The Computational Complexity of finding Game-Theoretic Solutions

Rahul Savani

University of Liverpool
and
The Alan Turing Institute

Outline

Major results we will cover:

- **PURE-NASH** for **congestion** games is **PLS**-complete **(2004)**
- **MIXED-NASH** for **bimatrix** games is **PPAD**-complete **(2006)**
- **CLS = PPAD ∩ PLS**

  (**2D-KKT** is (**PLS ∩ PPAD**)-complete) **(2021)**
- **MIXED-NASH** for **congestion games** is **CLS**-complete **(2021)**

Outline

Major results we will cover:

- **PURE-NASH** for **congestion** games is **PLS**-complete **(2004)**
- **MIXED-NASH** for **bimatrix** games is **PPAD**-complete **(2006)**
- **CLS = PPAD ∩ PLS**
  (**2D-KKT** is (**PLS ∩ PPAD**)-complete) **(2021)**
- **MIXED-NASH** for **congestion games** is **CLS**-complete **(2021)**

There are many important problems in **CLS** that are unlikely to be complete for it because they **always have a unique solution**

We finish by introducing **UEOPL**, a class within CLS that only contains problems that admit unique solutions...

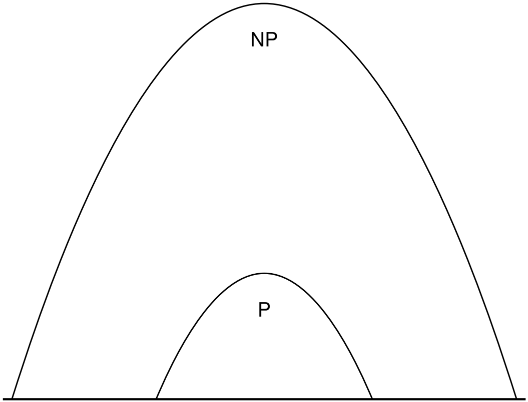
For **PPAD**, **PLS**, **CLS**, and **UEOPL**, we will discuss:

- **Inspiration and motivation for the classes**,
  e.g. via algorithmic approaches or properties of solutions
- **Technical definitions of the classes**
- **Examples of complete problems** for these classes
- **High-level ideas** of (the extremely technical) reductions
- **Open problems**

1. **Total Function problems in NP (TFNP)**
   - Totality and verifiability
   - Syntactic subclasses of TFNP
2. **Polynomial Parity Argument, Directed Version (PPAD)**
   - Bimatrix games, the Lemke-Howson algorithm, membership in PPAD
   - Sketch of PPAD-hardness
   - Nash to Brouwer
3. **Polynomial Local Search (PLS)**
   - Congestion games, potential functions, membership in PLS
   - PLS-hardness for congestion games
4. **Continuous Local Search (CLS)**
   - Gradient Descent
   - CLS = PPAD ∩ PLS
   - Candidates for CLS-hardness
   - Finding a mixed equilibrium of a congestion game is CLS-complete
5. **Unique End of Potential Line (UEOPL)**
   - Definition, example problems in UEOPL, and related open problems

Total Function problems in NP (TFNP)

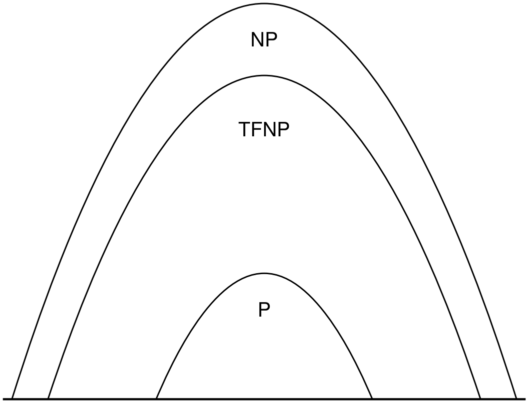
Complexity classes between P and NP



There are many problems that lie **between** P and NP

- Factoring, graph isomorphism, computing Nash equilibria, local max cut, simple-stochastic games, ...

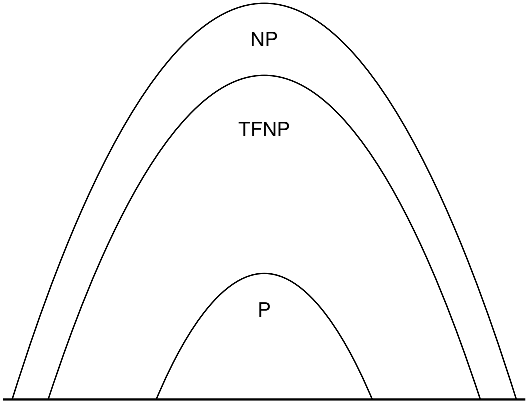
Complexity classes between P and NP



FNP is the class of **function** problems in NP

- Given polynomial time computable relation $\boldsymbol{R}$ and value $\boldsymbol{x}$
- Find $\boldsymbol{y}$ such that $(\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{R}$

Complexity classes between P and NP



TFNP is the subclass of problems that **always** have solutions

- Contains factoring, Nash equilibria, local max cut, simple-stochastic games, ...

Total search problems

A search problem is **total** if a solution is **guaranteed to exist**

Examples:

- **NASH:**
  Find a mixed Nash equilibrium of a game
- **PURE-CONGESTION:**
  Find a pure Nash equilibrium of a congestion game
- **FACTORING:**
  Find a prime factor of a number ≥ 2
- **BROUWER:**
  Find a fixed point of a continuous function $f : [0,1]^3 \mapsto [0,1]^3$
- **KKT (Karush-Kuhn-Tucker):**
  Find a KKT point of a $\mathcal{C}^1$ function $f : [0,1]^3 \mapsto [0,1]$

NP Total Search Problems (TFNP)

NASH, **PURE-CONGESTION**, **FACTORING**, **BROUWER**, **KKT**, . . .

In addition to being total, these problems have more in common:

They are **NP** function problems with **easy-to-verify solutions**

NP Total Search Problems (TFNP)

NASH, **PURE-CONGESTION**, **FACTORING**, **BROUWER**, **KKT**, . . .

In addition to being total, these problems have more in common:

They are **NP** function problems with **easy-to-verify solutions**

Can a **TFNP** problem be **NP**-hard?

NP Total Search Problems (TFNP)

NASH, **PURE-CONGESTION**, **FACTORING**, **BROUWER**, **KKT**, . . .

In addition to being total, these problems have more in common:

They are **NP** function problems with **easy-to-verify solutions**

Can a **TFNP** problem be **NP**-hard? Not unless **NP = co-NP ...**

[Megiddo-Papadimitriou, 1991]

NP Total Search Problems (TFNP)

NASH, **PURE-CONGESTION**, **FACTORING**, **BROUWER**, **KKT**, . . .

In addition to being total, these problems have more in common:

They are **NP** function problems with **easy-to-verify solutions**

Can a **TFNP** problem be **NP**-hard? Not unless **NP = co-NP ...**

[Megiddo-Papadimitriou, 1991]

It is believed that **TFNP** does **not** have complete problems

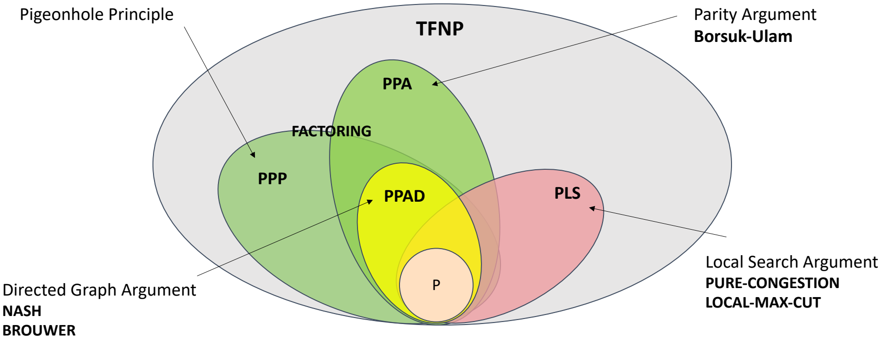
Syntactic subclasses of TFNP

To classify the complexity of problems within TFNP

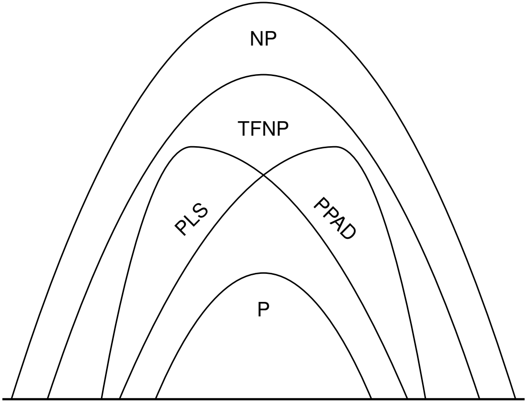
syntactic subclasses have been defined based on the (combinatorial) **proof principles of totality**:

- **PPP**: totality based on pigeonhole principle
- **PLS**: totality based on potential function (DAGs have sinks)
- **PPAD**: totality based on (reversible) line-following argument

TFNP Landscape

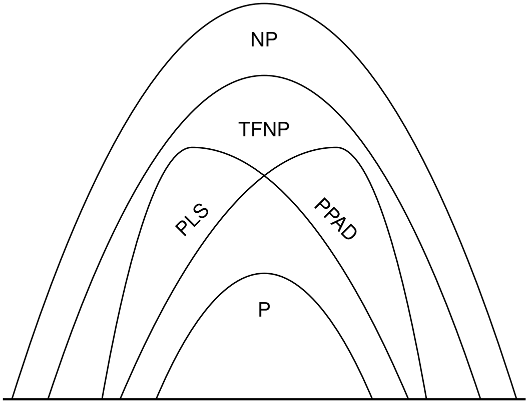


Complexity classes between P and NP



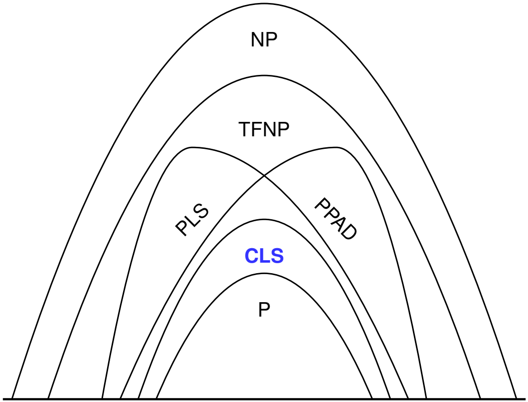
PPAD and PLS are two **subclasses** of TFNP

Complexity classes between P and NP



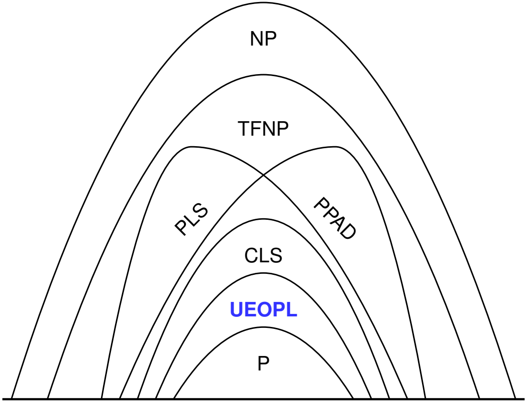
Are there interesting problems in PPAD and PLS?

Complexity classes between P and NP



CLS (Continuous Local Search) was defined
to capture these problems (Daskalakis and Papadimitriou, 2011)

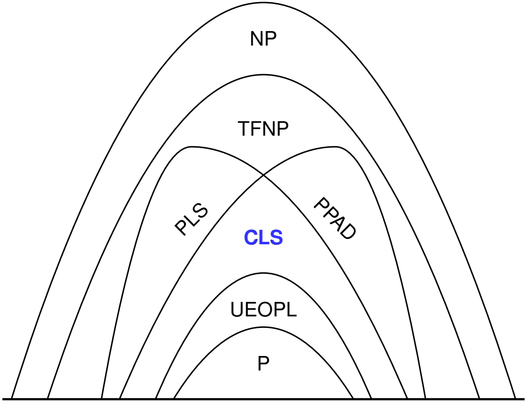
Complexity classes between P and NP



UEOPL – Unique End of Potential Line

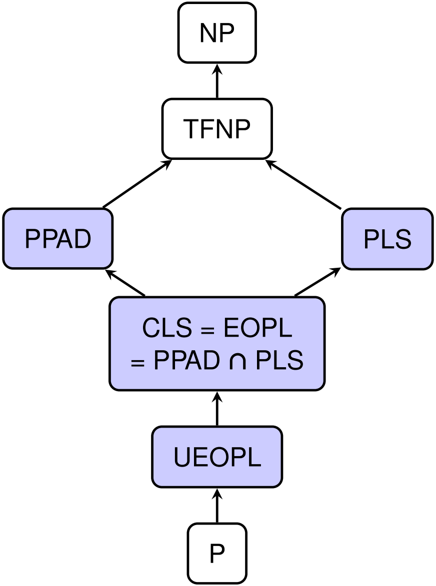
UEOPL ⊆ CLS defined to capture problems with **unique** solutions (2020)

Complexity classes between P and NP



Later CLS was surprisingly shown to equal PPAD ∩ PLS (2021)

Complexity classes: PPAD, PLS, CLS, UEOPL



Complexity classes: PPAD, PLS, CLS, UEOPL

- **PPAD**: Nash equilibrium of a strategic-form game; Brouwer fixed points; market equilibrium...
- **PLS**: Pure Nash equilibrium of a congestion game; Local Max Cut (and other “local” versions of NP-hard problems)...
- **CLS**: Continuous Local optima (found e.g. by Gradient Descent); **mixed** Nash equilibrium of a congestion game
- **UEOPL**: Parity Games; Simple Stochastic Games; P-matrix LCP; fixed points of contraction maps...

TFNP subclasses

Why believe that **PPAD** $\neq$ **P**, **PLS** $\neq$ **P**, etc. ?

- many seemingly hard problems lie in **PPAD**, **PLS**, ...
- oracle separations (in particular **PPAD** $\neq$ **PLS**)
- hard under cryptographic assumptions

References

On Total Functions, Existence Theorems and Computational Complexity by **Megiddo and Papadimitriou**
Theor. Comput. Sci. (1991) **TFNP definition and basic results**

On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence by **Papadimitriou**
J. Comput. Syst. Sci. (1994) **PPAD, PPA, PPP, memberships and relationships**

Propositional proofs and reductions between NP search problems by **Buss and Johnson**
Ann. Pure Appl. Log. (2012) **Oracle separations**

On the Cryptographic Hardness of Finding a Nash Equilibrium by **Bitansky, Paneth, Rosen**
FOCS (2015) **Example of cryptographic hardness (for PPAD)**

Polynomial Parity Argument, Directed Version (PPAD)

Nash equilibria of bimatrix games

I \ II	l	r
T	3, 1	3, 0
M	2, 0	5, 2
B	0, 4	6, 3

Nash equilibria of bimatrix games

I \ II	l	r
T	1 3	0 3
M	0 2	2 5
B	4 0	3 6

Nash equilibrium =

pair of strategies **x**, **y** with

x best response to **y** and
y best response to **x**

Mixed equilibria

I \ II	l	r
T	1 3	0 3
M	0 2	2 5
B	4 0	3 6

$$Ay = \begin{pmatrix} 3 & 3 \\ 2 & 5 \\ 0 & 6 \end{pmatrix} \begin{pmatrix} 1/3 & 2/3 \end{pmatrix}^{T} = \begin{pmatrix} 3 \\ 4 \\ 4 \end{pmatrix}$$

$$x^{T}B = \begin{pmatrix} 0 \\ 1/3 \\ 2/3 \end{pmatrix}^{T} \begin{pmatrix} 1 & 0 \\ 0 & 2 \\ 4 & 3 \end{pmatrix} = \begin{pmatrix} 8/3 & 8/3 \end{pmatrix}$$

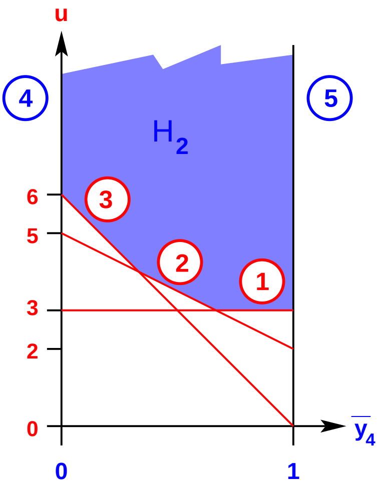
only **only pure best responses** can have probability > 0

Best response polyhedron H_2 for player 2

$$\begin{array}{c|cc|} & \bar{y}_4 & \bar{y}_5 \\ \hline ① & 3 & 3 \\ ② & 2 & 5 \\ ③ & 0 & 6 \\ \hline \end{array} = A$$

$H_2 = \{\ (\bar{y}_4, \bar{y}_5, u)\ |$

$$\begin{aligned} ①:&\quad 3\bar{y}_4 + 3\bar{y}_5 \le u \\ ②:&\quad 2\bar{y}_4 + 5\bar{y}_5 \le u \\ ③:&\quad \phantom{2\bar{y}_4 +{}} 6\bar{y}_5 \le u \\ &\quad \bar{y}_4 + \bar{y}_5 = 1 \\ ④:&\quad \bar{y}_4 \ge 0 \\ ⑤:&\quad \bar{y}_5 \ge 0 \ \} \end{aligned}$$



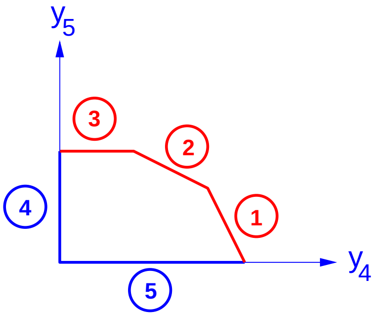
Best response polytope Q for player 2

$$\begin{array}{c|cc|} & y_4 & y_5 \\ \hline ① & 3 & 3 \\ ② & 2 & 5 \\ ③ & 0 & 6 \\ \hline \end{array} = A$$

$$Q = \{\, y \mid Ay \le 1,\ \ y \ge 0 \,\}$$

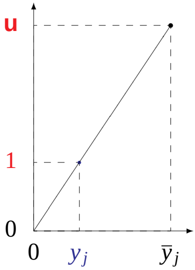
$$Q = \{\ (y_4, y_5) \ \mid$$

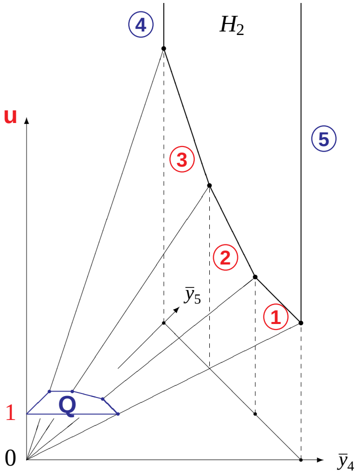
$$\begin{array}{lrcl} ①: & 3y_4 + 3y_5 & \le & 1 \\ ②: & 2y_4 + 5y_5 & \le & 1 \\ ③: & 6y_5 & \le & 1 \\ \\ ④: & y_4 & \ge & 0 \\ ⑤: & y_5 & \ge & 0 \ \ \} \end{array}$$



