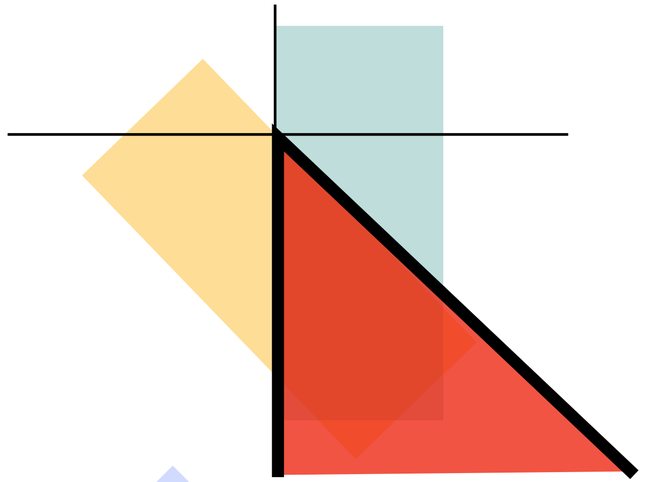




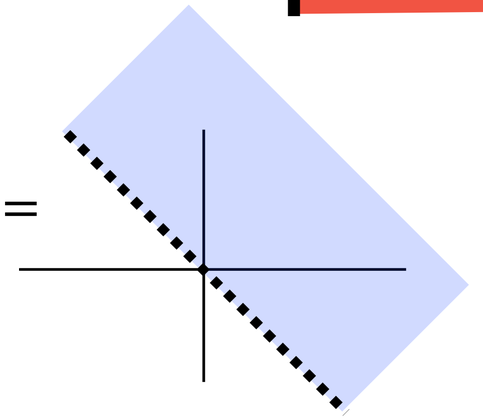
Combine into main solution graph.

