6.896: Topics in Algorithmic Game Theory

Lecture 7

Constantinos Daskalakis

Sperner's Lemma

Theorem [Sperner 1928]:

Suppose that the vertices of the canonical simplicization of the hypercube $[0,1]^n$ are colored with colors 0,1, …, n so that the following property is satisfied by the coloring on the boundary:

(P_n): For all $i \in \{1,\ldots, n\}$, none of the vertices on the face $x_i = 0$ uses color i; moreover, color 0 is not used by any vertex on a face $x_i = 1$, for some $i \in \{1,\ldots, n\}$.

Then there exists a panchromatic simplex in the simplicization. In fact, there is an odd number of those.

pan: from ancient Greek πᾶν = all, every

chromatic: from ancient Greek χρῶμα.= color

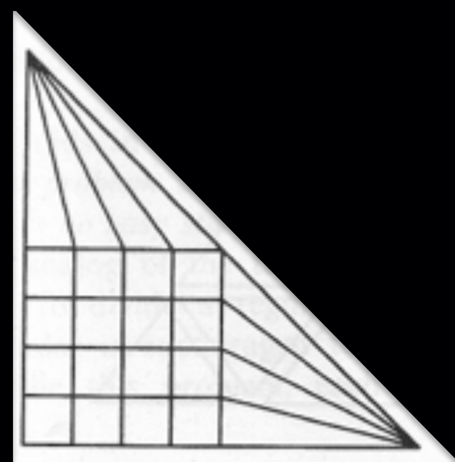
Remarks:

1. We need not restrict ourselves to the canonical simplicization of the hypercube shown above (that is, divide the hypercube into cubelets, and divide each cubelet into simplices in the canonical way shown in the previous lecture). The conclusion of the theorem is true for any partition of the cube into n-simplices, as long as the coloring on the boundary satisfies the property stated above.

   The reason we state Sperner's lemma in terms of the canonical triangulation is in an effort to provide an algorithmically-friendly version of the computational problem related to Sperner, in which the triangulation and its simplices are easy to define, the neighbors of a simplex can be computed efficiently etc. We follow-up on this shortly. Moreover, our setup allows us to make all the steps in the proof of Sperner's lemma "constructive" (except for the length of the walk, see below).

2. Sperner's Lemma was originally stated for a coloring of a triangulation of the n-simplex, (rather than the cube shown above). In that setting, we color the vertices of any triangulation of the n-simplex---a convex combination of points $v_0, v_1, \ldots, v_n$ in general position---with n colors, $0,1,\ldots,n,$ so that the facet not containing vertex v_i does not use color i. Then Sperner's lemma states that there exists a panchromatic simplex in the simplicization.

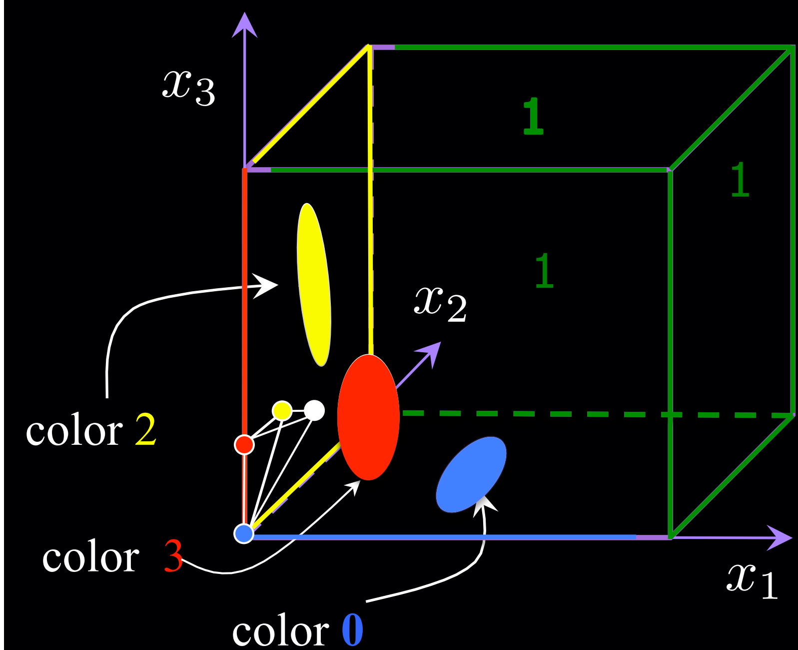
   Our coloring of the n-dimensional cube with $n+1$ colors is essentially mimicking the coloring of a simplex whose facets (except one) correspond to the facets of the cube around the corner 0^n, while the left-out facet corresponds to the "cap" of the hypercube around 1^n.



The structure of the Proof

Proof of Sperner

a. Start at the starting simplex; this has all the colors in {2,3,…,n, 0} but not color 1 and hence it is a colorful simplex. One of its colorful facets lies in $x_1=0$, while the other is shared with some neighboring simplex.



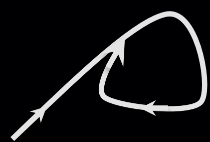
b. Enter into that simplex through the shared colorful facet. If the other vertex of that simplex has color 1 the walk is over, and the existence of a panchromatic simplex has been established. If the other vertex is not colored 1, the simplex has another colorful facet.

c. Cross that facet. Whenever you enter into a colorful simplex through a colorful facet, find the other colorful facet and cross it.

what are the possible evolutions of this walk?

Proof of Sperner

(i) Walk cannot loop into itself in a rho-shape, since that would require a simplex with three colorful facets.



(ii) Walk cannot exit the hypercube, since the only colorful facet on the boundary belongs to the starting simplex, and by (i) the walk cannot arrive to that simplex from the inside of the hypercube (this would require a third colorful facet for the starting simplex or a violation to (i) somewhere else on the path).

(iii) Walk cannot get into a cycle by coming into the starting simplex (since it would have to come in from outside of the hypercube)

The single remaining possibility is that the walk keeps evolving a path orbit, encountering a new simplex at every step while being restricted inside the hypercube. Since there is a finite number of simplices, walk must stop, and the only way this can happen is by encountering color 1 when entering into a simplex through a colorful facet.

$\Rightarrow$ a panchromatic simplex exists

Odd number of panchromatic simplices?

