# 6.890: Fun with Hardness Proofs

## Guest Lectures on PPAD-Part B

## November 2014

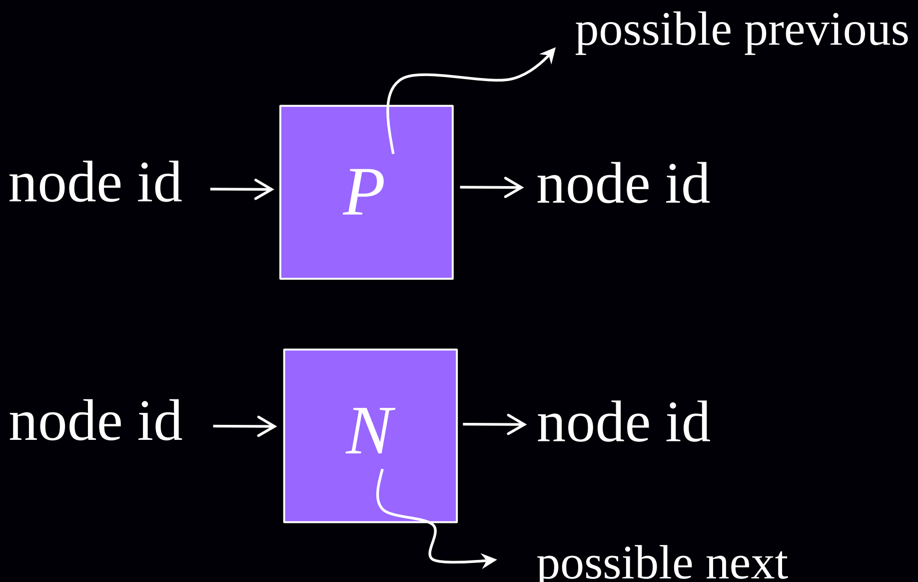
**Constantinos Daskalakis**
CSAIL, MIT

## Last time

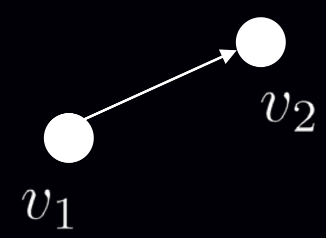
- Existence Theorems: Nash, Brouwer, Sperner
- Total Search Problems in NP
- Totality = Parity Argument on Directed Graphs
- Definition of PPAD
- ArithmCircuitSat

# The PPAD Class [Papadimitriou '94]

A graph with vertex set $\{0,1\}^n$ can be defined by two circuits:



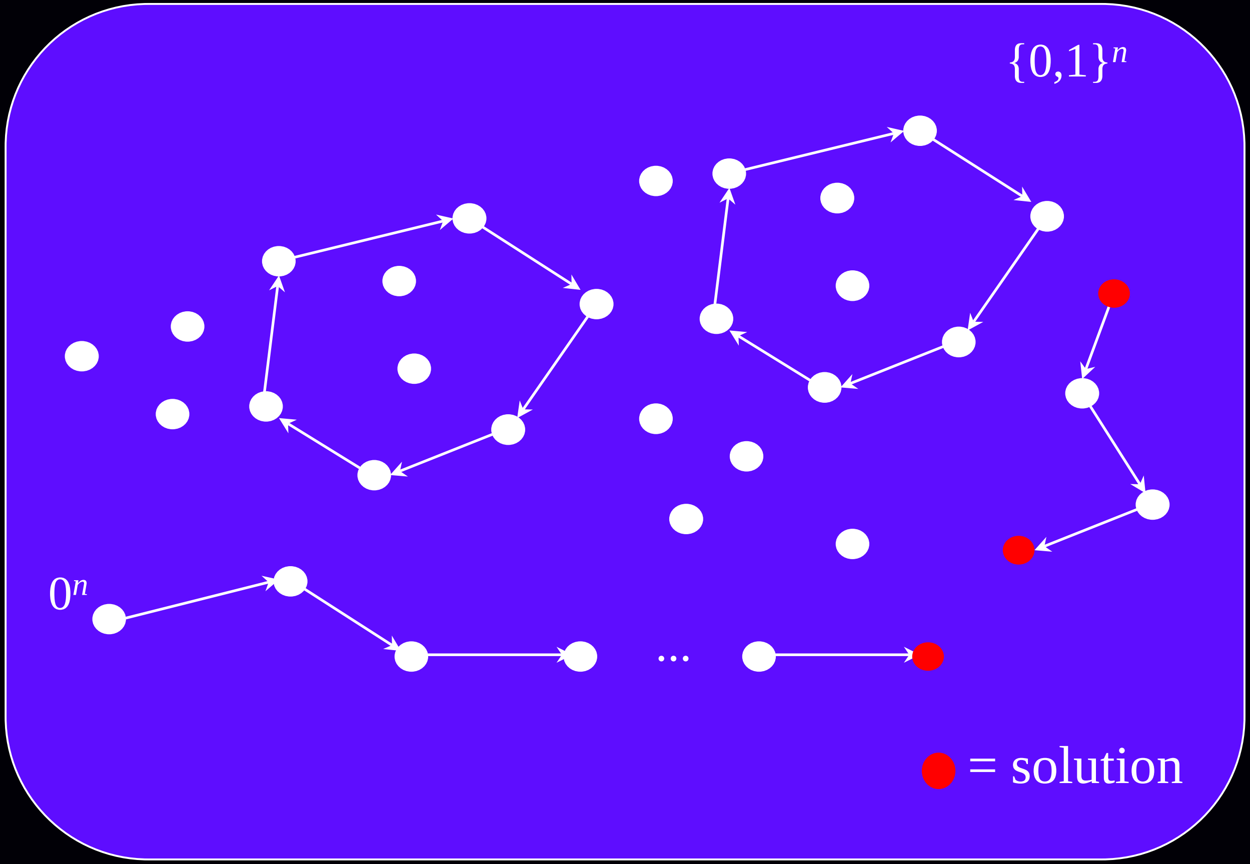

$$P(v_2) = v_1 \wedge N(v_1) = v_2$$

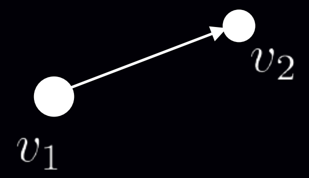



**END OF THE LINE**: Given $P$ and $N$: If $0^n$ is an unbalanced node, find another unbalanced node. Otherwise output $0^n$.

**PPAD =** { *Search problems in* FNP *reducible to* END OF THE LINE }

# END OF THE LINE



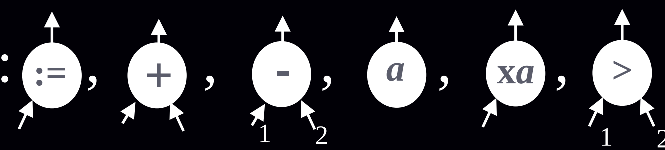




$\Leftrightarrow \quad P(v_2) = v_1 \wedge N(v_1) = v_2$

# ARITHMCIRCUITSAT

**[Daskalakis, Goldberg, Papadimitriou’06]**

INPUT: 1. A circuit comprising:

- variable nodes $v_1, \ldots, v_n$
- gate nodes $g_1, \ldots, g_m$ of types: 

- directed edges connecting variables to gates and gates to variables (loops are allowed);
- variable nodes have in-degree 1; gates have 0, 1, or 2 inputs depending on type as above; gates & nodes have arbitrary fan-out

2. $\varepsilon \geq 0$

OUTPUT: An assignment of values $v_1, \ldots, v_n \in [0,1]$ satisfying:

:= $\quad y == x_1 \pm \epsilon$

\+ $\quad y == \min\{1, x_1 + x_2\} \pm \epsilon$

\- $\quad y == \max\{0, x_1 - x_2\} \pm \epsilon$

a $\quad y == \max\{0, \min\{1, a\}\} \pm \epsilon$

xa $\quad y == \max\{0, \min\{1, a \cdot x_1\}\} \pm \epsilon$

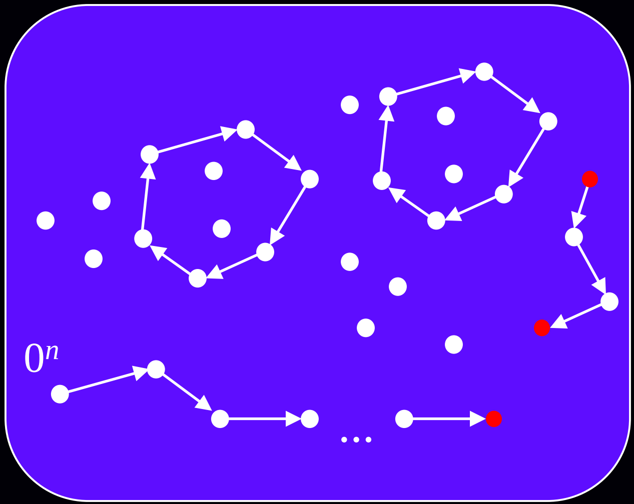
> $\quad y == \begin{cases} 1, & \text{if } x_1 > x_2 + \epsilon \\ 0, & \text{if } x_1 < x_2 - \epsilon \\ *, & \text{if } x_1 = x_2 \pm \epsilon \end{cases}$

[DGP’06]: Always exists satisfying assignment!