$$R = (R^1 \cap R^2) \bigcup (R^1 \cap (U^2)') \bigcup (R^2 \cap (U^1)')$$

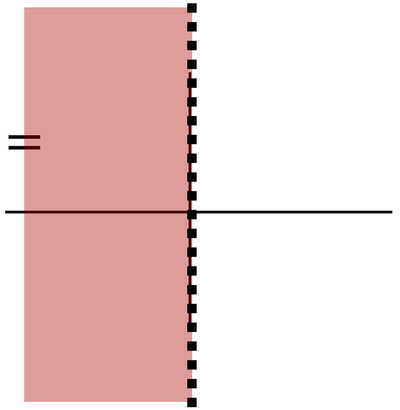
$R^1 \cap R^2 =$

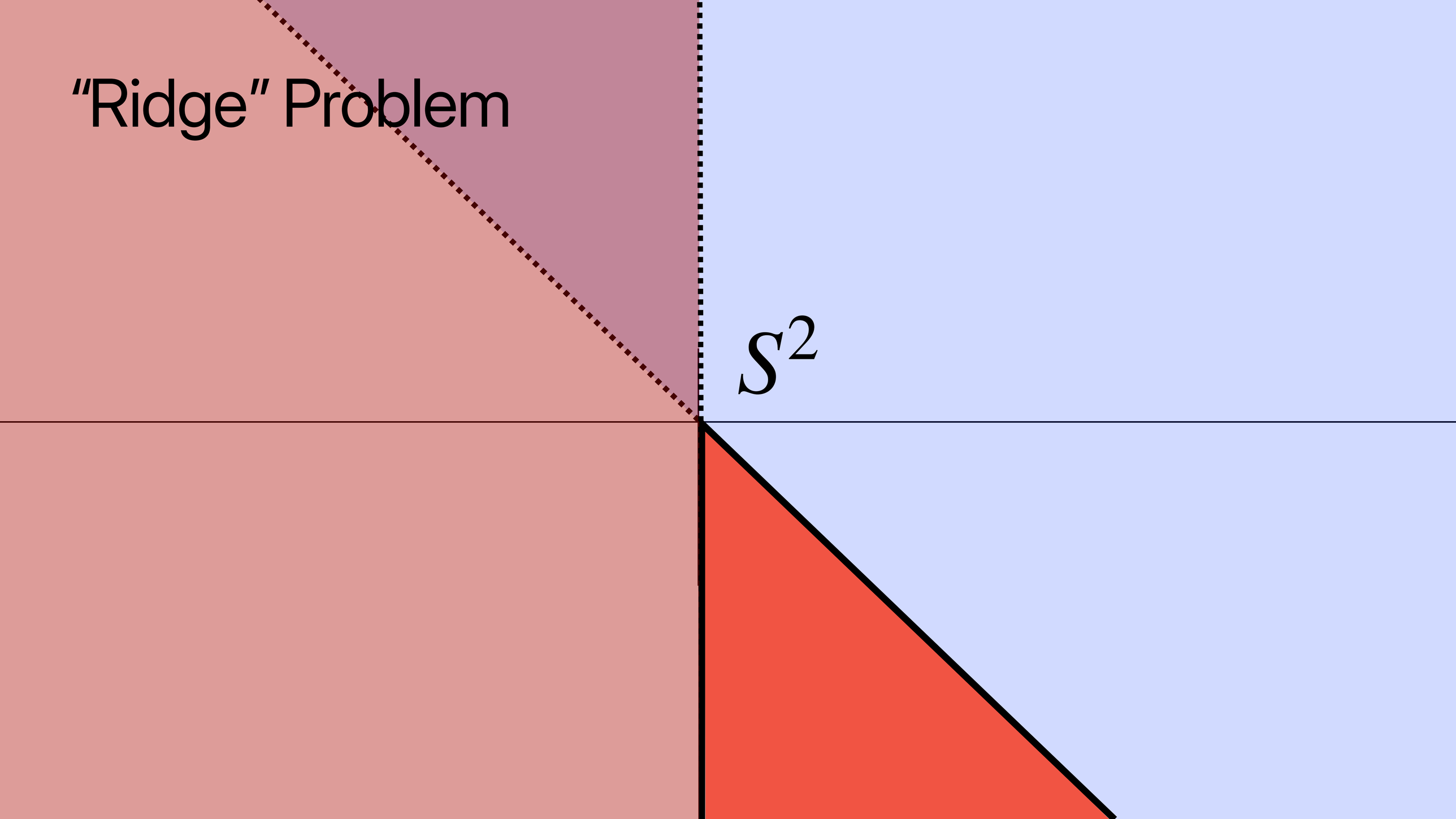


$R^1 \cap (U^2)' =$

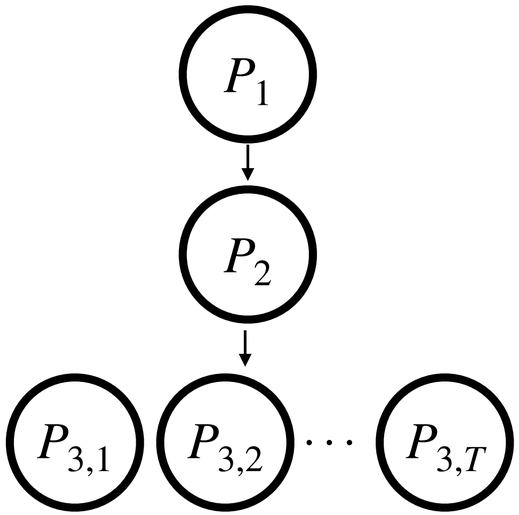


$R^2 \cap (U^1)' =$




"Ridge" Problem
S^2

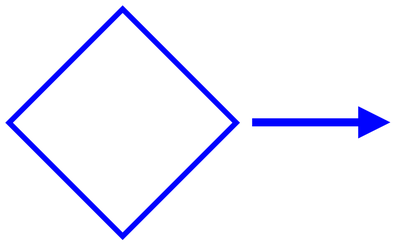
Robust Control × Collision Avoidance

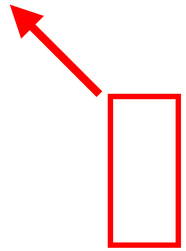


P_1 : Move ◇ to right as far as possible, subject to collision, dynamic, and control constraints.

P_2 : Move ▯ to minimize enlargement w.r.t ◇, subject to dynamic and control constraints.