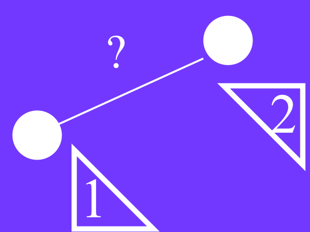
After original walk has settled, we can start a walk from some other simplex that is not part of the original walk.

- *If the simplex has no colorful facet, stop immediately* isolated node
- *If the simplex is colorful, start two simultaneous walks by crossing the two colorful facets of the simplex; for each walk: if S is a colorful simplex encountered, exit the simplex from the facet not used to come in; there are two cases:*
  - *either the two walks meet* cycle
  - *or the walks stop at a different panchromatic simplex each* path

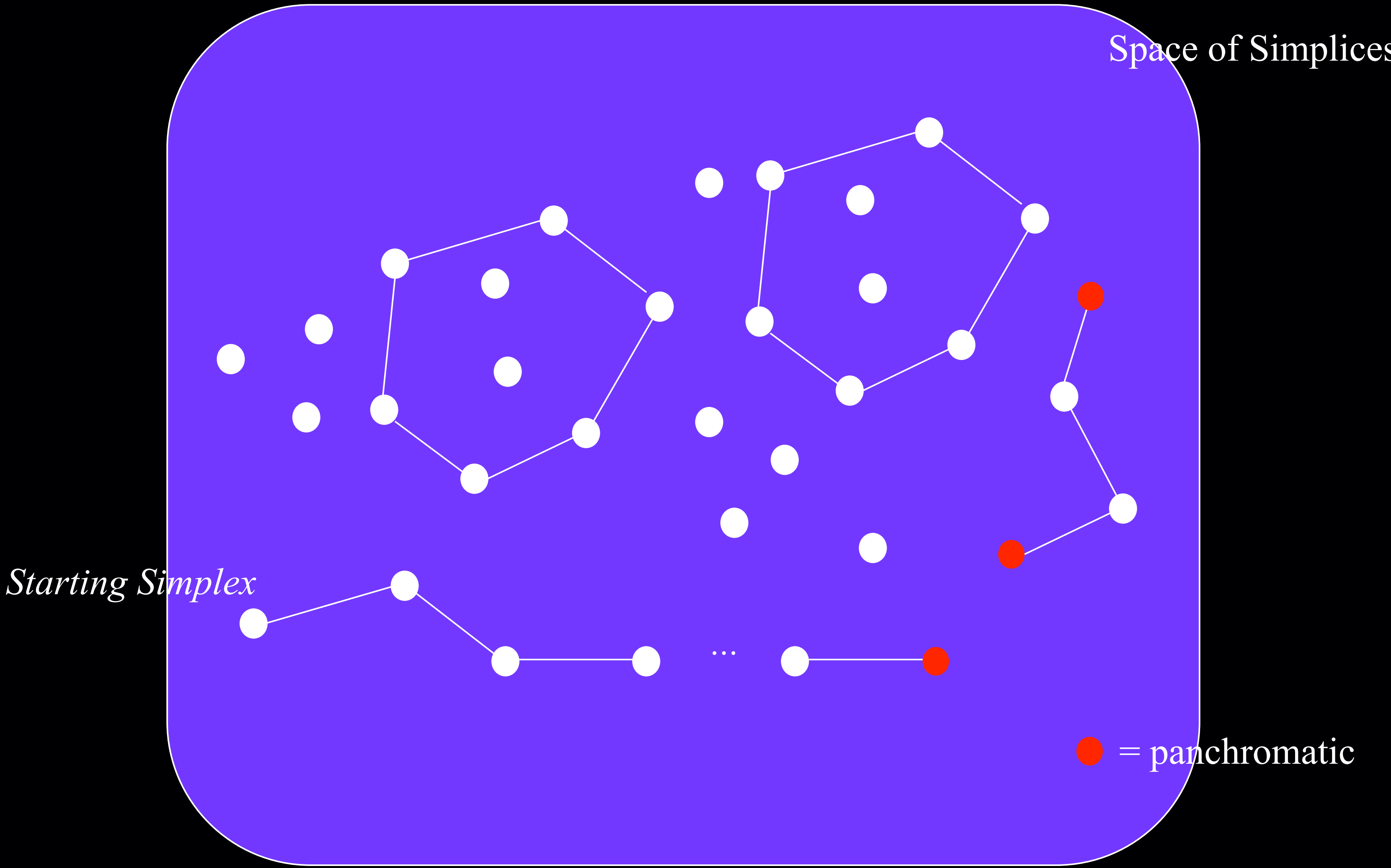
Abstractly…

Space of Simplices

Two simplices are ***Neighbors*** *iff they share a colorful facet*



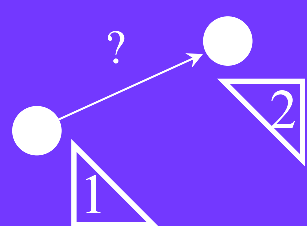
Proofs constructs a graph with degree $\leq$ 2



