COMP4141 Theory of Computation

Alternation

Ron van der Meyden

CSE, UNSW

Revision: 2014/05/14

(Credits: K Engelhardt, M Sipser, C Papadimitriou)

Motivation

Recall

- *nondeterministic* TM's, accept if some branch of the computation tree accepts (used for **NP**)
- *co-nondeterministic* TM's, accept if all branches of the computation tree accepts (used for **coNP**)

Alternating Turing machines combine the two acceptance modes into one type of machine ......

ATMs

Definition

An *alternating Turing Machine* (*ATM*)

$$M = (Q, \ell, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$$

consists of:

- Q, a finite set of states
- $\ell : Q \longrightarrow \{\forall, \exists\}$, a labelling of states as *universal* or *existential*
- Σ, the input symbol alphabet, $\sqcup \notin \Sigma$
- $\Gamma \supseteq \Sigma$, the tape symbol alphabet, $\sqcup \in \Gamma$
- $\delta : Q \times \Gamma \longrightarrow 2^{Q \times \Gamma \times \{\mathrm{L},\mathrm{R}\}}$, the transition function
- $q_0 \in Q$, the start state
- $q_{accept} \in Q$, the accept state
- $q_{reject} \in Q$, the reject state

L(ATM)

An ATM M runs on a word w as if it were an NTM, creating a tree $T_M(w)$ of TM configurations (or a directed graph if we identify nodes with identical configurations).

Definition (ATM acceptance)

Mark nodes of $T_M(w)$, proceeding from leaves to the root, as follows:

1. every (accepting) configuration $xq_{\text{accept}}y$ is marked,
2. c is marked if $\ell(c) = \exists$ and c has *some* marked successor in $T_M(w)$, and
3. c is marked if $\ell(c) = \forall$ and c *all* successors in $T_M(w)$ are marked .

If the root of $T_M(w)$ (the initial configuration) is marked at the end of this process, then M accepts w.

Definitions of time and space complexity need not be changed.

Definition

Let $t : \mathbb{N} \longrightarrow \mathbb{N}$.

The *alternating time complexity class*, $\mathbf{ATIME}(t(n))$ is the collection of all languages that are decidable by an $\mathcal{O}(t(n))$ time ATM.

The *alternating space complexity class*, $\mathbf{ASPACE}(t(n))$ is the collection of all languages that are decidable by an $\mathcal{O}(t(n))$ space ATM.

$$\mathbf{AP} = \bigcup_{k \in \mathbb{N}} \mathbf{ATIME}(n^k)$$

$$\mathbf{APSPACE} = \bigcup_{k \in \mathbb{N}} \mathbf{ASPACE}(n^k)$$

$$\mathbf{AL} = \mathbf{ASPACE}(\log n)$$

Example

Recall that it is not know whether

$$\mathit{MIN\text{-}F} = \{ \langle \phi \rangle \mid \phi \text{ is a minimal Boolean formula } \}$$

is in **NP** or **P**. All we know so far is that $\overline{\mathit{MIN\text{-}F}} \in \mathbf{NP}^{\mathit{SAT}}$.
"On input $\langle \phi \rangle$:

1. Universally select a shorter formula ψ.
2. Existentially select an interpretation π of ϕ
3. Evaluate ϕ and ψ on π
4. *Accept* if the results differ and *reject* otherwise."

proves that $\mathit{MIN\text{-}F} \in \mathbf{AP}$.

Time, Space, and Alternation

Theorem

1. *$\mathbf{ATIME}(t(n)) \overset{(a)}{\subseteq} \mathbf{SPACE}(t(n)) \overset{(b)}{\subseteq} \mathbf{ATIME}(t(n)^2)$ if $t(n) \geq n$.*
2. *$\mathbf{ASPACE}(t(n)) = \mathbf{TIME}(2^{O(t(n))})$ if $t(n) \geq \log n$.*

Corollary

$\mathbf{AL} = \mathbf{P}$, $\mathbf{AP} = \mathbf{PSPACE}$, and $\mathbf{APSPACE} = \mathbf{EXPTIME}$.

Proof Ideas

1.(a) $\mathbf{ATIME}(t(n)) \subseteq \mathbf{SPACE}(t(n))$:

simulate the ATM, do DFS to do the marking.

This gives $\mathbf{SPACE}(t(n)^2)$, one $t(n)$ for the recursion depth and one $t(n)$ for the configuration size.

Reduce the latter to constant size by merely recording the choices made at each non-deterministic step, and recomputing the configuration from the start when backtracking.

Proof Ideas

1.(b) $\mathbf{SPACE}(t(n)) \subseteq \mathbf{ATIME}(t^2(n))$:

As in Savitch's theorem, the ATM uses binary search to determine whether the simulated TM could reach the accepting configuration in $2^{dt(n)}$ steps.