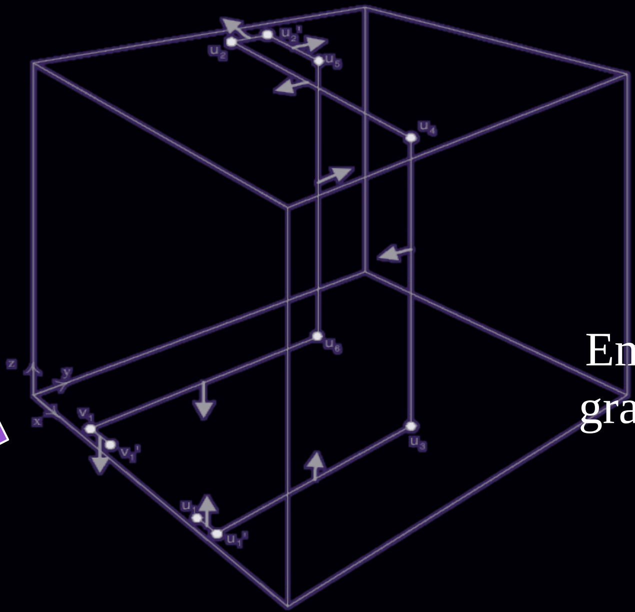
[DGP’06]: but is PPAD-complete to find

# PPAD-Completeness of NASH

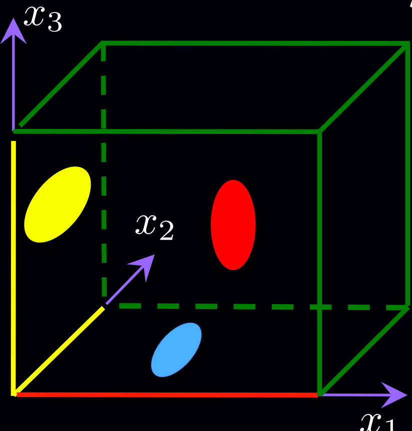
**[Daskalakis, Goldberg, Papadimitriou'06]**



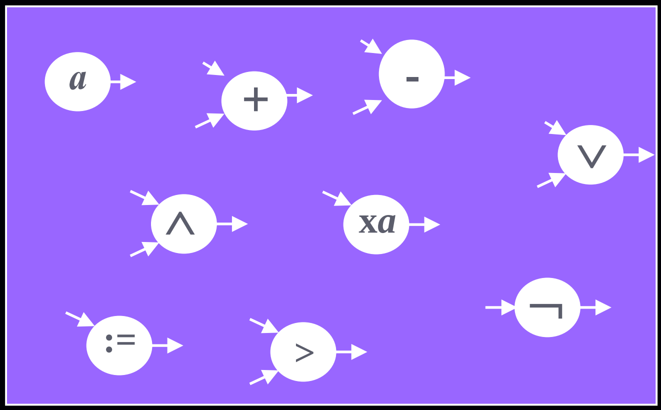

Generic PPAD



Embed PPAD graph in $[0,1]^3$




3D-SPERNER




ARITHMCIRCUITSAT

NASH

## Last time

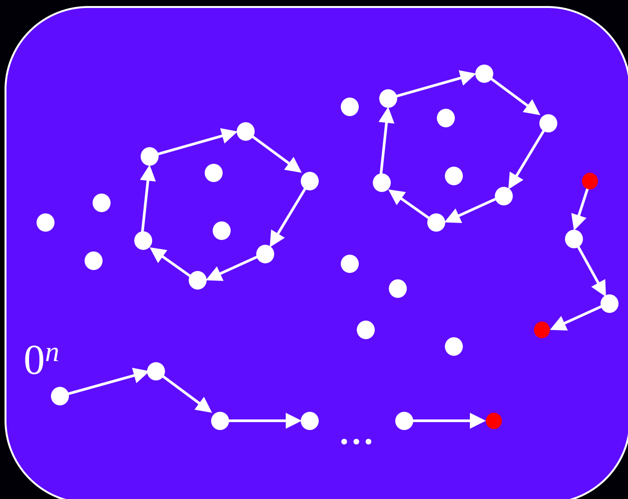
- Existence Theorems: Nash, Brouwer, Sperner
- Total Search Problems in NP
- Totality = Parity Argument on Directed Graphs
- Definition of PPAD
- ARITHMCIRCUITSAT

## This time

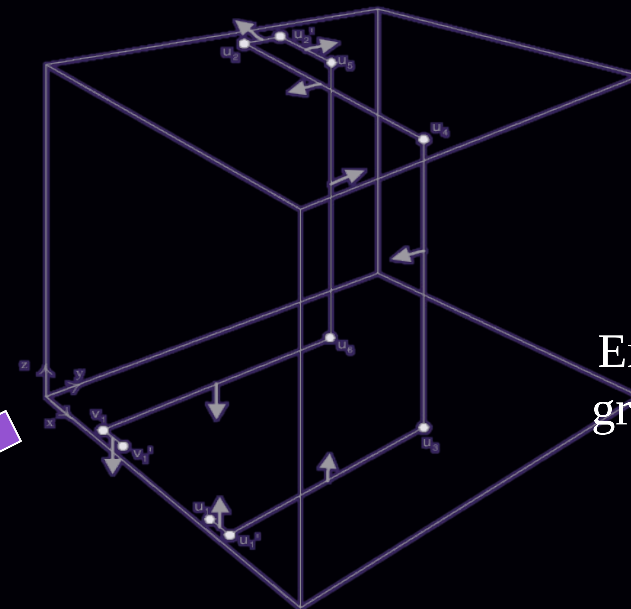
- PPAD-completeness of:
  - Nash Equilibrium
  - Preference Games, Stable Hypergraph Matching
- Other existence arguments: PPA, PPP, PLS

# PPAD-Completeness of NASH

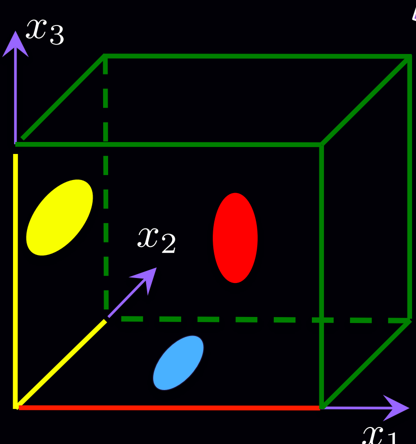
**[Daskalakis, Goldberg, Papadimitriou'06]**