In fact can assign directions…

Space of Simplices

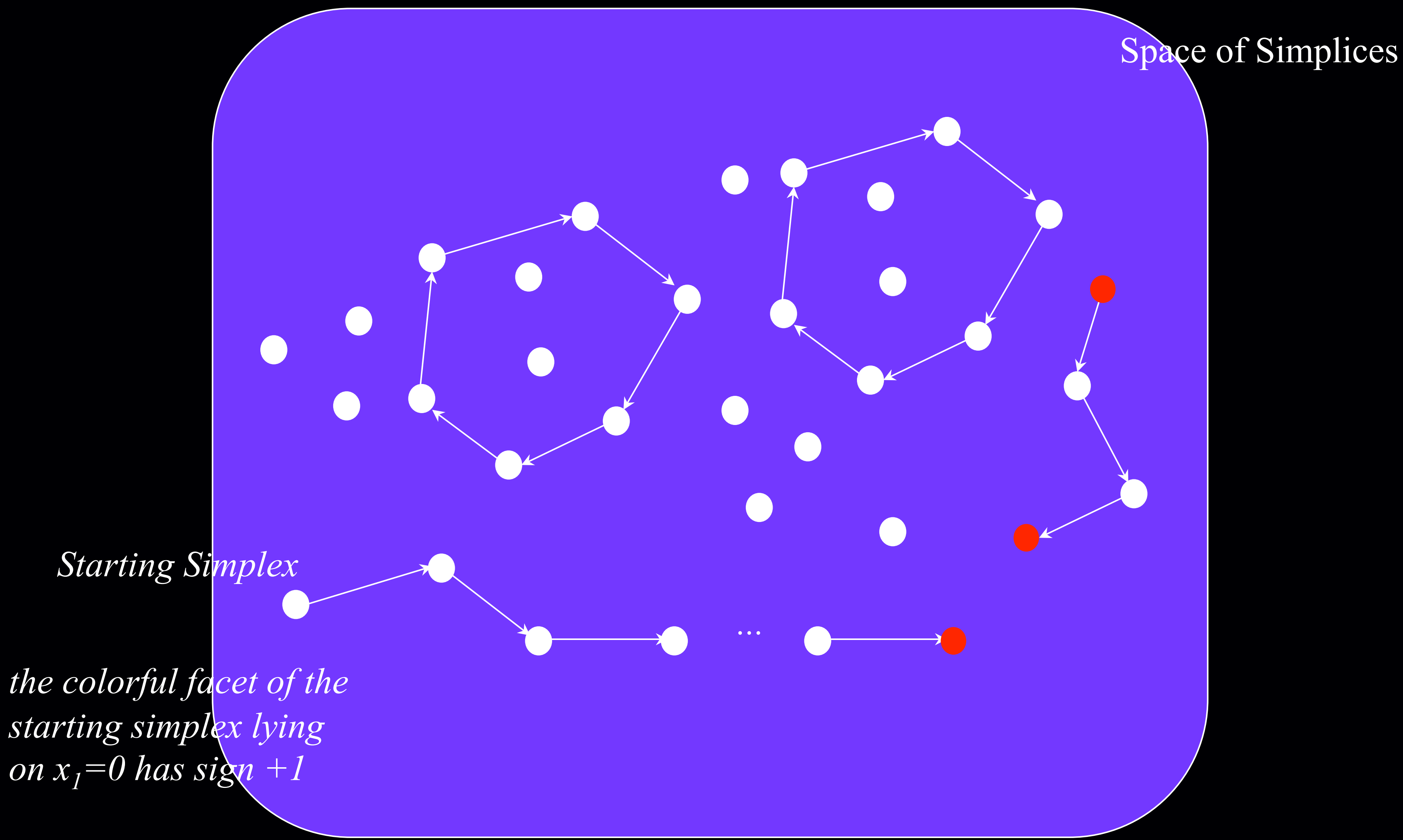
The simplices share a colorful facet, that has sign -1 for S_1 and sign +1 for S_2



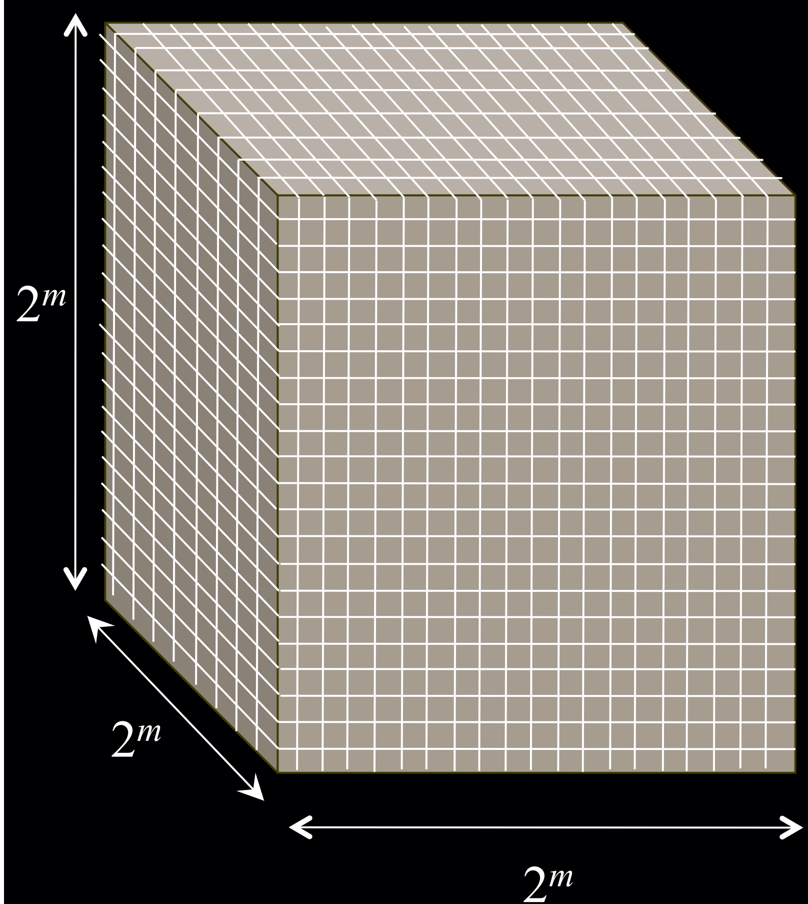
remarkable property of our sign function:

all nodes on a path agree on the direction of the path!

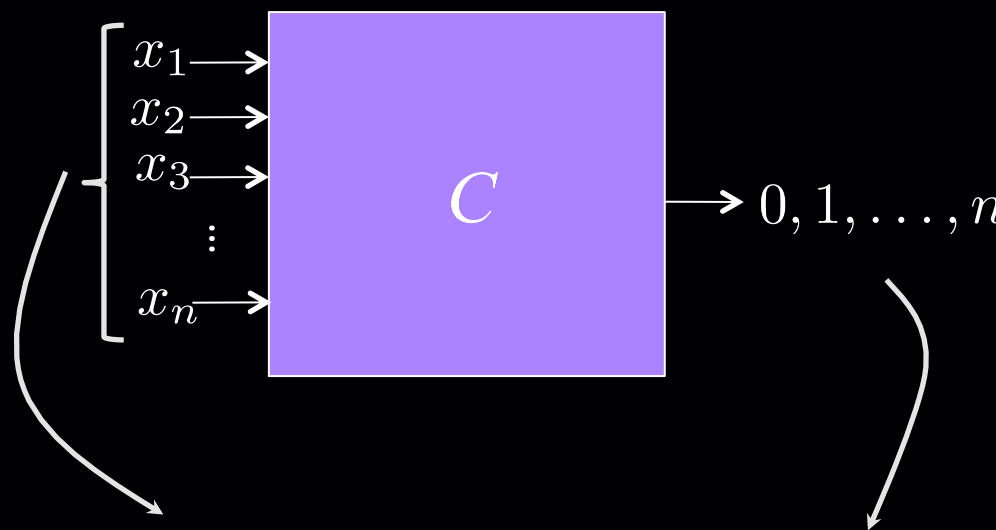
The Directed Graph



Computational Sperner Problem



INPUT:



