CS501: Formal Language Theory

L28: Alternation
The concept of a nondeterministic acceptor of a boolean query has a long and rich history, going back to various kinds of nondeterministic automata.

It is important to remember that these are fictitious machines: we suspect that they cannot be built.

**Open question:** \( \text{NP} \ ? = \text{co-NP} = \{ \overline{A} \mid A \in \text{NP} \} \)

If one could really build an NP machine, then one could, with a single gate to invert its answer, also build a co-NP machine.

From a practical point of view, the complexity of a problem \( A \) and its complement, \( \overline{A} \) are identical.
Nondeterminism

Value(ID) := Value(LeftChild(ID)) \lor Value(RightChild(ID))

weak communication pattern
The States of an Alternating TM are split into:

**Existential states** ($\exists$) and **Universal states** ($\forall$).

**Def:** An alternating TM in $\text{ID}_0$ accepts iff
1. $\text{ID}_0$ is in a final accepting state, or
2. $\text{ID}_0$ is in $\exists$ state and some next $\text{ID}'$ accepts, or
3. $\text{ID}_0$ is in $\forall$ state, exists at least one next ID, and all next ID’s accept.
Alternating TM’s have **random-access** read-only input.

The **index tape** can be written on and read. When the value $h$ in binary is on the index tape, the read head automatically scans bit $h$ of the input.
Def: \( \text{ASPACE}[s(n)] \) and \( \text{ATIME}[t(n)] \): the sets of problems accepted by alternating TM's using \( O(s(n)) \) tape cells and \( O(t(n)) \) time, respectively.

Main Alternation Thm: For \( s(n) \geq \log n \), and for \( t(n) \geq n \),

1. \( \text{ATIME}[(t(n))^{O(1)}] = \text{DSPACE}[(t(n))^{O(1)}] \)
2. \( \text{ASPACE}[s(n)] = \text{DTIME}[2^{O(s(n))}] \)

Cor: \( \text{ASPACE}[\log n] = \mathbb{P} \)

\( \text{ATIME}[n^{O(1)}] = \mathbb{PSPACE} \)
Def: Circuit Value Problem: \[ \text{CVP} = \{ C \mid \text{eval}(C) = 1 \} \]

Prop: \( \text{CVP} \in \mathbb{P} \).
**Def:** the monotone, circuit value problem (MCVP) is the subset of CVP in which no negation gates occur.

**Prop:** MCVP is recognizable in ASPACE[log $n$].

**Proof:** Let $G$ be a monotone boolean circuit. For $a \in V^G$, define “EVAL($a$),

1. if (InputOn($a$)) then **Accept**
2. if (InputOff($a$)) then **Reject**
3. if ($G \wedge (a)$) then universally choose child $b$ of $a$
4. if ($G \vee (a)$) then existentially choose child $b$ of $a$
5. Return(EVAL($b$))

A calls EVAL($r$). EVAL($a$) returns “**Accept** ” iff gate $a$ evaluates to one.

Space used for naming vertices $a, b$: $O(\log n)$. □
Def: The quantified satisfiability problem: QSAT = set of true formulas of form: $\psi = Q_1 x_1 Q_2 x_2 \cdots Q_r x_r \varphi$.

For any boolean formula $\varphi$ on variables $\overline{x}$,

$\varphi \in \text{SAT} \iff \exists \overline{x} (\varphi) \in \text{QSAT}$

$\varphi \in \overline{\text{SAT}} \iff \forall \overline{x} (\neg \varphi) \in \text{QSAT}$

Thus QSAT logically contains SAT and $\overline{\text{SAT}}$. 
Prop: QSAT is recognizable in ATIME[n].

Proof: Construct ATM, A, on input, \( \Phi \equiv \) 

\[ \exists x_1 \forall x_2 \ldots \exists x_{2k-1} \forall x_{2k} \bigwedge_{i=1}^{r} \bigvee_{j=1}^{s} \ell_{ij} \]

\[ b_1 b_2 \ldots b_{2k-1} b_{2k} i j \ell_{ij}(b_1, \ldots, b_{2k}) \]

Quantifiers:

- in \( \exists \) state, A writes a bit \( b_1 \) for \( x_1 \),
- in \( \forall \) state, A writes a bit \( b_2 \) for \( x_2 \), and so on.