$P_{3,t}$: Enlarge ◇ and ▯ as little as possible so that they intersect.



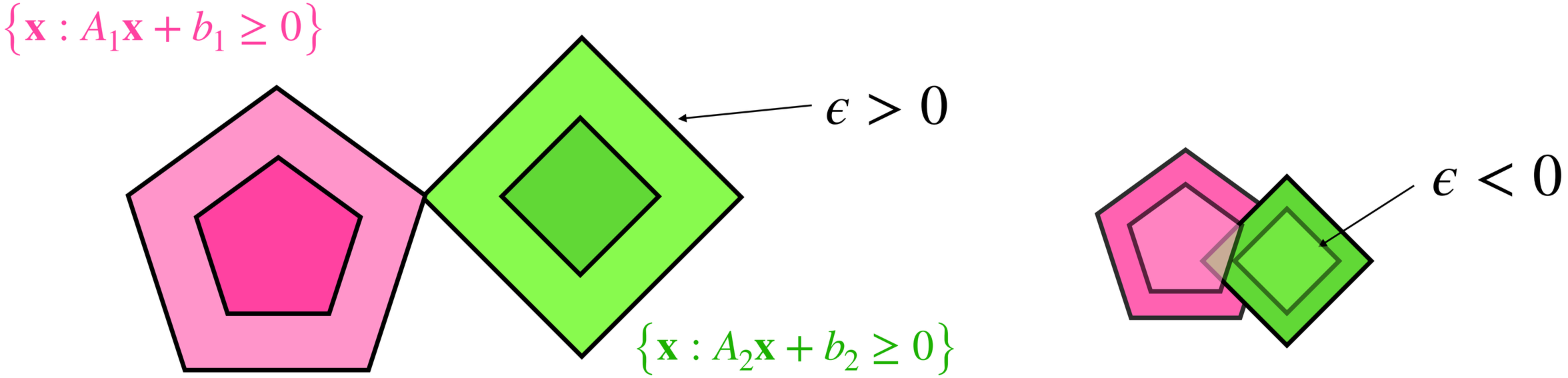


Robust Control × Collision Avoidance

$$\min_{\mathbf{x}\in\mathbb{R}^2,\epsilon} \quad \epsilon$$

$$\text{s.t.} \quad A_1\mathbf{x} + b_1 + \epsilon\mathbf{1} \geq 0$$

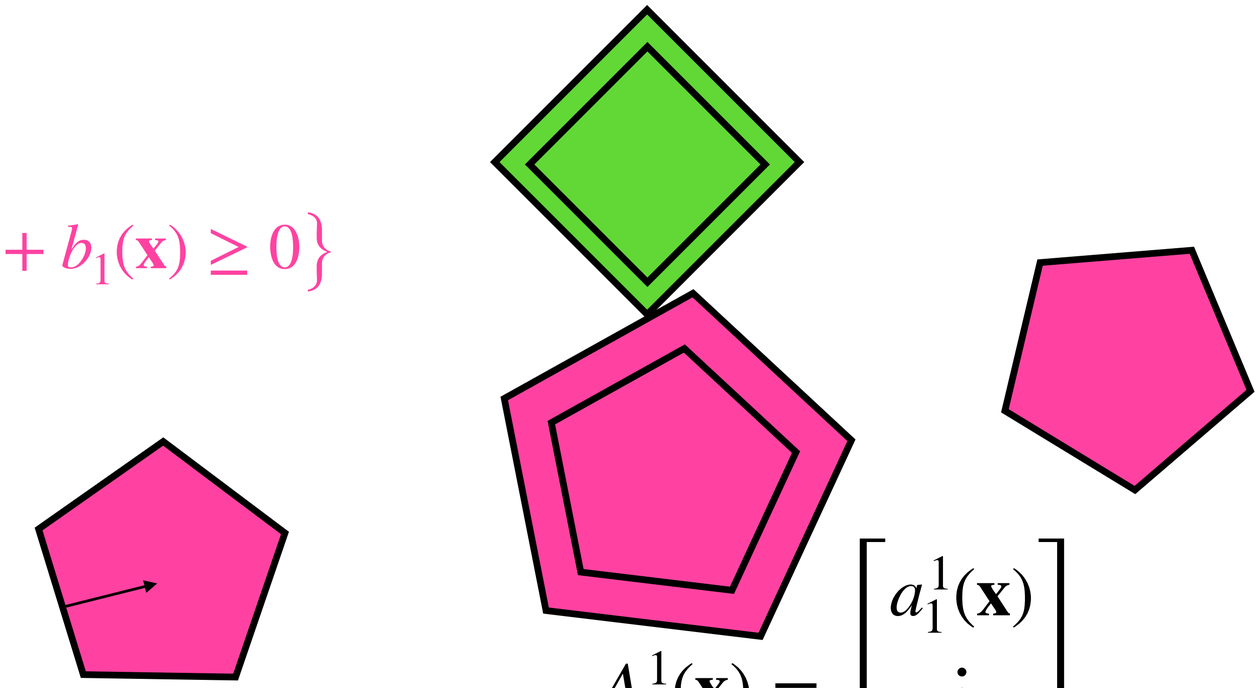
$$A_2\mathbf{x} + b_2 + \epsilon\mathbf{1} \geq 0$$