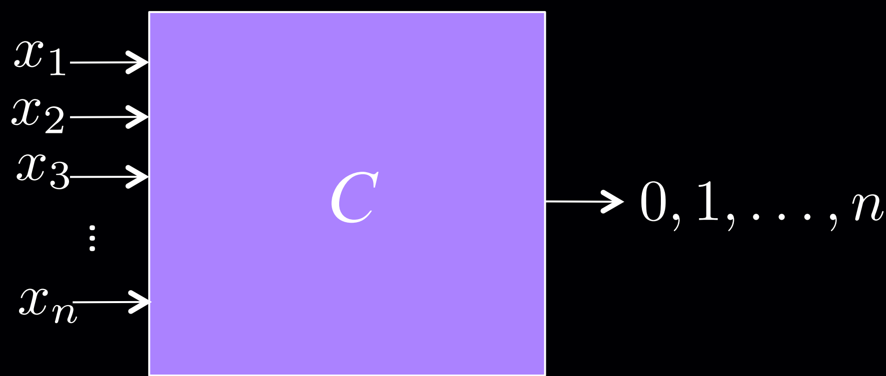
Coordinates of point
in the triangulation
m bits/coordinate

Color of node

SPERNER: Given a coloring circuit C, find a panchromatic simplex.

Trouble…

INPUT:



SPERNER: Given a coloring circuit C, find a panchromatic triangle.

The trouble is that the circuit may not produce a legal coloring. So there is no guarantee that there is a panchromatic simplex.

Definition of SPERNER

Two ways to circumvent this…

1st version

SPERNER: Given a coloring circuit C, either find a panchromatic triangle, or a point on the boundary that violates the legal coloring property.

2nd version

SPERNER: Given a coloring circuit C, find a panchromatic triangle in the coloring produced by another circuit C' that:

- agrees with C inside the hypercube;
- produces the "envelope coloring" at the boundary.

*Both versions correspond to **total problems**, that is for all inputs there is a solution.*

I can also solve them both, using the algorithm we have developed.

Function NP (FNP)

A *search problem* L is defined by a relation $R_L(x, y)$ such that

$$R_L(x, y)=1 \quad \text{iff} \quad y \text{ is a solution to } x$$

A search problem L belongs to FNP iff there exists an efficient algorithm $A_L(x, y)$ and a polynomial function $p_L(\cdot)$ such that

(i) if $A_L(x, z)=1 \quad \Rightarrow \quad R_L(x, z)=1$

(ii) if $\exists\, y$ s.t. $R_L(x, y)=1 \quad \Rightarrow \quad \exists\, z$ with $|z| \leq p_L(|x|)$ such that $A_L(x, z)=1$

Clearly, SPERNER $\in$ FNP.

Reductions between Problems

A search problem $L \in$ FNP, associated with $A_L(x, y)$ and p_L , is ***polynomial-time reducible*** to another problem $L' \in$ FNP, associated with $A_{L'}(x, y)$ and $p_{L'}$, iff there exist efficiently computable functions f, g such that

(i) x is input to L ➔ $f(x)$ is input to L'

(ii)

$A_{L'}(f(x), y)=1$ ➔ $A_L(x, g(y))=1$

$R_{L'}(f(x), y)=0, \forall y$ ➔ $R_L(x, y)=0, \forall y$

A search problem L is ***FNP-complete*** iff

$L \in$ FNP

L' is poly-time reducible to L, for all $L' \in$ FNP

$\boxed{\text{e.g. SAT}}$

Definition of BROUWER

Finding a Brouwer fixed point of a given function is not immediately a combinatorial problem. For one thing, the function could have a unique fixed point that is irrational. To define a combinatorial problem, we introduce approximation.

to guarantee that for given approximation ϵ can express an approximate fixed point with polynomial precision

Informally the Brouwer problem is the following:

Find an ***approximate fixed point*** of a continuous function $f : [0,1]^n \rightarrow [0,1]^n$, with some well-behaved modulus of continuity.

$$|f(x) - x| < \epsilon, \text{ for some } \epsilon$$

if well-behaved modulus of continuity, e.g. Lipschitz

Remark: An alternative (stronger) notion of approximation is to find x s.t.

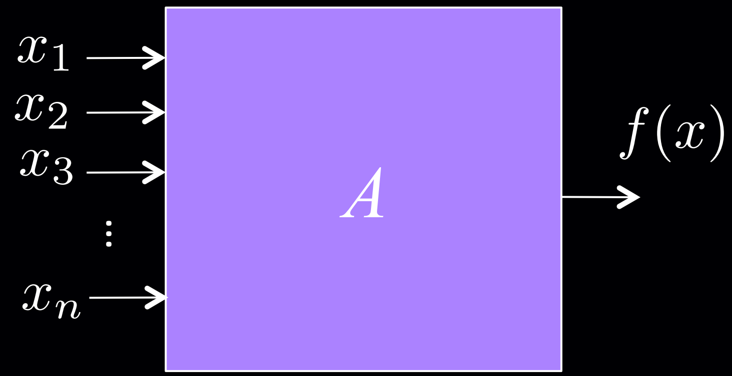
$$\exists\ y \text{ s.t. } |x - y| < \epsilon \text{ and } f(y) = y$$

Strong approximation

Definition of BROUWER

INPUT: a. an algorithm A (that is claimed to) evaluate a continuous function

$$f : [0,1]^n \rightarrow [0,1]^n.$$



b. an approximation requirement ϵ

c. a Lipschitz constant c claimed to be satisfied by the function

BROUWER: Find x such that $|f(x) - x| < \epsilon$

OR a pair of points x, y violating the Lipschitz constraint, i.e.

$$|f(x) - f(y)| > c|x - y|$$

OR a point that is mapped outside of $[0,1]^n$.