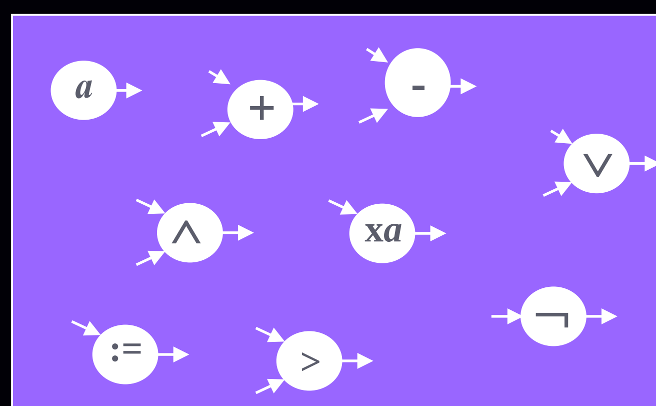

Generic PPAD



Embed PPAD graph in $[0,1]^3$




3D-SPERNER




ArithmCircuitSAT

NASH

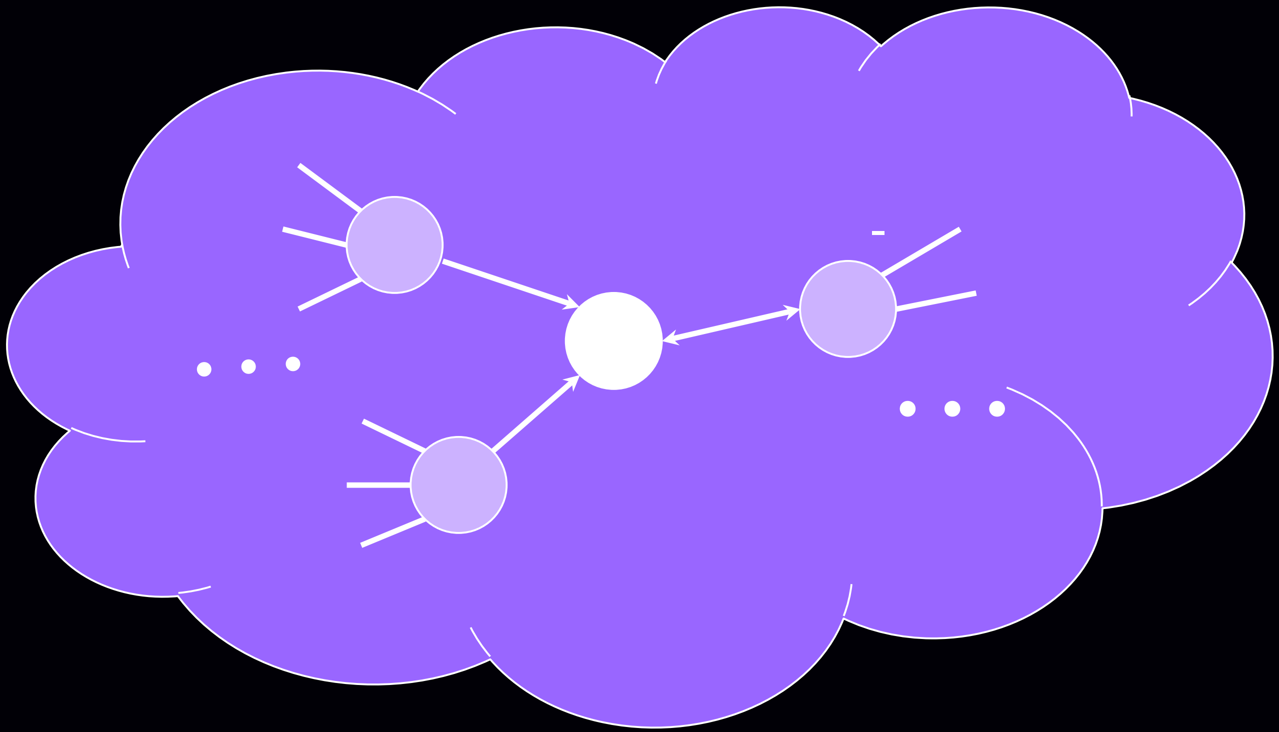
Menu

Graphical and Polymatrix Games

# Graphical Games

**Graphical Games** [Kearns-Littman-Singh'01]

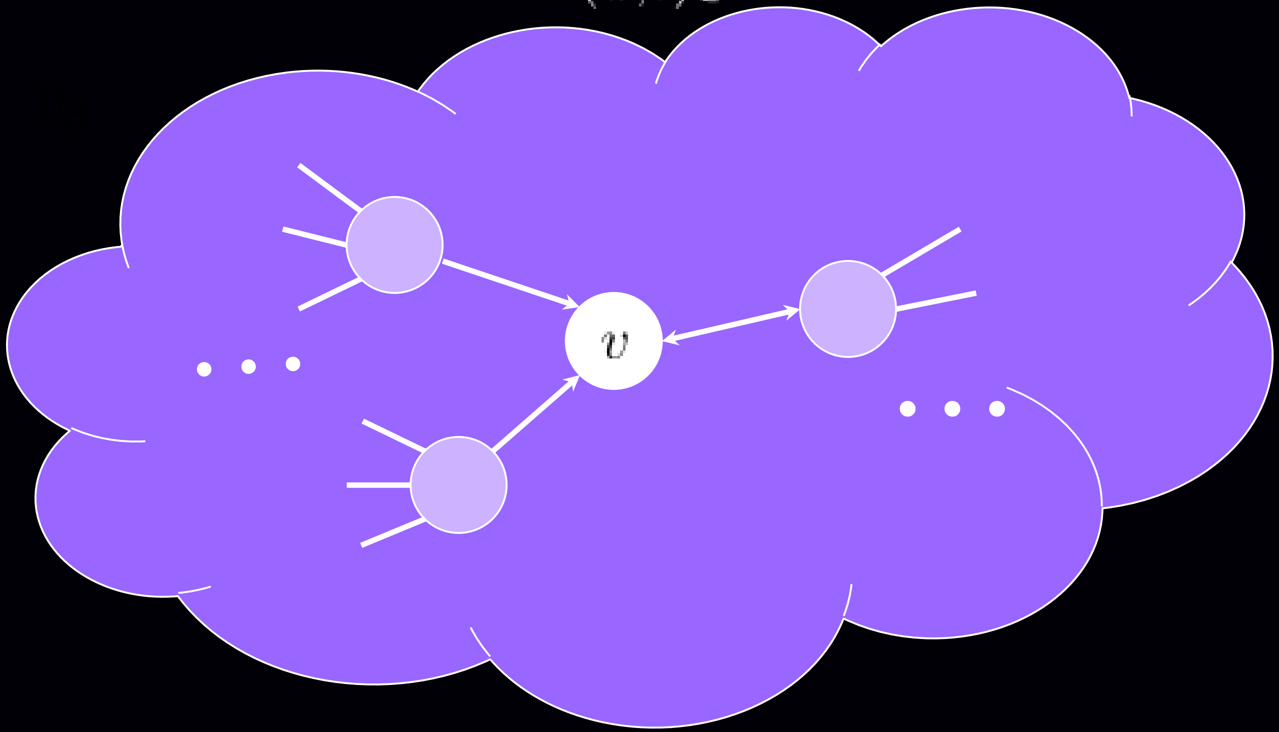
- Defined to capture sparse player interactions, such as those arising under geographical, communication or other constraints.
- Players are nodes in a directed graph.
- Player's payoff only depends on own strategy and the strategies of her in-neighbors in the graph (nodes pointing to her)



# Polymatrix Games

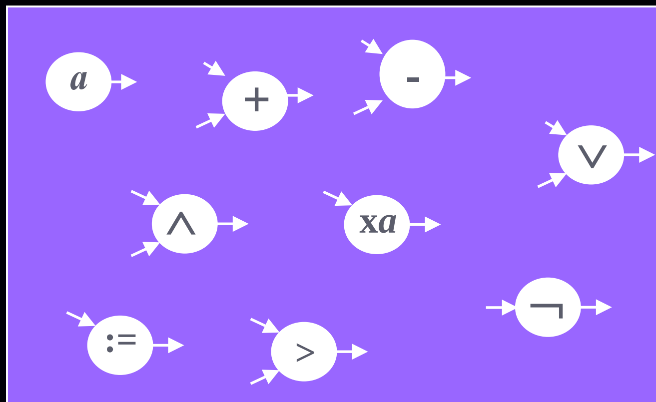
**Polymatrix Games** [Janovskaya'68]: Graphical games with edge-wise separable utility functions.

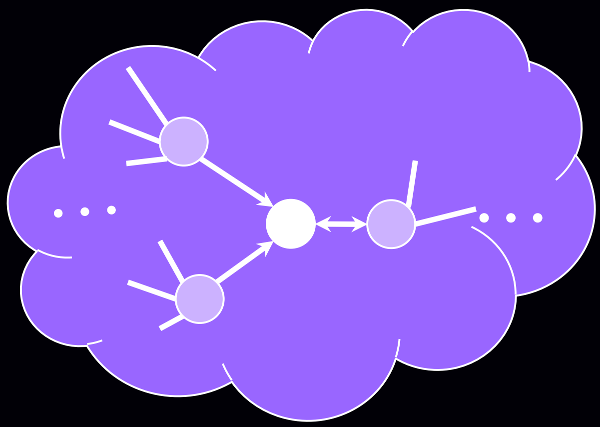
$$u_v(x_1, \ldots, x_n) = \sum_{(w,v)\in E} u_{w,v}(x_w, x_v)$$
$$= \sum_{(w,v)\in E} x_v^{\mathrm{T}} A^{(v,w)} x_w$$



