Bounded team private-information games:

NEXPTIME-complete \cite{Peterson, Reif, Azhar-C&M 2001}
- Dependency QBF (DQBF):
  $\forall x_1 : \forall x_2 : \exists y_1(x_1) : \exists y_2(x_2)$: CNF formula
  
  \begin{itemize}
  \item black player \hspace{2em} white 1
  \item only sees $x_1$ \hspace{2em} \textcolor{green}{\text{white player 2}} \hspace{2em} only sees $x_2$ variables
  \end{itemize}

  - can white force a win? (satisfied formula)
  - only one round! (multiple rounds don't help)
  - $\in$NEXPTIME: guess $y_1 \land x_1 \land y_2 \land x_2$ (exponential)

- Bounded Team Private Constraint Logic (TPCL)
  with 3 players & planar graph
  - moves must be known legal with visible information
  - $\in$NEXPTIME: guess strategy for all possible
    visible information (exp. # states)

- reduction from DQBF
- first black sets all vars. (white twiddles thumbs)
- chosen activates $\rightarrow$ long chain (black threat)
- white players set their vars.
- chosens $\rightarrow$ unlock all $\rightarrow$ formula activation
- white wins (just in time) if formula satisfied
Unbounded team private-information games: undecidable \cite{HearnDemaine} (based on work by Peterson & Reif – FOCS 1979)

Team Computation Game:
- instance = space-k algorithm/Turing machine (memory/tape initially blank)
- black move = run alg./machine for k more steps; output (if any) determines winner;
  else set $x_1, x_2 \in \{A, B\}$
- white i sees only $x_i$ & can set only $m_i$
- white i move = set $m_i$
- does white have a forced win?

- reduction from Halting problem: does this Turing machine ever terminate?
- build $O(1)$-space algorithm to check white players play valid computation history → halt of the form \#$ state_0 \#$ state_1 \#$ \ldots \#$ halt\_state
- in fact each white player must have in mind 2 pointers A&B into common history
- $x_i = A$ asks for character at A & advance A
- but white players have no idea of other's A/B
- alg. maintains whether 1's $x_1$ state = 2's $x_2$ state (identical from \#$ with (x_1, x_2)$ moves since)
- then if \((x_1, x_2)\) moves until \(1\) reports \(\Rightarrow 1 \ x_1\) ahead one
  and if \((x_1, x_2)\) moves then continue,
  then check this \(1\) state valid transition from \(a\)'s
  & vice versa with \(1 \Rightarrow a\) \(\Rightarrow O(1)\) space!
- white strategies must work for all possible
  black moves \(\Rightarrow\) valid computation history

- Team Formula Game:
  - black sets \(X\) such that \(F(X, X', Y_1, Y_2)\) (else lose)
  - black wins if \(G(x)\) \(\uparrow F \Rightarrow \neg F'\)
  - black sets \(X'\) such that \(F'(X, X')\) (else lose)
  - white 1 sets \(Y_1\), seeing only \(Y_1\) & \(x_1 \in X\)
  - white 2 sets \(Y_2\), seeing only \(Y_2\) & \(x_2 \in X\)
  - standard reduction from Team Computation Game

- (Unbounded) TPCL with 3 players, planar graph
Parallelism & P-completeness:
  "Limits to Parallel Computation: P-Completeness Theory"

**NC** (Nick's Class, after Nick Pippinger)
= problems solvable in \( \log^{O(1)} n \) time using \( n^{O(1)} \) processors (PRAM)
  i.e. circuit of size \( n^{O(1)} \) & depth \( \log^{O(1)} n \)
  - e.g. sorting: compare all pairs, compute rank = sum of '<'s
    via binary tree
  \( \{0(\log n)\) time on \( O(n^2) \) proc.