Remarks

1. Choice of norm: The norms used for the approximation requirement and the Lipschitz-ness in the definition of the problem are flexible.

2. Totality: No matter what the given algorithm *A* is the problem is total and in FNP.

proof: Reduce to SPERNER as follows

- define a sufficiently fine canonical simplicization of $[0,1]^n$, using cells of size 2^{-m} (m to be decided later);

- define a coloring of the vertices of the simplicization with (n+1)-colors depending on the direction of $f(x) - x$, as follows:

for all i=1,…,n, color *i* is allowed if $(f(x) - x)_i \leq 0$

color 0 is allowed if $(f(x) - x)_i \geq 0$, for all i

when coloring boundary nodes tie-break appropriately the above rules to avoid (if at all possible) violating the coloring requirements of Sperner (for example, if $(f(x)-x)_i=0$ for some x, s.t. $x_i=0$, do not use color *i*)

Remarks

proof (of totality continued)**:**

thus a valid instance of SPERNER is defined; solve this instance:

- if a point on the boundary violating the Sperner coloring requirements is returned, this corresponds to a point x mapped outside of the hypercube.

- if a panchromatic simplex is returned, it can be argued (similarly to the 2-D case in lecture 5) that

$$|(f(z^0) - z^0)_i| \leq |(f(z^0) - f(z^i))_i| + |(z^0 - z^i)_i|, \forall i$$

where $z^0, z^1, \ldots, z^n$ are the vertices of the simplex colored 0, 1,…, n

- if Lipschitz condition is satisfied for all pairs of points (z^0, z^i), the above implies (say we are working with the infinity norm) that

$$|f(z^0) - z^0|_\infty < (c+1)2^{-m} \qquad (**)$$

and similarly for other norms…

the only way (**) does not hold is when some pair (z^0, z^i) violates the Lipschitz condition. Such pair can be identified. ■

Definition of NASH

INPUT: A game described by

- the number of players n;
- an enumeration of the strategy set S_p of every player $p = 1,\ldots, n$;
- the utility function $u_p : S \longrightarrow \mathbb{R}$ of every player.

An approximation requirement ϵ

NASH: Compute an $\epsilon-\text{Nash}$ equilibrium of the game.

that is, an $\epsilon-\text{well supported Nash equilibrium}$

i.e. everything in the support of a player is an $\epsilon-\text{maximizer}$ of payoff for that player given the strategies of the other players.

Remarks

1. Approximation: Already in his 1951 paper, Nash provides a three-player game whose unique equilibrium is irrational. This motivates our definition of the problem in terms of approximation.

2. 2-player Games: We will see later, that two-player games always have a rational equilibrium of polynomial description complexity in the size of the game (assuming that the payoffs of the game are rationals). Hence, for two-player games we can also define the exact NASH problem.

3. Totality is guaranteed from Nash's theorem

4. Notion of approximation: We could define our problem in terms of the alternative notion of an $\epsilon-\text{approximate}$ Nash equilibrium. This won't affect the complexity of the problem given the following:

maximum absolute value in payoffs

Theorem [Daskalakis-Goldberg-Papadimitriou '09]

Given an ϵ - approximate Nash equilibrium of an n-player game, we can efficiently compute a $\sqrt{\epsilon} \cdot (\sqrt{\epsilon} + 1 + 4(n-1)u_{\max}) - \text{Nash equilibrium}$

Remarks

5. NASH ➔ BROUWER :

- use Nash's function (recall from Lecture 5), defined as follows

$$x \overset{f}{\longmapsto} y$$

$$y_p(j) = \frac{x_p(j) + \max\left(0, u_p(j; x_{-p}) - u_p(x)\right)}{1 + \sum_{j \in S_p} \max\left(0, u_p(j; x_{-p}) - u_p(x)\right)}$$

- f is Lipschitz:

Theorem [Daskalakis-Goldberg-Papadimitriou '09]:

For all pairs of mixed strategy profiles *x*, *y*:

$$|f(x) - f(y)|_\infty \leq [1 + 2u_{\max} n \cdot m \cdot (m+1)]|x - y|_\infty$$

where *m* is an upper bound on the number of strategies of a player.

Remarks

5. NASH ➔ BROUWER (cont.):

- approximation preservation

Theorem [Daskalakis-Goldberg-Papadimitriou '09]:

If a vector x satisfies

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

then x is a

$$m\sqrt{\epsilon(1 + m \cdot u_{\max})}\left(1 + \sqrt{\epsilon(1 + m \cdot u_{\max})}\right)\max\{u_{\max}, 1\}$$

approximate Nash equilibrium of the game.

Remarks

5. NASH ➔ BROUWER (cont.):

- Final Print:

We defined BROUWER for functions in the hypercube. But Nash's function is defined on the product of simplices. Hence, to properly reduce NASH to BROUWER we first embed the product of simplices in a hypercube, then extend Nash's function to points outside the product of simplices in a way that does not introduce approximate fixed points that do not correspond to approximate fixed points of Nash's function.

Our Reductions so far…

NASH $\rightsquigarrow$ BROUWER $\rightsquigarrow$ SPERNER $\in$ FNP

both Reductions are polynomial-time

Is then SPERNER FNP-complete?

- with our current notion of reduction no, because SPERNER always has a solution, while SAT not;