# PPAD-Completeness of NASH

**[Daskalakis, Goldberg, Papadimitriou'06]**



ARITHMCIRCUITSAT

POLYMATRIXNASH

NASH

# Menu

- Graphical and Polymatrix Games
- PPAD-completeness of PolymatrixNash

***Game Gadgets:*** Polymatrix games performing real arithmetic at their Nash equilibrium.

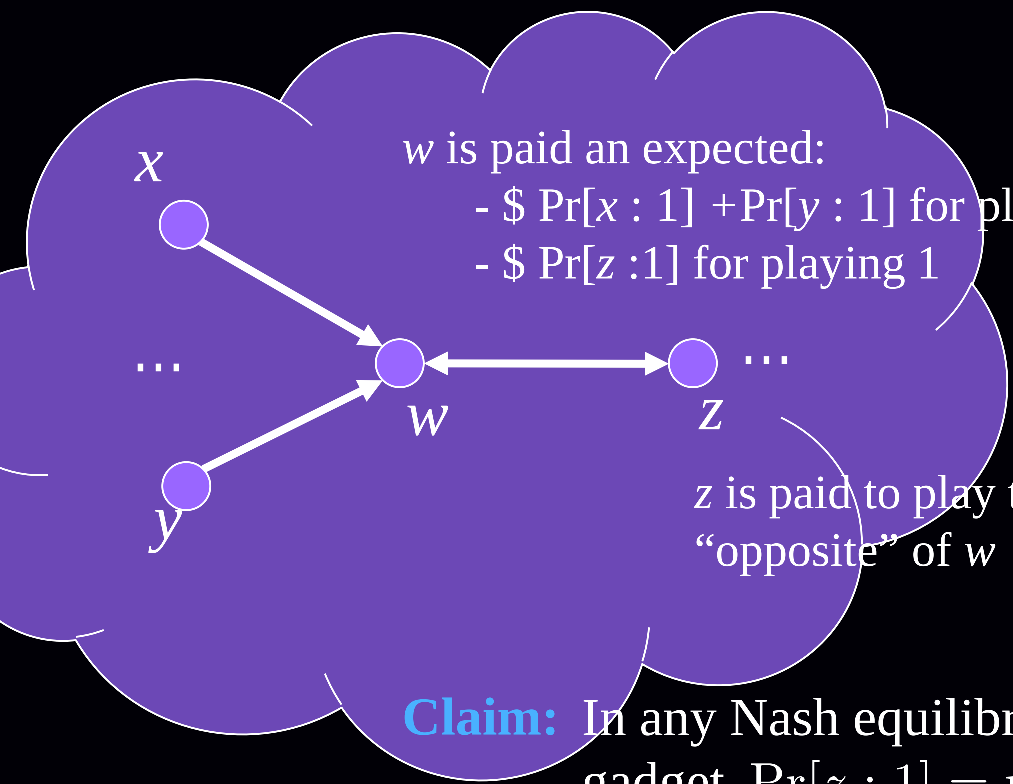
# Addition Gadget

Suppose two strategies per player: {0,1}

then mixed strategy ≡ a number in [0,1] (the probability of playing 1)

e.g. ***addition game***

$$u(w:0) = \Pr[x:1] + \Pr[y:1]$$
$$u(w:1) = \Pr[z:1]$$




*w* is paid an expected:
- \$ Pr[*x* : 1] +Pr[*y* : 1] for playing 0
- \$ Pr[z :1] for playing 1

*z* is paid to play the "opposite" of *w*

$$u(z:0) = 0.5$$
$$u(z:1) = 1 - Pr[w:1]$$

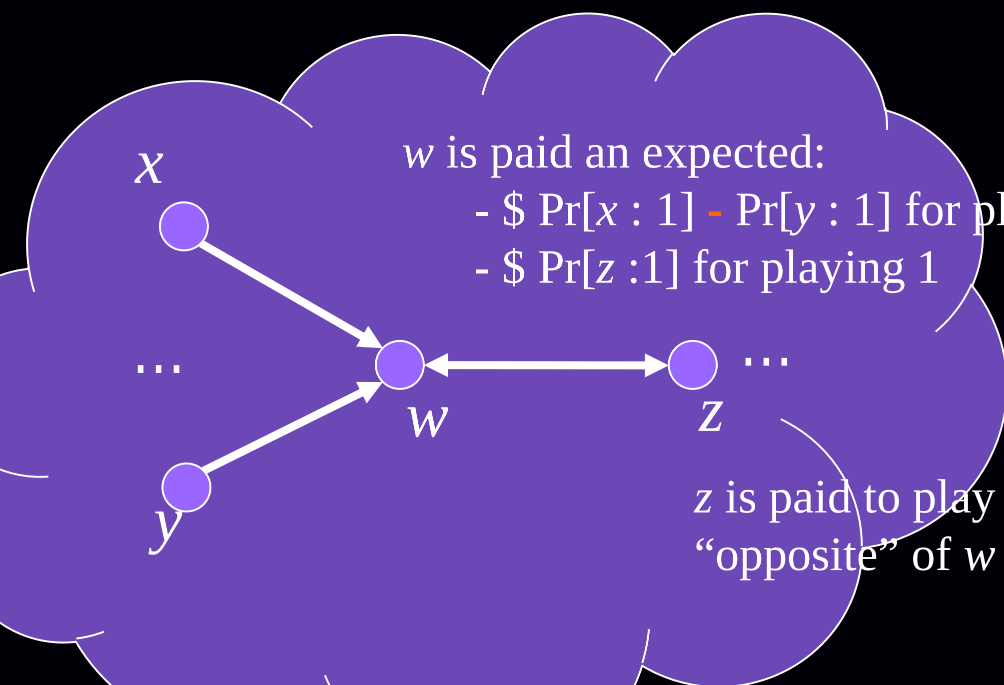
**Claim:** In any Nash equilibrium of a game containing the above gadget $\Pr[z:1] = \min\{\Pr[x:1] + \Pr[y:1], 1\}$.

# Subtraction Gadget

Suppose two strategies per player: {0,1}

then mixed strategy ≡ a number in [0,1] (the probability of playing 1)

e.g. ***subtraction***




*w* is paid an expected:

- \$ Pr[*x* : 1] - Pr[*y* : 1] for playing 0
- \$ Pr[z :1] for playing 1

$$u(w:0) = \Pr[x:1] - \Pr[y:1]$$
$$u(w:1) = \Pr[z:1]$$

*z* is paid to play the "opposite" of *w*

$$u(z:0) = 0.5$$
$$u(z:1) = 1 - Pr[w:1]$$

**Claim:** In any Nash equilibrium of a game containing the above gadget $\Pr[z:1] = \max\{0, \Pr[x:1] - \Pr[y:1]\}$.

**Notational convention:** Use the name of the player and the probability of that player playing 1 interchangeably.

$$x \quad \leftrightarrow \quad \Pr[x : 1]$$

# List of Game Gadgets

copy : $z = x$

addition : $z = \min\{1, x + y\}$

subtraction : $z = \max\{0, x - y\}$

set equal to a constant : $z = \max\{0, \min\{1, \alpha\}\}$

multiply by constant : $z = \max\{0, \min\{1, \alpha \cdot x\}\}$

comparison :
$$z = \begin{cases} 1, & \text{if } x > y \\ 0, & \text{if } x < y \\ *, & \text{if } x = y \end{cases}$$

If any of these gadgets is contained in a bigger polymatrix game, these conditions hold at ***any*** Nash eq. of that bigger game.

Bigger game can only have edges into the "input players" and out of the "output players."