Rather than iterating over all possible midpoint configurations c_m, the ATM can use one big existential guess of $t(n)$ steps to construct c_m, (and then universally branch into two recursive calls).

Proof Ideas

2.("$\subseteq$") **ASPACE**$(t(n)) \subseteq$ **TIME**$(2^{O(t(n))})$:

similar to the proof of **PSPACE** $\subseteq$ **EXPTIME**, on input w, construct the directed acyclic computation graph containing the configurations of the simulated ATM; first mark the accepting nodes and then accept the word according to the definition of ATM acceptance.

Proof Ideas cont.

2.("$\supseteq$") $\mathbf{ASPACE}(t(n)) \supseteq \mathbf{TIME}(2^{O(t(n))})$:

Simulate a deterministic $2^{f(n)}$ machine M by an $\mathcal{O}(f(n))$ space ATM S, where $f(n) = \mathcal{O}(t(n))$. W.l.o.g. M accepts after erasing all tape content and moving all the way to the left.

On input w, M goes through a sequence of configurations. We can't even store a single one of those in S because it could be too long!

Let's encode the sequence of configurations as a $2^{f(|w|)} \times 2^{f(|w|)}$ grid where a cell contains either a tape symbol or a tape symbol and a state if that's where the head of M is.

S then uses a recursive procedure $R(i, j, d)$ where i, j are pointers of size $\lceil f(|w|) \rceil$ in binary to a cell and d is a cell contents, to check the bottom left grid cell $i = 2^{f(|w|)}$, $j = 1$ has contents $d = (q_{\text{accept}}, \sqcup)$.

Proof Ideas cont.

$R =$ "on input $\langle i, j, d\rangle$

1. if $i = 1$ then *accept* if d is consistent with the j'th cell of the initial configuration $q_0 w$ in this representation; else *reject*.
2. $\exists$-guess the contents a, b, c of the parent cells $[i-1, j-1]$, $[i-1, j]$, $[i-1, j+1]$
    1. if the parent cells with contents a, b, c shouldn't have the child cell $[i, j]$ with contents d, *reject*
    2. $\forall$-recurse into $R(i-1, j-1, a)$, $R(i-1, j, b)$, $R(i-1, j+1, c)$."

Even though the recursion depth is $2^{f(|w|)}$, the ATM can do with $\mathcal{O}(f(w))$ space because it need not store any of the arguments of previous recursive calls: R only ever returns *accept* or *reject*.

Alternative Proof of PSPACE $\subseteq$ AP

We show $QBF \in$ **AP** by giving a polynomial time ATM.
$M =$ "on input $Q_1 x_1 \dots Q_k x_k \phi(x_1, \dots, x_k)$ where the $Q_i \in \{\exists, \forall\}$

1. for $1 \leq i \leq k$
   1. Q_i-guess a value $v_i \in \{\text{FALSE}, \text{TRUE}\}$ for x_i
2. evaluate $\phi(v_1, \dots, v_k)$ and *accept* if it's true; *reject* otherwise."

Polynomial Time Hierarchy #1

Definitions

A Σ_i-*ATM* is an ATM whose runs begin at an $\exists$ state, and alternate, i.e., switch the $\{\exists, \forall\}$-type of state, at most $i-1$ times.

$\Sigma_i\mathbf{TIME}(f(n))$ is the class of languages Σ_i-ATMs can decide in $\mathcal{O}(f(n))$ time.

$\Sigma_i\mathbf{P} = \bigcup_k \Sigma_i\mathbf{TIME}(n^k)$.

Similarly, define Π_i-*ATMs*, $\Pi_i\mathbf{TIME}(f(n))$, and $\Pi_i\mathbf{P}$ by starting with $\forall$ instead of $\exists$.

$\mathbf{PH} = \bigcup_i \Sigma_i\mathbf{P} = \bigcup_i \Pi_i\mathbf{P}$ is called the *polynomial time hierachy*.

NB

$\mathbf{NP} = \Sigma_1\mathbf{P}$ *and* $\mathbf{coNP} = \Pi_1\mathbf{P}$.

Polynomial Time Hierarchy #2

Alternatively, we could have defined

Definitions

$\Delta_0\mathbf{P} = \Sigma_0\mathbf{P} = \Pi_0\mathbf{P} = \mathbf{P}$; and for $i \geq 0$

$$\Delta_{i+1}\mathbf{P} = \mathbf{P}^{\Sigma_i\mathbf{P}}$$
$$\Sigma_{i+1}\mathbf{P} = \mathbf{NP}^{\Sigma_i\mathbf{P}}$$
$$\Pi_{i+1}\mathbf{P} = \mathbf{coNP}^{\Sigma_i\mathbf{P}}$$