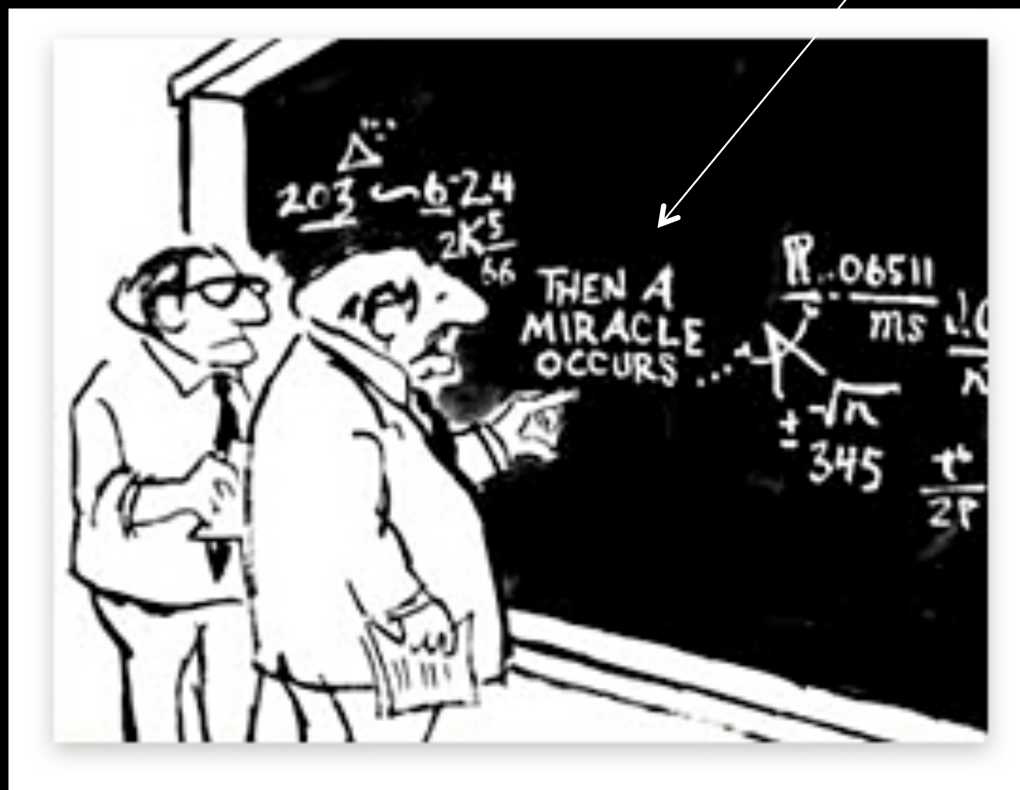
- we could try to change our notion of reduction, e.g., require that a solution to SPERNER informs us about whether the SAT instance is satisfiable or not, and provides us with solution to the SAT instance in the yes case;

  but this can be turned into a non-deterministic algorithm for checking "no" answers to SAT: guess the solution to SPERNER; this will inform you about whether the answer to the SAT instance is "yes" or "no", leading to $NP = co-NP$ …

- finally, we could turn SPERNER into a non-total problem, by removing the boundary conditions; this way, SPERNER can be easily shown FNP-complete, but all the structure of the original problem is lost in the reduction.

A Complexity Theory of Total Search Problems ?

??



A Complexity Theory of Total Search Problems ?

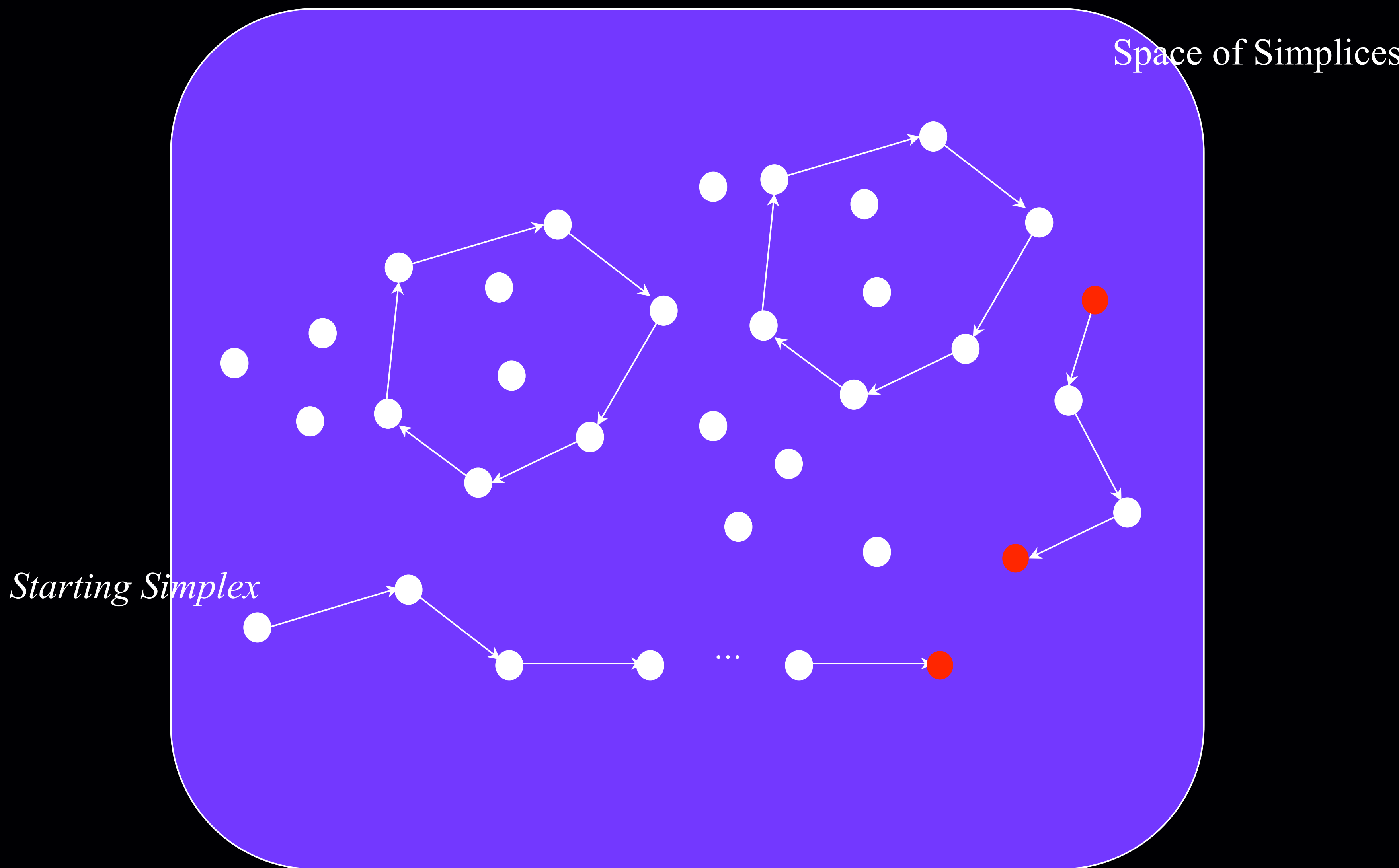
100-feet overview of our methodology:

1. identify the combinatorial argument of existence, responsible for making the problem total;

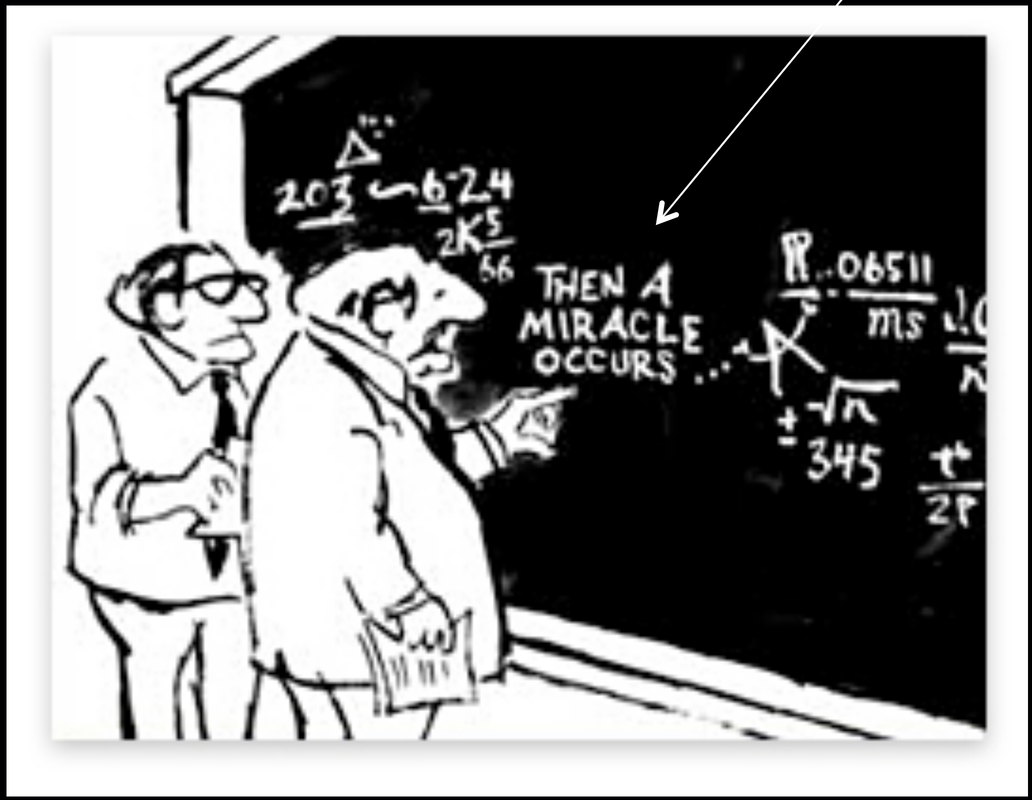
2. define a complexity class inspired by the argument of existence;

3. make sure that the complexity of the problem was captured as tightly as possible (via a completeness result).

Recall Proof of Sperner's Lemma



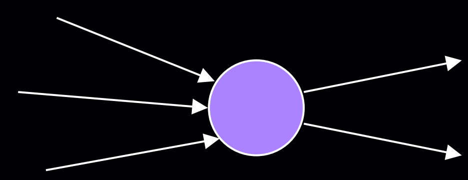
Combinatorial argument of existence?



The Non-Constructive Step

an easy parity lemma:

a directed graph with an unbalanced node (a node with indegree ≠ outdegree) must have another.



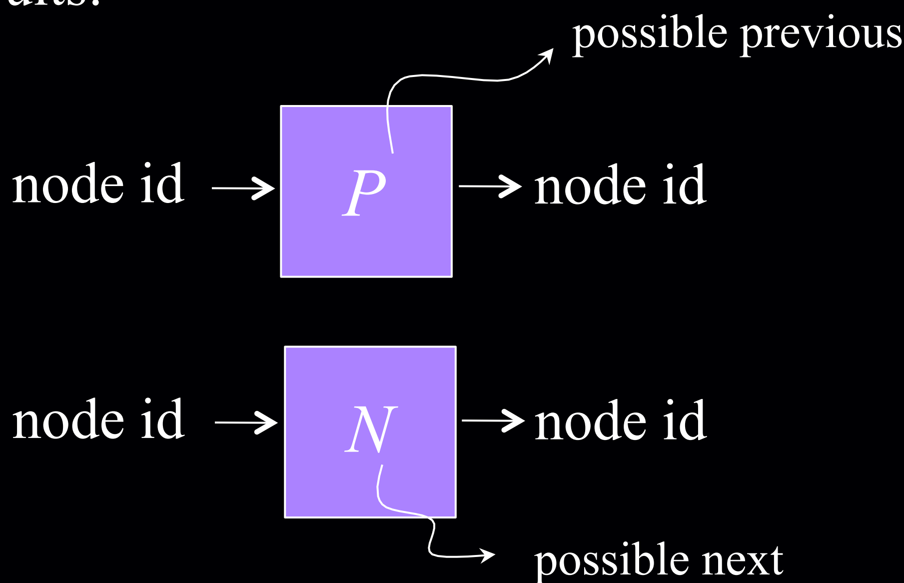
but, why is this non-constructive?

given a directed graph and an unbalanced node, isn't it trivial to find another unbalanced node?

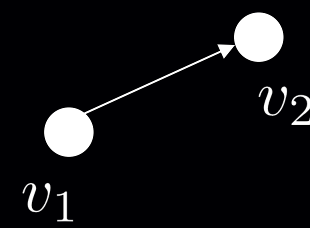
the graph can be exponentially large, but has succinct description…

The PPAD Class [Papadimitriou '94]

Suppose that an exponentially large graph with vertex set $\{0,1\}^n$ is 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 say "yes".