Robust Control × Collision Avoidance

$$\{\mathbf{y} : A_2(\mathbf{x})\mathbf{y} + b_2(\mathbf{x}) \geq 0\}$$

$$\{\mathbf{y} : A_1(\mathbf{x})\mathbf{y} + b_1(\mathbf{x}) \geq 0\}$$

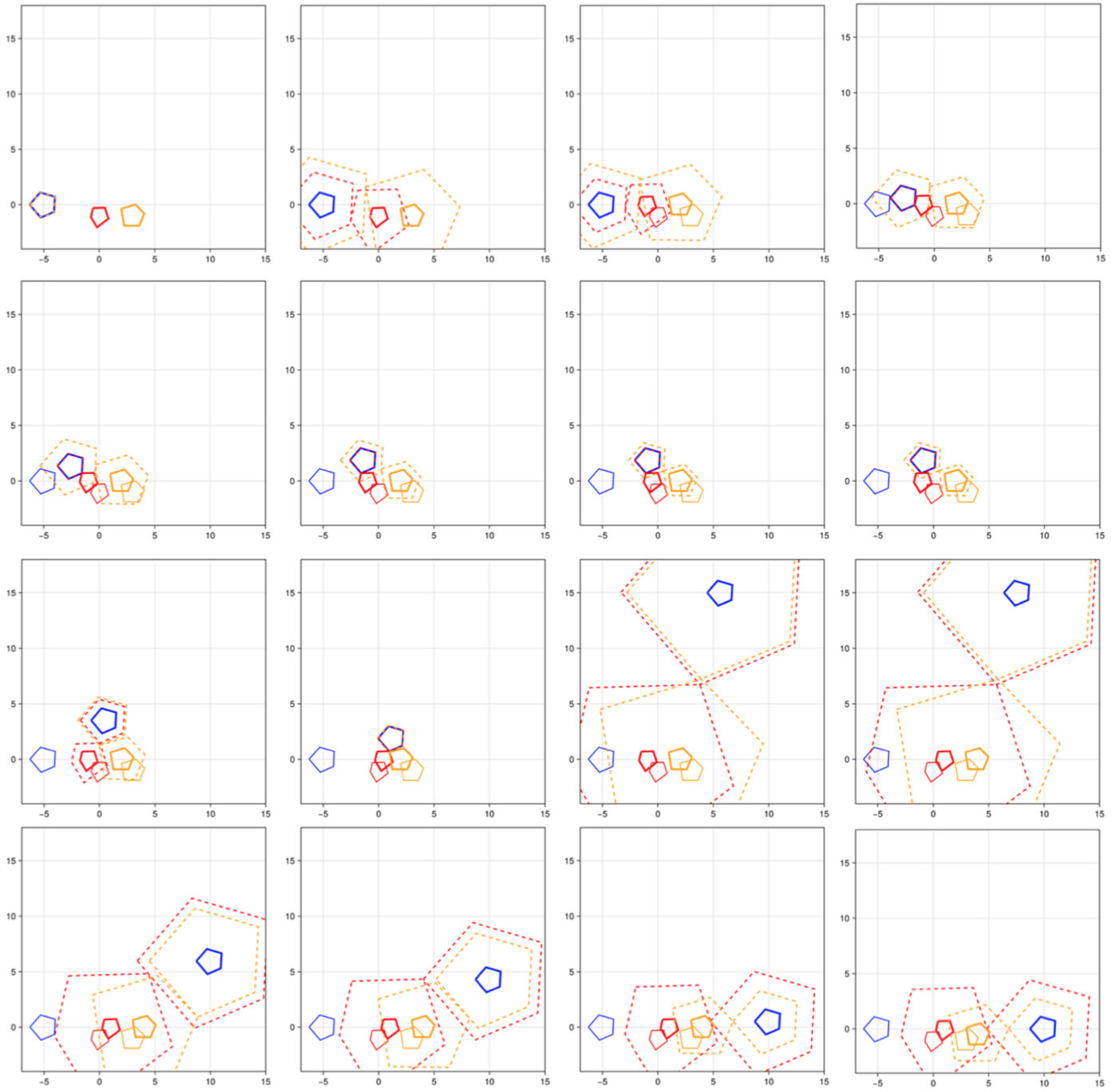


$$A^1(\mathbf{x}) = \begin{bmatrix} a_1^1(\mathbf{x}) \\ \vdots \\ a_m^1(\mathbf{x}) \end{bmatrix}$$

