Computational Complexity; slides 9, HT 2022
PSPACE-complete problems, alternating TMs

Prof. Paul W. Goldberg (Dept. of Computer Science, University of Oxford)

HT 2022

Coming up

Example of PSPACE-completeness (the "geography" game)
Then, alternative characterisation of PSPACE (as poly-time "alternating" TM). Recall general point that when there are various characterisations of a complexity class, it suggests the class is important.
Afterwards, polynomial hierarchy (classes between NP/co-NP and PSPACE)

The Formula Game

Players: Played by two Players $\exists$ and $\forall$

Board: A formula φ in conjunctive normal form with variables $X_1, \ldots, X_n$

Moves: Players take turns in assigning truth values to $X_1, \ldots, X_n$ in order.

That is, player $\exists$ assigns values to "odd" variables $X_1, X_3, \ldots$

Winning condition: After all variables have been instantiated, $\exists$ wins if the formula evaluates to true. Otherwise $\forall$ wins.

The Formula Game

Players: Played by two Players $\exists$ and $\forall$

Board: A formula φ in conjunctive normal form with variables $X_1, \ldots, X_n$

Moves: Players take turns in assigning truth values to $X_1, \ldots, X_n$ in order.

That is, player $\exists$ assigns values to "odd" variables $X_1, X_3, \ldots$

Winning condition: After all variables have been instantiated, $\exists$ wins if the formula evaluates to true. Otherwise $\forall$ wins.

Formula Game	
Input:	A CNF formula φ in the variables $X_1, \ldots, X_n$
Problem:	Does $\exists$ have a winning strategy in the game on φ?

Theorem. FORMULA GAME is PSPACE-complete.

Formula Game (extended version)

Board: A formula φ in conjunctive normal form with variables $X_1, \ldots, X_n$

- After players have chosen values for the variables, player $\forall$ chooses a clause
- Then player $\exists$ chooses a literal within that clause
- *exists* wins if the literal is satisfied, else $\forall$ wins

Example

$$\exists X_1 \forall X_2 \exists X_3 \forall X_4 \forall X_5 \Big((X_1 \vee 0 \vee \neg X_5) \ \wedge \ (\neg X_2 \vee 1 \vee \neg X_5) \ \wedge \ (X_2 \vee X_3 \vee X_4) \Big)$$

if $\exists$-player makes right choices, for all clauses C, there exists, within C, a satisfied literal

A generalised version of "Geography":

The board is a directed graph G and a start node $s \in V(G)$

Initially the token is on the start node.

Players take turns in pushing this token along a directed edge.

Edges may not be used more than once. If a player cannot move, he loses.

A generalised version of "Geography":

The board is a directed graph G and a start node $s \in V(G)$

Initially the token is on the start node.

Players take turns in pushing this token along a directed edge.

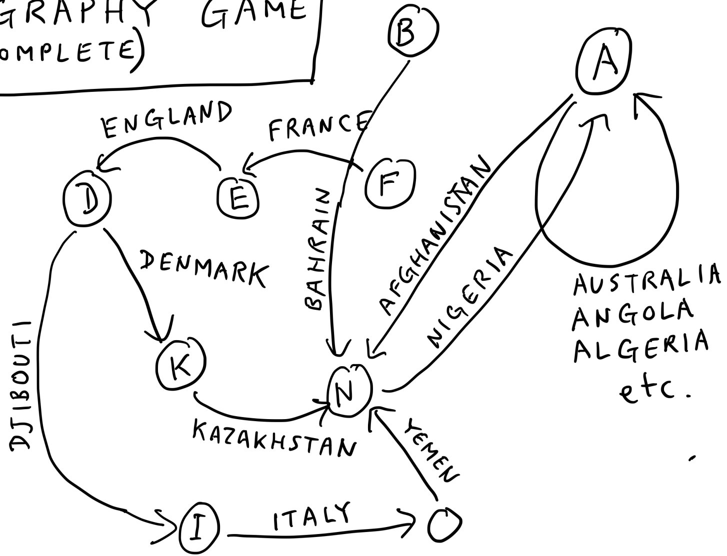
Edges may not be used more than once. If a player cannot move, he loses.

GEOGRAPHY	
Input:	Directed graph G, start node $s \in V(G)$
Problem:	Does Player 1 have a winning strategy?

Theorem. GEOGRAPHY is PSPACE-complete.

