Geography Game

**Geography.** Alice names capital city $c$ of country she is in. Bob names a capital city $c'$ that starts with the letter on which $c$ ends. Alice and Bob repeat this game until one player is unable to continue. Does Alice have a forced win?

**Ex.** Budapest → Tokyo → Ottawa → Ankara → Amsterdam → Moscow → Washington → Nairobi → ...

**Geography on graphs.** Given a directed graph $G = (V, E)$ and a start node $s$, two players alternate turns by following, if possible, an edge out of the current node to an unvisited node. Can first player guarantee to make the last legal move?

**Remark.** Some problems (especially involving 2-player games and AI) defy classification according to P, EXPTIME, NP, and NP-complete.

---

9.1 PSPACE

**P.** Decision problems solvable in polynomial time.

**PSPACE.** Decision problems solvable in polynomial space.

**Observation.** $P \subseteq PSPACE$.

poly-time algorithm can consume only polynomial space
Binary counter. Count from 0 to \(2^n - 1\) in binary.

Algorithm. Use \(n\) bit odometer.

Claim. 3-SAT is in PSPACE.

Pf.

- Enumerate all \(2^n\) possible truth assignments using counter.
- Check each assignment to see if it satisfies all clauses.

Theorem. \(NP \subseteq PSPACE\).

Pf. Consider arbitrary problem \(Y\) in \(NP\).

- Since \(Y \leq 3\text{-SAT}\), there exists algorithm that solves \(Y\) in poly-time plus polynomial number of calls to 3-SAT black box.
- Can implement black box in poly-space.

Quantified Satisfiability

QSAT. Let \(\Phi(x_1, \ldots, x_n)\) be a Boolean CNF formula. Is the following propositional formula true?

\[
\exists x_1 \, \forall x_2 \, \exists x_3 \, \forall x_4 \, \ldots \, \forall x_{n-1} \, \exists x_n \, \Phi(x_1, \ldots, x_n)
\]

Assume \(n\) is odd

Intuition. Amy picks truth value for \(x_1\), then Bob for \(x_2\), then Amy for \(x_3\), and so on. Can Amy satisfy \(\Phi\) no matter what Bob does?

Ex. \((x_1 \lor x_2) \land (x_2 \lor \neg x_1) \land (\neg x_1 \lor \neg x_2 \lor x_3)\)

Yes. Amy sets \(x_1\) true; Bob sets \(x_2\); Amy sets \(x_3\) to be same as \(x_2\).

Ex. \((x_1 \lor x_2) \land (\neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3)\)

No. If Amy sets \(x_1\) false; Bob sets \(x_2\) false; Amy sets \(x_3\) false; if Amy sets \(x_1\) true; Bob sets \(x_2\) true; Amy loses.

QSAT is in PSPACE

Theorem. QSAT \(\in PSPACE\).

Pf. Recursively try all possibilities.

- Only need one bit of information from each subproblem.
- Amount of space is proportional to depth of function call stack.
9.4 Planning Problem

**Conditions.** Set \( C = \{ C_1, \ldots, C_n \} \).

**Initial configuration.** Subset \( c_0 \subseteq C \) of conditions initially satisfied.

**Goal configuration.** Subset \( c^* \subseteq C \) of conditions we seek to satisfy.

**Operators.** Set \( O = \{ O_1, \ldots, O_k \} \).
- To invoke operator \( O_i \), must satisfy certain prereq conditions.
- After invoking \( O_i \), certain conditions become true, and certain conditions become false.

**Planning Problem.** Is it possible to apply sequence of operators to get from initial configuration to goal configuration?

**Examples.**
- 15-puzzle.
- Rubik’s cube.
- Logistical operations to move people, equipment, and materials.

---

**Planning Problem: 8-Puzzle**

**Planning example.** Can we solve the 8-puzzle?

**Conditions.** \( C_{ij} \), \( 1 \leq i, j \leq 9 \). \( \leftarrow \) \( C_{ij} \) means tile \( i \) is in square \( j \)

**Initial state.** \( c_0 = \{ C_{11}, C_{22}, \ldots, C_{66}, C_{77}, C_{88}, C_{99} \} \)

**Goal state.** \( c^* = \{ C_{11}, C_{22}, \ldots, C_{66}, C_{77}, C_{88}, C_{99} \} \)

**Operators.**
- Precondition to apply \( O_i = \{ C_{11}, C_{22}, \ldots, C_{66}, C_{77}, C_{88}, C_{99} \} \).
- After invoking \( O_i \), conditions \( C_{77} \) and \( C_{99} \) become true.
- After invoking \( O_i \), conditions \( C_{77} \) and \( C_{99} \) become false.

**Solution.** No solution to 8-puzzle or 15-puzzle!
8-puzzle invariant. Any legal move preserves the parity of the number of pairs of pieces in reverse order (inversions).