Projective transformation

H_2, **Q** same face incidences





Best response polytope Q for player 2

	y_4	y_5
①	3	3
②	2	5
③	0	6

$= A$

$$Q = \{\, y \mid Ay \le 1,\ \ y \ge 0 \,\}$$

$$Q = \{\ (y_4, y_5)\ \mid$$

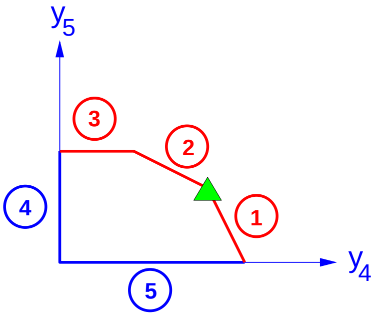
① : $3y_4 + 3y_5 \le 1$

② : $2y_4 + 5y_5 \le 1$

③ : $6y_5 \le 1$

④ : $y_4 \ge 0$

⑤ : $y_5 \ge 0\ \}$

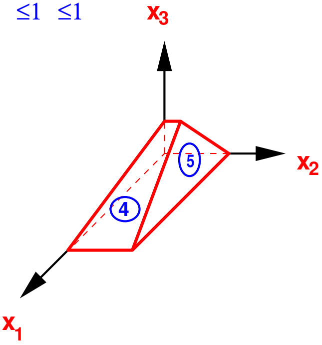


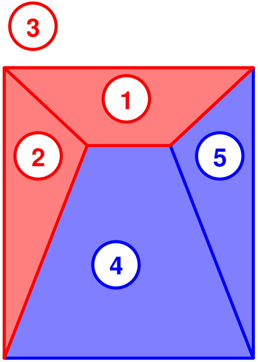
Best response polytope P for player 1

x_1	1	0
x_2	0	2
x_3	4	3
	≤ 1	≤ 1

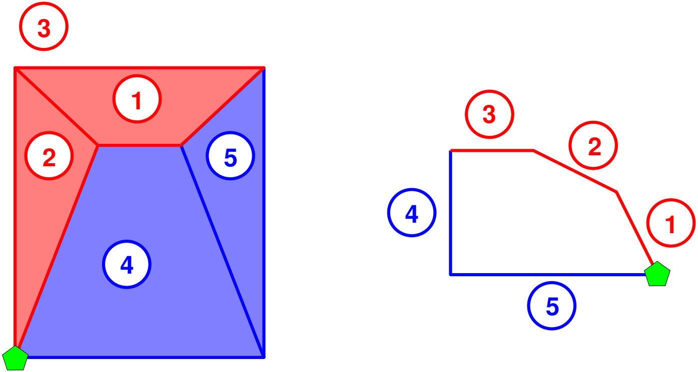
$= B$

$$P = \{\, x \mid x \ge 0,\ x^{\top} B \le 1 \,\}$$



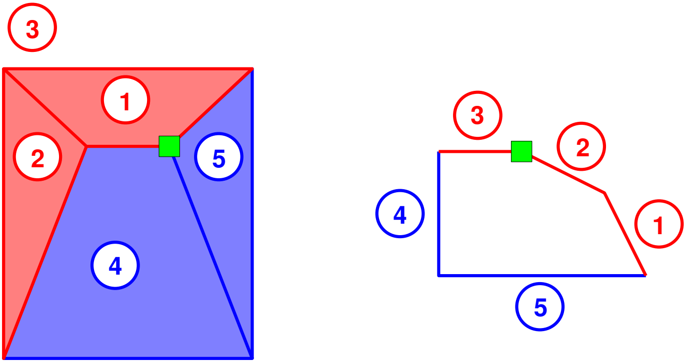


Equilibrium = completely labeled pair



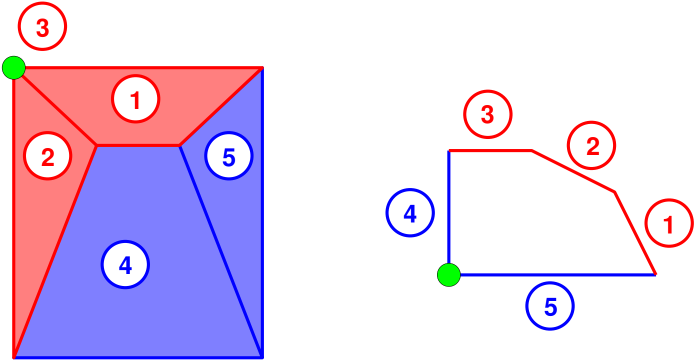
pure equilibrium

Equilibrium = completely labeled pair

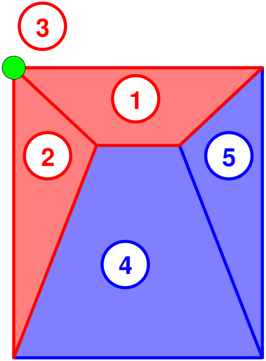


mixed equilibrium

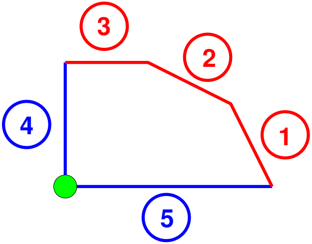
The Lemke–Howson algorithm



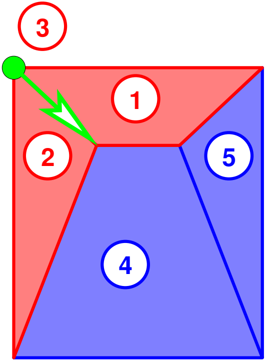
The Lemke–Howson algorithm



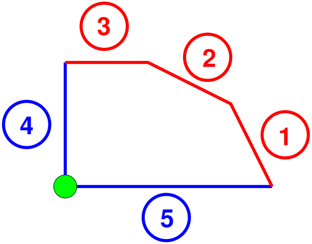
Drop label 3



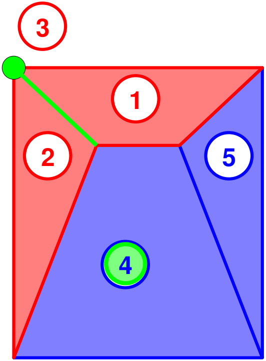
The Lemke–Howson algorithm



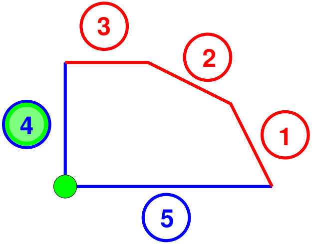
Drop label 3



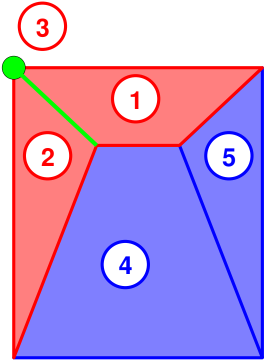
The Lemke–Howson algorithm

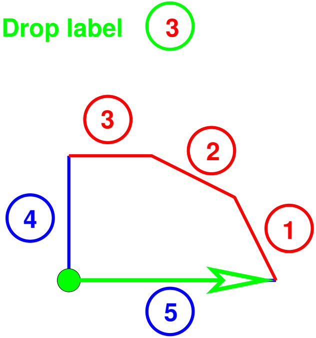


Drop label 3



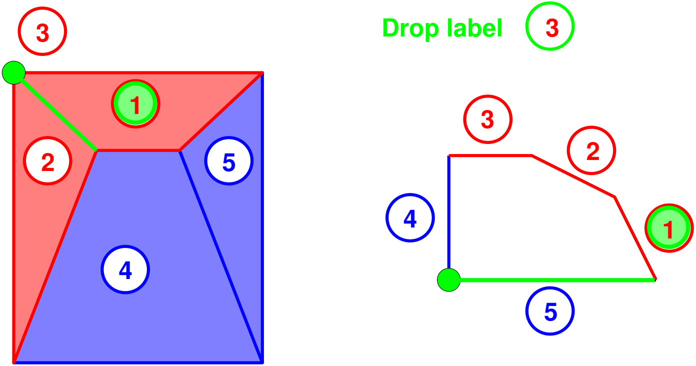
The Lemke–Howson algorithm



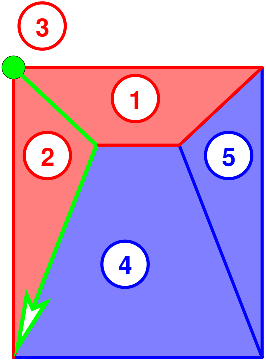


The Lemke–Howson algorithm

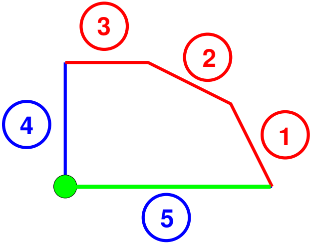
Drop label 3



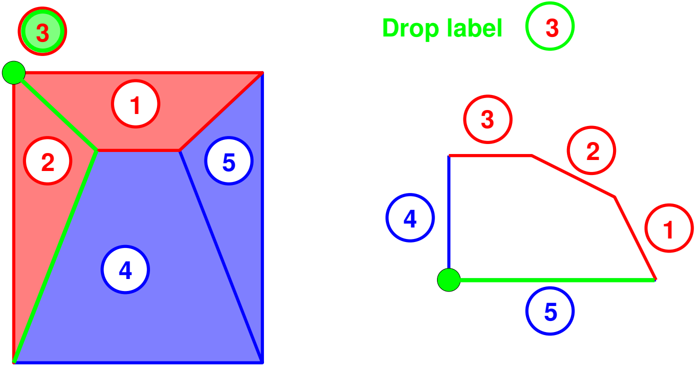
The Lemke–Howson algorithm



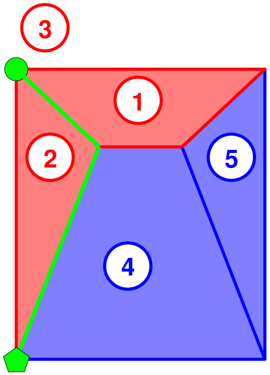
Drop label 3

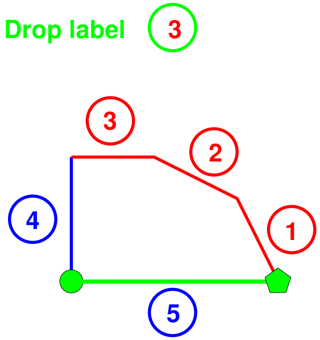


The Lemke–Howson algorithm

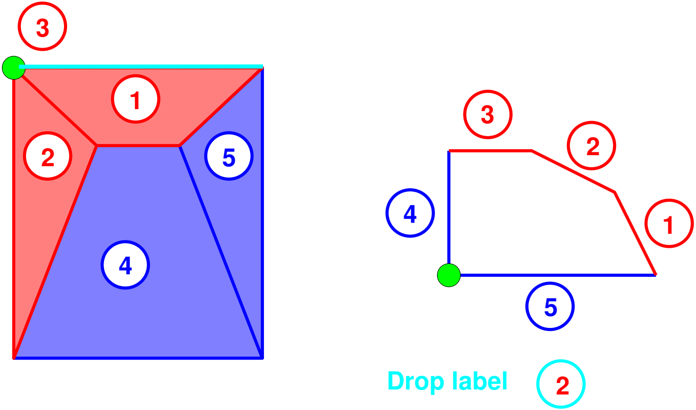


The Lemke–Howson algorithm

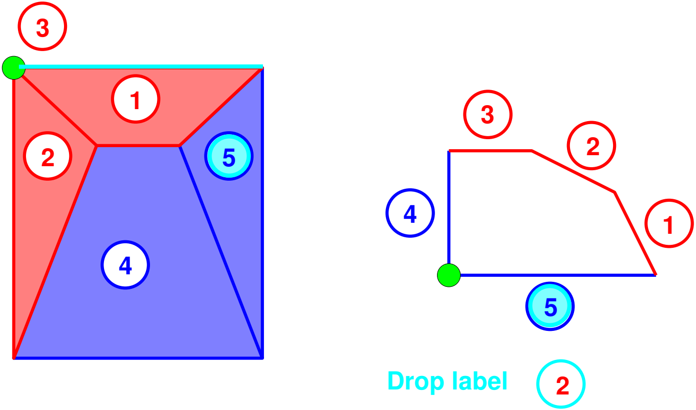




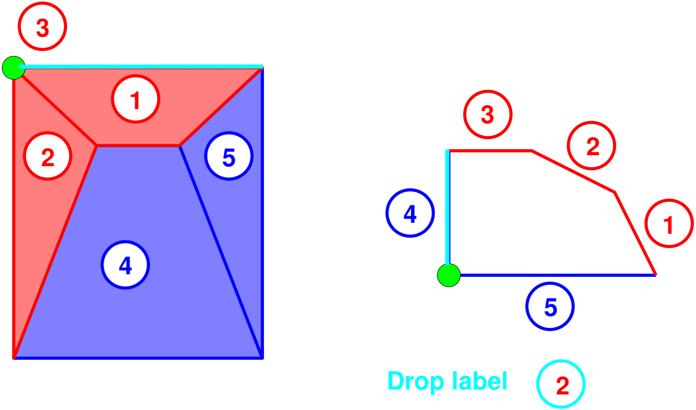
The Lemke–Howson algorithm



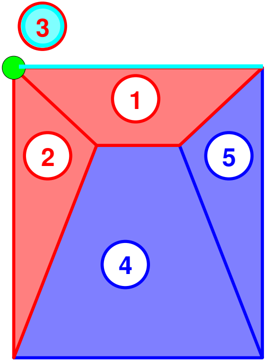
The Lemke–Howson algorithm

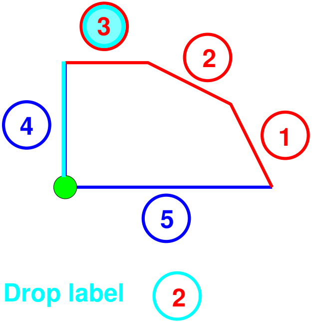


The Lemke–Howson algorithm

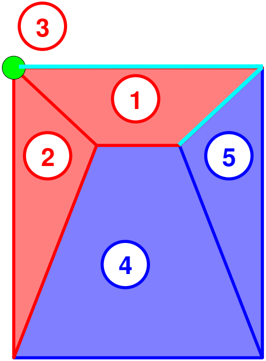


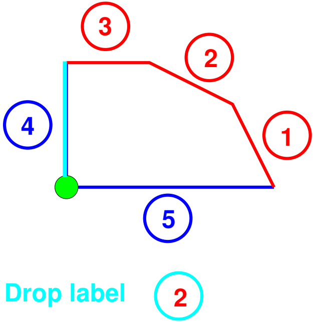
The Lemke–Howson algorithm



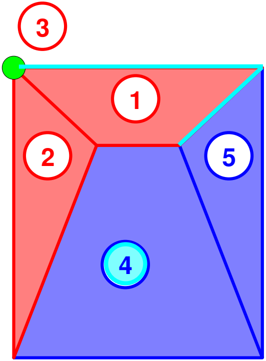


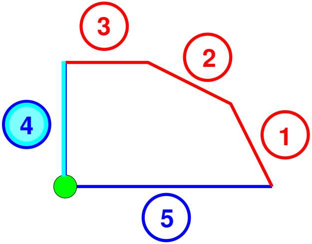
The Lemke–Howson algorithm





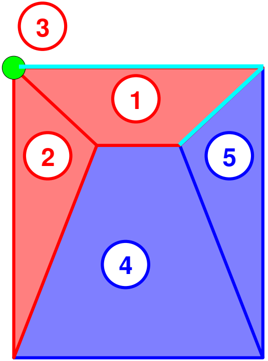
The Lemke–Howson algorithm

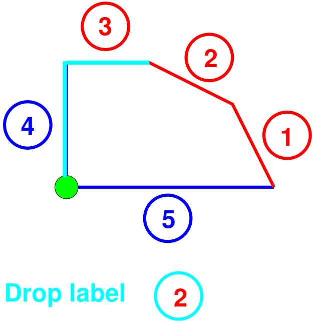




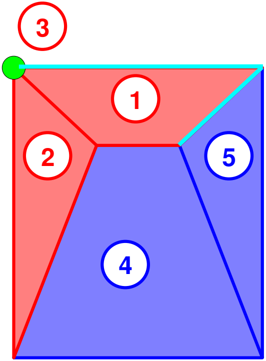
Drop label 2

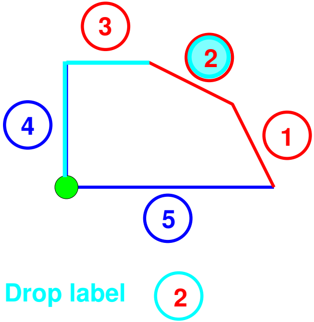
The Lemke–Howson algorithm



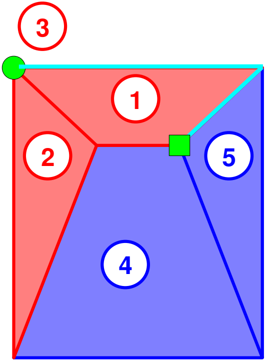


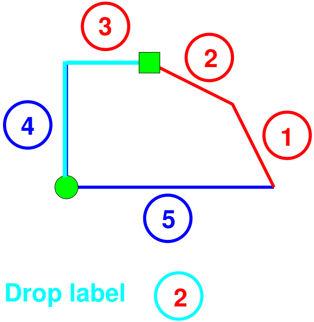
The Lemke–Howson algorithm



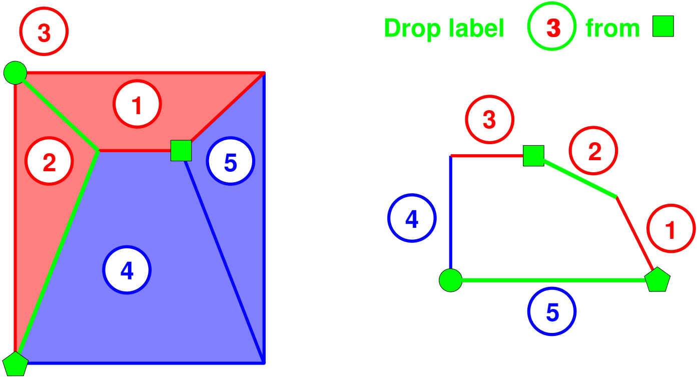


The Lemke–Howson algorithm



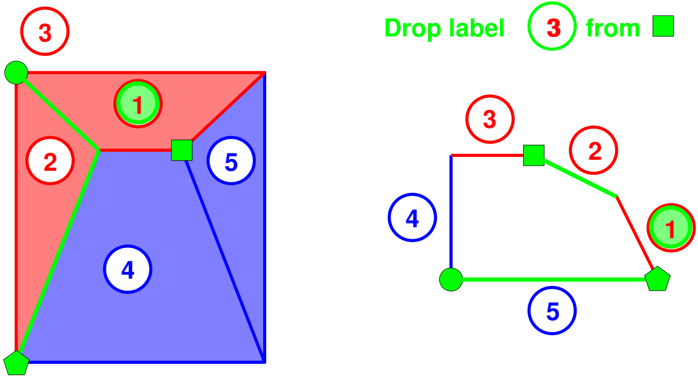


The Lemke–Howson algorithm



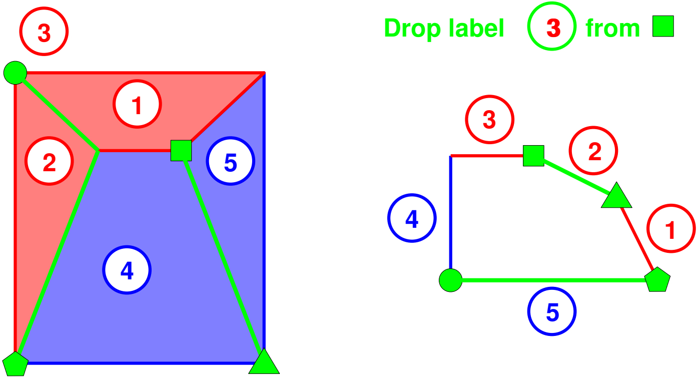
The Lemke–Howson algorithm

Drop label 3 from ■



The Lemke–Howson algorithm

Drop label 3 from



Why Lemke-Howson works

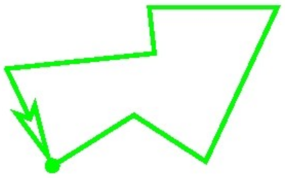
LH finds at least one Nash equilibrium because

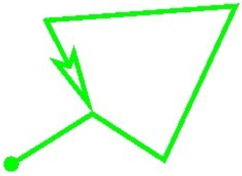
- finitely many "vertices"

for nondegenerate (generic) games:

- **unique** starting edge given missing label
- **unique** continuation

⇒ precludes "coming back" like here:





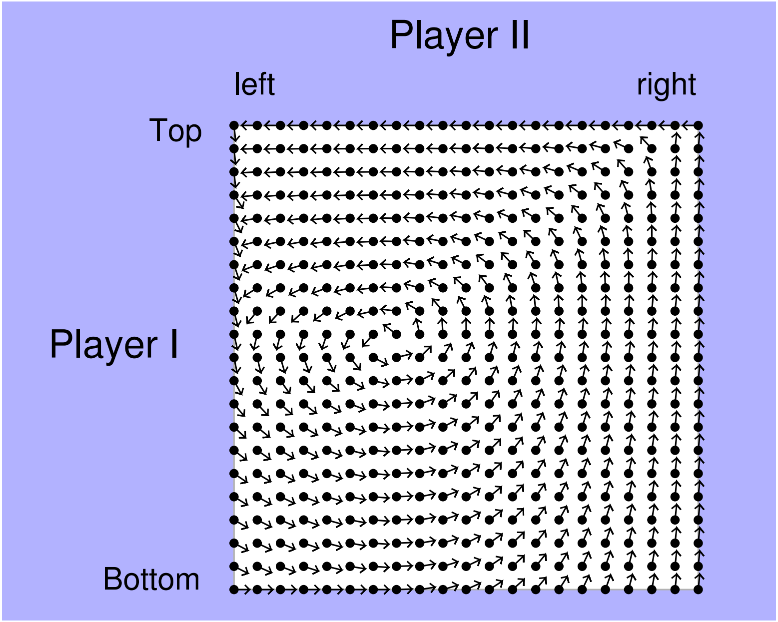
Lemke-Howson (LH) summary

- LH implies **non-degenerate** bimatrix game has **odd number of equilibria**, in particular **at least one**
- Extendable to full existence proof via **degeneracy resolution**
- From artificial equilibrium, LH can find upto $n + m$ equilibria of an $n \times m$ game; by chaining LH paths it might be able to find more
- The **shortest path** can be **exponentially long**
  [**S** and von Stengel (2004)]
- LH was the main motivation for the complexity class **PPAD**
- Next: alternative **existence proof** via **fixed points**

Existence of Nash equilibria

I \ II	left	right
Top	1, 2	2, 0
Bottom	2, 4	0, 6

“Incentive direction” of the players



Nash equilibrium

We are reducing the search for NE to search for a *Brouwer fixpoint*...