(Sipser Theorem 8.14)

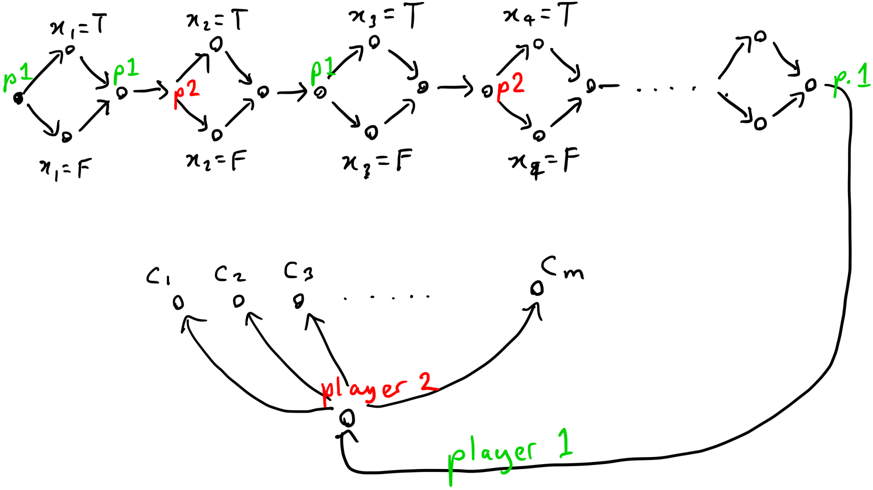
GEOGRAPHY GAME (INCOMPLETE)



$$\exists x_1 \forall x_2 \exists x_3 \forall x_4 \; \ldots . \; \phi(x)$$

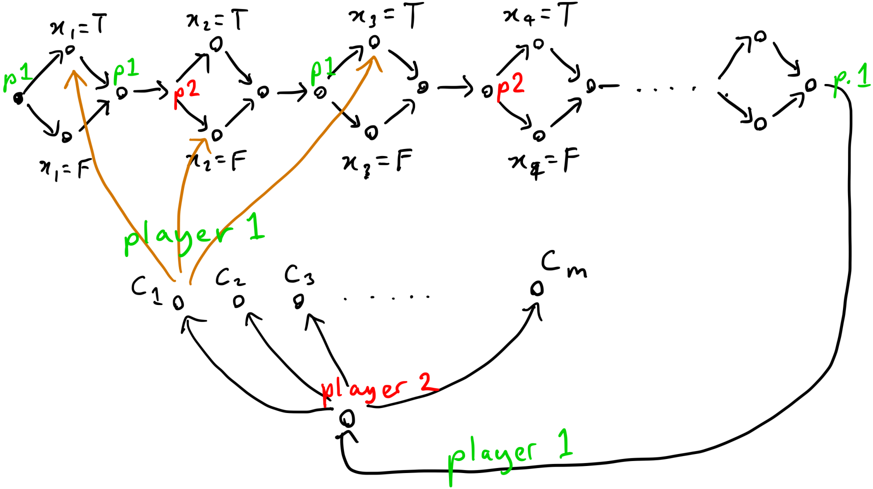
$$\phi = C_1 \wedge C_1 \wedge \ldots \wedge C_m$$

player 1 wants T
player 2 wants F
winner plays last



Suppose $C_1 = x_1 \vee \neg x_2 \vee x_3$

player 1 wants T
player 2 wants F
winner plays last



Next: Alternating Turing Machines

general idea: a class of automata whose languages are all the PSPACE languages. They can be a useful way to prove membership of problems in PSPACE.

They also give alternative characterisations of P, EXPTIME

Alternating Turing Machines

Definition. An alternating Turing machine M is a non-deterministic Turing accepter whose set of non-final states is partitioned into existential and universal states.

$Q_\exists$: set of existential states $\qquad Q_\forall$: set of universal states

Acceptance: Consider the computation tree $\mathcal{T}$ of M on w

Alternating Turing Machines

Definition. An alternating Turing machine M is a non-deterministic Turing accepter whose set of non-final states is partitioned into existential and universal states.

$Q_\exists$: set of existential states $\qquad$ $Q_\forall$: set of universal states

Acceptance: Consider the computation tree $\mathcal{T}$ of M on w

A configuration C in $\mathcal{T}$ is eventually accepting if

- C is an accepting stop configuration: an accepting leaf of $\mathcal{T}$
- $C = (q, p, w)$ with $q \in Q_\exists$ and there is at least one eventually accepting successor configuration in $\mathcal{T}$
- $C = (q, p, w)$ with $q \in Q_\forall$ and all successor configurations of C in $\mathcal{T}$ are eventually accepting

M accepts w if start configuration on w is eventually accepting.