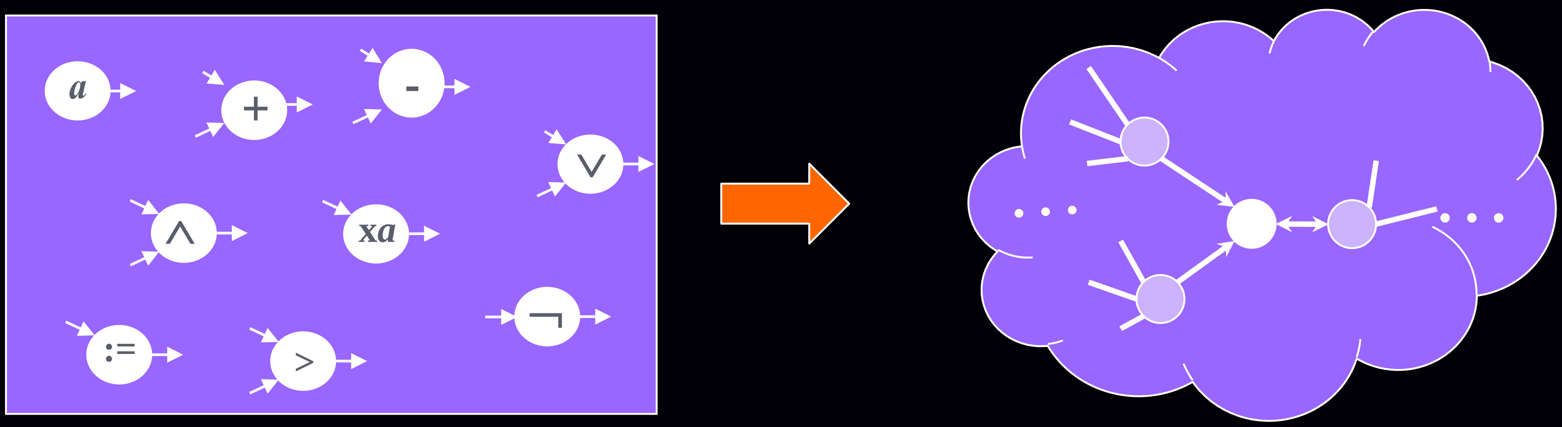
***z***: "output player" of the gadget

***x***, ***y***: "input players" of the gadget

gadgets may have additional players; their graph can be made bipartite

# PPAD-Completeness of POLYMATRIXNASH

**[Daskalakis, Goldberg, Papadimitriou'06]**




ARITHMCIRCUITSAT

POLYMATRIXNASH

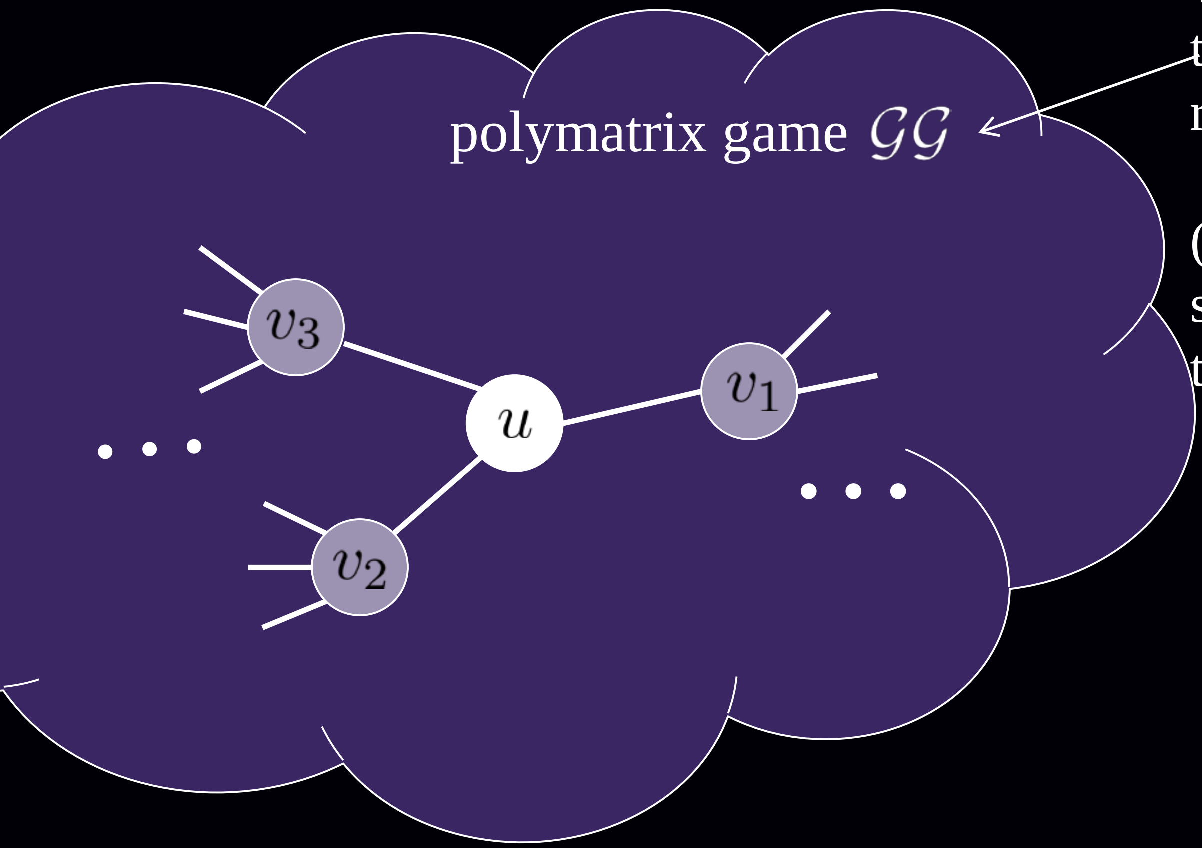
Given arbitrary instance of ARITHMCIRCUITSAT can create polymatrix game by appropriately composing game gadgets corresponding to each of the gates.

At any Nash equilibrium of resulting polymatrix game, the gate conditions are satisfied.

# Menu

- Graphical and Polymatrix Games
- PPAD-completeness of PolymatrixNash
- PPAD-completeness of Nash

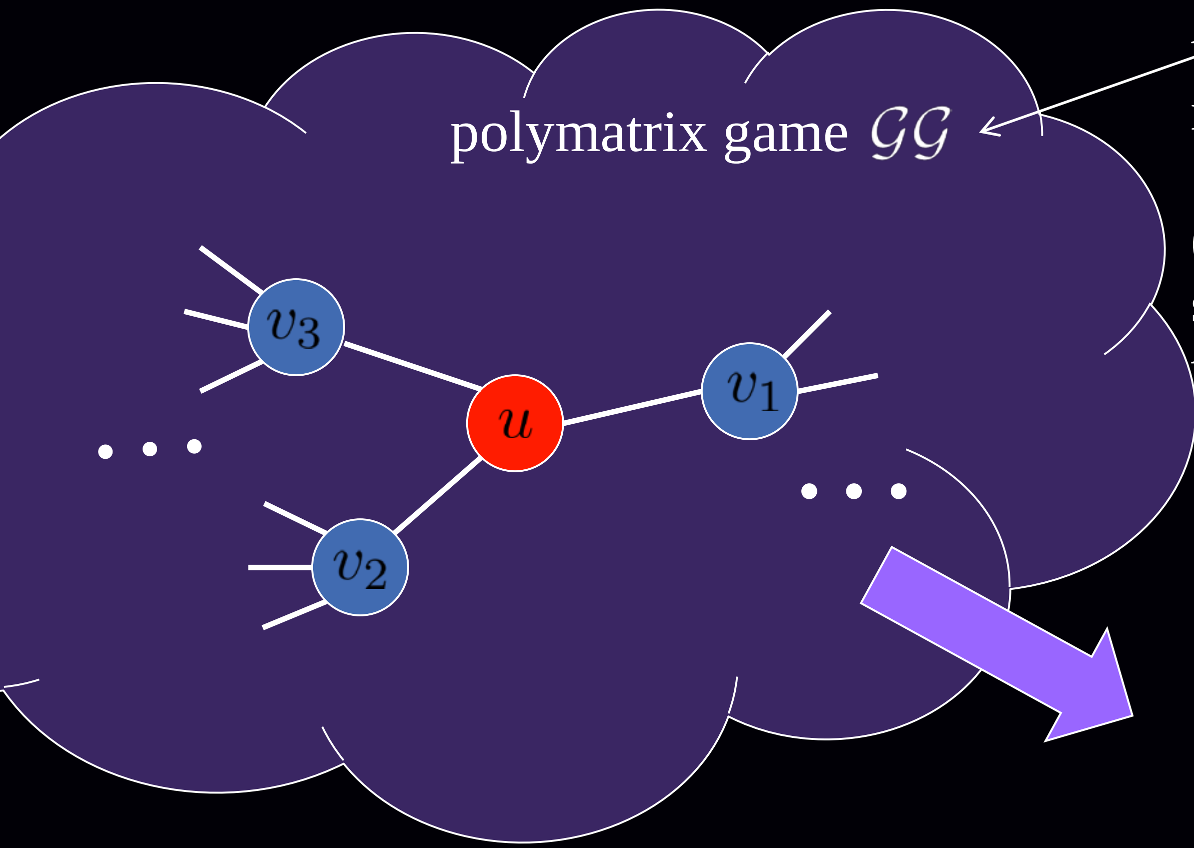
# Reducing Polymatrix to 2-player Games




w.l.o.g. can assume bipartite, by turning every gadget used in the reduction into a bipartite game

(input&output players are on one side and auxiliary player is on the other side)

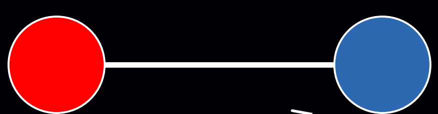
# Reducing Polymatrix to 2-player Games




w.l.o.g. can assume bipartite, by turning every gadget used in the reduction into a bipartite game

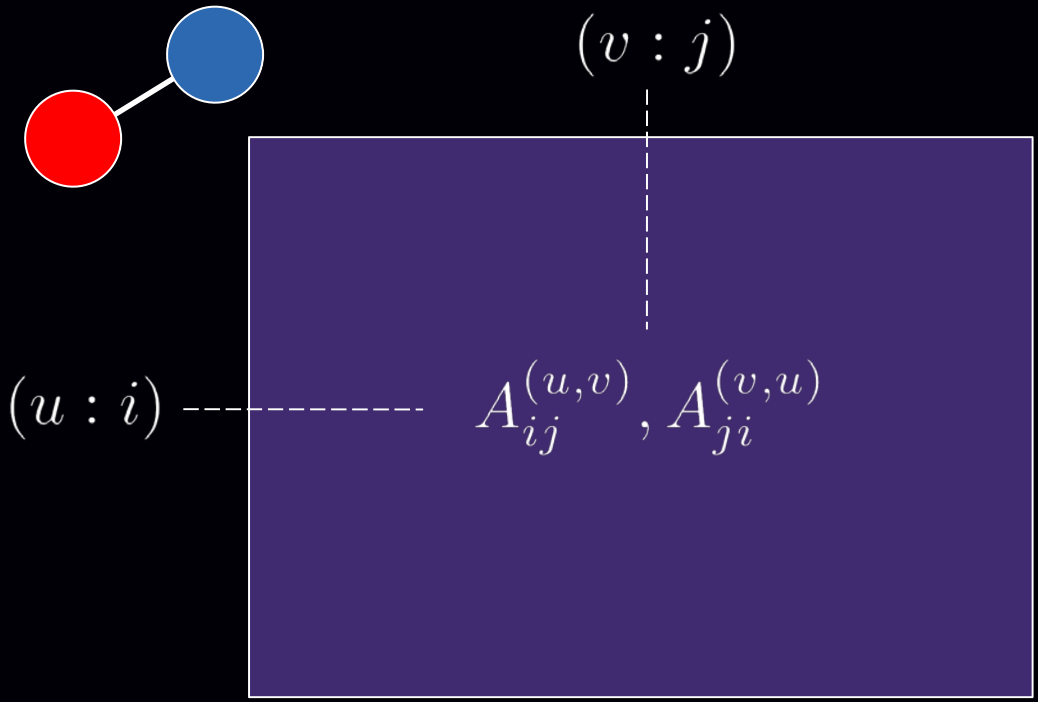
(input&output players are on one side and auxiliary player is on the other side)