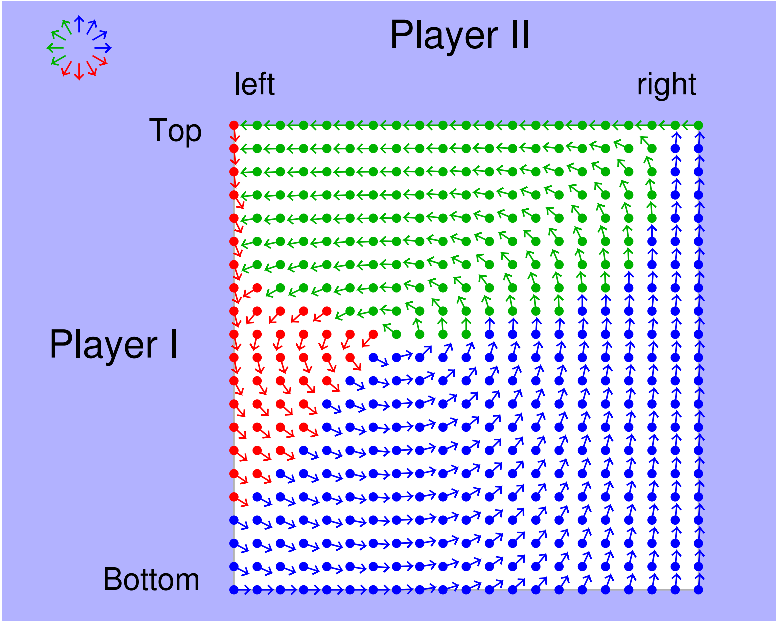
Brouwer's fixpoint theorem

continuous functions from a compact domain to itself, have fixpoints.

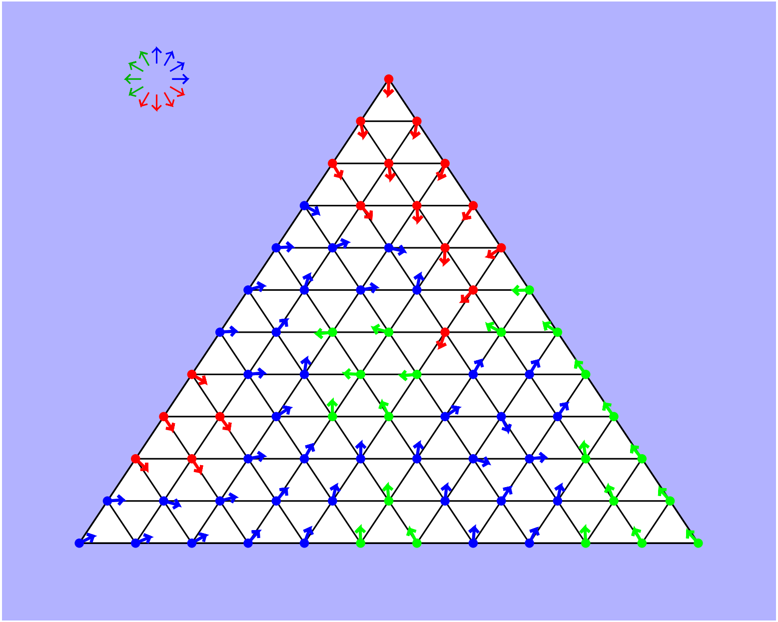
proof. construct *approximate* fixpoints (in a computationally inefficient manner) ...in a way that reduces computation of approx fixpoints to search on large graphs...

L.E.J. Brouwer (1881-1966)

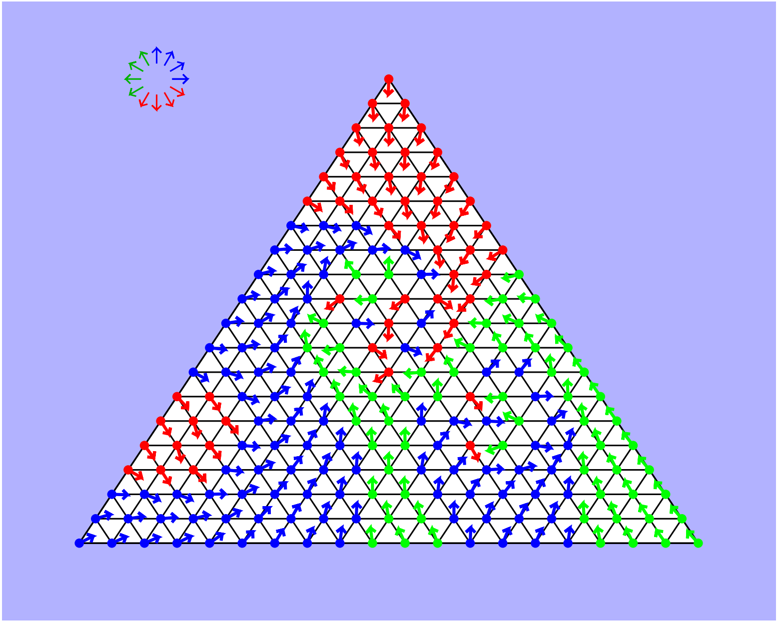
“Incentive direction”, colour-coded



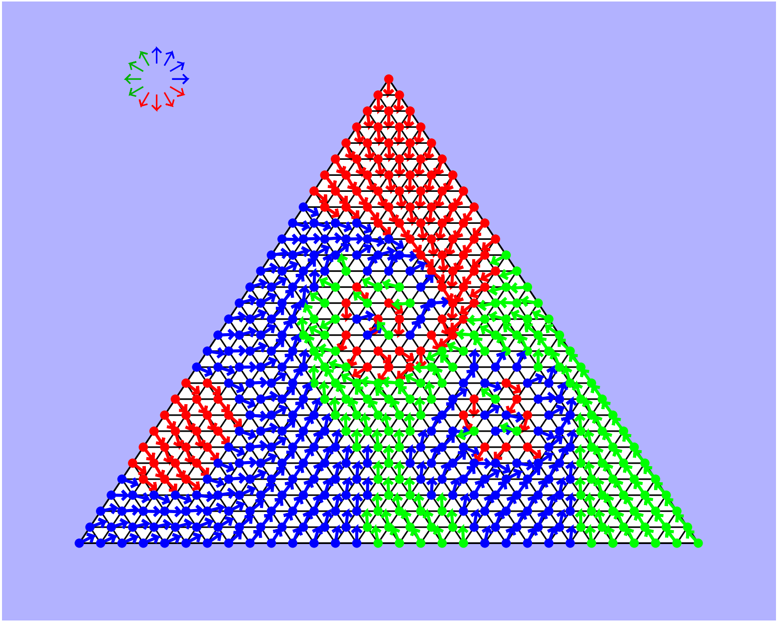
Now, pretend this triangle is high-dimension domain



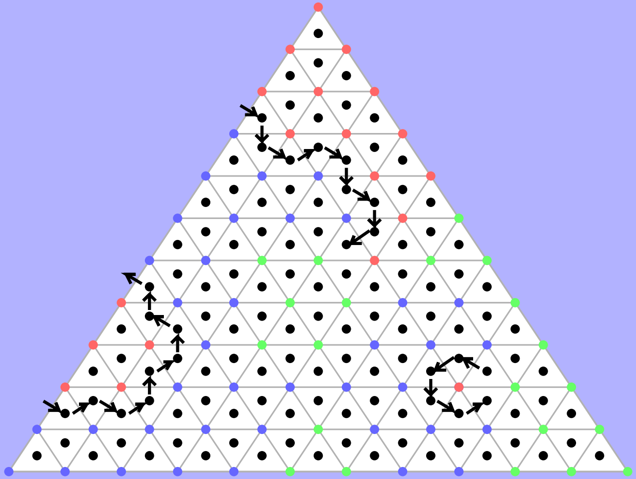
Search for “trichromatic triangles”



...converges to Brouwer fixpoint



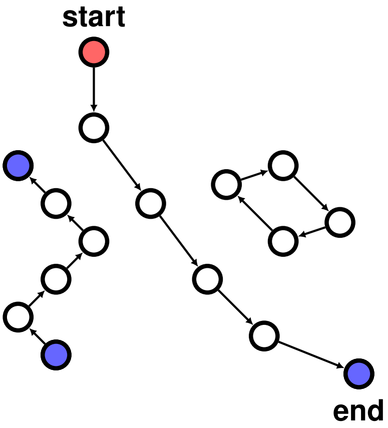
The corresponding graph



Motivation for PPAD

Both Lemke-Howson paths and the “Sperner paths” we just saw (as part of the proof of Brouwers fixed point theorem) **motivate** the definition of **PPAD** via the problem **End-of-Line**

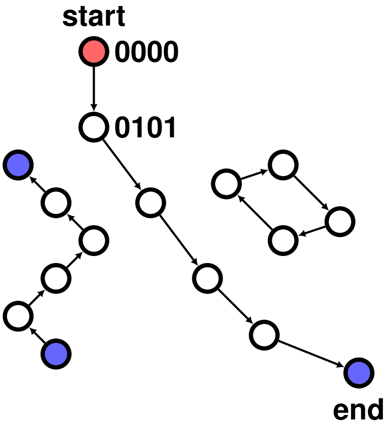
PPAD and End-of-Line (Papadimitriou 1991)



End-of-Line:

Given graph ***G*** of in/out degree at most 1 and a **source start** vertex **find another vertex of degree 1**

PPAD and End-of-Line (Papadimitriou 1991)



Catch:
The graph is **exponentially large**

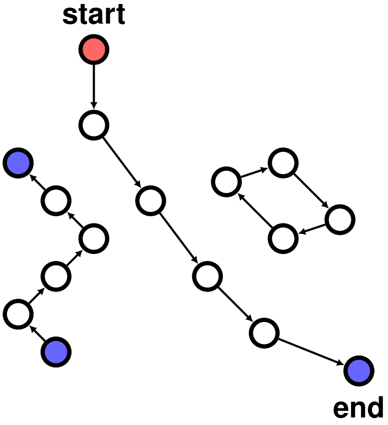
It is defined by

- Boolean successor circuit S
- Boolean predecessor circuit P

$$S(0000) = 0101$$

$$P(0101) = 0000$$

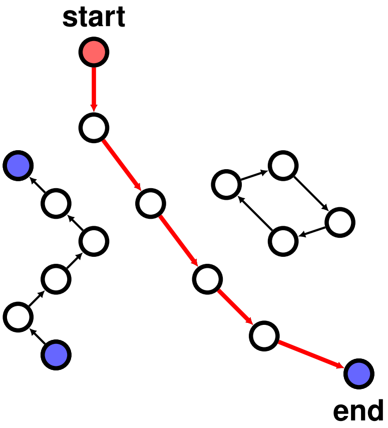
PPAD and End-of-Line (Papadimitriou 1991)



Problem ***A*** is

- in PPAD if ***A*** reduces to EOL
- PPAD-complete if EOL also reduces to it

PPAD and End-of-Line (Papadimitriou 1991)



Not to be confused with

OTHER END OF THIS LINE
output **unique sink** found by
“following the line” from the start
– this is **PSPACE**-hard

A view from the past

Christos Papadimitrou [STOC 2001]:

Together with factoring, the complexity of finding a Nash equilibrium is in my opinion the most important concrete open question on the boundary of P today.

MIXED-NASH of bimatrix games is PPAD-hard

Christos Papadimitrou [STOC 2001]:

Together with factoring, the complexity of finding a Nash equilibrium is in my opinion the most important concrete open question on the boundary of P today.

Resolved in 2006, NASH is PPAD-hard and thus unlikely to be in P:

The Complexity of Computing a Nash Equilibrium
Daskalakis, Goldberg, Papadimitriou

Settling the Complexity of Computing 2-player Nash Equilibria
Chen, Deng, Teng

From graph search to Nash equilibrium computation

Daskalakis, Goldberg and Papadimitriou '06, Chen, Deng and Teng '06

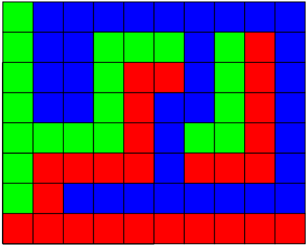
Intermediate step:

search for a **panchromatic point** of a **discrete Brouwer function** — in 2D,

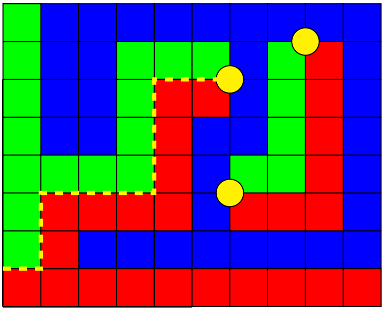
$$\mathbf{f : N \times N \longrightarrow \{red, green, blue\}}$$

where

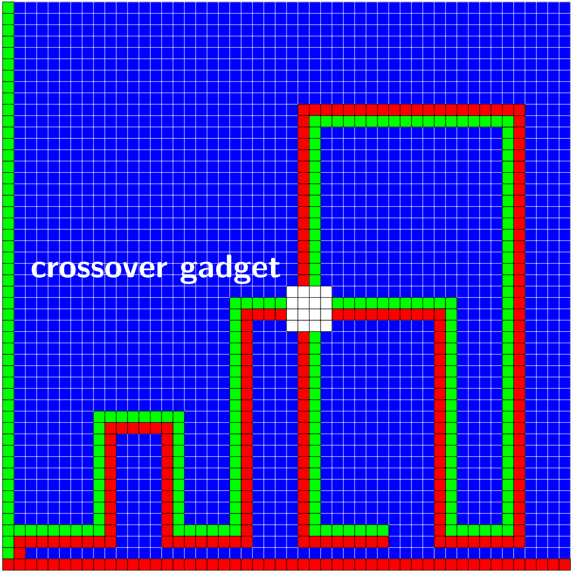
- the **bottom** is all **red**
- the **LHS** is all **green**
- the **top** and **RHS** is **blue**
- internal cells colored by poly-size boolean circuit



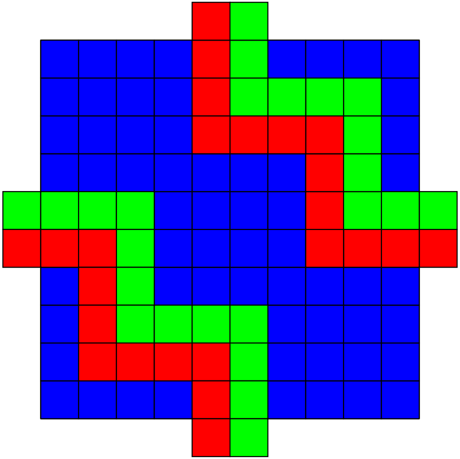
From graph search to finding Nash equilibria



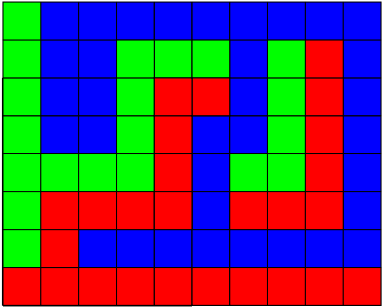
The reduction from END OF LINE in more detail

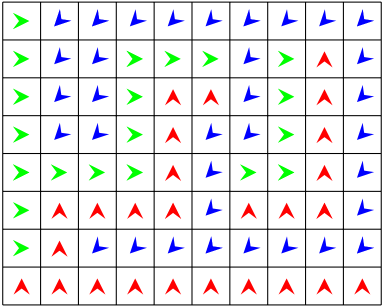


Crossover gadget



From discrete to continuous Brouwer functions





Gates for continuous Brouwer functions

Linear-FIXP (= **PPAD**) **[Etessami Yannakakis 2006]**

INPUT: algebraic circuit (straight-line program) over basis **{+, max, ×c, introduce c}**

OUTPUT: (approximate) fixed point of the circuit

Gates for continuous Brouwer functions

Linear-FIXP (= **PPAD**) **[Etessami Yannakakis 2006]**

INPUT: algebraic circuit (straight-line program) over basis $\{+, \max, \times c, \text{introduce } c\}$

OUTPUT: (approximate) fixed point of the circuit

For games, we work with a small variant of the problem:

INPUT: our basis $\{\text{bounded } +, \text{ bounded} \times c, \text{ introduce } c\}$
where: $\text{bounded}(x) = \max(\min(1, x), 0)$ "clips" output to $[0, 1]$

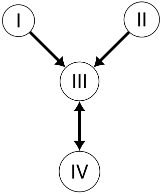
Polymatrix Games

- So far we have only looked at **two-player** bimatrix games
- PPAD-hardness of finding a Nash equilibrium first went via many-player games
- However, a general many-player strategic-form game has **exponential size** (in the number of players)
- Instead we use a special type of many-player game called a **polymatrix game**

Polymatrix games

- **many-player** graphical game
- **interaction graph** with
  nodes = players
  edges = **bimatrix games**
- single strategy for all player's bimatrix games
- player gets **sum of payoffs** from bimatrix games

Introduced by **Janovskaya (1968)**



III \ I	ℓ	r
T	0 / 0	0 / 1
B	0 / 0	0 / 0

III \ II	a	b
T	0 / 0	0 / 1
B	0 / 0	0 / 0

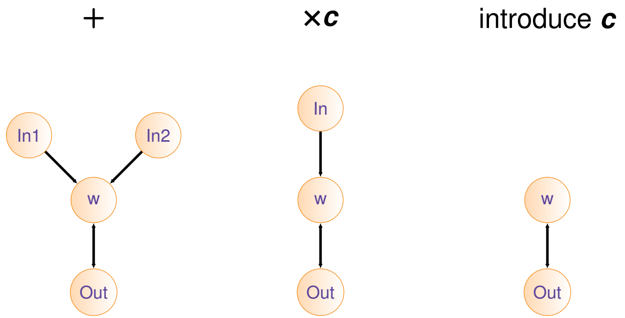
III \ IV	x	y
T	0 / 0	1 / 0
B	1 / 0	0 / 1

Succinct representation

	# players	# actions per player	# payoff entries
strategic-form	n	k	exponential: $n \cdot k^n$
polymatrix			quadratic: $2k^2 \cdot \binom{n}{2}$

DGP gadgets

Gadgets from **Daskalakis Goldberg Papadimitriou** [2006]:



- All these gadgets use 2 actions/player
- They all implement the **bounded** versions of these gates

EXERCISE: Addition gadget example

$$\ell = \min(p + q, 1)$$

In 1

w \ In 1	$(1-p)$	p
	0, 0	1, 0
	0, 0	0, 0

In 2

w \ In 2	$(1-q)$	q
	0, 0	1, 0
	0, 0	0, 0

Out

w \ Out	$(1-\ell)$	ℓ
	0, 0	0, 1
	0, 1	1, 0

ANSWER: Addition gadget example

$$\ell = \min(p + q, 1)$$

W \ Out	$1-\ell$	ℓ
	0 $p+q$	1 $p+q$
	1 0	0 1

ANSWER: Addition gadget example

Case 1/4: $p + q > 1, \quad \ell = \min(p+q, 1) = 1$

W \ Out	$1-\ell$	ℓ
	0 $p+q$	1 $p+q$
	1 0	0 1

ANSWER: Addition gadget example

Case 2/4: $p + q = 1, \quad \ell = \min(p + q, 1) = 1$

W \ Out	$1-\ell$	ℓ
	$p+q$, 0	$p+q$, 1
	0, 1	1, 0

ANSWER: Addition gadget example

Case 3/4: $p + q \in (0, 1)$, $\ell = p + q$

W \ Out	$1-\ell$	ℓ
	0 $p+q$	1 $p+q$
	1 0	0 1

ANSWER: Addition gadget example

Case 4/4: $\quad p + q = 0, \quad \ell = p + q = 0$

W \ Out	$1-\ell$	ℓ
	$p+q$, 0	$p+q$, 1
	0, 1	1, 0

Final step: polymatrix to bimatrix games

- The polymatrix game interaction graph can be made **bipartite**
- **Two players** in bimatrix game = **two parts** of interaction graph
- Additional **lawyer game** ensures that **all gates matter**

Recent advances: Pure Circuit

- Nice new PPAD-complete problem that reduces to games very natural with tight hardness of approximation

  Pure-Circuit: Strong Inapproximability for PPAD

  Deligkas, Fearnley, Hollender, Melissourgos

References

Exponentially Many Steps for Finding a Nash Equilibrium in a Bimatrix Game by **Savani and von Stengel** FOCS (2004) **Long shortest LH paths**

On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence by **Papadimitriou**
J. Comput. Syst. Sci. (1994) **PPAD, PPA, PPP, memberships and relationships**

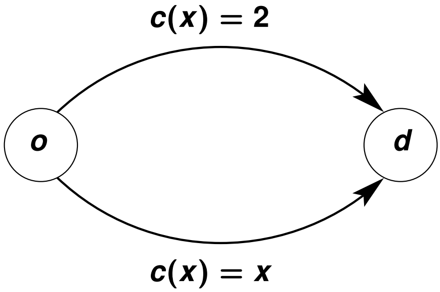
Pure-Circuit: Strong Inapproximability for PPAD by **Deligkas, Fearnley, Hollender, Melissourgos**
FOCS (2022) **Tight inapproximability results for bimatrix/polymatrix/graphical**

The Complexity of Computing a Nash Equilibrium by **Daskalakis, Goldberg, Papadimitriou**
STOC (2006) **PPAD-hardness for 3-NASH and then 2-NASH (bimatrix games)**

Settling the Complexity of Computing 2-player Nash Equilibria by **Chen, Deng, Teng** (2006) **PPAD-hardness for 2-NASH**

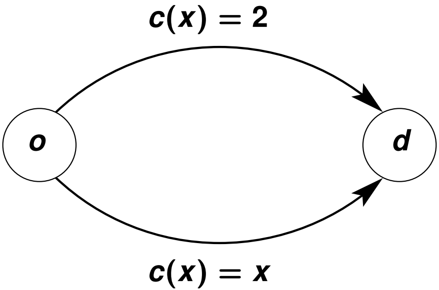
Polynomial Local Search (PLS)

A congestion network



2 users who want to travel from origin ***o*** to destination ***d***.