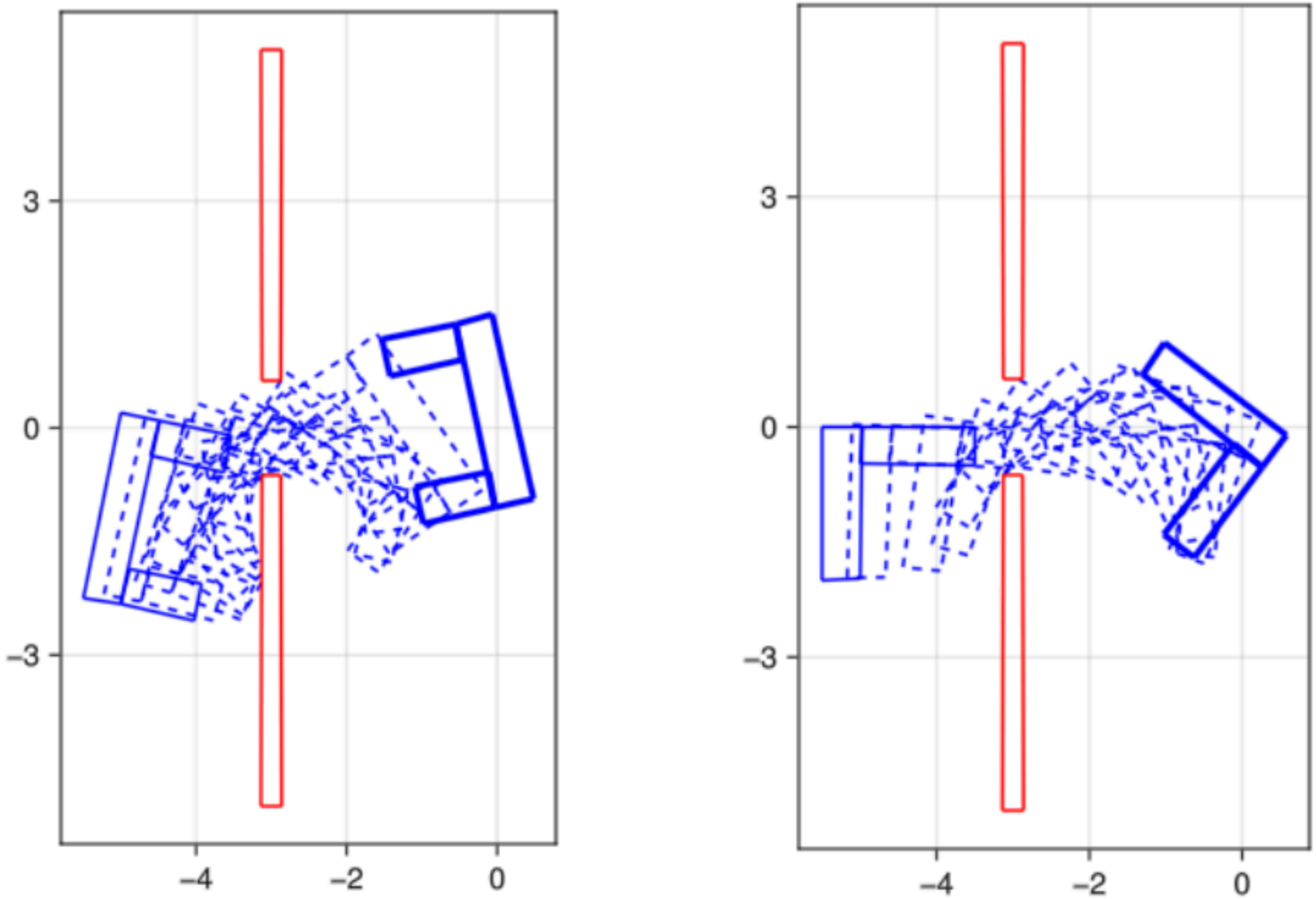
$$\begin{aligned} \min_{\mathbf{y},\mathbf{x},\epsilon,t} \quad & \epsilon \\ \text{s.t.} \quad & A_1(\mathbf{x})\mathbf{y} + b_1(\mathbf{x}) + \epsilon\mathbf{1} \geq 0 \\ & A_2\mathbf{y} + b_2 + \epsilon\mathbf{1} \geq 0 \\ & \mathbf{x} = (1-t)\mathbf{x^1} + t\mathbf{x^2} \\ & 0 \leq t \leq 1 \end{aligned}$$

