Subclasses of TFNP and stuff

Psomas Christos-Alexandros

Why bother?

- FNP: Given an input x and a polynomial-time predicate F(x,y), if there exists a y satisfying F(x,y) then output any such y, otherwise output 'no.'
- TFNP: y always exists
- TFNP is the analogue of NP ∩ coNP
- TFNP is semantic => No Complete Problems

PLS: Polynomial Local Search

- A problem ***A*** in PLS is defined in terms of two polynomial algorithms ***N*** and ***c***.
- For each input x, S(x) is the set of all solutions
- ***N*** and ***c*** compute for each input x and node s∈S(x) the set of neighbors *N(x,s)* and the cost *c(x,s)*.
- Find a local optimum ***s**** (a solution s.t. no neighbor has better cost).

PLS: Polynomial Local Search

- Why is it in TFNP?
- Every natural member of a semantic class is equipped with a mathematical proof that it belongs to that class!
- Graph G, s.t. the adjacency lists are the N(x,s) with arcs leading to nodes with worst c
- Proof of existence: Every directed acyclic graph has a sink

G can be exponentially large!

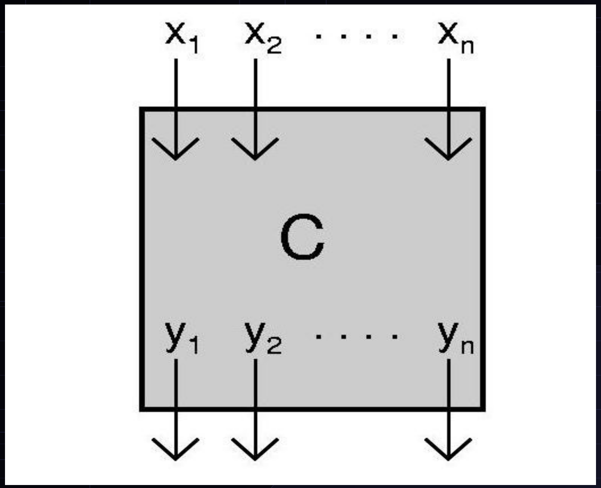
PLS: Polynomial Local Search

- Complete Problems:
  - GRAPH PARTITIONING under the swap neighborhood
  - MAX CUT under the flip neighborhood
  - MAX 2 SAT under the flip neighborhood
  - FLIP

PPP: Polynomial Pigeonhole Principle

- The class of all total search problems polynomially reducible to the following problem:

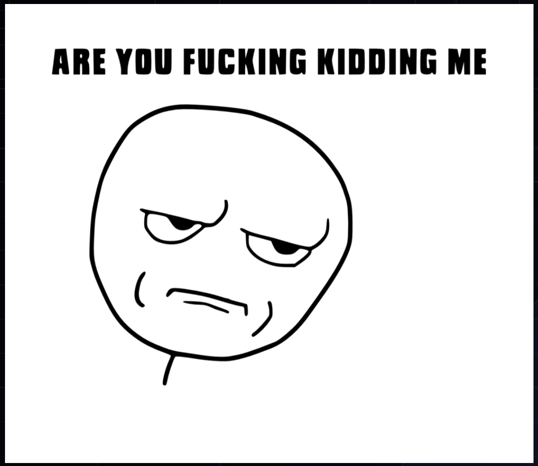
  PIGEONHOLE CIRCUIT: Given a boolean circuit C having the same number of inputs as outputs, find either x s.t $C(x) = 0^n$ or x and x' s.t $C(x) = C(x')$



If there are no inputs that map to the all-zeroes vector, then by the pigeonhole principle, there must be two inputs that map to the same output. So there must always exist a solution.

PPP: Polynomial Pigeonhole Principle

- Proof of existence: Pigeon Principle!
- Complete Problems:
  - Pigeonhole Circuit (surprising!)
  - That's pretty much it...



In PPP:

- Discrete Logarithm
- Factoring

PPA: Polynomial Parity Argument

- Let A be a problem and M the (associated) poly-time TM
- Let x be an input of A
- Let $C_x = \Sigma^{p(|x|)}$ be the configuration space of x, i.e. the set of all strings of length at most p(|x|)
- On input c ∊ C_x machine M outputs in time O(p(n)) a set of at most two configurations M(x,c)
  - M(x,c) may well be empty if c is "rubbish"
- c and c' are neighbors ([c,c'] ∊ G(x)) iff c ∊ M(x,c) and c' ∊ M(x,c')
  - G(x) is symmetric with degree at most 2
  - It is the **implicit search graph** of the problem
- Let M(x, 0…0) = {1…1} and 0…0 € M(x, 1…1), hence 0…0 is the standard leaf
- PPA is the class of problems defined as follows:

"Given x, find a leaf of G(x) other than the standard leaf 0…0"

PPA: Polynomial Parity Argument

- Define PPA' by allowing the degree of G(x) to be polynomially large.
- PPA' = PPA!

