COMP3630/6360: Theory of Computation
Semester 1, 2022
The Australian National University

Alternating Time

This Lecture Covers Material Beyond the Textbook

- APTIME
- APTIME vs PSPACE

The Geography Game

Rules of Geography given a designated starting city (e.g. London)

1. Player 1 names a city that begins with the last letter of the designated city (e.g. Newcastle) and makes this the designated city.
2. Player 2 names a city that begins with the last letter of the city named by player 2 (e.g. Edinburgh) and makes this the designated city, continue with rule 1

Winning Conditions.

- The game is lost by the player that cannot name a city . . .
- and won by the other player.

Question.

Does Player 1 have a winning strategy (i.e. can always win irrespective of the moves of player one)?

The Proof Game

Background.

- A formula A is *provable* if there is a proof rule with conclusion A, all of whose premisses are provable (e.g $\frac{B \to A \qquad B}{A}$)

Rules of the Proof Game for a given designated formula A_0:

1. Player 1 chooses a proof rule $A_1, \ldots, A_n / A_0$ whose conclusion is the designated formula
2. Player 2 chooses a premiss A_i of the rule, and makes A_i the designated formula, continue with rule 1

Winning conditions.

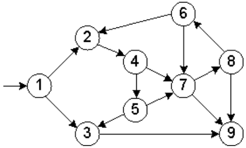
- the player who cannot move loses the game
- infinite plays are lost by Player 1

Question.

Does Player 1 have a winning strategy (i.e. can always win irrespective of the moves of player one) so that A is provable?

Generalised Geography.

Replace *cities* with *directed graph*:



Winning Conditions.

- who cannot move, looses
- Player 2 wins infinite plays

Rules.

- the indicated node is the designated node
- Player 1 chooses a successor of the designated node which is the new designated node
- Player 2 chooses a successor of the designated node which is the new designated node, continue with rule 1.

Question.

What is the complexity that – given graph G with designated initial node – of determining whether Player 1 has a winning strategy?

Mapping

From Geography to Generalised Geography. Construct a graph where:

- the nodes are the names of cities
- there is an edge between city 1 and city 2 if the name of city 2 begins with the last letter of the name of city 1

From Proof to Generalised Geography. Construct a graph where:

- nodes are either formulae, or proof rules
- there is an edge between a formula node A and a proof rule node $A_1, \ldots, A_n/A_0$ if $A = A_0$
- there is an edge between a proof rule node $A_1, \ldots, A_n/A_0$ and a formula node A if $A = A_1$, some $1 \leq i \leq n$.

Winning Strategies

For Player 1 to win from starting node n:

- there *exists* a move such that for *all* moves of player 2 to node n' ...
- Player 1 has a winning strategy from node n'

Pattern for winning strategy:

- *existential choice* for player 1
- *universal choice* for player 2

Nondeterministic Machines

Complexity Class NP. Have non-deterministic machine

- where every run takes at most polynomially many steps
- there *exists* an accepting sequence of IDs

Complexity Class co-NP. Have non-determninistic machine

- where every run takes at most polynomially many steps
- *every* sequence of IDs is accepting

Alternating Turing machines *combine* existential and universal runs

Alternating Turing Machines

Definition. An *alternating Turing machine* is a non-deterministic Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ where additionally $Q = Q_e \cup Q_u$ is partitioned into a set of Q_e of *existential states* and Q_u of *universal* states.

Instantaneous Descriptions (IDs)

- are defined as for non-determnninistic machines, and contain tape content, head position, and state
- the *transition relation* $I \vdash J$ between IDs is defined as for non-deterministic machines
- an ID is *existential* if the state is existential, and *universal*, if the state is universal.

Q. What about acceptance . . . ?

Acceptance

Informally. An ATM M accepts string w if there is a *finite* tree whose nodes are IDs and

- the root node is the initial ID (w on tape, state q_0)
- every existential ID E has *one* child J with $E \vdash J$
- every universal ID U has *all* IDs J with $U \vdash J$ as children
- all leaf nodes are universal.

Informal Example. Generalised Geography

- On tape: Graph and designated node
- two states, q_0 (initial and *existential*) and q_1 (*universal*)
- from q_i to $q_(1 - i)$: replace designated node by successor in graph

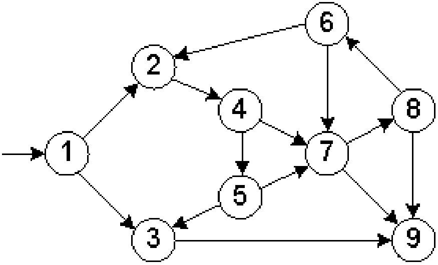
(omitting intermediate states that are needed to change designated node)

Idea.

- q_0 are the states where player 1 moves, and q_1 is a state of player 2
- universal leaf nodes = player two can't move and player 1 wins

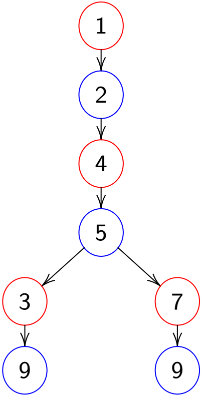
Informal Example.

Geography Graph.



- existential states are red
- universal states are blue

Winning Strategy.



ATM Acceptance, Formally

Definition. Given an ATM M and string w, then the set of *accepting IDs* is the *least* set A of IDs such that

- for every existential ID E there is an ID $I \in A$ with $E \vdash I$
- for every universal ID $U \in A$ and every ID I with $U \vdash I$ we have $I \in A$.