Example: Alternating Algorithm for GEOGRAPHY

Input: Directed graph G $s \in V(G)$ start node.

Set $\textsc{Visited} := \{s\}$ Mark s as current node.

repeat

existential move: choose successor $v \notin \textsc{Visited}$ of current node s

if not possible **then** reject.

$\textsc{Visited} := \textsc{Visited} \cup \{v\}$

set current node $s := v$

universal move: choose successor $v \notin \textsc{Visited}$ of current node s

if not possible **then** accept.

$\textsc{Visited} := \textsc{Visited} \cup \{v\}$

set current node $s := v$

Note. This algorithm runs in alternating polynomial time.

Basic definitions of alternating time/space complexity

Recall $\mathcal{L}(M)$ denotes words (in Σ^*) accepted by M.

For function $T : \mathbb{N} \to \mathbb{N}$, an alternating TM is T time-bounded if every computation of M on input w of length n halts after $\leq T(n)$ steps.

Analogously for T space-bounded.

Basic definitions of alternating time/space complexity

Recall $\mathcal{L}(M)$ denotes words (in Σ^*) accepted by M.

For function $T : \mathbb{N} \to \mathbb{N}$, an alternating TM is T time-bounded if every computation of M on input w of length n halts after $\leq T(n)$ steps.

Analogously for T space-bounded.

For $T : \mathbb{N} \to \mathbb{N}$ a monotone increasing function, define

1. ATIME(T) as the class of languages $\mathcal{L}$ for which there is a T-time bounded k-tape alternating Turing accepter deciding $\mathcal{L}$, $k \geq 1$.
2. ASPACE(T) as the class of languages $\mathcal{L}$ for which there is a T-space bounded alternating k-tape Turing accepter deciding $\mathcal{L}$, $k \geq 1$.

Alternating Complexity Classes:

Time classes:

- APTIME $:= \bigcup_{d\in\mathbb{N}} \mathsf{ATIME}(n^d)$ — alternating poly time
- AEXPTIME $:= \bigcup_{d\in\mathbb{N}} \mathsf{ATIME}(2^{n^d})$ — alternating exp. time
- 2-AEXPTIME $:= \bigcup_{d\in\mathbb{N}} \mathsf{ATIME}(2^{2^{n^d}})$

Space classes:

- ALOGSPACE $:= \bigcup_{d\in\mathbb{N}} \mathsf{ASPACE}(d \log n)$
- APSPACE $:= \bigcup_{d\in\mathbb{N}} \mathsf{ASPACE}(n^d)$
- AEXPSPACE $:= \bigcup_{d\in\mathbb{N}} \mathsf{ASPACE}(2^{n^d})$

Examples.

GEOGRAPHY $\in$ APTIME.

MONOTONE CVP (coming up next) $\in$ ALOGSPACE.

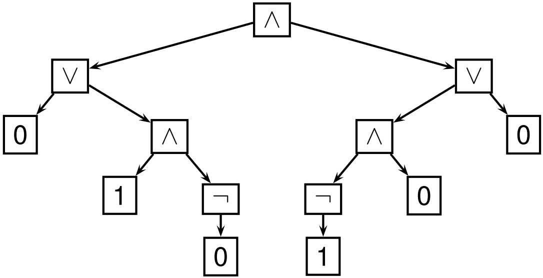
Similar alg.: CVP $\in$ ALOGSPACE.

Circuit. A connected directed acyclic graph with exactly one vertex of in-degree 0.

The vertices are labelled by:

label	no. of successors
$\wedge$	2
$\vee$	2
$\neg$	1
1	0
0	0

Example.



Evaluation of Circuits. A node v in a circuit C evaluates to 1 if

- v is a leaf labelled by 1
- v is a node labelled by $\vee$ and one successor evaluates to 1
- v is a node labelled by $\neg$ and its successor evaluates to 0
- v is a node labelled by $\wedge$ and both successors evaluate to 1

C evaluates to 1 if its root evaluates to 1.

Evaluation of Circuits. A node v in a circuit C evaluates to 1 if

- v is a leaf labelled by 1
- v is a node labelled by $\vee$ and one successor evaluates to 1
- v is a node labelled by $\neg$ and its successor evaluates to 0
- v is a node labelled by $\wedge$ and both successors evaluate to 1

C evaluates to 1 if its root evaluates to 1.