- Proof of existence: If an undirected graph has an odd-degree node, then it has another

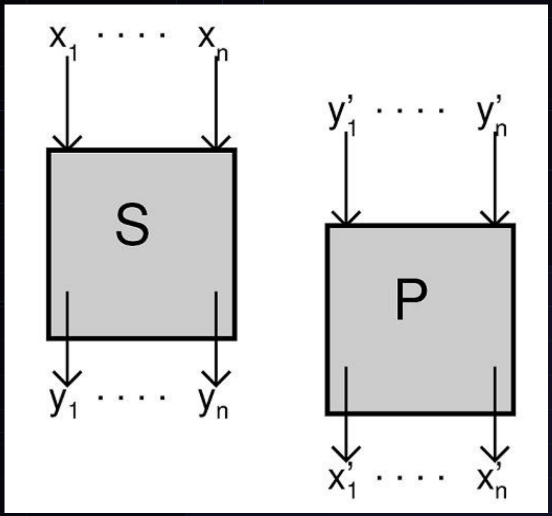
PPA: Polynomial Parity Argument

- Problems:
  - Another Hamilton Path
  - Cubic Subgraph
  - Chevalley's theorem for p=2
  - …
- Complete Problems:
  - Sperner's lemma for non-orientable 3-manifolds

PPAD: Polynomial Parity Argument Directed

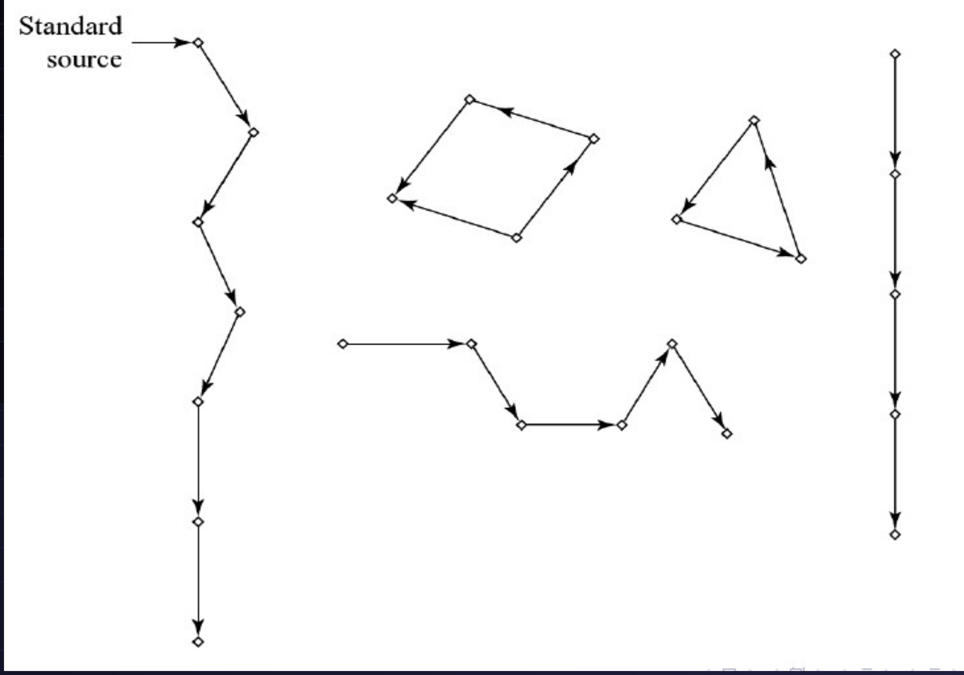
- The class of all total search problems polynomially reducible to the following problem:

  END OF THE LINE: Given two circuits S and P, each with n input and output bits, such that $S(0^n) \neq 0^n = P(0^n)$, find an input $x \in \{0, 1\}^n$ s.t. $P(S(x)) \neq x$ or $S(P(x)) \neq x \neq 0^n$



PPAD

- END OF LINE:
  - Exponentially large G
  - We are given an algorithm that returns for every node his successor
  - Find a sink!



Proof of Existence: If a directed graph has an unbalanced node, then it has another

A bit of game theory

Embrace Yourselves

What's game?

- Prisoners' Dilemma

P1 \ P2	Confess	Silent
Confess	4 / 4	1 / 5
Silent	5 / 1	2 / 2

- N players
- Each has a set of strategies S_i
- Strategy profile σ is a vector of n strategies: $S_1 \times S_2 \times \ldots \times S_n$
- Each player has a utility function $u_i: S_1 \times \ldots \times S_n \longrightarrow R$

What's an equilibrium?