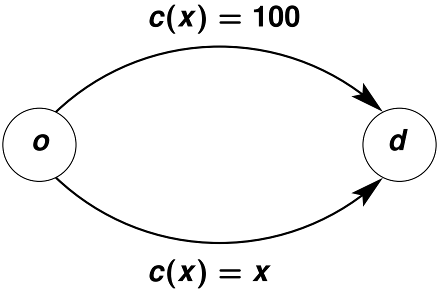
A congestion network



2 users who want to travel from origin $\boldsymbol{o}$ to destination $\boldsymbol{d}$.

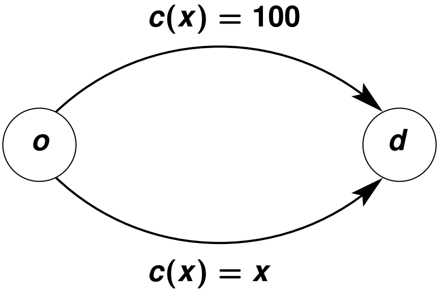
Possible routes:
both users on top edge,
1 user on top edge and 1 user on bottom edge,
both users on bottom edge

A similar “Pigou” congestion network



100 users who want to travel from origin o to destination d.

A similar "Pigou" congestion network

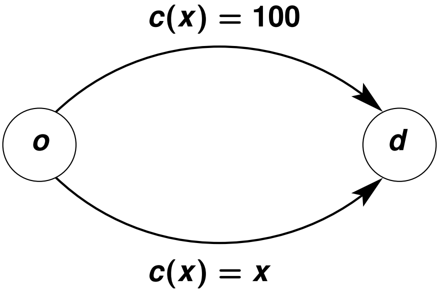


100 users who want to travel from origin ***o*** to destination ***d***.

Assume ***y*** users on bottom edge, **100 – *y*** on top edge.

Equilibrium?

A similar "Pigou" congestion network



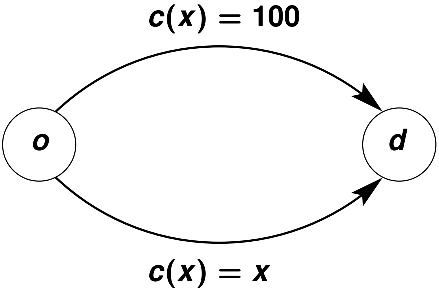
100 users who want to travel from origin ***o*** to destination ***d***.

Assume ***y*** users on bottom edge, **100 – *y*** on top edge.

Equilibrium? $y = 99$ or $y = 100$

Optimum?

A similar "Pigou" congestion network



100 users who want to travel from origin o to destination d.

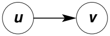
Assume y users on bottom edge, $100 - y$ on top edge.

Equilibrium? $y = 99$ or $y = 100$

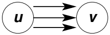
Optimum? $y = 50$

Congestion network – components

- finite set of **nodes**

- finite collection $\boldsymbol{E}$ of **edges** $\boldsymbol{e = uv}$ 

,

  parallel edges 

allowed.

- For each $\boldsymbol{e \in E}$ a **cost function** $\boldsymbol{c_e(x)}$ for **flow** (usage) $\boldsymbol{x}$.

- $\boldsymbol{n}$ **users** $\boldsymbol{i = 1, 2, \ldots, n}$ with origin $\boldsymbol{o_i}$ and destination $\boldsymbol{d_i}$

- **strategy** of user $\boldsymbol{i}$ = route (path) $\boldsymbol{P_i}$ from $\boldsymbol{o_i}$ to $\boldsymbol{d_i}$.

- Given strategies $\boldsymbol{P_1, \ldots, P_n}$, **flow** on $\boldsymbol{e}$ is $\boldsymbol{f_e = |\{i \mid e \in P_i\}|}$ and resulting cost $\boldsymbol{c_e(f_e)}$ for **every** user of $\boldsymbol{e}$.

- Cost to user $\boldsymbol{i}$ for strategy $\boldsymbol{P_i}$ is

$$\sum_{e \in P_i} c_e(f_e)$$

Best responses and equilibrium

Given $P_1, \ldots, P_n$ with resulting flow f, strategy P_i of user i is a

best response $\Leftrightarrow$ for any other deviating strategy Q_i

$$\sum_{e \in P_i} c_e(f_e) \leq \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

Best responses and equilibrium

Given $P_1, \ldots, P_n$ with resulting flow f, strategy P_i of user i is a

best response $\Leftrightarrow$ for any other deviating strategy Q_i

$$\sum_{e \in P_i} c_e(f_e) \leq \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

Definition

strategy **profile** $P_1, \ldots, P_n$ **is an equilibrium**

$\Leftrightarrow$ every strategy P_i is a best response to the others.

Every congestion game has an equilibrium

Proof

Given $P_1, \ldots, P_n$ and flow f, define the **potential function**

$$\Phi(f) = \sum_{e \in E} \Big(c_e(1) + c_e(2) + \cdots + c_e(f_e) \Big).$$

Every congestion game has an equilibrium

Proof

Given $P_1, \ldots, P_n$ and flow f, define the **potential function**

$$\Phi(f) = \sum_{e \in E} \Big(c_e(1) + c_e(2) + \cdots + c_e(f_e) \Big).$$

Let Q_i be any other strategy of user i with flow f^{Q_i}. Will show:

$$\Phi(f^{Q_i}) - \Phi(f) = \sum_{e \in Q_i} c_e(f_e^{Q_i}) - \sum_{e \in P_i} c_e(f_e). \tag{2.4}$$

Every congestion game has an equilibrium

Proof

Given $P_1, \ldots, P_n$ and flow f, define the **potential function**

$$\Phi(f) = \sum_{e \in E} \Big(c_e(1) + c_e(2) + \cdots + c_e(f_e) \Big).$$

Let Q_i be any other strategy of user i with flow f^{Q_i}. Will show:

$$\Phi(f^{Q_i}) - \Phi(f) \;=\; \sum_{e \in Q_i} c_e(f_e^{Q_i}) \;-\; \sum_{e \in P_i} c_e(f_e)\,. \qquad (2.4)$$

$\Rightarrow$ changes in Φ reflect changes in cost for (any) user i

$\Rightarrow$ minimum of Φ defines an equilibrium. □

Proof of potential function property (2.4)

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) = \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

Proof of potential function property (2.4)

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) = \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

$$\sum_{e \in P_i} c_e(f_e) = \sum_{e \in P_i \cap Q_i} c_e(f_e) + \sum_{e \in P_i \setminus Q_i} c_e(f_e)$$

Proof of potential function property (2.4)

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) = \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

$$\sum_{e \in P_i} c_e(f_e) = \sum_{e \in P_i \cap Q_i} c_e(f_e) + \sum_{e \in P_i \setminus Q_i} c_e(f_e)$$

so

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) - \sum_{e \in P_i} c_e(f_e) = \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1) - \sum_{e \in P_i \setminus Q_i} c_e(f_e)$$

Proof of potential function property (2.4)

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) = \sum_{e \in Q_i \cap P_i} c_e(f_e) + \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1)$$

$$\sum_{e \in P_i} c_e(f_e) = \sum_{e \in P_i \cap Q_i} c_e(f_e) + \sum_{e \in P_i \setminus Q_i} c_e(f_e)$$

so

$$\sum_{e \in Q_i} c_e(f_e^{Q_i}) - \sum_{e \in P_i} c_e(f_e) = \sum_{e \in Q_i \setminus P_i} c_e(f_e + 1) - \sum_{e \in P_i \setminus Q_i} c_e(f_e)$$

$= \Phi(f^{Q_i}) - \Phi(f)$ because

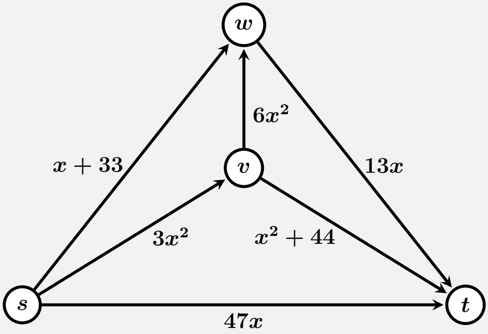
$$\Phi(f) = \sum_{e \in E} \Big(c_e(1) + c_e(2) + \cdots + c_e(f_e) \Big).$$

Remark

- Pure equilibrium may fail to exist with **weighted** users (e.g. **1** for passenger car, **2** for lorry)

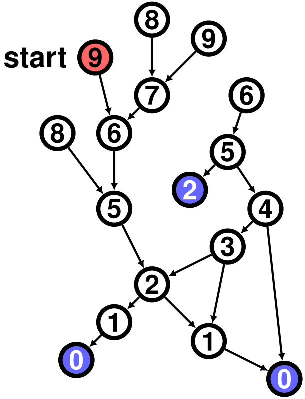
Exercise

- Consider the following **two-player** routing game. Both players want to go from s to t. They have **weights** w_1, w_2 respectively.



- Consider two cases:
  (i) $w_1 = 1, w_2 = 2$ (weighted); (ii) $w_1 = w_2 = 1$ (unweighted)
- For each case, **convert the game to a bimatrix game** and **compute all equilibria** (pure and mixed). Show your working.
  Hint: For case (i), you can dramatically simplify the game with

Polynomial Local Search (PLS)



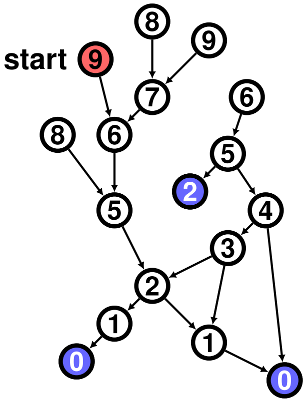
Given

- a DAG
- a starting vertex

Find

- a sink vertex

Polynomial Local Search (PLS)



Catch:
The graph is **exponentially large**

Defined by

- A circuit ***S*** giving the successor vertices
- A circuit ***p*** giving a **potential**

Every edge decreases the potential

$$p(S(v)) < p(v)$$

Complexity results for congestion games

Finding a pure Nash equilibrium in a congestion game is

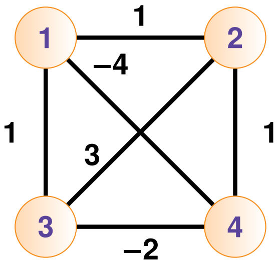
- **Polynomial-time** solvable for **symmetric network games**
- **PLS-complete** for **asymmetric network games**
- **PLS-complete** for **symmetric general games**
- **PLS-complete** for **asymmetric general games**

Local Max Cut

- Find local optimum of **Max Cut with the FLIP-neighbourhood** (exactly one node can change sides)
- Schäffer and Yannakakis **[SICOMP, 1991]** showed that **Local Max Cut is PLS-complete** (via an extremely involved reduction)
- **Local Max Cut is to PLS what 3-SAT is to NP**

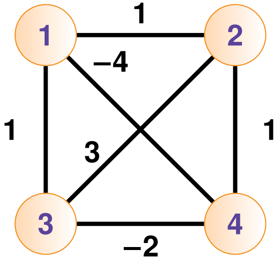
Local Max Cut

- Find local optimum of **Max Cut with the FLIP-neighbourhood** (exactly one node can change sides)
- Schäffer and Yannakakis **[SICOMP, 1991]** showed that **Local Max Cut is PLS-complete** (via an extremely involved reduction)
- **Local Max Cut is to PLS what 3-SAT is to NP**



Local Max Cut

- Find local optimum of **Max Cut with the FLIP-neighbourhood** (exactly one node can change sides)
- Schäffer and Yannakakis **[SICOMP, 1991]** showed that **Local Max Cut is PLS-complete** (via an extremely involved reduction)
- **Local Max Cut is to PLS what 3-SAT is to NP**

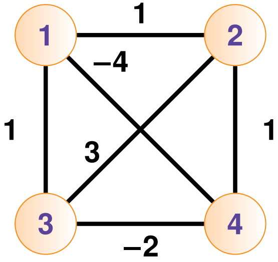


Solutions:

{{1, 3, 4}, {2}} (actual Max Cut)

Local Max Cut

- Find local optimum of
  Max Cut with the FLIP-neighbourhood (exactly one node can change sides)
- Schäffer and Yannakakis **[SICOMP, 1991]** showed that **Local Max Cut is PLS-complete** (via an extremely involved reduction)
- **Local Max Cut is to PLS what 3-SAT is to NP**



Solutions:

{{1, 3, 4}, {2}} (actual Max Cut)
{{3}, {1, 2, 4}}

Local-Max-Cut as the Party Affiliation Game

Players correspond to nodes in weighted graph $G = (V, E)$:

- Every player has 2 strategies: **left** or **right.**
- Strategy profile yields a cut, i.e., partition of V into left/right nodes
- **Edge weights** represent **antisympathy**
- Players **maximize sum of weights of incident cut edges**
- **Nash equilibria** in 1-1 correspondence with local max cuts

Minimization Variant of Party Affiliation Game

- For the congestion game we want **costs**:

  sum of incident edges on the **same side of the cut**

- This is equivalent because, for each node and strategy profile:

  > Total weight of all incident edges =
  > **incident cut edges** + **incident edges on same side**

  where the left-hand-side is a constant

General congestion game for Minimization Party Affiliation Game

- Represent each edge e by two resources:
  e_{left} , e_{right} with delay functions $d(1) = 0$ and $d(2) = w_e$
- For each player:
  - strategy S_{left} contains resource e_{left} for all incident edges;
  - strategy S_{right} contains resources e_{right} for all incident edges
- Players in the **congestion game** have **exactly the same cost** as players in the minimization variant of the **party affiliation game**
- Hence, the Nash equilibria of this congestion game coincide with local max cuts, QED

PLS-hardness for congestion games

Results from **Fabrikant, Papadimitriou, Talwar [2004]**

	network games	general games
symmetric	In P-time	PLS-complete
asymmetric	PLS-complete	**PLS-complete**

We presented **simplest case** of **asymmetric congestion games**

PLS-hardness for congestion games

Results from **Fabrikant, Papadimitriou, Talwar [2004]**

	network games	general games
symmetric	In P-time	PLS-complete
asymmetric	PLS-complete	**PLS-complete**

We presented **simplest case** of **asymmetric congestion games**

Why is the resulting game

- **asymmetric** and
- **not a network congestion** game?

References

A class of games possessing pure-strategy Nash equilibria by **Rosenthal**
Int. J. of Game Theory (1973) — **Congestion games have pure equilibria**

Potential Games by **Monderer and Shapley**
Games & Economic Behavior (1996) — **Congestion ≡ potential games**

How Easy is Local Search? by **Johnson, Papadimitriou, Yannakakis**
J. Comput. Syst. Sci (1998) — **Introduced PLS**

The complexity of pure Nash equilibria by **Fabrikant, Papadimitriou, Talwar**
STOC 2004 — **PLS-completeness in congestion games**

On the impact of combinatorial structure on congestion games by **Ackermann, Röglin, Vöcking** Journal of the ACM (2008) — **Further PLS-hardness**

Continuous Local Search (CLS)

Gradient descent

$$\textbf{minimise } f(x) \quad \text{s.t.} \quad x \in [0,1]^n$$

assume *f* continuously differentiable, but not necessarily convex

Gradient descent

$$\textbf{minimise } f(x) \quad \text{s.t.} \quad x \in [0,1]^n$$

NP-hard even for a quadratic polynomial given explicitly

Gradient descent

minimise $f(x)$ s.t. $x \in [0, 1]^n$ **NP−hard**

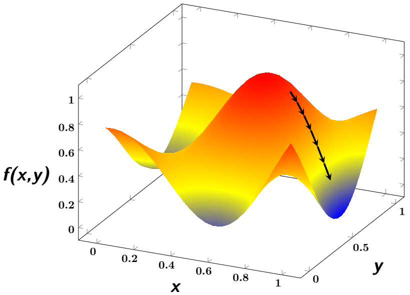
Gradient Descent: $x_{k+1} \leftarrow x_k - \eta \nabla(f(x_k))$ **(**η **: step size)**

Intuition: “move in the direction of steepest descent”

Gradient descent

(1) : **minimise $f(x)$** s.t. $x \in [0,1]^n$ **NP–hard**

Gradient Descent: $x_{k+1} \leftarrow x_k - \eta\nabla(f(x_k))$ **(η : step size)**



Gradient descent being applied to a function $f : [0,1]^2 \mapsto [0,1]$

Gradient descent

(1) : **minimise** $\boldsymbol{f(x)}$ s.t. $\boldsymbol{x \in [0,1]^n}$ **NP−hard**

Gradient Descent: $\boldsymbol{x_{k+1} \leftarrow x_k - \eta \nabla (f(x_k))}$ **(η : step size)**

Doesn’t actually solve (1); can get stuck in any **stationary point**

Gradient descent

minimise $f(x)$ s.t. $x \in [0,1]^n$ **NP−hard**

Gradient Descent: $x_{k+1} \leftarrow x_k - \eta\nabla(f(x_k))$ **(**η **: step size)**

Doesn't actually solve (1); can get stuck in any **stationary point**

actually a Karush-Kuhn-Tucker point (due to boundaries)

Gradient descent

minimise $f(x)$ s.t. $x \in [0,1]^n$ **NP−hard**

Gradient Descent: $x_{k+1} \leftarrow x_k - \eta \nabla (f(x_k))$ **(**η **: step size)**

What is the complexity of finding a solution where gradient descent terminates?

Gradient descent

minimise $f(x)$ s.t. $x \in [0,1]^n$ **NP−hard**

Gradient Descent: $x_{k+1} \leftarrow x_k - \eta\nabla(f(x_k))$ **(**η **: step size)**

What is the complexity of finding a solution where gradient descent terminates?

Let's explore how to formalise this...

Gradient descent problem

Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

Gradient descent problem

Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

$[x' := x - \eta \nabla f(x))]$

GD-Local-Search: find x s.t. $f(x') \geq f(x) - \epsilon$

limited improvement

Gradient descent problem

Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

GD-Local-Search: find x s.t. $f(x') \geq f(x) - \epsilon$
limited improvement

GD-Fixed-Point: find x s.t. $\|x' - x\| \leq \epsilon$
x not moved by much

Gradient descent problem

Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

GD-Local-Search: find x s.t. $f(x') \geq f(x) - \epsilon$
limited improvement

GD-Fixed-Point: find x s.t. $\|x' - x\| \leq \epsilon$
x not moved by much

These two problems are **polynomial-time equivalent**

Gradient descent problem

Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

One way to solve this problem: **run Gradient Descent**!