**P-hard** = all problems \( \in \text{NC} \) can be reduced via NC algorithm to your problem
  Karp-style reduction
  \( \Rightarrow \in \text{NC} \) if NC \( \neq \) P

**P-complete** = \( \in \text{P} + \text{P-hard} \)
Base P-complete problems:

General Machine Simulation Problem:
given a sequential algorithm & time bound written in unary, does it say YES within? to make \( P \sim \) else \( \text{EXPTIME-complete} \)

Circuit Value Problem (CVP): [Ladner-SIGACT 1975]
given an (acyclic) Boolean circuit & input bits, is the output TRUE?

\[ \text{NAND CVP: } \text{just NAND gates} \]
\[ \text{NOR CVP: } \text{just NOR gates} \]
\[ \text{Monotone CVP: } \text{just AND \& OR gates} \]
\[ \text{Alternating monotone CVP: (AMCVP)} \]
\[ \text{input \to output path alternates AND/OR, starting \& ending with OR} \]
\[ \text{Fanin-2, fanout-2 AMCVP: (AM2CVP)} \]
\[ \text{all gates have in \& out degree 2 (allow outputs other than one of interest)} \]

Synchronous AM2CVP: (SAM2CVP)
all inputs to each gate have same depth

Planar CVP: planar circuit [Goldschlager-SIGACT 1977]
- use NAND crossover
- but: planar monotone \( \in \text{NC} \) [Yang-FOCS 1991]
- start & end with ORs
- reduce fan out to \( \leq 2 \) (also fanin \( \leq 2 \))
- make AND & OR alternate
- fanin 1 \( \rightarrow \) fanin 2
  (preserving alternation & start with OR)
- fanout 1 \( \rightarrow \) fanout 2
  by duplicating circuit \( x \rightarrow x \& x' \)
  & combining extra outputs
  (preserving alternation & end with OR)
- synchronization: \( n = \# \text{gates} \)
  - \( \frac{n}{2} \) copies of circuit
  - \( i \)th copy = levels \( 2i \& 2i + 1 \)
    inputs & ANDs ORs
  - OR takes inputs from \( i \)th copy,
    sends outputs to \( (i+1) \)st copy
    (determining ANDs by alternation)
  - AND in 0th copy become 0 input
    \( \Rightarrow \) level 0 = inputs
  - inputs fed to \( i \)th copy by input gadget
  - output in \( \frac{n}{2} \) copy
Bounded DCL:
- edges are active (just flipped) or inactive
- vertex active if its active incoming edges have total weight ≥ 2
- round = reverse unreversed edges pointing to active vertices
  (& these are the new active edges)

- P-complete for AND, SPLIT, OR graphs
  (but not necessarily planar)
- reduction from Monotone CVP
- even easier from SAM2CVP
Lexically first maximal independent set:
- as found by greedy algorithm: \( e \in P \)
  \[ S = \emptyset \]
  for \( v = 1, 2, \ldots, |V| \):
    if \( v \) not adjacent to \( S \):
      \[ S = S \cup \{ v \} \]
- decision question: is \( v \in S \) ?
- \( P \)-hard:
  - reduction from NOR CVP
  - number gates & inputs in topological order
  - drop edge orientations
  - add vertex \( \emptyset \) connected to all \( \emptyset \) inputs
  \[ \Rightarrow v \in S \iff v = \emptyset \text{ or gate } v \text{ outputs true} \]
- computing whether size \( \leq k \) also \( P \)-complete:
  - reduction from previous problem
  - connect \( v \) to \( n+1 \) new vertices, set \( k = n \)
  \[ \Rightarrow \text{size} \leq n \iff v \in S \]
- gap-producing reduction: \( n+1 \rightarrow n^c \)
  \[ \Rightarrow n^{1-\varepsilon} \text{-gap problem is } P \text{-complete} \]
  \[ \Rightarrow n^{1-\varepsilon} \text{-approximation is } P \text{-complete} \]
More P-complete problems: \cite{Greenlaw, Hoover, Ruzzo - book 1995}
- Game of Life: cell \((x,y)\) alive at unary time \(t\)?
- 1D cellular automata
- acyclic Generalized Geography
- is point \(p\) on \(k\)th convex hull of point set?
- multiset ranking: given \(k\) lists, is \(x\) the \(k\)th smallest in the union?
- \(a \mod b_1 \mod b_2 \ldots \mod b_n = 0\)?
- first fit decreasing bin packing \(\not\exists\) strongly P-complete
- LP with coefficients 0 & 1
- max flow \(\not\exists \) has fully RNC approx. scheme

OPEN:
- are two numbers relatively prime?
- \(a^b \mod c\)
- feasibility of LP with \(\leq 2\) variables per inequality
- maximum edge-weighted matching
  - pseudo RNC algorithm
- bounded-degree graph isomorphism