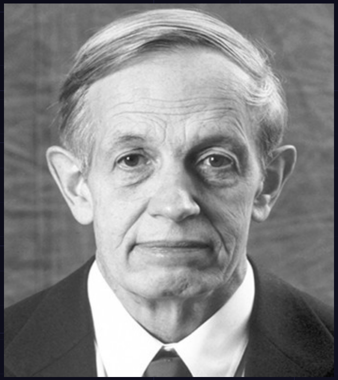
- A strategy profile σ is a Nash-equilibrium if no player has anything to gain by changing only his strategy

P1 \ P2	Confess	Silent
Confess	4 4	5 1
Silent	1 5	2 2

	1 \ 2	½ H	½ T
½	H	-1 1	1 -1
½	T	1 -1	-1 1

Existence of Nash equilibria

- Nash(1951) : Every finite game has an equilibrium.



Finding a Nash equilibrium

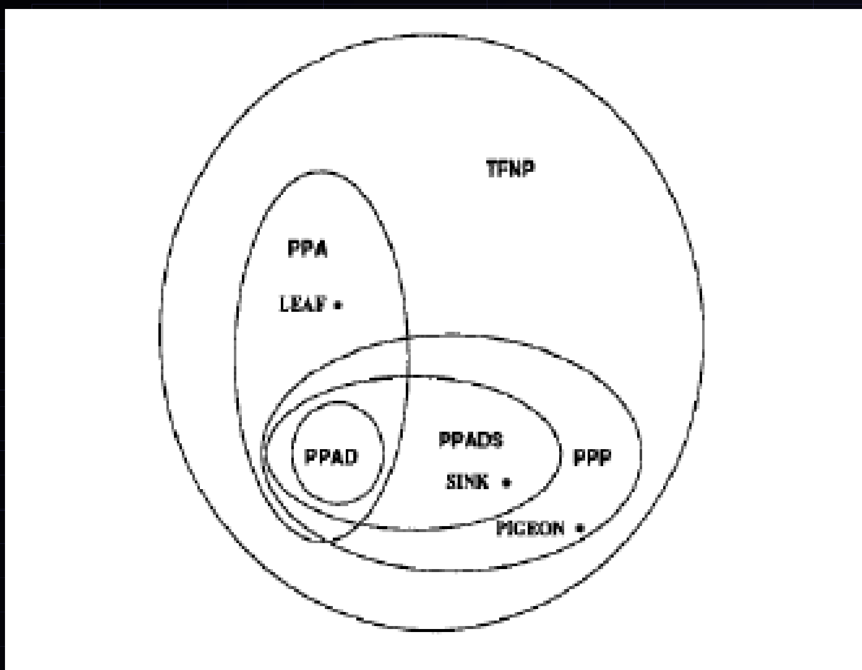
- If finding of a Nash equilibrium is NP-complete then NP = coNP!
- "Does it have two?" ← NP-Complete
- [Daskalakis,Goldberg,Papadimitriou, 2006] : Nash is PPAD-Complete

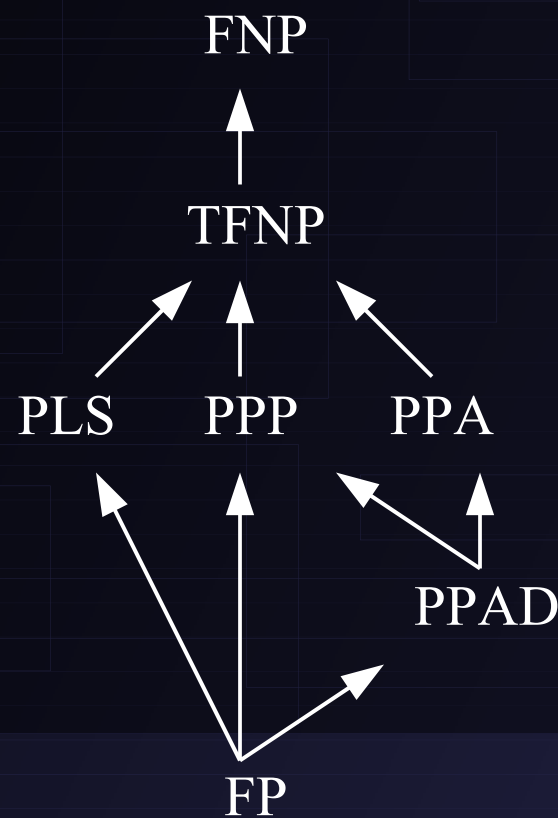


Back to PPAD

- Complete Problems:
  - End of line
  - Finding a Nash equilibrium
  - Finding a fixed point (Brower)
  - Kakutani
  - Price equilibrium

Hierarchy


TFNP
PPA
LEAF •
PPAD
PPADS
SINK •
PPP
PIGEON •


FNP
TFNP
PLS
PPP
PPA
PPAD
FP

References

- Algorithmic Game Theory , Nisan
- On the complexity of the parity argument and other inefficient proofs of existence, Papadimitriou, 1994
- How easy is local search?, Johnson, Yiannakakis, Papadimitriou