Running time: **polynomial in $1/\epsilon$, not in input size**

Gradient descent problem

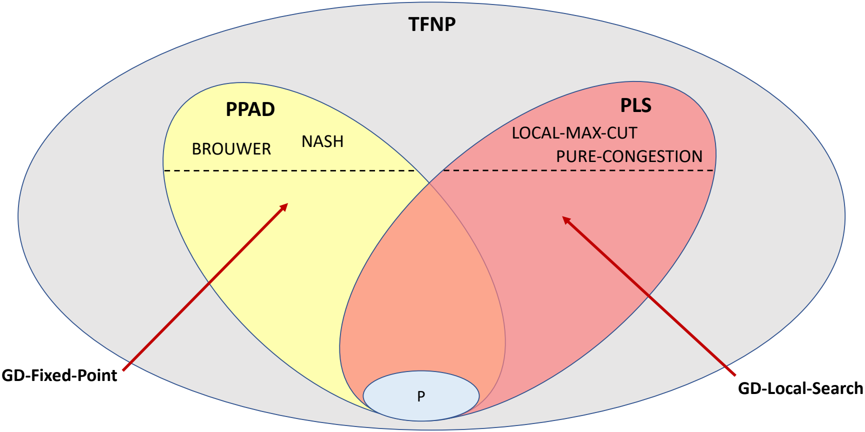
Input: C^1 function $f : [0,1]^n \mapsto \mathbb{R}$, stepsize $\eta > 0$, precision $\epsilon > 0$
(f and ∇f given as arithmetic circuits)

Goal: find a point where gradient descent terminates

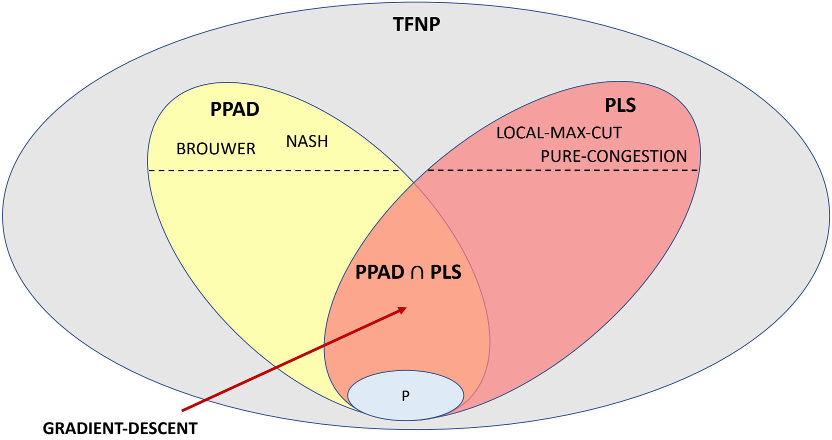
Can it be solved in time polynomial in $\log(1/\epsilon)$?

(f convex: yes, e.g., via the Ellipsoid method)

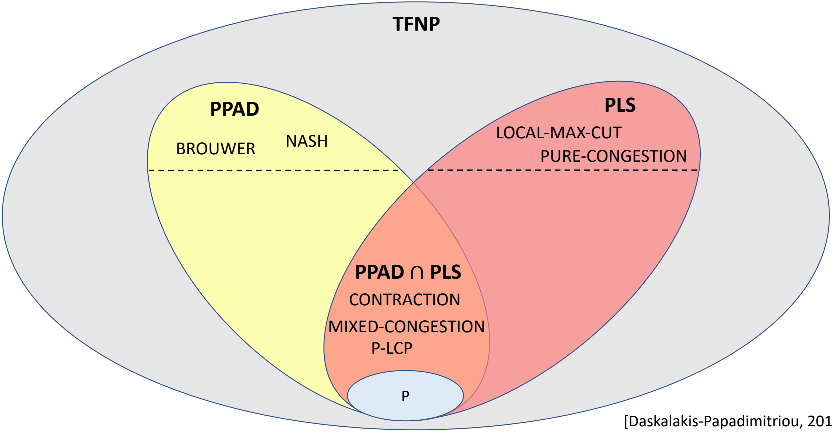
PPAD ∩ PLS


TFNP
PPAD
BROUWER
NASH
PLS
LOCAL-MAX-CUT
PURE-CONGESTION
P
GD-Fixed-Point
GD-Local-Search

PPAD ∩ PLS


TFNP
PPAD
BROUWER
NASH
PLS
LOCAL-MAX-CUT
PURE-CONGESTION
PPAD ∩ PLS
P
GRADIENT-DESCENT

PPAD ∩ PLS



[Daskalakis-Papadimitriou, 2011]

Unlikely containments

Consider a problem A in PPAD $\cap$ PLS

Since A is in both classes:

- If A is PPAD-hard then PPAD $\subseteq$ PLS
- If A is PLS-hard then PLS $\subseteq$ PPAD

Unlikely containments

Consider a problem ***A*** in PPAD ∩ PLS

Since ***A*** is in both classes:

- If ***A*** is PPAD-hard then PPAD ⊆ PLS
- If ***A*** is PLS-hard then PLS ⊆ PPAD

We do not believe that either containments holds, so
we do not believe *A* is PPAD-hard or PLS-hard

PPAD ∩ PLS seems unnatural...

Suppose problem ***A*** is **PPAD**-complete

Suppose problem ***B*** is **PLS**-complete

The following problem is **PPAD ∩ PLS**-complete:

EITHER(*A*,*B*)

Input: an instance I_A of ***A***, an instance I_B of ***B***

Output: a solution of I_A, or a solution of I_B

PPAD ∩ PLS seems unnatural...

BROUWER (PPAD-complete):
Input: continuous function $f : [0, 1]^3 \mapsto [0, 1]^3$, precision $\epsilon > 0$
Output: approximate fixpoint x:

$$||f(x) - x|| \leq \epsilon$$

PPAD ∩ PLS seems unnatural...

BROUWER (PPAD-complete):
Input: continuous function $f : [0, 1]^3 \mapsto [0, 1]^3$, precision $\epsilon > 0$
Output: approximate fixpoint x:

$$||f(x) - x|| \leq \epsilon$$

LOCAL-OPT (PLS-complete):
Input: continuous function $p : [0, 1]^3 \mapsto [0, 1]$, (non-continuous) function $g : [0, 1]^3 \mapsto [0, 1]^3$, precision $\epsilon > 0$
Output: local minimum x of p w.r.t. g:

$$p(g(x)) \geq p(x) - \epsilon$$

PPAD ∩ PLS seems unnatural...

BROUWER (PPAD-complete):
Input: continuous function $f : [0,1]^3 \mapsto [0,1]^3$, precision $\epsilon > 0$
Output: approximate fixpoint x:

$$\|f(x) - x\| \leq \epsilon$$

LOCAL-OPT (PLS-complete):
Input: continuous function $p : [0,1]^3 \mapsto [0,1]$, (non-continuous) function $g : [0,1]^3 \mapsto [0,1]^3$, precision $\epsilon > 0$
Output: local minimum x of p w.r.t. g:

$$p(g(x)) \geq p(x) - \epsilon$$

EITHER(BROUWER,LOCAL-OPT) is **PPAD ∩ PLS**-complete

Continuous Local Search (CLS)

Daskalakis & Papadimitriou [SODA 2011] defined a new class via:

CONTINUOUS-LOCAL-OPT

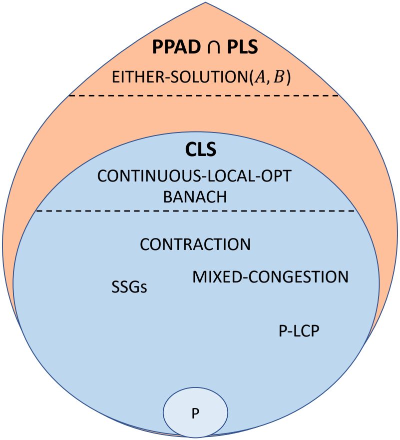
Input:
continuous $p : [0,1]^3 \mapsto [0,1]$ and
continuous $f : [0,1]^3 \mapsto [0,1]^3$, precision $\epsilon > 0$

Output: local minimum x of p w.r.t. f:

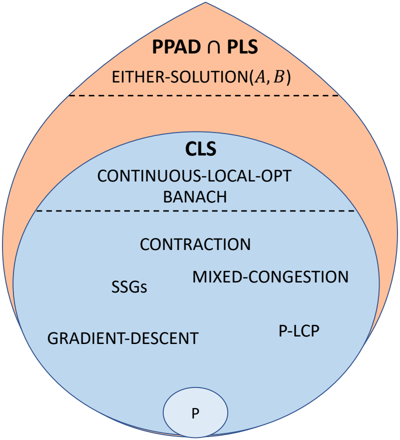
$$p(f(x)) \geq p(x) - \epsilon$$

CLS is the class of all problems that are polynomial-time reducible to **CONTINUOUS-LOCAL-OPT**

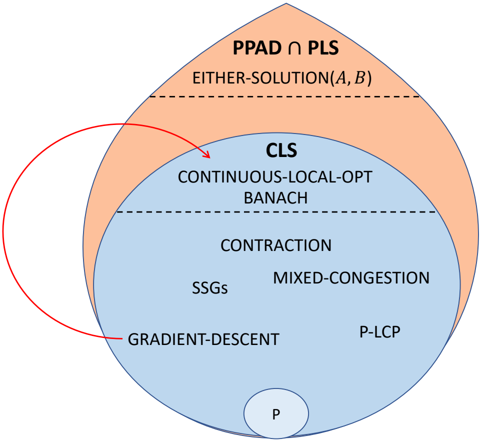
PPAD ∩ PLS and CLS



PPAD ∩ PLS and CLS

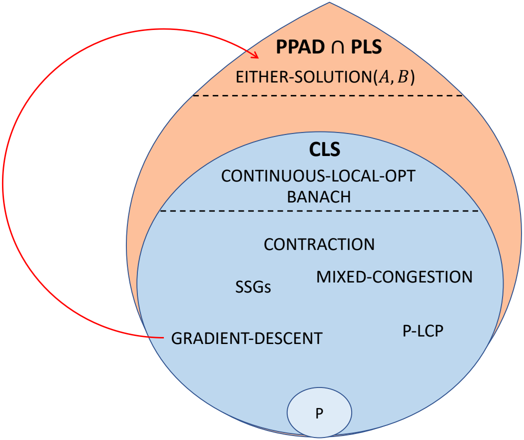


Collapse

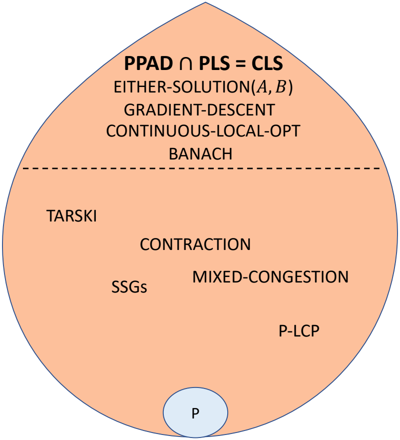


PPAD ∩ PLS
EITHER-SOLUTION(A, B)
CLS
CONTINUOUS-LOCAL-OPT
BANACH
CONTRACTION
SSGs
MIXED-CONGESTION
GRADIENT-DESCENT
P-LCP
P

Collapse


PPAD ∩ PLS
EITHER-SOLUTION(A, B)
CLS
CONTINUOUS-LOCAL-OPT
BANACH
CONTRACTION
SSGs
MIXED-CONGESTION
GRADIENT-DESCENT
P-LCP
P

Collapse



PPAD ∩ PLS = CLS
EITHER-SOLUTION(A, B)
GRADIENT-DESCENT
CONTINUOUS-LOCAL-OPT
BANACH
TARSKI
CONTRACTION
SSGs
MIXED-CONGESTION
P-LCP
P

Main Result

GRADIENT-DESCENT is **PPAD** ∩ **PLS** – hard

Main Result

Reduction from **EITHER(A, B)** to **2D-GRADIENT-DESCENT**

where

A is the **PPAD**-complete problem **End-of-Line**
B is the **PLS**-complete problem **ITER**

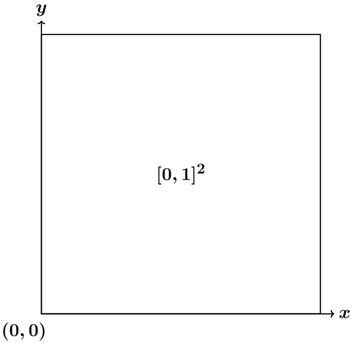
Proof Sketch

Reduction from **EITHER(A, B)** to **2D-GRADIENT-DESCENT**
where
A is the **PPAD**-complete problem **End-of-Line**
B is the **PLS**-complete problem **ITER**

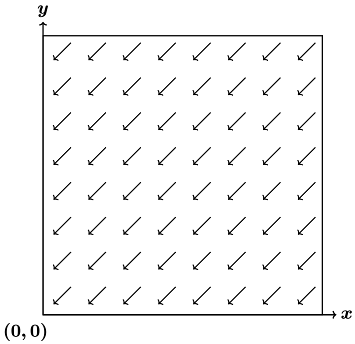
Constructing a 2D-GRADIENT-DESCENT instance *f*

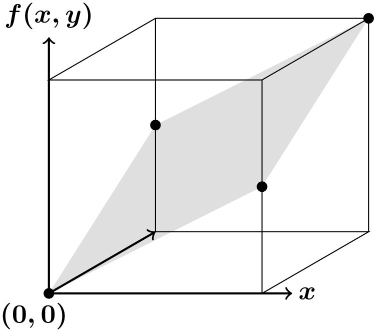
- Domain is the **square** $[0, 1]^2$
- Overlay grid and **assign values for *f* and ∇f at grid points**
- Use **bicubic interpolation** to produce smooth function
- All **stationary points** are either **End-Of-Line** or **ITER** solutions

Background “landscape”

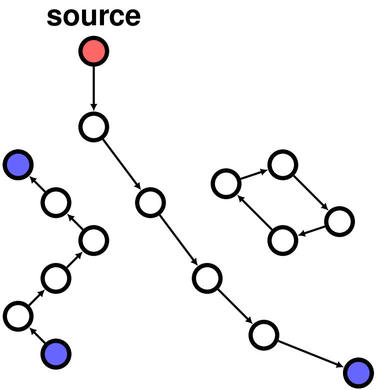


Background “landscape”





PPAD-complete problem: End-Of-Line

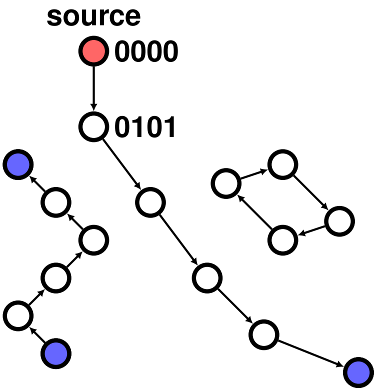


Given a graph of
indegree/outdegree at most 1

and a **source**
(indegree 0, outdegree 1)

find another vertex of degree 1

PPAD-complete problem: End-Of-Line



Catch:

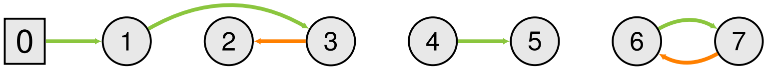
graph is **exponentially large**

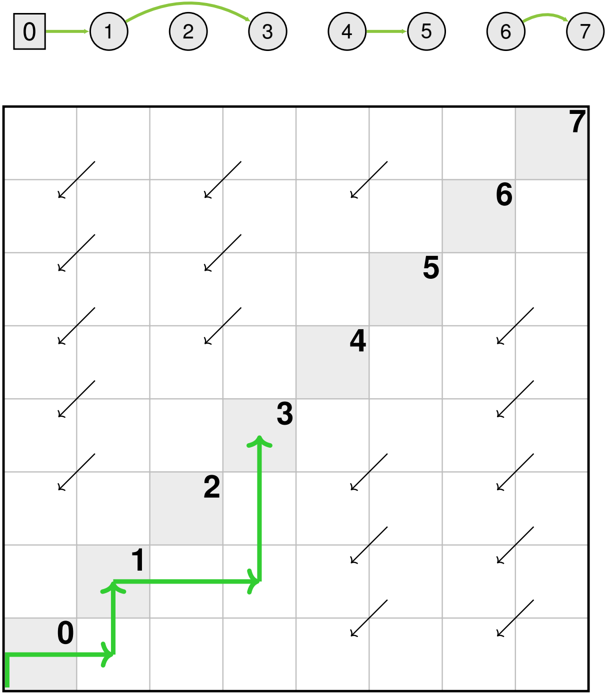
defined by boolean circuits $\boldsymbol{S}$, $\boldsymbol{P}$ that map a vertex $\{0,1\}^n$ to its successor and predecessor

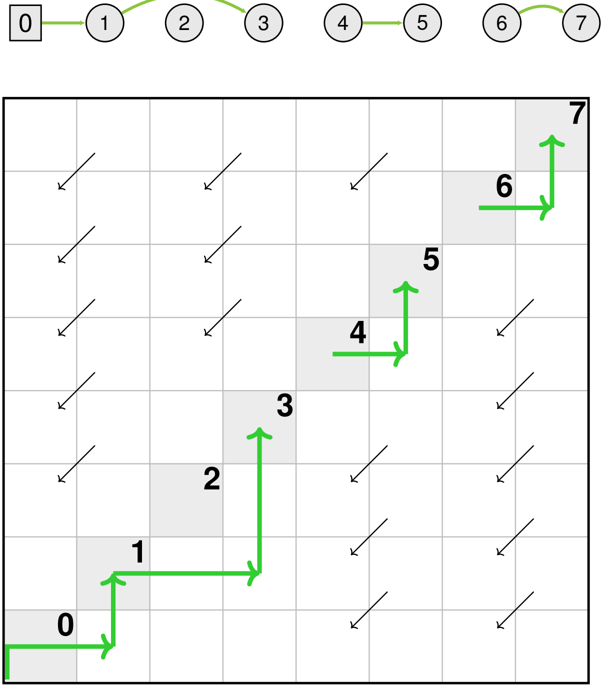
$$S(0000) = 0101$$

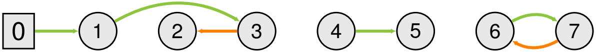
$$P(0101) = 0000$$

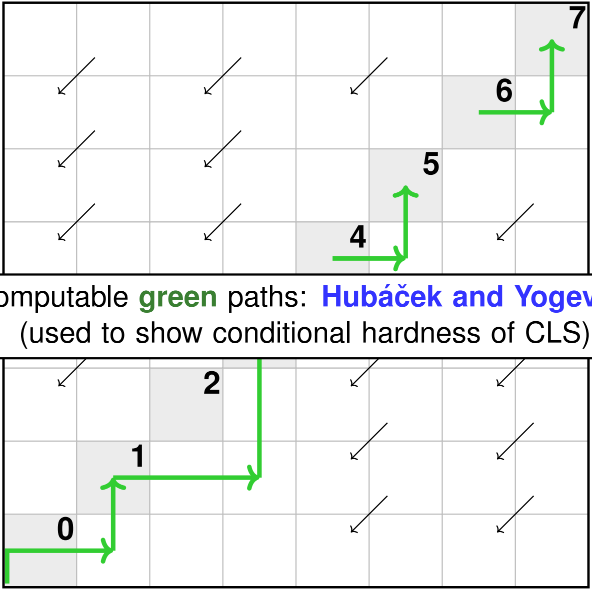
PPAD-complete problem: End-Of-Line



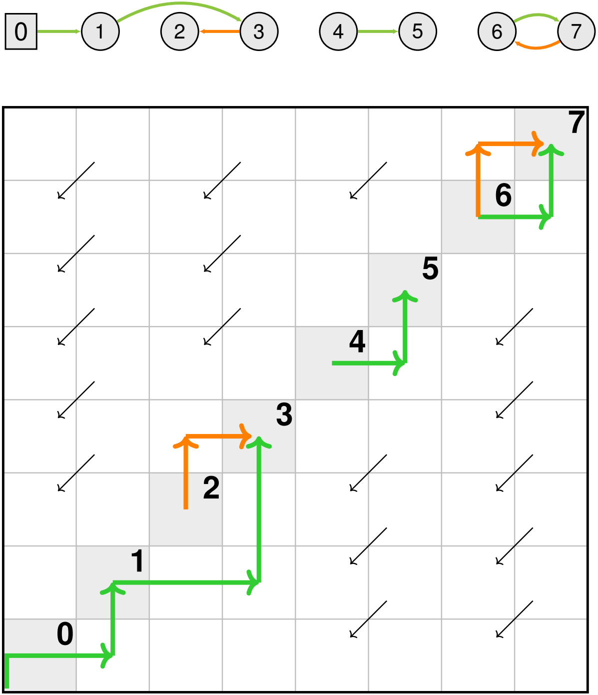

0
1
2
3
4
5
6
7

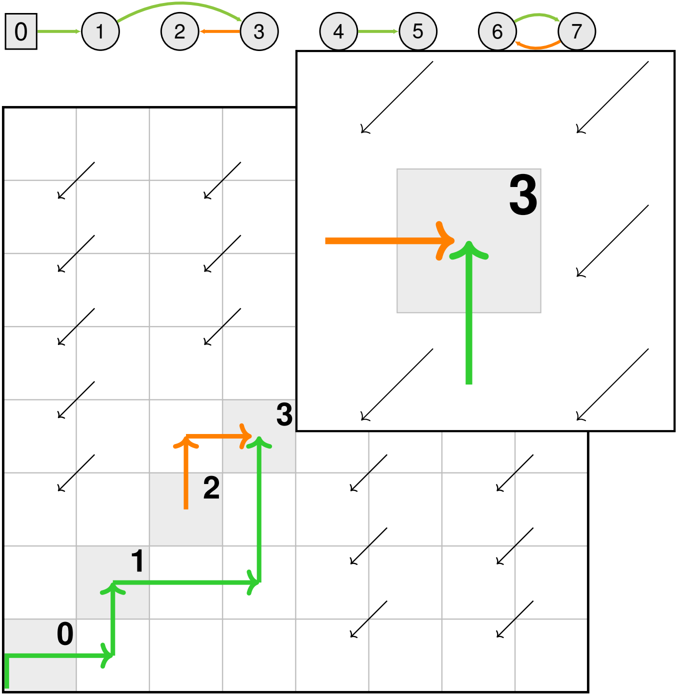

0
1
2
3
4
5
6
7



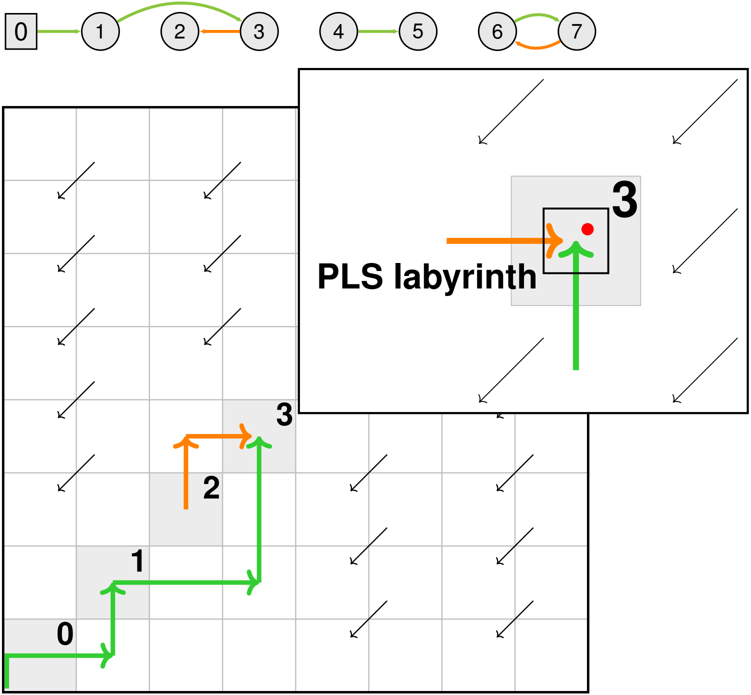


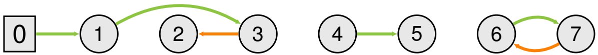
Locally-computable **green** paths: **Hubáček and Yogev SODA'17**
(used to show conditional hardness of CLS)