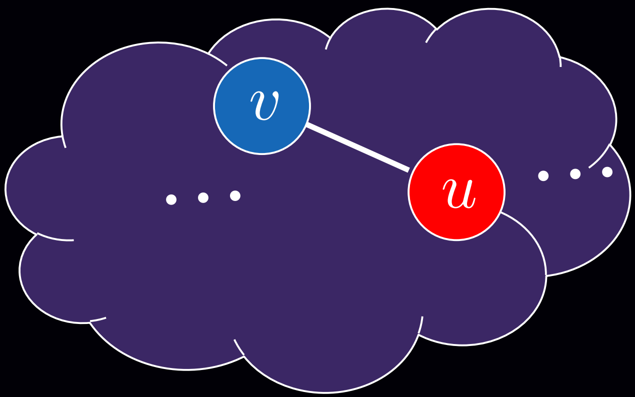
2-player game $\mathcal{G}$



red lawyer "represents" red nodes, while
blue lawyer "represents" blue nodes

# Payoffs of the Lawyer-Game




But why would a lawyer play strategies of every node he represents?

- Each lawyer's set of pure strategies is the union of the pure strategy sets of her clients

- *wishful thinking*: if ($x$ , $y$) is a Nash equilibrium of the lawyer-game, then the marginal distributions that $x$ assigns to the strategies of the red nodes and the marginals that $y$ assigns to the blue nodes, comprise a Nash equilibrium.

# Enforcing Equal Representation

- The lawyers play on the side a high-stakes game.

- W.l.o.g. assume that each lawyer represents *n* clients. Label each lawyer's clients 1,…,*n*, arbitrarily

- Payoffs of the high-stakes game:

Suppose the red lawyer plays any strategy of client *j*, and blue lawyer plays any strategy of client *k*, then

$\|$

***M***

If $j \neq k$, then both players get **0**.

If $j = k$, then red lawyer gets **+*M***, while blue lawyer gets **–*M***.

**Claim:** In any Nash equilibrium of the high stakes game, each lawyer assigns probability 1/***n*** to the set of pure strategies of each of his clients.

# Enforcing Equal Representation

naïve lawyer game

$(v : j)$

$(u : i)$ $A_{ij}^{(u,v)}, A_{ji}^{(v,u)}$

+

high stakes game

| | | | | |
|---|---|---|---|---|
| M, –M | … | 0, 0 | … | 0, 0 |
| | | | | |
| 0, 0 | | M, –M | | 0, 0 |
| | | | | |
| 0, 0 | … | 0, 0 | … | M, –M |

payoff table addition

M = 

**Choose:** $M > 2n \cdot u_{\max}$

$u_{\max}$ : maximum absolute value of payoffs in original game

# Analyzing the Lawyer Game

- when it comes to distributing the total probability mass among the different nodes of $\mathcal{GG}$, essentially only the high-stakes game is relevant to the lawyers…

*Lemma* 1: if ($x$, $y$) is an equilibrium of the lawyer game, for all $u$, $v$ :

$$x_u = \frac{1}{n} \cdot \left(1 \pm \frac{2u_{\max} n^2}{M}\right) \qquad y_v = \frac{1}{n} \cdot \left(1 \pm \frac{2u_{\max} n^2}{M}\right)$$

Proof: exercise

total probability mass assigned by lawyers on nodes $u$, $v$ respectively

- when it comes to distributing the probability mass $x_u$ among the different strategies of node $u$, only the payoffs of the game $\mathcal{GG}$ are relevant…

*Lemma* 2: The payoff difference for the red lawyer from strategies $(u:i)$ and $(u:j)$ is

$$\sum_v \sum_\ell \left(A_{i,\ell}^{(u,v)} - A_{j,\ell}^{(u,v)}\right) \cdot y_{v:\ell}$$

# Analyzing the Lawyer Game (cont.)

*Lemma* 2 ➔ if $x_{u:i} > 0$, then for all $j$:

$$\sum_v \sum_\ell \left( A_{i,\ell}^{(u,v)} - A_{j,\ell}^{(u,v)} \right) \cdot y_{v:\ell} \geq 0$$

- define $\hat{x}_u(i) := \frac{x_{u:i}}{x_u}$ and $\hat{y}_v(j) := \frac{y_{v:j}}{y_v}$ (marginals given by lawyers to different nodes)

**Observation:** if we had $x_u$ =1/$n$, for all $u$, and $y_v$ =1/n, for all $v$, then

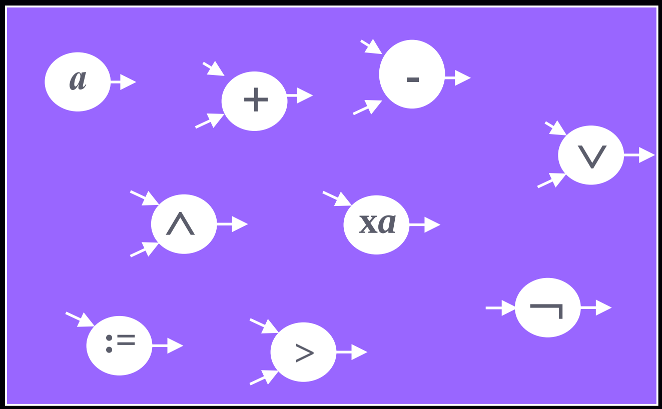
$\{\{\hat{x}_u\}_u, \{\hat{y}_v\}_v\}$ would be a Nash equilibrium.

- the $\pm \frac{2u_{\max}n}{M}$ deviation from uniformity results in an approximate Nash equilibrium of the polymatrix game.

- but $M$ can be chosen as large as we'd like, and APPROXIMATE-POLYMATRIXNASH is still PPAD-complete, as ARITHMCICRUITSAT allows approximation ε. ∎

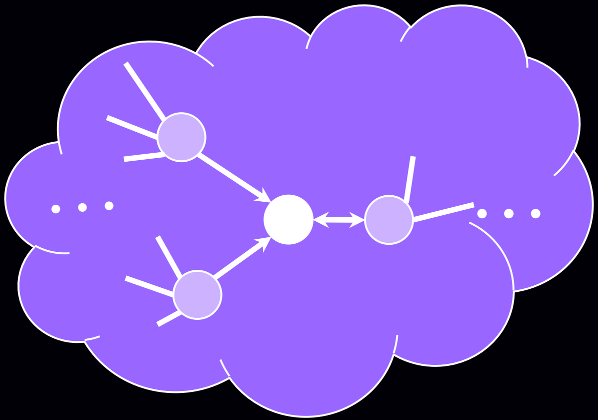
# PPAD-Completeness of NASH

DGP=Daskalakis-Goldberg-Papadimitriou




ArithmCircuitSAT

[DGP'06]



PolymatrixNash

[DGP'06]

[Chen-Deng'06]

4-player Nash

3-player Nash

2-player Nash

# Menu

- Graphical and Polymatrix Games
- PPAD-completeness of PolymatrixNash
- PPAD-completeness of Nash
- PPAD-completeness of
  - PreferenceGames
  - StableHypergraphMatching

# PREFERENCE GAMES

**Preference Game [Kintali et al'09]:**
- Player set = Strategy Set = $[\boldsymbol{n}]$
- Each player $\boldsymbol{i}$ has preference relation $\geq_i$ over $[\boldsymbol{n}]$

**Player Strategies:** Each player $\boldsymbol{i}$ chooses: $\boldsymbol{w_i}: [\boldsymbol{n}] \rightarrow [0, 1]$.
Strategies are feasible iff: (i) $\sum_j \boldsymbol{w_i(j)} = 1$ , $\forall\ \boldsymbol{i}$
(ii) $\boldsymbol{w_i(j)} \leq \boldsymbol{w_j(j)}$ , $\forall\ \boldsymbol{i, j}$