That is, every existential ID in A needs to have one successor in A, and every universal ID in A needs to have all successors in A.

What about accepting states?

- An existential ID with no successors is never accepting
- A universal ID with no successors is accepting

(Hence accepting states are not needed, and we just mention the for compatibility with the original definition)

How about infinite loops?

- in the tree-definition we have insisted on *finite* trees
- here, *least* set makes sure that infinite loops never accept.

First Algorithm for Geography

```
Algorithm Geography (Graph G, start node n):
  let cur = n;
  forever do {
    existentially guess (a successor node e of cur);
    // if this is not possible, we don't accept

    universally guess (a successor node u of e);
    // if there are none, we accept

    let cur := u;
  }
```

Comments.

- This shows (modulo a translation to TM) that Geography is solvable using an ATM
- However the number of steps that this ATM takes is possibly infinite if there are loops in the graph

Restrictions of ATMs

Definition. An ATM is *polytime bounded* if there exists a polynomial p such every sequence of IDs from an initial ID (q_0, w) is at most $p(|w|)$ steps long.
The class *APTime* of *alternating polytime languages* is the class of languages accepted by an ATM that is polytime bounded.

Observation.

- NP $\subseteq$ APTime (just make every state existential)
- co-NP $\subseteq$ APTime (just make every state universal)

Reductions. If L is polytime red'e to L' and $L' \in \mathrm{APTime}$ then so is L.

(In the combined ATM, make every state of the transducer that reduces L to L' either universal or existential. As the trasducer is deterministic, this doesn't matter.)

Example: Geography

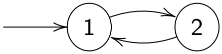
Earlier Algorithm.

```
Algorithm Geography (Graph G, start node n):
  let cur = n;
  forever do {
    existentially guess (a successor node e of cur);
    // if this is not possible, we don't accept

    universally guess (a successor node u of e);
    // if there are none, we accept

    let cur := u;
  }
```

- not necessarily terminating, e.g.



- let alone in polynomially many steps!

Geography, Terminating

Idea. Existential nodes don't need to repeat

```
Algorithm Geography2 (Graph G, start node cur):
  let seen := { cur };
  forever do {  // Player 1:
    existentially guess (cur := unseen successor of cur)
    // if this fails, we terminate and don't accept

    // Player 2:
    universally guess (cur := successor of cur);
    // if this fails, we terminate and accept

    seen := seen u { cur } // never visit twice
  }
```

Geography in APTime.

- branches of tree at most twice as long as number of nodes in graph
- every computation path takes polynomially many steps

APTime vs co-APTime

Observation. Given polytime bounded ATM M, construct ATM M' by swapping existential and universal states

- then M' accepts w if and only if M rejects w
- requires that all runs are *terminating*

Corollary. co-APTime = APTime

Example. What are the strings accepted by the TM and it's dual version below

where * indicates any letter?

QBF Revisited

Idea. $\exists \rightsquigarrow$ existential guess, $\forall \rightsquigarrow$ universal guess

```
Algorithm evalqbf (formula A):
  case A of {
    A_1 \/ A_2: if (evalqbf A_1) = 1 then 1 else evalqnf(A_2)
    A_1 /\ A_2: if (evalqbf A_1) = 0 then 0 else evalqbf(A_2)
    ~ A_1 : return 1 - evalqnf (A_1)
    exists x A : existentially guess v in {0, 1};
                 evalqbf A [ x := v]
    forall x A : universally guess v in {0, 1};
                 evalqbf A [ x := v]
  }
```

where `A [x := v]` replaces all free occurrences of `x` in `A` with `v`.

Theorem.

- QBF is in APTime (by algorithm above)
- PSPACE $\subseteq$ APTime (as QBF is PSPACE-hard)

From APTime to PSpace

Theorem. APTime $\subseteq$ PSpace.
Proof (Idea). Depth-first search simulates ATM M on standard TM.

```
Algorithm ATMaccept (ATM-ID I):
  if (I is existential) {
    let accept := false;
    foreach J with I |- J { accept := accept \/ ATMaccept(J); }
    return (accept);
  } else if (I is universal) {
    let accept := true;
    foreach J with I |- J { accept := accept /\ ATMaccept(J); }
    return (accept);
  }
```

For polynomial bound p and input of length n:

- recursion depth is polynomial as M is APTime
- argument in recursive calls is of size $O(p(n))$

So space in $O(p^2(n))$.

Why PSPACE is "harder" than NP

- True NP instances can (at least) be easily verified:
  Provide witness = accepting-ID-path of NTM.
  Has polynomial length and can be verified in polynomial time.
- Example: Powerful Prover (in PSPACE) provides satisfying assignment for ϕ.
  Verifier can check correctness in polytime.
  Not possible for unsatisfiable ϕ.
- Example: Composite numbers:
  Prover provides factors. Verifier checks by multiplication.
  Remarkably also possible for Primes (Primes $\in$ NP).
- No short proofs=certificates for PSPACE complete problems:
  Prover even of unlimited power cannot convince poly-time verifier that some language is in some class (if PSPACE $\neq$ NP)
- Examples: Prover cannot convince Verifier that white (or black) has winning strategy in many zero-sum games such as n $\times$ n Chess or Go, or that QBF formula is true.

What we know and don't know

Inclusions

$$P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXP$$

- but $P \neq EXP$, and don't know which inclusions are non-strict

Co-Classes.

$$NP \subseteq PSPACE \text{ and } co-NP \subseteq PSPACE$$

- but we don't know whether NP = co-NP (however this would follow if P = NP)

Equalities

$$PSPACE = NPSPACE = APTIME$$