0
1
2
3
4
5
6
7
0
1
2
3
4
5
6
7

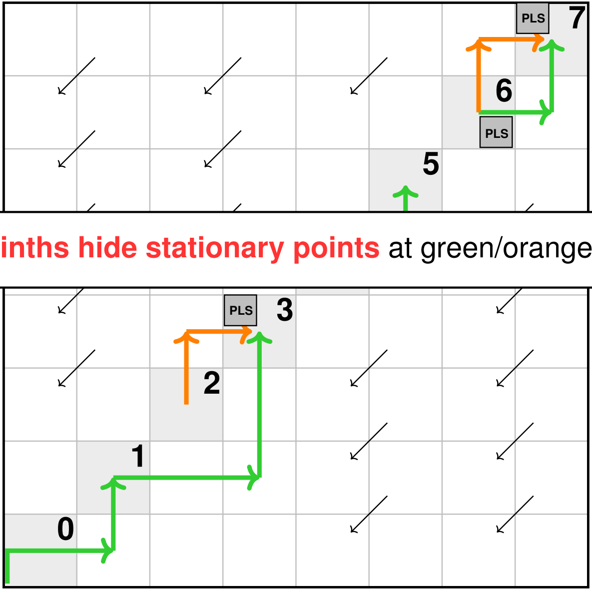


0
1
2
3
4
5
6
7
3
3
2
1
0

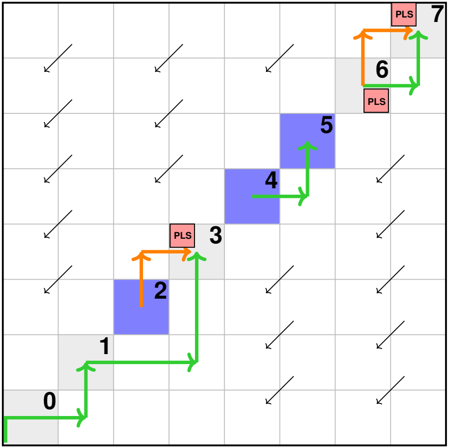

0
1
2
3
4
5
6
7
PLS labyrinth
0
1
2
3
3



PLS labyrinths hide stationary points at green/orange meetings



All stationary points are:
solutions of **End-of-Line** instance; or
solutions of **PLS-complete** labyrinth



We have shown: **2D-GRADIENT-DESCENT** is **PPAD ∩ PLS** – hard

Increasing lines: EOPL

- After our result in a further collapse it was proved that:

$$\text{EOPL} = \text{PPAD} \cap \text{PLS}$$

- EOPL is closely related to UEOPL; more later/tomorrow...
- For now the key point is that the paths are **monotone**
- Hubacek and Yogev had already shown that EOPL $\subseteq$ CLS
- Thus combining these two results:

$$\text{CLS} = \text{EOPL} = \text{PPAD} \cap \text{PLS}$$

- This means that:
  for an **alternative** way to get our **CLS-hardness** results for **2D-KKT**, one can assume monotone paths
- I.e., no need for PLS labyrinths

Take home message: PPAD ∩ PLS

Before:

- **PPAD** and **PLS** both successful classes
- **PPAD ∩ PLS** not believed to have interesting complete problems
- **CLS** introduced as "natural" (presumed distinct) counterpart

Now:

- **PPAD ∩ PLS** is a **natural class with complete problems**
- Captures complexity of problems solved by **gradient descent**
- **PPAD ∩ PLS** = **CLS**
- **Many important problems are now candidates for hardness**

Motivation behind classes

PPAD: all problems that can be solved by path following
(the Lemke-Howson algorithm for Nash equilibria)

PLS: all problems that can be solved by local search

CLS: all problems that can be solved by continuous local search

Motivation behind classes

PPAD: all problems that can be solved by path following
(the Lemke-Howson algorithm for Nash equilibria)

PLS: all problems that can be solved by local search

CLS: all problems that can be solved by continuous local search

GD = CLS: all problems that can be solved by gradient descent

Open Problems

The following are candidates for **PPAD** $\cap$ **PLS**-completeness:

- **POLYNOMIAL-KKT**
- **MIXED-CONGESTION**
- **CONTRACTION**
- **TARSKI**
- **COLORFUL-CARATHEODORY**

Open Problems

The following are candidates for **PPAD ∩ PLS**-completeness:

- ~~**POLYNOMIAL-KKT**~~
- ~~**MIXED-CONGESTION**~~ **[Babichenko, Rubinstein STOC'21]**
- **POLYNOMIAL-KKT for degree < 5**
- **MIXED-NETWORK-CONGESTION**
- **CONTRACTION**
- **TARSKI**
- **COLORFUL-CARATHEODORY**

References

The Complexity of Gradient Descent: CLS = PPAD ∩ PLS by **Fearnley, Goldberg, Hollender, Savani** STOC 2021

Settling the complexity of Nash equilibrium in congestion games by **Babichenko and Rubinstein** STOC 2021

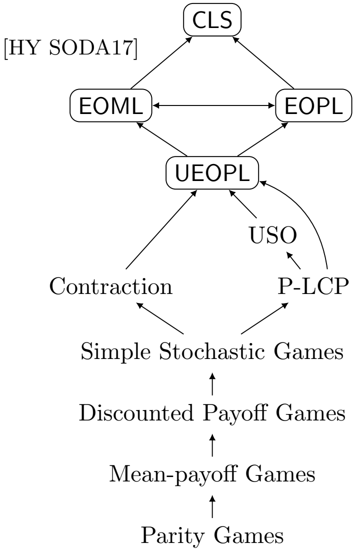
Further Collapses in TFNP by **Göös, Hollender, Jain, Maystre, Pires, Robere, Tao**
CCC 2022 **EOPL = PPAD ∩ PLS**

Hardness of Continuous Local Search by **Hubácek and Yogev**
SICOMP 2020 **EOPL in CLS, query/crypto hardness of (U)EOPL**

Unique End of Potential Line (UEOPL)

Outline

- **P-matrix Linear Complementarity Problem (P-LCP)**
  - Complementary cones view
- **Unique Sink Orientations (USO) of cubes**
  - Reduction from P-LCP to USOs as an **exercise**
- Two-player zero-sum turn-based **discounted games**
  - Optimality equations characterize unique values
  - **Reduction to P-LCP**
  - **Reduction to USO via strategy improvement algorithms**
  - **Reduction to Contraction via strategy iteration**
- **Unique End of Potential Line** (the problem and the class)
  - **Piecewise-linear Contraction in UEOPL**
  - **P-LCP in UEOPL**
  - **Open problems**


CLS
[HY SODA17]
EOML
EOPL
UEOPL
USO
Contraction
P-LCP
Simple Stochastic Games
Discounted Payoff Games
Mean-payoff Games
Parity Games

Linear Complementarity Problem (LCP)

Given: $q \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$ Find: z, $w \in \mathbb{R}^n$ so that

$$z \geq 0 \quad \perp \quad w = q + Mz \geq 0$$

$\perp$ means orthogonal:

$$z^T w = 0$$

$$\Leftrightarrow \quad z_i w_i = 0 \quad \text{all } i = 1, \ldots, n$$

Linear Complementarity Problem (LCP)

Given: $q \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$ Find: z, $w \in \mathbb{R}^n$ so that

$$z \geq \mathbf{0} \quad \perp \quad w = q + Mz \geq \mathbf{0}$$

$\perp$ means orthogonal:

$$z^T w = 0$$

$$\Leftrightarrow \quad z_i w_i = 0 \quad \text{all } i = 1, \ldots, n$$

If $q \geq \mathbf{0}$, the LCP has trivial solution $w = q$, $z = \mathbf{0}$.

LP in inequality form

primal : **max** $c^T x$

subject to $Ax \le b$

$x \ge 0$

dual : **min** $y^T b$

subject to $y^T A \ge c^T$

$y \ge 0$

LP in inequality form

primal : **max** $c^T x$

subject to $Ax \leq b$

$x \geq 0$

dual : **min** $y^T b$

subject to $y^T A \geq c^T$

$y \geq 0$

Weak duality: x, y feasible (fulfilling constraints)

$$\Rightarrow \quad c^T x \leq y^T A x \leq y^T b$$

LP in inequality form

primal :	**max**	$c^T x$
	subject to	$Ax \le b$
		$x \ge 0$

dual :	**min**	$y^T b$
	subject to	$y^T A \ge c^T$
		$y \ge 0$

Weak duality: x, y feasible (fulfilling constraints)

$$\Rightarrow \quad c^T x \le y^T A x \le y^T b$$

Strong duality: primal and dual feasible

$$\Rightarrow \exists \text{ feasible } x, y : \quad c^T x = y^T b \quad (x, y \text{ optimal})$$

LCP generalizes LP

LCP encodes **complementary slackness** of strong duality:

$$\begin{aligned} c^T x = \quad & y^T A x & = y^T b \\ \Leftrightarrow \quad (y^T A - c^T) x = 0, \quad & y^T (b - Ax) & = 0. \end{aligned}$$

$\geq 0 \quad \geq 0 \qquad\qquad \geq 0 \quad \geq 0$

LCP generalizes LP

LCP encodes **complementary slackness** of strong duality:

$$c^T x = y^T A x = y^T b$$

$$\Leftrightarrow \quad (y^T A - c^T) x = 0, \qquad y^T (b - Ax) = 0.$$

$$\geq 0 \quad \geq 0 \qquad \geq 0 \quad \geq 0$$

LP $\Leftrightarrow$ LCP

$$\underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{z} \geq 0 \quad \perp \quad \underbrace{\begin{pmatrix} -c \\ b \end{pmatrix}}_{q} + \underbrace{\begin{pmatrix} 0 & A^T \\ -A & 0 \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{z} \geq 0$$

LCPs and complementary cones

Given: $q \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$ Find: $z \in \mathbb{R}^n$ so that

$$z \geq 0 \perp w = q + Mz \geq 0$$

LCPs and complementary cones

Given: $\boldsymbol{q} \in \mathbb{R}^{\boldsymbol{n}}$, $\boldsymbol{M} \in \mathbb{R}^{\boldsymbol{n \times n}}$ Find: $\boldsymbol{z} \in \mathbb{R}^{\boldsymbol{n}}$ so that

$$\boldsymbol{z} \geq \mathbf{0} \perp \boldsymbol{w} = \boldsymbol{q} + \boldsymbol{M}\boldsymbol{z} \geq \mathbf{0}$$

$$\Leftrightarrow \quad \boldsymbol{z} \geq \mathbf{0} \perp \boldsymbol{w} \geq \mathbf{0} \quad \boxed{\boldsymbol{q} = \boldsymbol{I}\boldsymbol{w} - \boldsymbol{M}\boldsymbol{z}}$$

LCPs and complementary cones

Given: $q \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$ Find: $z \in \mathbb{R}^n$ so that

$$z \geq 0 \perp w = q + Mz \geq 0$$

$$\Leftrightarrow \quad z \geq 0 \perp w \geq 0 \quad \boxed{q = Iw - Mz}$$

$\Leftrightarrow$ q belongs to a **complementary cone:**

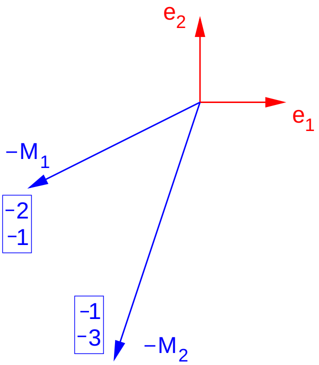
$$\boxed{q \in C(\alpha) = \textit{cone}\,\{-M_i, e_j \mid i \in \alpha, j \notin \alpha\}}$$

for some $\alpha \subseteq \{1, \ldots, n\}$, $M = [M_1 M_2 \cdots M_n]$

$\alpha = \{i \mid z_i > 0\}$

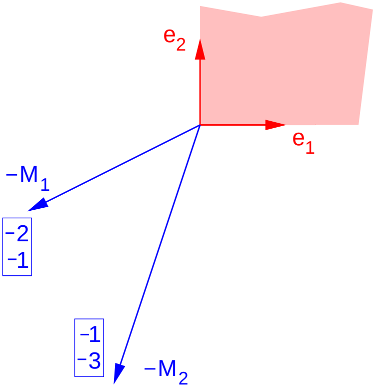
LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



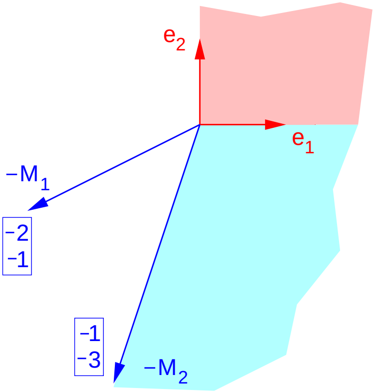
LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



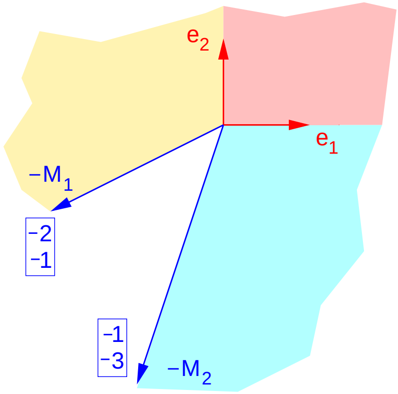
LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



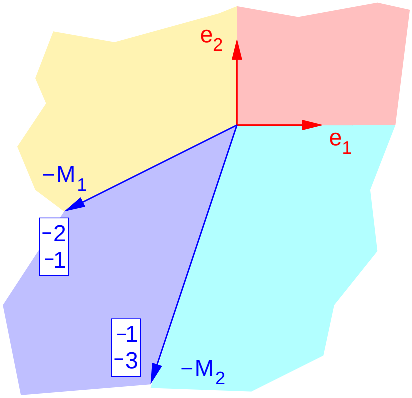
LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



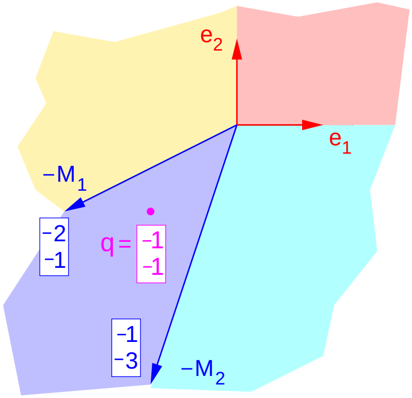
LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



LCPs and complementary cones

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



P-matrices

Def: $M \in \mathbb{R}^{n\times n}$ is a **P-matrix** if **all** its **principal minors** are **positive**.

Thm: M is a **P-matrix** $\Leftrightarrow$ LCP (M, q) has **unique solution** $\forall q \in \mathbb{R}^n$.

P-matrices

Def: $M \in \mathbb{R}^{n \times n}$ is a **P-matrix** if **all** its **principal minors** are **positive**.

Thm: M is a **P-matrix** $\Leftrightarrow$ LCP (M, q) has **unique solution** $\forall q \in \mathbb{R}^n$.

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix} \qquad M' = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix}$$

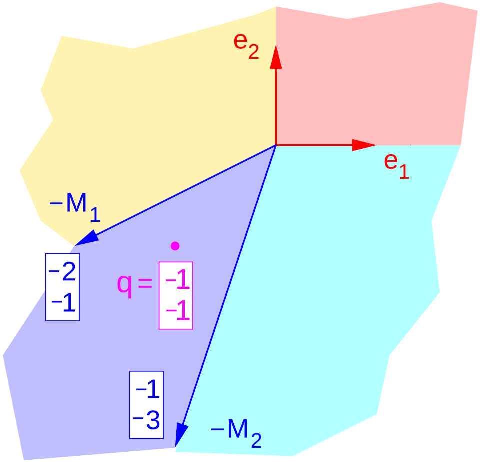
M is a P-matrix, as

$$\begin{aligned} det(M_{11}) &= 2 > 0 \\ det(M_{22}) &= 3 > 0 \\ det(M) &= 5 > 0 \end{aligned}$$

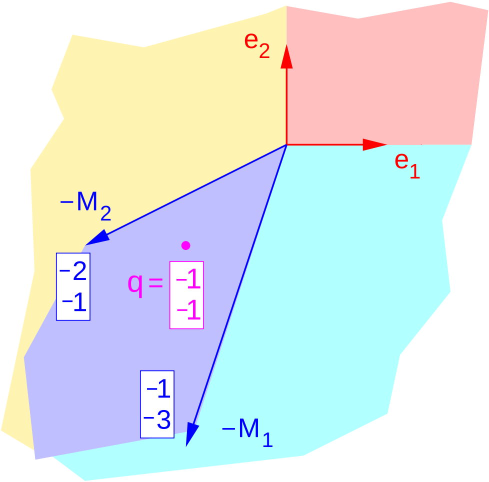
M' is not a P-matrix, as $det(M') = -5 < 0$

Complementary cones: P-matrix

$$M = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$



Multiple solutions



Binary zero-sum discounted games

- Finite directed graph on states $S = \{1, \ldots, n\}$
- Partition $S = S_{Max} \cup S_{Min}$

Binary zero-sum discounted games

- Finite directed graph on states $S = \{1, \ldots, n\}$
- Partition $S = S_{Max} \cup S_{Min}$
- Every state has a left successor $\lambda(s)$ and right successor $\rho(s)$

Binary zero-sum discounted games

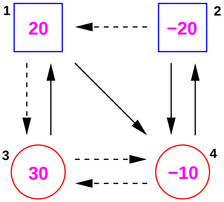
- Finite directed graph on states $S = \{1, \ldots, n\}$
- Partition $S = S_{Max} \cup S_{Min}$
- Every state has a left successor $\lambda(s)$ and right successor $\rho(s)$
- Every state has a reward - $r : S \mapsto \mathbb{Z}$

Binary zero-sum discounted games

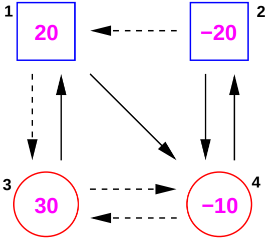
- Finite directed graph on states $S = \{1, \ldots, n\}$
- Partition $S = S_{Max} \cup S_{Min}$
- Every state has a left successor $\lambda(s)$ and right successor $\rho(s)$
- Every state has a reward - $r : S \mapsto \mathbb{Z}$
- Discount factor $\delta \in (0,1)$ (same for both players)

Binary zero-sum discounted games

- Finite directed graph on states $S = \{1, \ldots, n\}$
- Partition $S = S_{Max} \cup S_{Min}$
- Every state has a left successor $\lambda(s)$ and right successor $\rho(s)$
- Every state has a reward - $r : S \mapsto \mathbb{Z}$
- Discount factor $\delta \in (0,1)$ (same for both players)

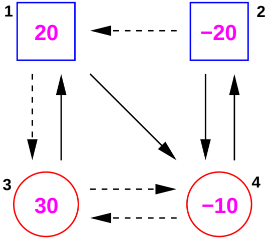


Player objectives



- A play is an infinite path $\pi = s_0, s_1, s_3, \ldots$
  - initial state s_0
  - owner of s_i chooses $s_{i+1} \in \{ \lambda(s_i), \rho(s_i) \}$

Player objectives



- A play is an infinite path $\pi = s_0, s_1, s_3, \ldots$
  - initial state s_0
  - owner of s_i chooses $s_{i+1} \in \{ \lambda(s_i), \rho(s_i) \}$
- ***Max*** maximizes and ***Min*** minimizes

$$\sum_{i=0}^{\infty} \delta^i r(s_i)$$

Optimality equations

- Every state has a **value** $v(s)$ characterized by:

$$\forall s \in S_{Max}: \quad v(s) = \max_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

$$\forall s \in S_{Min}: \quad v(s) = \min_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

Optimality equations

- Every state has a **value** $v(s)$ characterized by:

$$\forall s \in S_{Max}: \quad v(s) = \max_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

$$\forall s \in S_{Min}: \quad v(s) = \min_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

- Proofs:
  - **Banach fixed point theorem** for **contraction** mappings
  - **Strategy improvement** algorithm (constructive)

Optimality equations

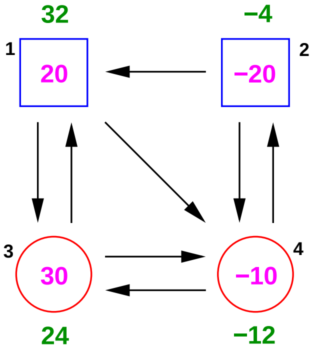
- Every state has a **value** $\boldsymbol{v(s)}$ characterized by:

$$\forall s \in S_{Max}: \quad v(s) = \max_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

$$\forall s \in S_{Min}: \quad v(s) = \min_{t \in \{\lambda(s), \rho(s)\}} (r(s) + \delta v(t))$$

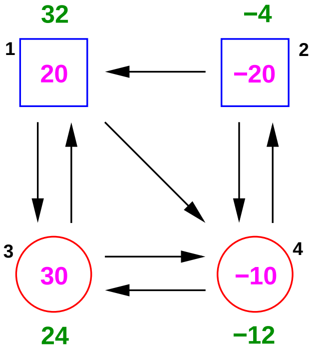
- Proofs:
  - **Banach fixed point theorem** for **contraction** mappings
  - **Strategy improvement** algorithm (constructive)
- Values give *pure* and *positional* optimal strategies:
  Max (***Min***) picks succesor with largest (smallest) value.