![8-Puzzle State Transition]

3 inversions: 1-3, 2-3, 7-8

5 inversions: 1-3, 2-3, 7-8, 5-8, 5-6

3 inversions: 1-3, 2-3, 7-8

1 inversion: 7-8

Planning Problem: Binary Counter

Planning example. Can we increment an n-bit counter from the all-zeroes state to the all-ones state?

Conditions. $C_1, ..., C_n$. $C_i$ corresponds to bit $i = 1$

Initial state. $C_0 = \phi$. All 0s

Goal state. $c^* = \{C_1, ..., C_n\}$. All 1s

Operators. $O_1, ..., O_n$.

- To invoke operator $O_i$, must satisfy $C_1, ..., C_{i-1}$.
- After invoking $O_i$, condition $C_i$ becomes true.
- After invoking $O_i$, conditions $C_1, ..., C_{i-1}$ become false.

Solution. $\{\} \Rightarrow \{C_1\} \Rightarrow \{C_2\} \Rightarrow \{C_1, C_2\} \Rightarrow \{C_3\} \Rightarrow \{C_3, C_1\} \Rightarrow ...$

Observation. Any solution requires at least $2^n - 1$ steps.

Planning Problem: In Exponential Space

Configuration graph $G$.
- Include node for each of $2^n$ possible configurations.
- Include an edge from configuration $c'$ to configuration $c''$ if one of the operators can convert from $c'$ to $c''$.

PLANNING. Is there a path from $c_0$ to $c^*$ in configuration graph?

Claim. PLANNING is in EXPTIME.

Pf. Run BFS to find path from $c_0$ to $c^*$ in configuration graph.

Note. Configuration graph can have $2^n$ nodes, and shortest path can be of length $= 2^n - 1$.

Planning Problem: In Polynomial Space

Theorem. PLANNING is in PSPACE.

Pf.
- Suppose there is a path from $c_1$ to $c_2$ of length $L$.
- Path from $c_1$ to midpoint and from $c_2$ to midpoint are each $\leq L/2$.
- Enumerate all possible midpoints.
- Apply recursively. Depth of recursion $= \log_2 L$.

```java
boolean hasPath(c1, c2, L) {
    if (L > 1) return correct answer
    enumerate using binary counter

    foreach configuration c' {
        boolean x = hasPath(c1, c', L/2)
        boolean y = hasPath(c2, c', L/2)
        if (x and y) return true
    }
    return false
}
```
9.5 PSPACE-Complete

PSPACE. Decision problems solvable in polynomial space.

PSPACE-Complete. Problem Y is PSPACE-complete if (i) Y is in PSPACE and (ii) for every problem X in PSPACE, $X \leq_p Y$.


Theorem. PSPACE $\subseteq$ EXPTIME.

Pf. Previous algorithm solves QSAT in exponential time, and QSAT is PSPACE-complete.

Summary. $P \subseteq NP \subseteq PSPACE \subseteq EXPTIME$.

It is known that $P \neq EXPTIME$, but unknown which inclusion is strict; conjectured that all are.

PSPACE-Complete Problems

More PSPACE-complete problems.
- **Competitive facility location.**
- Natural generalizations of games.
  - Othello, Hex, Geography, Rush-Hour, Instant Insanity
  - Shanghai, go-moku, Sokoban
- Given a memory restricted Turing machine, does it terminate in at most k steps?
- Do two regular expressions describe different languages?
- Is it possible to move and rotate complicated object with attachments through an irregularly shaped corridor?
- Is a deadlock state possible within a system of communicating processors?

Competitive Facility Location

Input. Graph with positive edge weights, and target B.

Game. Two competing players alternate in selecting nodes. Not allowed to select a node if any of its neighbors has been selected.

Competitive facility location. Can second player guarantee at least B units of profit?

Yes if B = 20; no if B = 25.
**Claim.** COMPETITIVE-FACILITY is PSPACE-complete.

**Pf.**

- To solve in poly-space, use recursion like QSAT, but at each step there are up to $n$ choices instead of $2$.
- To show that it’s complete, we show that QSAT polynomial reduces to it. Given an instance of QSAT, we construct an instance of COMPETITIVE-FACILITY such that player 2 can force a win iff QSAT formula is true.

**Construction.** Given instance $\Phi(x_1, ..., x_n) = C_1 \land C_1 \land ... C_k$ of QSAT.

- Include a node for each literal and its negation and connect them.
- Choose $c \geq k+2$, and put weight $c^i$ on literal $x^i$ and its negation; set $B = c^{n-1} + c^{n-3} + ... + c^4 + c^2 + 1$.
- Ensures variables are selected in order $x_n, x_{n-1}, ..., x_1$.
- As is, player 2 will lose by $1$ unit: $c^{n-1} + c^{n-3} + ... + c^4 + c^2$.

Assume $n$ is odd.