Circuit Value Problem.

CVP

Input:	Circuit C
Problem:	Does C evaluate to 1?

Monotone Circuit Value Problem.

Monotone CVP

Input:	Monotone circuit C without negation $\neg$.
Problem:	Does C evaluate to 1?

Monotone Circuit Value Problem

Input: Monotone circuit C with root s.

Set Current $:= s$.

while Current is not a leaf **do**

    if current node v is a $\vee$-node **then**

        existential move: choose successor v' of v

    else if current node v is a $\wedge$-node **then**

        universal move: choose successor v' of v

    end if

    set current node Current $:=$ v'

if Current is labelled by 1 **then** accept **else** reject.

Monotone Circuit Value Problem

Input: Monotone circuit C with root s.

Set Current $:= s$.

while Current is not a leaf **do**

if current node v is a $\vee$-node **then**

existential move: choose successor v' of v

else if current node v is a $\wedge$-node **then**

universal move: choose successor v' of v

end if

set current node Current $:=$ v'

if Current is labelled by 1 **then** accept **else** reject.

Note. This algorithm runs in alternating logarithmic space. Can be extended to general CVP

Non-determinism. A non-deterministic Turing accepter **is** an alternating TM (without universal states).

$$\mathcal{L} \in \mathsf{NP} \Longrightarrow \mathcal{L} \in \mathsf{APTIME}$$

Basic general properties of alternating TMs/complexity

Non-determinism. A non-deterministic Turing accepter **is** an alternating TM (without universal states).

$$\mathcal{L} \in \mathsf{NP} \implies \mathcal{L} \in \mathsf{APTIME}$$

Reductions. If $\mathcal{L} \in \mathsf{ATIME}(T)$ and $\mathcal{L}' \leq_p \mathcal{L}$ then $\mathcal{L}' \in \mathsf{ATIME}(T + f)$ where f is a polynomial.

Since GEOGRAPHY is PSPACE-complete and also in APTIME we have PSPACE $\subseteq$ APTIME

Basic general properties of alternating TMs/complexity

Non-determinism. A non-deterministic Turing accepter **is** an alternating TM (without universal states).

$$\mathcal{L} \in \mathsf{NP} \implies \mathcal{L} \in \mathsf{APTIME}$$

Reductions. If $\mathcal{L} \in \mathsf{ATIME}(T)$ and $\mathcal{L}' \leq_p \mathcal{L}$ then $\mathcal{L}' \in \mathsf{ATIME}(T + f)$ where f is a polynomial.

Since GEOGRAPHY is PSPACE-complete and also in APTIME we have $\mathsf{PSPACE} \subseteq \mathsf{APTIME}$

Complementation. Alternating Turing accepters are easily "negated".

Let M be an alternating TM accepting language $\mathcal{L}$

Let M' be obtained from M by swapping

- the accepting and rejecting state
- swapping existential and universal states.

Then $\mathcal{L}(M') = \overline{\mathcal{L}(M)}$

Example of complementation

Satisfiability for formulae $\varphi := \exists X_1 \forall X_2 \psi$, where ψ is quantifier-free:

Algorithm 1:

existential move. choose assignment $\beta : X_1 \mapsto 1$ or $\beta : X_1 \mapsto 0$.

universal move.

choose assignment $\beta := \beta \cup \{X_2 \mapsto 1\}$ and $\beta := \beta \cup \{X_2 \mapsto 0\}$.

if β satisfies ψ **then** accept **else** reject.

Example of complementation

Satisfiability for formulae $\varphi := \exists X_1 \forall X_2 \psi$, where ψ is quantifier-free:

Algorithm 1:

existential move. choose assignment $\beta : X_1 \mapsto 1$ or $\beta : X_1 \mapsto 0$.

universal move.

choose assignment $\beta := \beta \cup \{X_2 \mapsto 1\}$ and $\beta := \beta \cup \{X_2 \mapsto 0\}$.

if β satisfies ψ **then** accept **else** reject.

Its complement is defined as:

Algorithm 2:

universal move. choose assignment $\beta : X_1 \mapsto 1$ or $\beta : X_1 \mapsto 0$.

existential move.

choose assignment $\beta := \beta \cup \{X_2 \mapsto 1\}$ or $\beta := \beta \cup \{X_2 \mapsto 0\}$.

if β satisfies ψ **then** reject **else** accept.