**Equilibrium:** Strategies $w_1, w_2, \ldots, w_n$ such that for all $i$, and all $w_i'$ such that $w_i', w_{-i}$ remains feasible:

$$\sum_{k \geq_i j} w_i(k) \geq \sum_{k \geq_i j} w_i'(k), \ \forall\ j$$

**[Kintali et al'09]**: Finding equilibrium is PPAD-complete.
Hardness reduction from ArithmCircuitSat.

# STABLEHYPERGRAPHMATCHING

**Input:**

- Hypergraph $\boldsymbol{H}=(\boldsymbol{V}, \boldsymbol{E})$
- For all $\boldsymbol{i} \in \boldsymbol{V}$ a linear order $\geq_{\boldsymbol{i}}$ over edges $\boldsymbol{e} \ni \boldsymbol{i}$

**Stable Matching:**

Set of edges $M$ that (i) is a matching

(ii) $\forall\, e, \exists\, i \in e, m \in M$ s.t. $\boldsymbol{m} \geq_{\boldsymbol{i}} \boldsymbol{e}$

**Fractional Stable Matching:**

Function $\boldsymbol{w}$: E$\rightarrow$ [0, 1] s.t. (i) $\sum_{i\,\in\, e} \boldsymbol{w}(\boldsymbol{e}) \leq 1$

(ii) $\forall\, e, \exists\, i \in e$ s.t. $\sum_{e' \ni i,\, e \leq_i e'} w(e') = 1$

**[Aharoni-Fleiner'03]:** Fractional stable matching exists.

**[Kintali et al'09]**: Finding fractional stable matching is PPAD-complete.
Hardness reduction from PREFERENCEGAMES.

# Menu

- Graphical and Polymatrix Games
- PPAD-completeness of PolymatrixNash
- PPAD-completeness of Nash
- PPAD-completeness of
  - PreferenceGames
  - StableHypergraphMatching
- Other Arguments of Existence in TFNP

# Other arguments of existence, and resulting complexity classes

"If a graph has a node of odd degree, then it must have another."

**PPA**

"Every directed acyclic graph must have a sink."

**PLS**

"If a function maps $n$ elements to $n$-1 elements, then there is a collision."

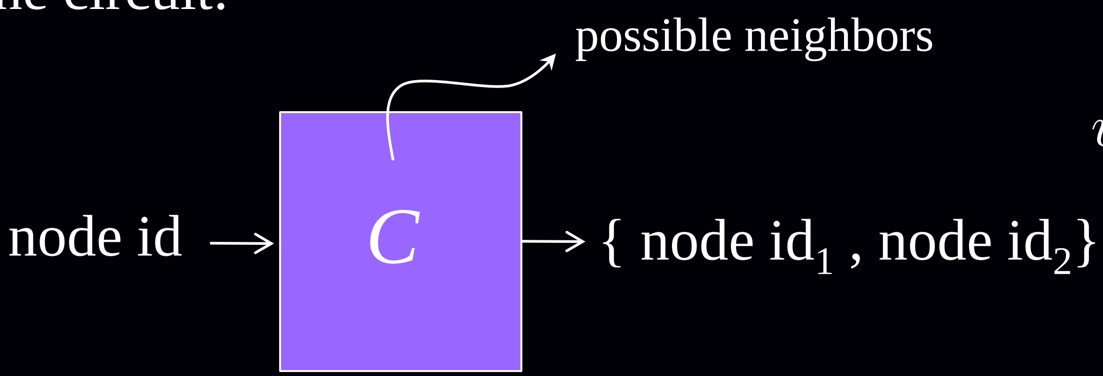
**PPP**

Formally?

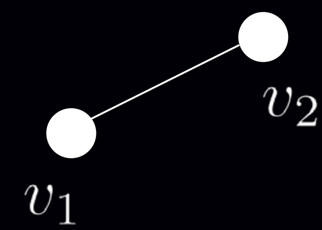
# The Class PPA [Papadimitriou '94]

*"If a graph has a node of odd degree, then it must have another."*

Suppose that an exponentially large graph with vertex set $\{0,1\}^n$ is defined by one circuit:



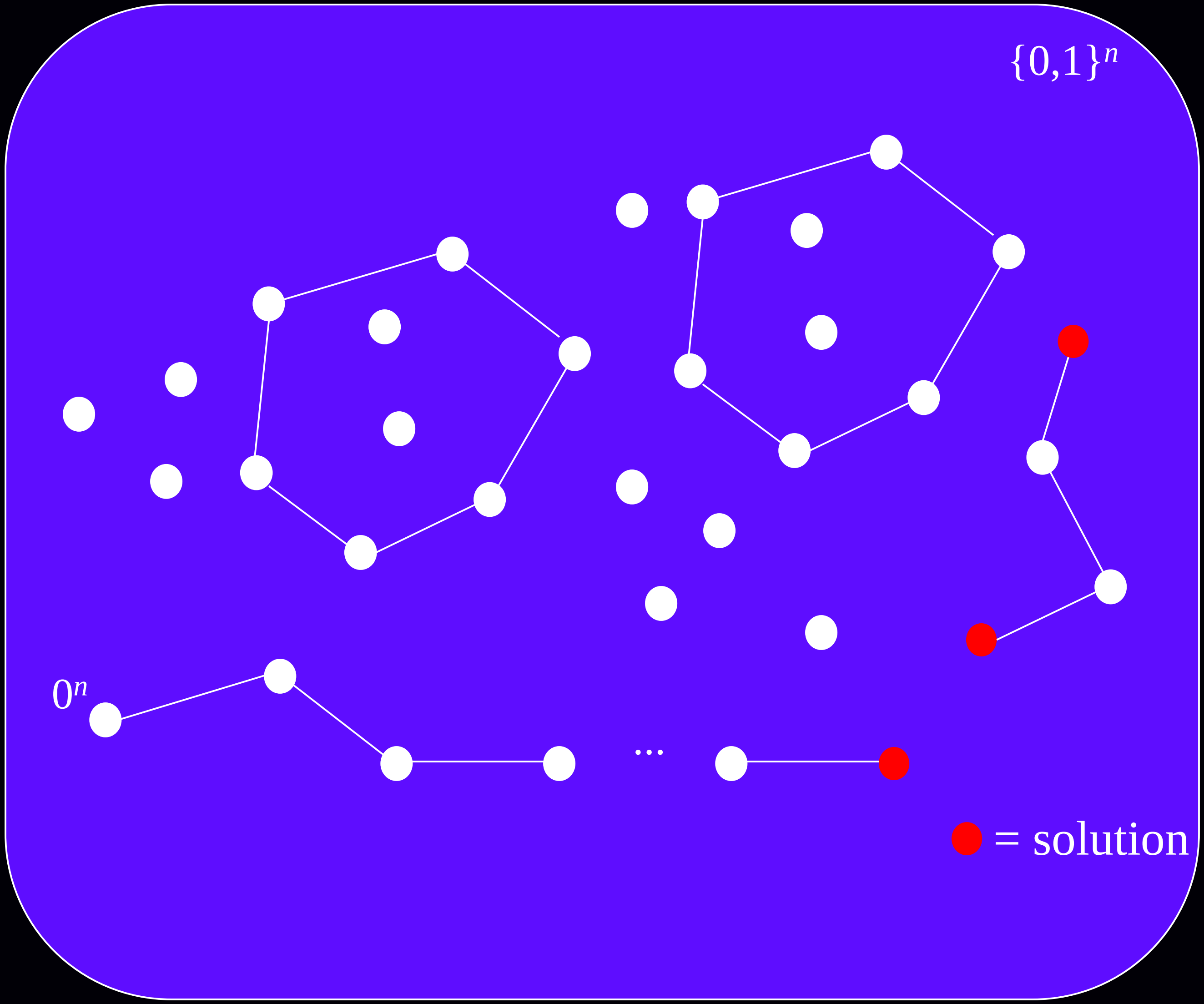

$$v_1 \in C(v_2) \quad \wedge \quad v_2 \in C(v_1)$$




**ODDDEGREENODE**: Given $C$: If $0^n$ has odd degree, find another node with odd degree. Otherwise say "yes."

**PPA =** { *Search problems in* FNP *reducible to* ODDDEGREENODE }

# ODDDEGREENODE



# SMITH ∈ PPA

**SMITH**: Given Hamiltonian cycle in 3-regular graph, find another one.

**[Smith]**: There must be another one.

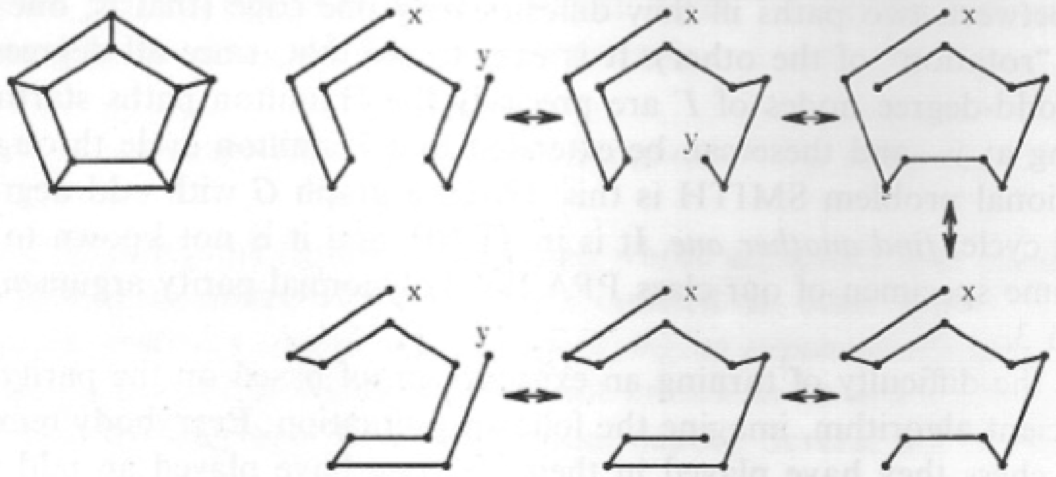



FIG. 1. Smiths theorem in the case of a cubic graph.