Unique values for $\delta = 1/2$



$$v(1) = 32 \qquad = r(1) + \delta \max(v(3), v(4)) = \qquad 20+1/2(24)$$

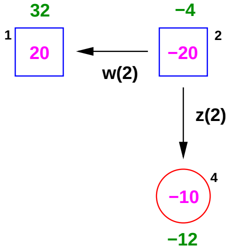
Unique values for $\delta = 1/2$



$$v(1) = 32 \quad = r(1) + \delta \max(v(3), v(4)) = \quad 20 + 1/2(24)$$
$$v(2) = -4 \quad = r(2) + \delta \max(v(1), v(4)) = \quad -20 + 1/2(32)$$
$$v(3) = 24 \quad = r(3) + \delta \min(v(1), v(4)) = \quad 30 + 1/2(-12)$$
$$v(4) = -12 \quad = r(4) + \delta \min(v(2), v(3)) = \quad -10 + 1/2(-4)$$

Nonnegative slacks and complementarity

$$v(2) = r(2) + \delta \max(v(1), v(4))$$



$$v(2) = \quad w(2) + \quad r(2) + \delta v(1)$$
$$v(2) = \quad z(2) + \quad r(2) + \delta v(4)$$

$$w(2), z(2) \geq 0, \quad w(2) \cdot z(2) = 0$$

Reduction to LCP

$$\forall s \in S_{Max} : \quad v(s) = \max_{t \in \{\lambda(s),\, \rho(s)\}} (r(s) + \delta v(t))$$

Replace **max**/**min** with **slacks** and **complementarity condition**

Reduction to LCP

$$\forall s \in S_{Max} : \quad v(s) = \max_{t \,\in\, \{\lambda(s),\, \rho(s)\}} (r(s) + \delta v(t))$$

Replace **max/min** with **slacks** and **complementarity condition**

$$\forall s \in S_{Max} : \quad v(s) = w(s) + r(s) + \delta v(\lambda(s))$$

$$v(s) = z(s) + r(s) + \delta v(\rho(s))$$

$$\forall s \in S : \quad w(s) \geq 0 \perp z(s) \geq 0$$

Reduction to LCP

$$\forall s \in S_{Max}: \quad v(s) = \max_{t \in \{\lambda(s),\, \rho(s)\}} (r(s) + \delta v(t))$$

$$\forall s \in S_{Min}: \quad v(s) = \min_{t \in \{\lambda(s),\, \rho(s)\}} (r(s) + \delta v(t))$$

Replace **max/min** with **slacks** and **complementarity condition**

$$\forall s \in S_{Max}: \quad v(s) = w(s) + r(s) + \delta v(\lambda(s))$$

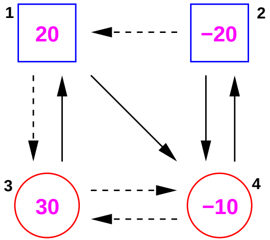
$$v(s) = z(s) + r(s) + \delta v(\rho(s))$$

$$\forall s \in S_{Min}: \quad v(s) = -w(s) + r(s) + \delta v(\lambda(s))$$

$$v(s) = -z(s) + r(s) + \delta v(\rho(s))$$

$$\forall s \in S: \quad w(s) \geq 0 \perp z(s) \geq 0$$

Example



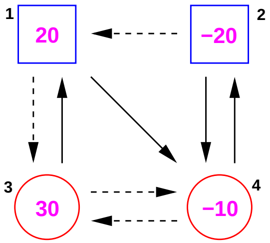
$\forall s \in S:$

$w(v) \geq 0 \perp z(v) \geq 0$

$$\begin{pmatrix} v(1) \\ v(2) \\ -v(3) \\ -v(4) \end{pmatrix} = \begin{pmatrix} w(1) \\ w(2) \\ w(3) \\ w(4) \end{pmatrix} + \begin{pmatrix} r(1) \\ r(2) \\ -r(3) \\ -r(4) \end{pmatrix} + \delta \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} v(1) \\ v(2) \\ v(3) \\ v(4) \end{pmatrix}$$

$$\begin{pmatrix} v(1) \\ v(2) \\ -v(3) \\ -v(4) \end{pmatrix} = \begin{pmatrix} z(1) \\ z(2) \\ z(3) \\ z(4) \end{pmatrix} + \begin{pmatrix} r(1) \\ r(2) \\ -r(3) \\ -r(4) \end{pmatrix} + \delta \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} v(1) \\ v(2) \\ v(3) \\ v(4) \end{pmatrix}$$

Example



$$w \geq 0 \perp z \geq 0$$

$$A := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$Av = w + Ar + \delta A \overbrace{\begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}}^{L} v$$

$$Av = z + Ar + \delta A \underbrace{\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}}_{R} v$$

Eliminate v

$$A(I - \delta L)v = w + Ar$$
$$A(I - \delta R)v = z + Ar$$

Eliminating v we get

$$w + Ar = A(I - \delta L)(A(I - \delta R))^{-1}(z + Ar)$$

$$\boxed{w = Mz + q}$$

$$\boxed{w \geq 0 \perp z \geq 0}$$

$$M = A(I - \delta L)(I - \delta R)^{-1}A, \quad q = (M - I)Ar$$

Example

$$\boxed{w = Mz + q}$$

$$\boxed{w \geq 0 \perp z \geq 0}$$

$$M = A(I - \delta L)(I - \delta R)^{-1} A, \quad q = (M - I)Ar$$

$$A(I - \delta L) = \begin{pmatrix} 1 & 0 & -\delta & 0 \\ -\delta & 1 & 0 & 0 \\ 0 & 0 & -1 & \delta \\ 0 & 0 & \delta & -1 \end{pmatrix} \quad A(I - \delta R) = \begin{pmatrix} 1 & 0 & 0 & -\delta \\ 0 & 1 & 0 & -\delta \\ \delta & 0 & -1 & 0 \\ 0 & \delta & 0 & -1 \end{pmatrix}$$

Levy-Desplanques Theorem

If $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is strictly diagonally dominant, i.e., $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ for all i, then $\boldsymbol{A}$ is non-singular.

Levy-Desplanques Theorem

If $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is strictly diagonally dominant, i.e., $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ for all i, then $\boldsymbol{A}$ is non-singular.

- $A(I - \delta L)$ and $A(I - \delta R)$ are strictly diagonally dominant. E.g.

$$A(I-\delta L) = \begin{pmatrix} 1 & 0 & -\delta & 0 \\ -\delta & 1 & 0 & 0 \\ 0 & 0 & -1 & \delta \\ 0 & 0 & \delta & -1 \end{pmatrix} \quad A(I-\delta R) = \begin{pmatrix} 1 & 0 & 0 & -\delta \\ 0 & 1 & 0 & -\delta \\ \delta & 0 & -1 & 0 \\ 0 & \delta & 0 & -1 \end{pmatrix}$$

- So $M = A(I - \delta L)(I - \delta R)^{-1} A$ is well defined

Theorem (Johnson and Tsatsomeros (1995))

Let $\boldsymbol{M} = \boldsymbol{B}\boldsymbol{C}^{-1}$, where $\boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^{n \times n}$. Then, $\boldsymbol{M}$ is a P-matrix if $\boldsymbol{T}\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{T})\boldsymbol{B}$ is invertible for all $\boldsymbol{T} \in [0, \boldsymbol{I}]$.

Theorem (Johnson and Tsatsomeros (1995))

Let $M = BC^{-1}$, where $B, C \in \mathbb{R}^{n\times n}$. Then, M is a P-matrix if $TC + (I - T)B$ is invertible for all $T \in [0, I]$.

$$w = Mz + q$$

$$w \geq 0 \perp z \geq 0$$

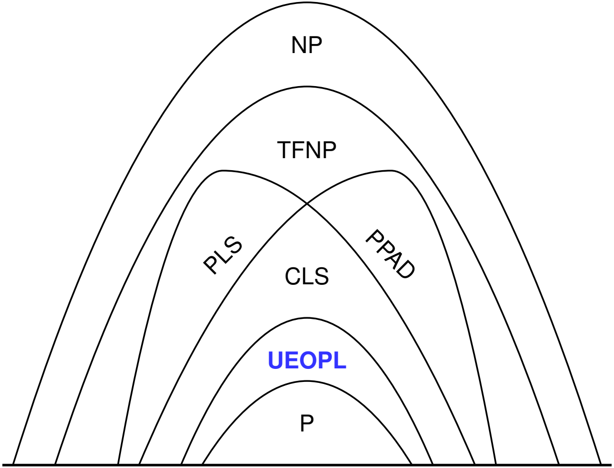
$$M = A(I - \delta L)(I - \delta R)^{-1}A, \quad q = (M - I)Ar$$

$B = A(I - \delta L)$ and $C = A(I - \delta R)$ are strictly diagonally dominant.

Thus, $TC + (I - T)B$ is s.d.d., and hence invertible, for all $T \in [0, I]$.

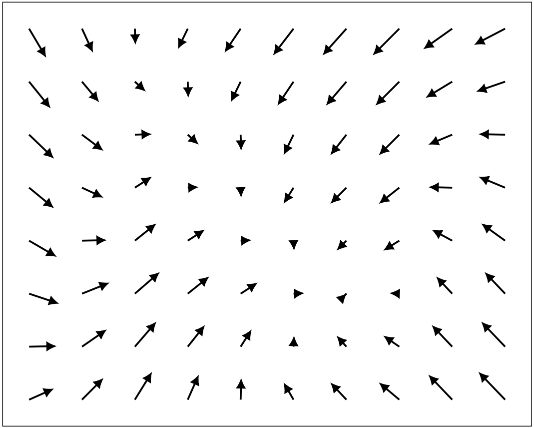
Thus, $M = BC^{-1}$ is a P-matrix.

Unique End of Potential Line (UEOPL)



UEOPL ⊆ EOPL = CLS = PPAD ∩ PLS

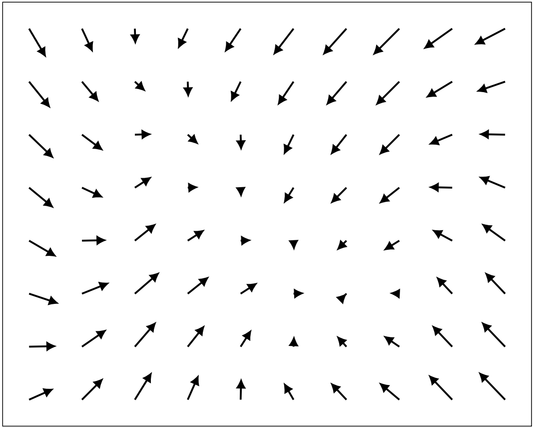
UEOPL 2nd motivation: Contraction Maps



f is **contracting** if

$$||f(x) - f(x')|| \leq c \cdot ||x - x'|| \quad \text{for } c < 1$$

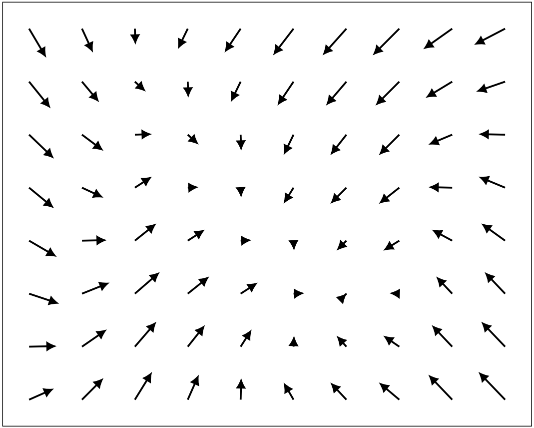
UEOPL 2nd motivation: Contraction Maps



Banach's fixpoint theorem

- Every contraction map has a **unique** fixpoint

UEOPL 2nd motivation: Contraction Maps



Problem: given a contraction map as an arithmetic circuit

- Find a **fixpoint** or a **violation** of contraction

No violations ⇒ the problem has a **unique** solution

The **three problems**

- Contraction (for piecewise-linear circuits)
- Unique sink orientation (definition to come later)
- P-matrix LCP

Each can be formulated so that there are

- **proper solutions**
- **violation solutions**

When there are no **violations** there is a **unique** solution

UEOPL is intended to capture problems like this

Defining (U)EOPL

CLS combines

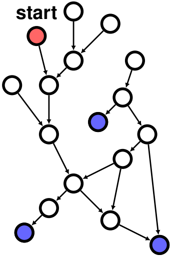
- the **continuous** PPAD-complete problem Brouwer
- the **canonical** PLS-complete problem

EOPL

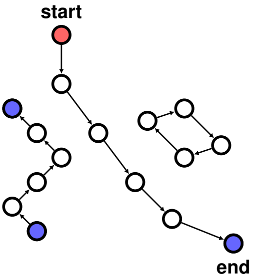
> Why not combine both **canonical** problems?

End Of Potential Line (EOPL)

PLS



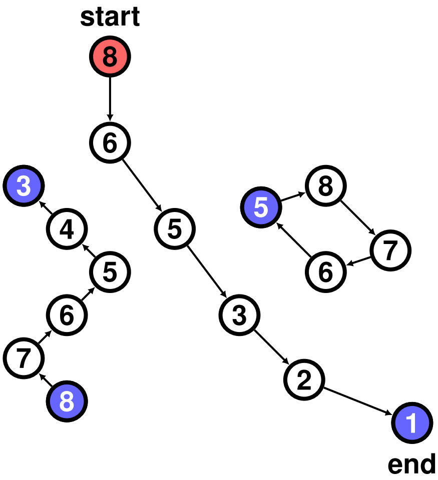
PPAD



Hardness of CLS: Query Complexity and Cryptographic Lower Bounds
Hubáček and Yogev [SODA 2017]

CLS: New Problems and Completeness (arXiv)
[Fearnley, Gordon, Mehta, S. 2017–]

End of Potential Line (EOPL)



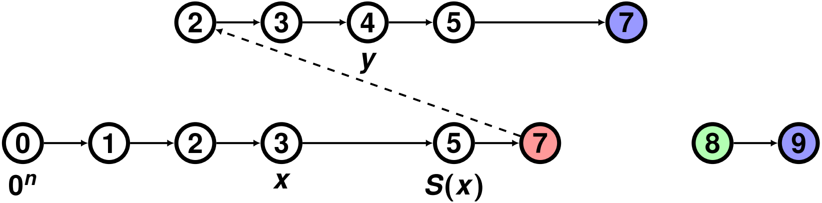
Combines the two **canonical** complete problems

- An End-of-the-Line instance
- That has a potential

Find

- The end of a line
- A vertex where the potential increases

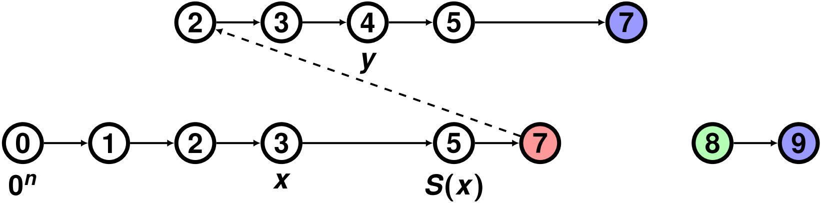
Unique End of Potential Line (UEOPL)



- **Proper solution**: The end of a line
- **Violation 1**: The start of a line other than $\mathbf{0}^n$
- **Violation 2**: An edge that increases the potential
- **Violation 3**: Any pair of vertices v and u satisfying

$$V(x) < V(y) < V(S(x))$$

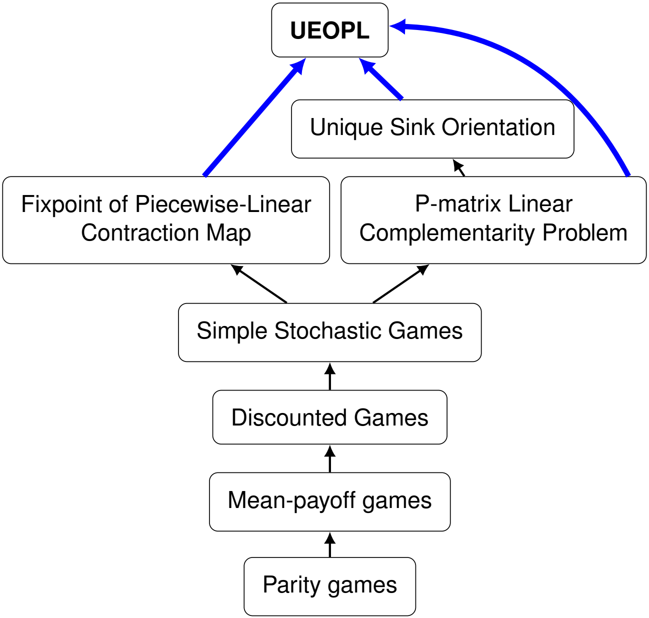
Unique End of Potential Line (UEOPL)



If there are no violations then there is a **unique** line

- That starts at at $\mathbf{0}^n$
- And ends at the **unique** proper solution to the problem

Main results



Main results

One Permutation Discrete Contraction is **UEOPL-complete**

- A technical tool used in our reductions
- USO reduces to OPDC
- Contraction reduces to OPDC
- OPDC is “close“ to both problems

OPDC is **not very natural**...