$$\mathbf{x^1} = \begin{bmatrix} p_1^1 \\ p_2^1 \\ \theta^1 \end{bmatrix} \quad \mathbf{x^2} = \begin{bmatrix} p_1^2 \\ p_2^2 \\ \theta^2 \end{bmatrix}$$

$$a_i^1(\mathbf{x}) = R(\theta)\hat{a}_i^1$$

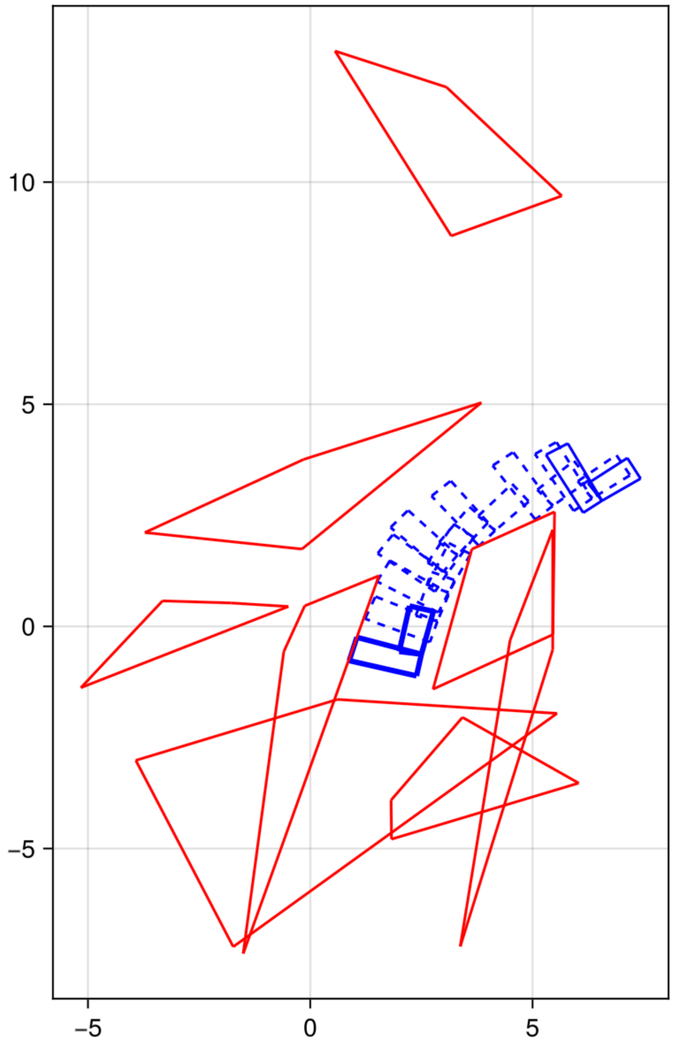


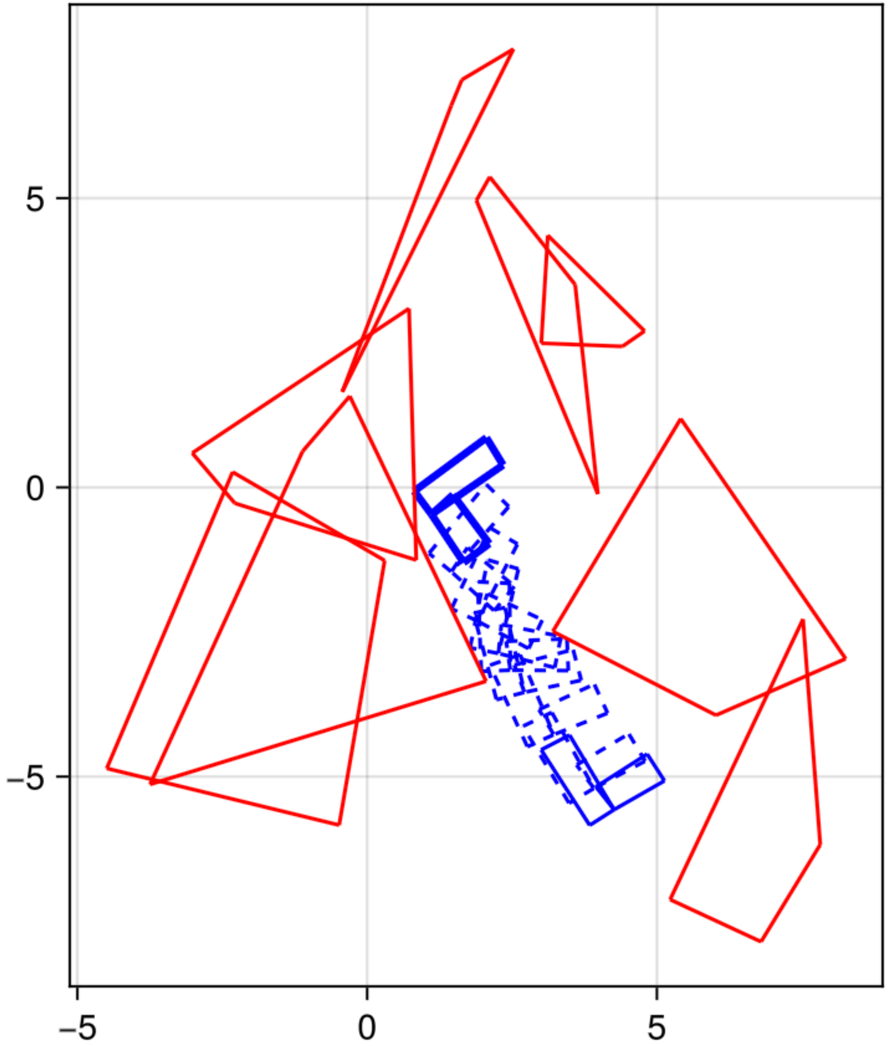
MPN Collision Avoidance



Polyhedral Collision Detection via Vertex Enumeration: https://arxiv.org/abs/2501.13201

MPN Collision Avoidance





Polyhedral Collision Detection via Vertex Enumeration: https://arxiv.org/abs/2501.13201

Takeaways

- Mathematical Program Networks are a graph-based framework for defining games
- One equilibrium concept is needed to encompass most existing definitions
- Algorithms can be defined to compute equilibrium points under different network topologies
- This is very appealing for applied game theory where information structure is not fundamental but rather a modeling choice

https://github.com/forrestlaine/QuadraticProgramNetworks.jl

https://arxiv.org/abs/2404.03767