Boolean operators:

- in \( \forall \) state, A chooses \( i \),
- in \( \exists \) state, A chooses \( j \)

Final state: accept iff \( \ell_{ij}(b_1, \ldots, b_{2k}) \) is true.

\[ A \text{ accepts } \Phi \iff \Phi \text{ is true.} \]
**Thm:** For any $s(n) \geq \log n$,

\[
\text{NSPACE}[s(n)] \subseteq \text{ATIME}[s(n)^2] \subseteq \text{DSPACE}[s(n)^2]
\]

**Proof:** \text{NSPACE}[s(n)] \subseteq \text{ATIME}[s(n)^2]:

Let $N$ be an \text{NSPACE}[s(n)] Turing machine.

Let $w$ be an input to $N$, $n = |w|$.

\[
w \in \mathcal{L}(N) \iff \text{CompGraph}(N, w) \in \text{REACH}
\]
\( w \in \mathcal{L}(N) \iff \text{CompGraph}(N, w) \in \text{REACH} \)

\[
P(d, x, y) \equiv \text{“In } \text{CompGraph}(N, w), \text{ dist}(x, y) \leq 2^d \text{”}
\]

\[
P(0, x, y) \equiv x = y \lor E(x, y)
\]

\[
P(d, x, y) \equiv \exists z \left( P(d - 1, x, z) \land P(d - 1, z, y) \right)
\]

1. **Existentially:** choose middle ID \( z \).
2. **Universally:** \((x, y) := (x, z) \text{ AND } (z, y)\)
3. Return(\(P(d - 1, x, y)\))

\[
T(d) = O(s(n)) + T(d - 1) = O(d \cdot s(n))
\]

\[
d = O(s(n))
\]

\[
T(d) = O((s(n))^2)
\]
Let $A$ be an ATIME[$t(n)$] machine, input $w$, $n = \|w\|$.

CompGraph($A, w$) has depth $c(t(n))$ and size $2^{c(t(n))}$, for some constant $c$.

Search this and/or graph systematically using $c(t(n))$ extra bits of space.
Evaluate computation graph of \( \text{ATIME}[t(n)] \) machine using \( t(n) \) space to cycle through all possible computations of \( A \) on input \( w \).
Example: \( \text{ATIME}[t(n)] \subseteq \text{DSPACE}[t(n)] \)
Thm: $\text{ASPACE}[s(n)] = \text{DTIME}[2^{O(s(n))}]$

**Proof:** $\text{ASPACE}[s(n)] \subseteq \text{DTIME}[2^{O(s(n))}]$:
Let $A$ be an $\text{ASPACE}[s(n)]$ machine, $w$ an input, $n = |w|$. $\text{CompGraph}(A(w))$ has size $\leq 2^{O(s(n))}$
Marking algorithm evaluates this in $\text{DTIME}[2^{O(s(n))}]$. 

\[ \begin{array}{c}
\text{CompGraph}(A(w)) \\
\text{Marking algorithm evaluates this in DTIME}[2^{O(s(n))}]
\end{array} \]
Let $M$ be $\text{DTIME}[2^{k(s(n))}]$ TM, $w$ an input, $n = |w|$. alternating procedure $C(t, p, a)$ accepts iff contents of cell $p$ at time $t$ in $M$'s computation on input $w$ is symbol $a$.

$C(t + 1, p, b)$ holds iff the three symbols $a_{-1}, a_0, a_1$ in tape positions $p - 1, p, p + 1$ lead to a “b” in position $p$ in one step of $M$’s computation.

$$C(t + 1, p, b) \equiv \bigvee (a_{-1}, a_0, a_1)^M b \bigwedge_{i \in \{-1, 0, 1\}} C(t, p + i, a_i)$$

Space needed is $O(\log 2^{k(s(n))}) = O(s(n))$.

Note that $M$ accepts $w$ iff $C(2^{k(s(n))}, 1, \langle q_f, 1 \rangle)$
This completes the proof of the Alternation Thm. □