Piecewise-Linear Contraction

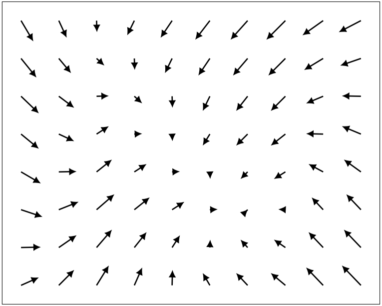
Input

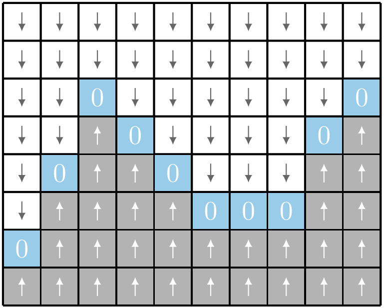
- contraction map $\boldsymbol{f}$ given as an *arithmetic circuit*
- gates: $\mathbf{max}, \mathbf{min}, +, -$, and $\times\zeta$ (multiplication by a constant)
- a LinearFIXP circuit defines a *piecewise linear* function
- we seek a fixpoint, i.e., $\boldsymbol{x}^*$ such that $\boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{x}$
- $\boldsymbol{x}^*$ is unique and has **polynomial bit complexity**

Find

- A **fixpoint** of $\boldsymbol{f}$ (which will be unique if $\boldsymbol{f}$ is contracting)
- A **violation** that shows $\boldsymbol{f}$ is not contracting

PL-Contraction to UEOPL

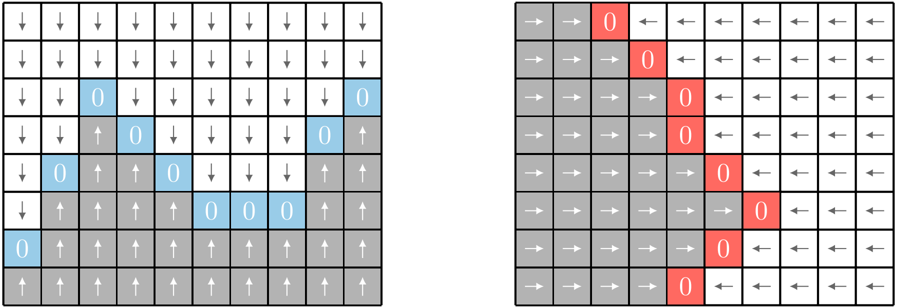




First we discretize the problem

- Lay a grid of points over the space
- For each dimension construct a **direction** function

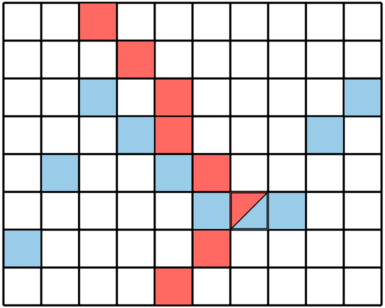
PL-Contraction to UEOPL



Discrete contraction

- Find a point that is **0** in all dimensions

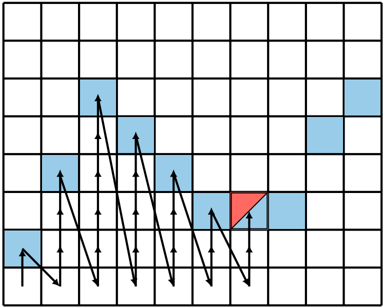
PL-Contraction to UEOPL



A point is on the **surface** if it is **0** for some direction

- Every vertical slice has a unique point on the blue surface
- At each of these, we can follow the red direction function

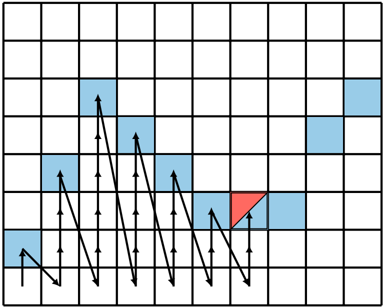
PL-Contraction to UEOPL



The path

1. Start at $(0, 0)$
2. Find the blue surface
3. If not at red surface, move across one, return to bottom, go to 2

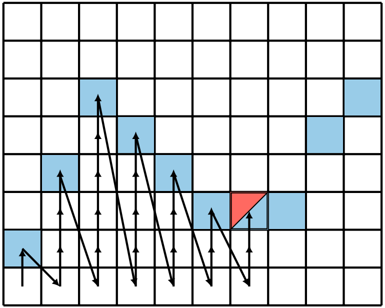
PL-Contraction to UEOPL



The potential

- The path never moves left
- In every slice, it either moves moves up or down

PL-Contraction to UEOPL

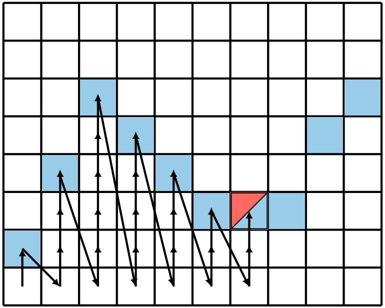


So we can use a pair **(*a, b*)** ordered lexicographically where

- ***a*** is the ***x*** coordinate of the vertex
- ***b*** is
  - ***y*** if we are moving up
  - ***−y*** if we are moving down

This monotonically increases along the line

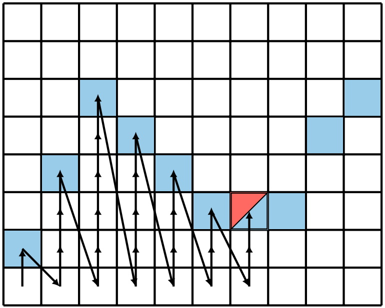
PL-Contraction to UEOPL



Actually, this formulation only gives us a **forward** circuit

- But the line is unique
- So we can apply a technique of Hubáček and Yogev (2017) to make the line reversible

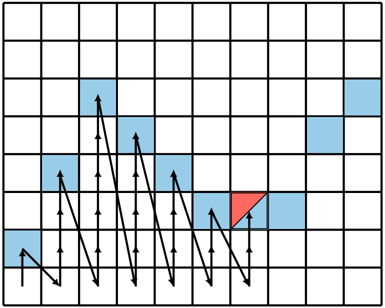
PL-Contraction to UEOPL



This generalises to arbitrary dimension

- We walked along the blue surface to reach the red surface

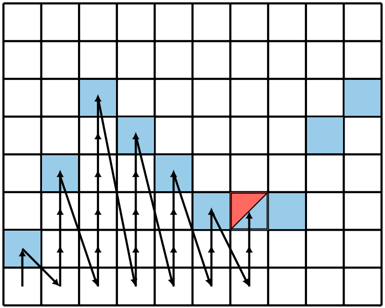
PL-Contraction to UEOPL



In 3D

- Walk along the red/blue surface to find the green surface
- Between any two points on the red/blue surface
    - Walk along the blue surface to find the red surface

PL-Contraction to UEOPL



Theorem

Contraction is in UEOPL

Consequences for contraction

Theorem

Given an arithmetic circuit C encoding a contraction map

$$f : [0,1]^d \to [0,1]^d$$

with respect to any ℓ_p norm

there is an algorithm, based on a **nested binary search**

that finds a fixpoint of f in time

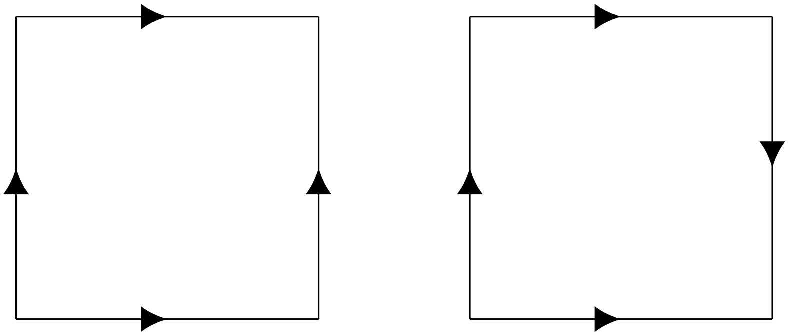
- polynomial in size(C)
- exponential in d

Before, **such algorithms were only known for ℓ_2 and ℓ_∞**

Unique Sink Orientations of Cubes

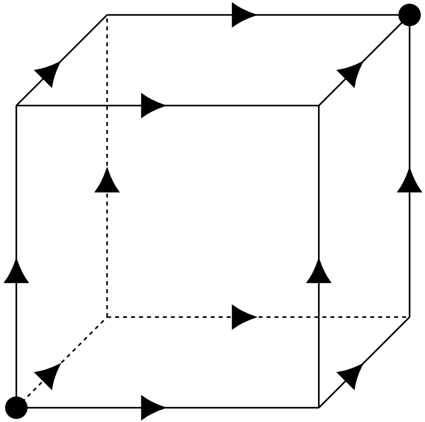
Orient the edges of an ***n***-dimensional cube

- So that every face has a **unique** sink



Unique Sink Orientations of Cubes

A 3-dimensional USO



Unique Sink Orientations of Cubes

Can be **cyclic** (EXERCISE)

UniqueSinkOrientation

Given a **polynomial-time boolean circuit**

$$C : \{0, 1\}^n \mapsto \{0, 1\}^n$$

that maps a vertex v of then n-cube to the orientation at v:

- **find the sink of the cube**
- or a violation to the USO property

Why is USO interesting?

Long line of work on UniqueSinkOrientation:

P-matrix LCP reduces to UniqueSinkOrientation

[Stickney and Watson ’78]

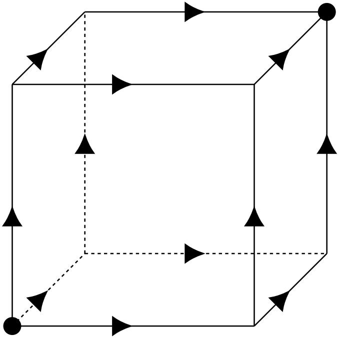
Non-trivial USO algorithms (previously best for P-matrix LCP)

[Szabó and Welzl ’01]

Some problems reduce to **acyclic** USO

- parity games
- mean-payoff games
- discounted games
- simple-stochastic games

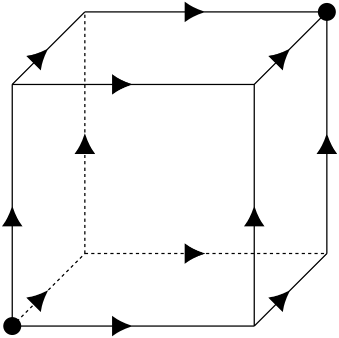
USO in UEOPL



Previously

- USO was known to be in TFNP
- But not PPAD or PLS

USO in UEOPL

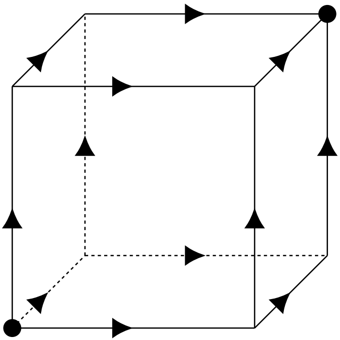


Theorem

USO is in UEOPL

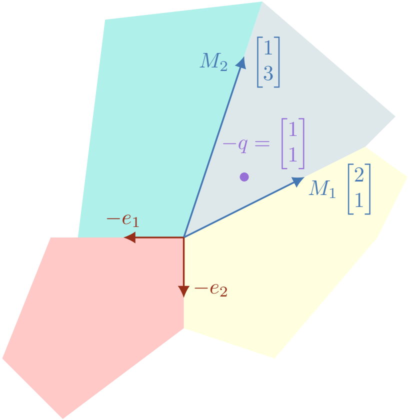
(USO is a “width 2” instance of discrete contraction)

USO in UEOPL



So we put USO in UEOPL, CLS, PPAD, and PLS

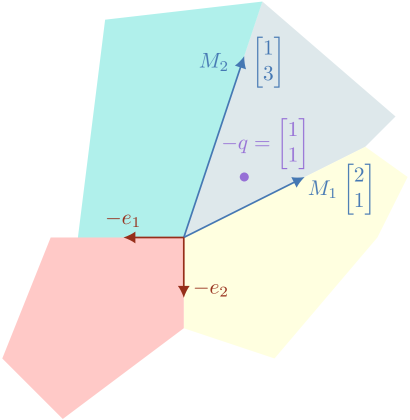
P-matrix Linear Complementarity Problem



Input:

- Vectors $\boldsymbol{M_1}, \boldsymbol{M_2}, \ldots, \boldsymbol{M_d}$
- A vector $\boldsymbol{q}$

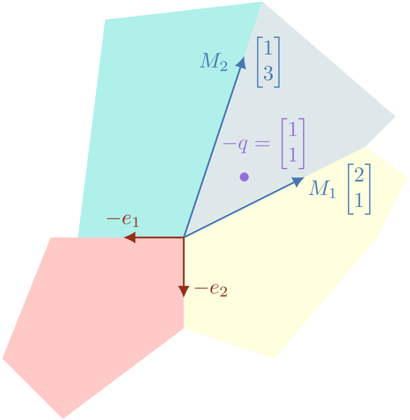
P-matrix Linear Complementarity Problem



A **complementary cone** is all non-negative linear combinations of

- A subset of $M_1, M_2, \ldots, M_d$, with
- $-e_i$ in place of each vector not chosen

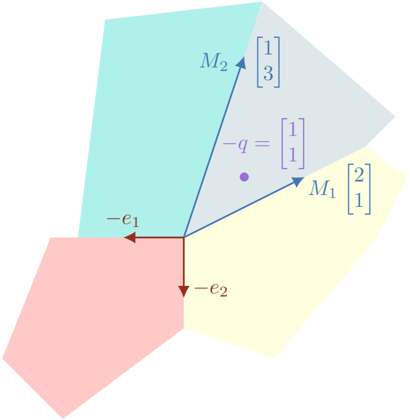
P-matrix Linear Complementarity Problem



The **linear complementarity problem** (LCP)

- Find a cone that contains $\boldsymbol{q}$

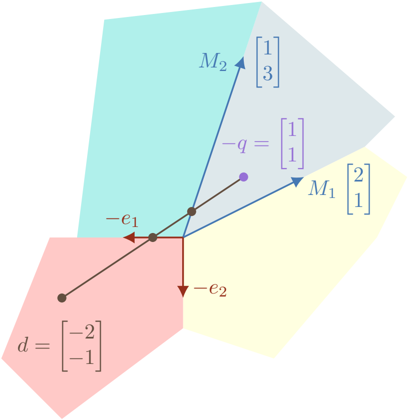
P-matrix Linear Complementarity Problem



P-matrix LCPs

- The cones are guaranteed to exactly partition the space

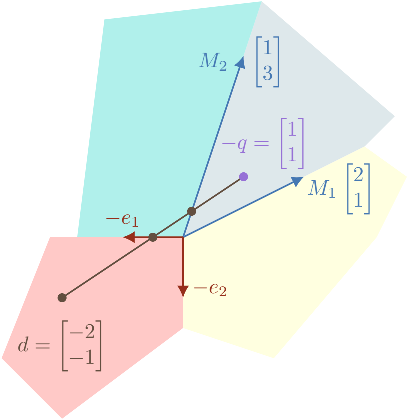
P-matrix Linear Complementarity Problem



We reduce P-matrix LCP to UEOPL using **Lemke's algorithm**

- Start at the vector $\boldsymbol{d}$ in the cone $-\boldsymbol{e}_1, -\boldsymbol{e}_2$
- Walk through the sequence of cones from $\boldsymbol{d}$ to $\boldsymbol{q}$

P-matrix Linear Complementarity Problem



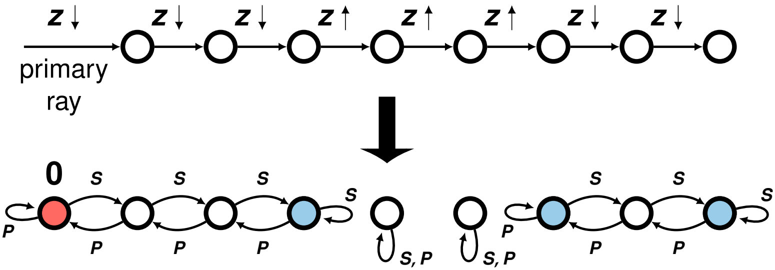
The progress along the path gives us a **potential**

- The algorithm has a variable ***z***
- ***z*** corresponds to distance along the path
- it monotonically decreases

P-matrix LCP → UEOPL

If the input is not a P-matrix, then ***z*** may **increase**

- We deal with this by introducing **new solutions**



P-matrix LCP → UEOPL

Theorem
P-matrix LCP is in UEOPL

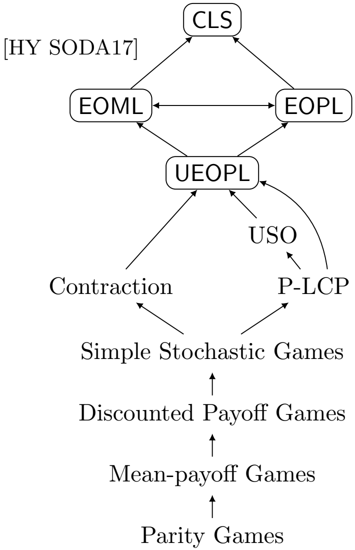
Consequences for P-matrix LCP

Blowup of reduction to UEOPL is only **linear**

This allows us to apply an algorithm of **Aldous (1983)**

Gives **fastest-known (randomized) algorithm** for P-matrix LCP, with running time

$$2^{\frac{n}{2}} \cdot \mathrm{poly}(n)$$


CLS
[HY SODA17]
EOML
EOPL
UEOPL
USO
Contraction
P-LCP
Simple Stochastic Games
Discounted Payoff Games
Mean-payoff Games
Parity Games

Conjectures

USO is complete for UEOPL

Contraction is complete for UEOPL

PLCP is complete for UEOPL

Conjectures

USO is complete for UEOPL

Contraction is complete for UEOPL

PLCP is complete for UEOPL

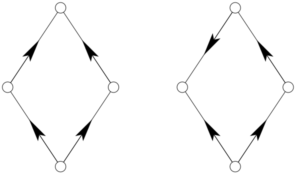
EOPL = CLS $\neq$ UEOPL

Unique sink orientations of cubes

[Stickney and Watson (1978)][Szabó and Welzl (2001)]

- n-dimensional hypercube
- edges oriented such that **every face** has a **unique sink**
- thus unique global sink

The two USOs for $n = 2$:



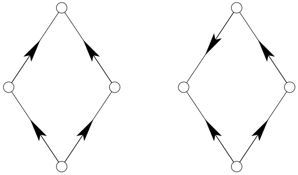
Fact: Every one of 2^d outmaps occurs at some vertex

Unique sink orientations of cubes

[Stickney and Watson (1978)][Szabó and Welzl (2001)]

- n-dimensional hypercube
- edges oriented such that **every face** has a **unique sink**
- thus unique global sink

The two USOs for $n = 2$:



Fact: Every one of 2^d outmaps occurs at some vertex

In particular, there's also a single source on each face too

EXERCISES

Reduce the promise version of the P-matrix LCP problem to the USO problem.

Construct a USO in 3 dimensions that contains a cyclic.
Hints:

1. Recall that the cycle cannot exist within a 2 face
2. Recall that the USO must have an overall source and an overall sink

ANSWER 1: USO for P-matrix LCP

LCP: $z \geq 0 \perp w \geq 0$, $\quad q = Iw - Mz$

For every $\alpha \subseteq \{1, \ldots, n\}$, define $B^\alpha \in \mathbb{R}^{n \times n}$ by

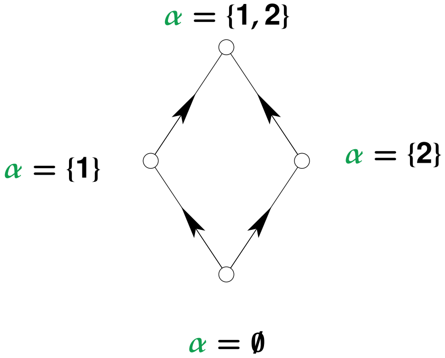
$$(B^\alpha)_i = \begin{cases} -M_i, & i \in \alpha \\ e_i, & i \notin \alpha \end{cases}$$

Orient edges at vertex α oriented according to

$$sign\left((B^\alpha)^{-1} q \right)$$

ANSWER 1: PLCP USO example

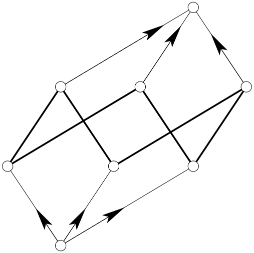
$$-1/5\begin{pmatrix} 3 & -1 \\ -1 & 2 \end{pmatrix} z' + I w' = q' = \begin{pmatrix} 2/5 \\ 1/5 \end{pmatrix} \geq 0$$



$$Iw - Mz = Iw - \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix} z = q = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$

Cyclic USO

Antipodal sink and source; remaining form cycle (two directions possible)



Note:

- this cyclic USOs arises from a P-matrix LCP
- **subexponential** algorithms ($2^{O(\sqrt{(n)})}$) known, but rely on **acyclicity**
- none known for P-LCP, major open problem

P-LCP in UEOPL two ways

- We presented a **direct reduction** from P-LCP to UEOPL possible via **Lemke's algorithm**
- P-LCP can be reduced to USO by a rather straightforward reduction (exercise)
- This gives an alternative (but less "efficient") proof of membership in UEOPL for P-LCP

References

Unique end of potential line by **Fearnley, Gordon, Mehta, Savani**
ICALP 2019 / JCSS 2020 **Definition of UEOPL and containment results**

Hardness of Continuous Local Search by **Hubácek and Yogev**
SODA 2017 / SICOMP 2020 **EOPL in CLS, query/crypto hardness of (U)EOPL**

Further Collapses in TFNP by
Göös, Hollender, Jain, Maystre, Pires, Robere, Tao
CCC 2022 **EOPL = PPAD ∩ PLS**

Thanks!