# The Class PLS  [JPY '89]

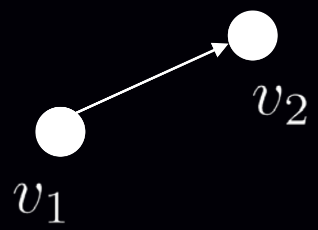
*"Every DAG has a sink."*

Suppose that a DAG with vertex set $\{0,1\}^n$ is defined by two circuits:

node id $\longrightarrow$ $C$ $\longrightarrow$ $\{\text{node id}_1, \ldots, \text{node id}_k\}$

$$v_2 \in C(v_1) \quad \wedge \quad F(v_2) > F_{(}v_1)$$

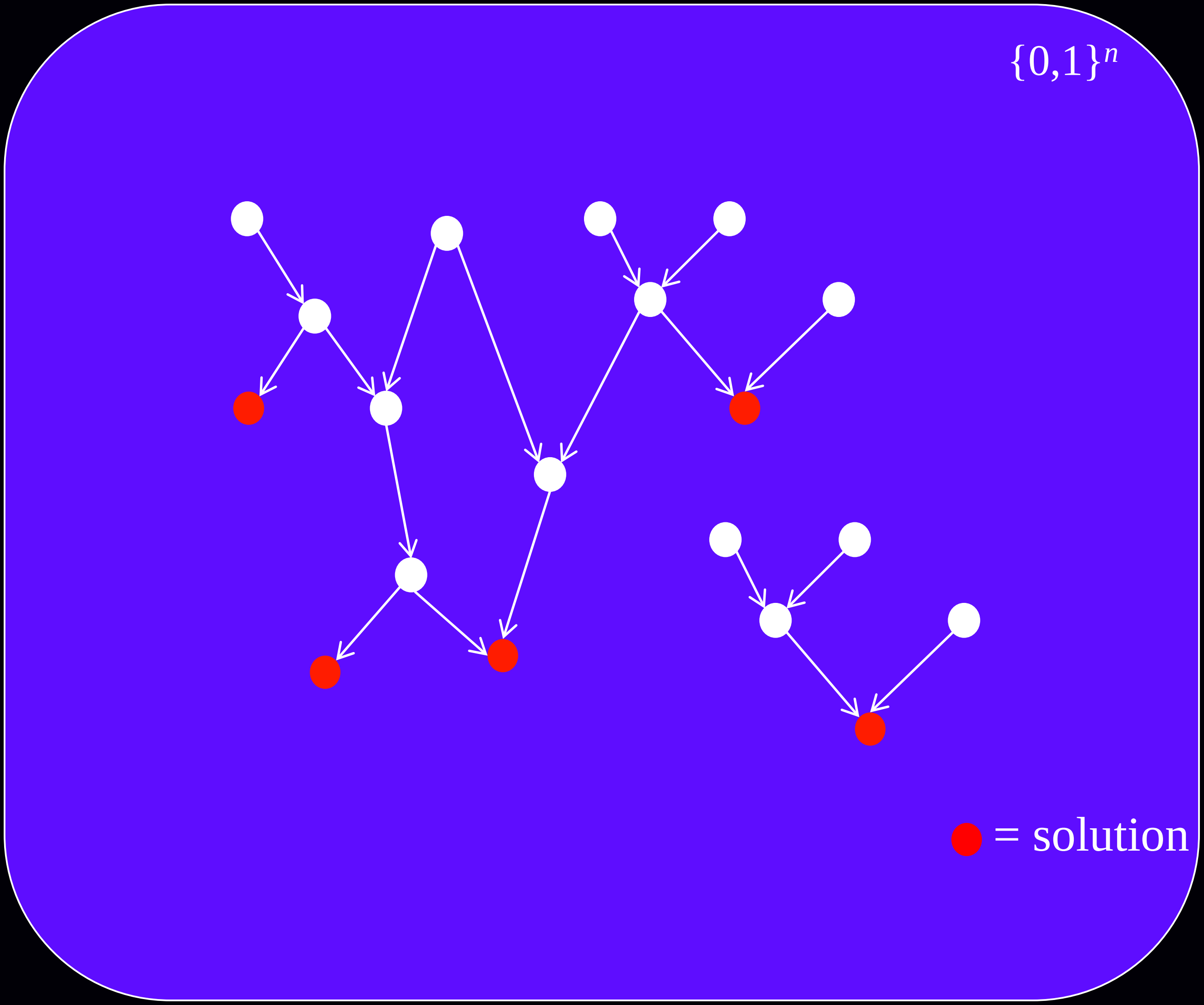
node id $\longrightarrow$ $F$ $\longrightarrow$ $\mathbb{R}$




**FINDSINK**: Given $C$, $F$: Find $x$ s.t. $F(x) \geq F(y)$, for all $y \in C(x)$.

**PLS =** { *Search problems in* FNP *reducible to* FINDSINK }

# FINDSINK



# LOCALMAXCUT is PLS-complete

**LOCALMAXCUT**: Given weighted graph $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{w})$, find a partition $\boldsymbol{V}=\boldsymbol{V}_1 \cup \boldsymbol{V}_2$ that is locally optimal (i.e. can't move any single vertex to the other side to increase the cut size.

**[Schaffer-Yannakakis'91]:** LocalMaxCut is PLS-complete.

# The Class PPP [Papadimitriou '94]

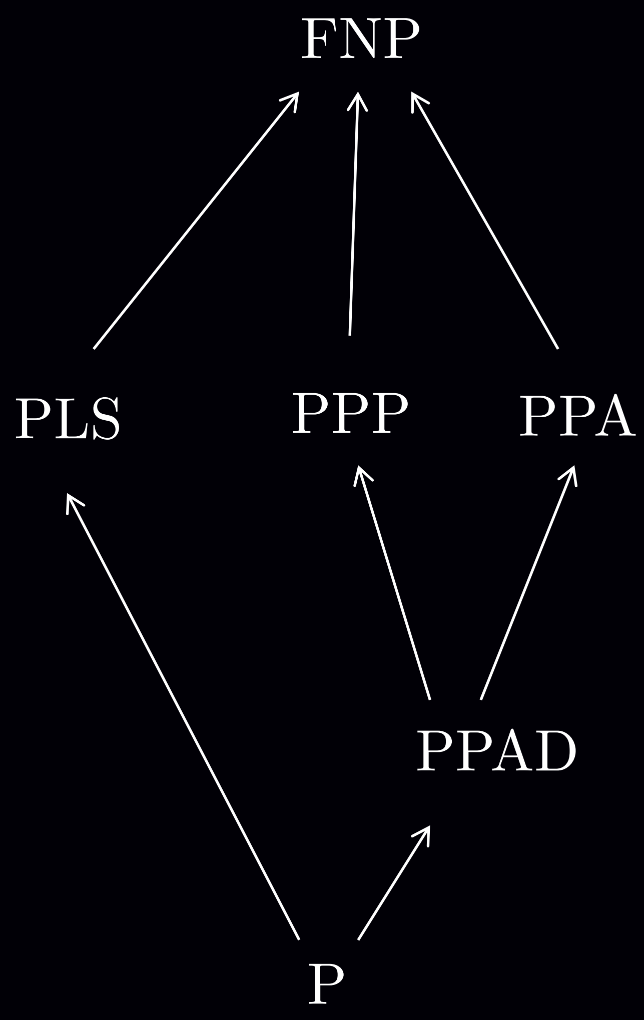
*"If a function maps n elements to n-1 elements, then there is a collision."*

Suppose that an exponentially large graph with vertex set $\{0,1\}^n$ is defined by one circuit:

$$\text{node id} \longrightarrow \boxed{C} \longrightarrow \text{node id}$$

**COLLISION**: Given $C$: Find $x$ s.t. $C(x) = 0^n$; or find $x \neq y$ s.t. $C(x)=C(y)$.

**PPP =** $\{$ *Search problems in FNP reducible to* COLLISION $\}$


FNP
PLS
PPP
PPA
PPAD
P

# Advertisement

**6.891:** Games, Decision, and Computation

Games, Auctions, and more fun