Example of complementation

Satisfiability for formulae $\varphi := \exists X_1 \forall X_2 \psi$, where ψ is quantifier-free:

Algorithm 1:

existential move. choose assignment $\beta : X_1 \mapsto 1$ or $\beta : X_1 \mapsto 0$.

universal move.

choose assignment $\beta := \beta \cup \{X_2 \mapsto 1\}$ and $\beta := \beta \cup \{X_2 \mapsto 0\}$.

if β satisfies ψ **then** accept **else** reject.

Its complement is defined as:

Algorithm 2:

universal move. choose assignment $\beta : X_1 \mapsto 1$ or $\beta : X_1 \mapsto 0$.

existential move.

choose assignment $\beta := \beta \cup \{X_2 \mapsto 1\}$ or $\beta := \beta \cup \{X_2 \mapsto 0\}$.

if β satisfies ψ **then** reject **else** accept.

Note: Algorithm 1 accepts φ iff Algorithm 2 rejects φ

Theorem

APTIME = PSPACE

Proof.

1. We have already seen that GEOGRAPHY $\in$ APTIME.
   As GEOGRAPHY is PSPACE-complete,

$$\text{PSPACE} \subseteq \text{APTIME}.$$

Alternating vs. Sequential Time and Space

Theorem

APTIME = PSPACE

Proof.

1. We have already seen that GEOGRAPHY $\in$ APTIME.
   As GEOGRAPHY is PSPACE-complete,

$$\text{PSPACE} \subseteq \text{APTIME}.$$

2. APTIME $\subseteq$ PSPACE follows from the following more general result.

   Lemma. For $f(n) \geq n$ we have

$$\text{ATIME}(f(n)) \subseteq \text{DSPACE}(f(n))$$

   To prove this, explore configuration tree of ATM of depth $f(n)$

Alternating vs. Sequential Time and Space

Theorem.

1. For $f(n) \geq n$ we have

$$\mathrm{ATIME}(f(n)) \subseteq \mathrm{DSPACE}(f(n)) \subseteq \mathrm{ATIME}(f^2(n))$$

2. For $f(n) \geq \log n$ we have $\mathrm{ASPACE}(f(n)) = \mathrm{DTIME}(2^{\mathcal{O}(f(n))})$

(see Sipser Thm. 10.21)

(c.f. Savitch's theorem)

Lemma. For $f(n) \geq n$ we have $\mathsf{DSPACE}(f(n)) \subseteq \mathsf{ATIME}(f^2(n))$.

Proof. Let $\mathcal{L}$ be in $\mathsf{DSPACE}(f(n))$ and M be an $f(n)$ space-bounded TM deciding $\mathcal{L}$.

On input w, M makes at most $2^{\mathcal{O}(f(n))}$ computation steps.

Alternating Algorithm. *Reach*(C_1, C_2, t)

Returns 1 if C_2 is reachable from C_1 in $\leq 2^t$ steps.

if $t = 0$
  if $C_1 = C_2$ or $C_1 \vdash C_2$ **do** return 1 **else** return 0
else
  existential step. choose configuration C with $|C| \leq \mathcal{O}(f(n))$
  universal step. choose $(D_1, D_2) = (C_1, C)$ or $(D_1, D_2) = (C, C_2)$
  return $Reach(D_1, D_2, t - 1)$.

Alternating vs. Sequential Time and Space

Theorem.

1. For $f(n) \geq n$ we have

$$\mathsf{ATIME}(f(n)) \subseteq \mathsf{DSPACE}(f(n)) \subseteq \mathsf{ATIME}(f^2(n))$$

2. For $f(n) \geq \log n$ we have $\mathsf{ASPACE}(f(n)) = \mathsf{DTIME}(2^{\mathcal{O}(f(n))})$

(see Sipser Thm. 10.21)

Alternating vs. Sequential Time and Space

Theorem.

1. For $f(n) \geq n$ we have

$$\text{ATIME}(f(n)) \subseteq \text{DSPACE}(f(n)) \subseteq \text{ATIME}(f^2(n))$$

2. For $f(n) \geq \log n$ we have $\text{ASPACE}(f(n)) = \text{DTIME}(2^{\mathcal{O}(f(n))})$

(see Sipser Thm. 10.21)

Corollaries.

- ALOGSPACE = PTIME
- APTIME = PSPACE
- APSPACE = EXPTIME

To conclude

Alternating TMs give us a different characterisation of complexity classes we have seen.

Next: the polynomial hierarchy: a sequence of classes that are intermediate between NP and PSPACE. They represent some important problems that are "above" NP and "below" PSPACE