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

Inclusions

(i) $\text{PPAD} \subseteq \text{FNP}$

(ii) $\text{SPERNER} \in \text{PPAD}$

PROOF: sufficient to define appropriate circuits *P* and *N*

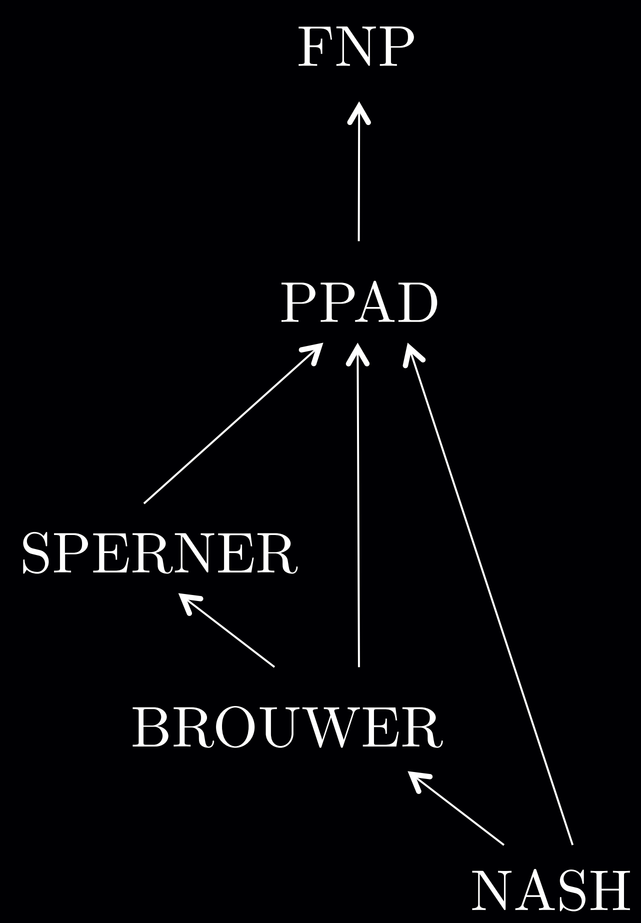
- starting simplex $\rightsquigarrow$ 0^n

- $P(0^n) = 0^n$, and $N(0^n)$ outputs the simplex sharing the colorful facet with the starting simplex

- if a simplex *S* is neither colorful nor panchromatic, then *P* outputs *S*, while *N* outputs 0^n (this makes sure that S is isolated)

important that the directions are locally computable, and consisten

- if a simplex S has a colorful facet shared with another simplex S', then if the sign of the facet is -1 then N(S)=S'; if the sign is +1 then P (S)=S'


FNP
PPAD
SPERNER
BROUWER
NASH

Other arguments of existence

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

PPAD

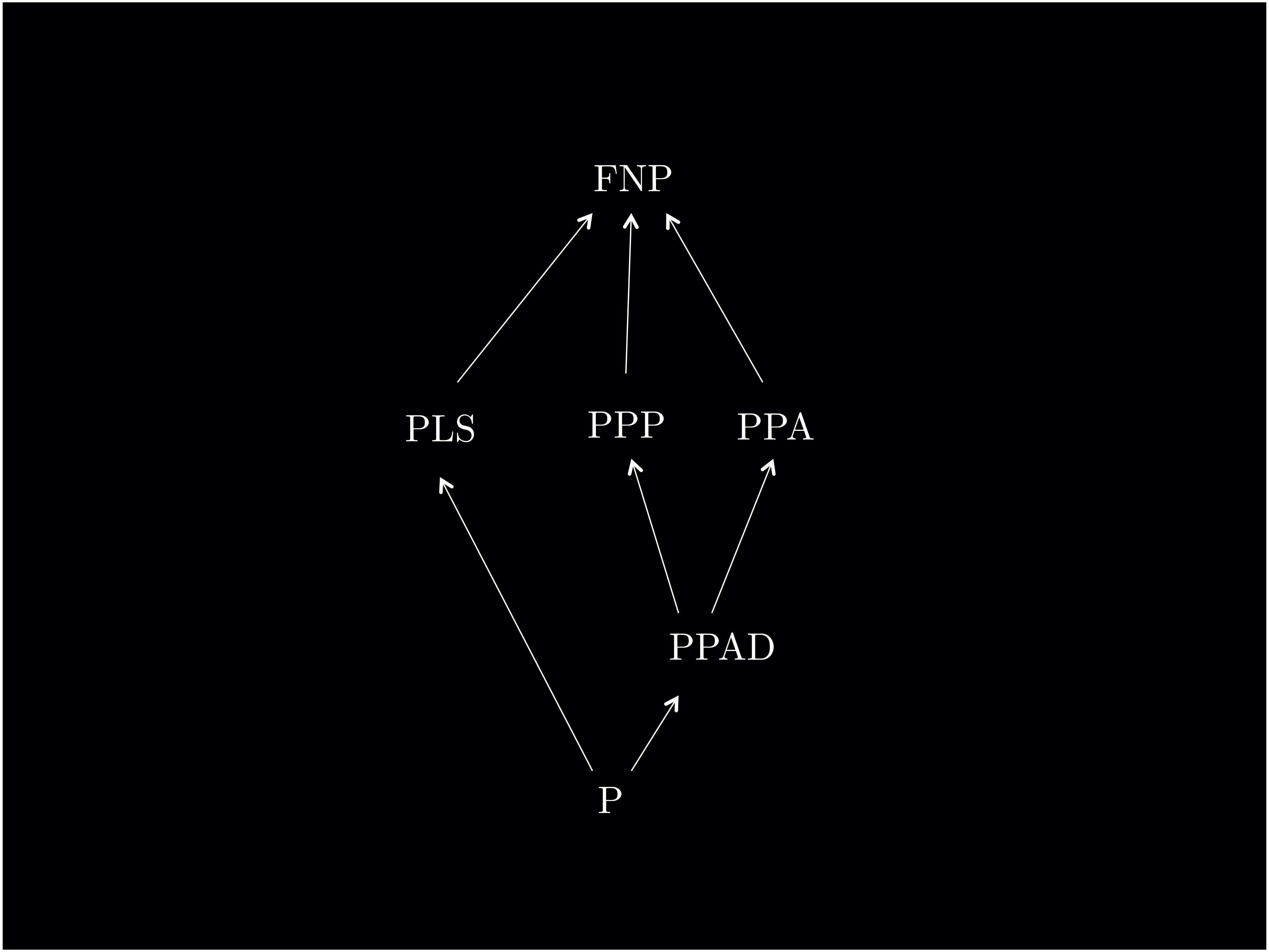
"Every directed acyclic graph must have a sink."

PLS

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

PPP

Formally?


FNP
PLS
PPP
PPA
PPAD
P