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))
The states of an alternating Turing machine are split into: Existential states ($\exists$) and Universal states ($\forall$).

**Def:** An alternating TM in $ID_0$ accepts iff

1. $ID_0$ is in a final accepting state, or
2. $ID_0$ is in an $\exists$ state and some next $ID'$ accepts, or
3. $ID_0$ is in a $\forall$ state, has at least one next ID, and all next ID’s accept.
From now on assume that our Turing machines have a **random access** read-only input. There is an **index tape** which can be written on and read like other tapes. Whenever the value $h$, written in binary, appears on the index tape, the read head will automatically scan bit $h$ of the input.
**Def:** \( \text{ASPACE}[s(n)] \) and \( \text{ATIME}[t(n)] \) to be the set of problems accepted by alternating TM’s using \( O(s(n)) \) tape cells, \( O(t(n)) \) time, respectively, in any computation path on any input of length \( n \).

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

\[
\bigcup_{k=1}^{\infty} \text{ATIME}[(t(n))^k] = \bigcup_{k=1}^{\infty} \text{DSPACE}[(t(n))^k]
\]

\[
\text{ASPACE}[s(n)] = \bigcup_{k=1}^{\infty} \text{DTIME}[k^{s(n)}]
\]

**Cor:**

\[
\text{ASPACE}[\log n] = \text{P}
\]

\[
\text{ATIME}[n^{O(1)}] = \text{PSPACE}
\]
**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 ($G \land (a)$) then universally choose child $b$ of $a$
4. if ($G \lor (a)$) then existentially choose child $b$ of $a$
5. Return(EVAL($b$))

$M$ simply 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) is the set of true formulas of the following form:

\[ \Psi = Q_1 x_1 Q_2 x_2 \cdots Q_r x_r (\varphi) \]

For any boolean formula \( \varphi \) on variables \( \pi \),

\[
\begin{align*}
\varphi \in \text{SAT} & \iff \exists \pi (\varphi) \in \text{QSAT} \\
\varphi \notin \text{SAT} & \iff \forall \pi (\neg \varphi) \in \text{QSAT}
\end{align*}
\]

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.

$$A \text{ accepts } \Phi \iff \Phi \text{ is true.}$$

$\square$
**Thm:** For any $s(n) \geq \log n$,

$$\text{NSPACE}[s(n)] \subseteq \text{A TIME}[s(n)^2] \subseteq \text{DSPACE}[s(n)^2]$$

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

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

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

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

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

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

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

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

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

\[
T(d) = O((s(n))^2)
\]
\( \text{ATIME}[t(n)] \subseteq \text{DSPACE}[t(n)] \)

Let \( A \) be an \( \text{ATIME}[t(n)] \) machine, input \( w \), \( n = |w| \).

\( \text{CompGraph}(A, w) \) 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.

\[ \text{ATIME}[t(n)] \subseteq \text{DSPACE}[t(n)] \]
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, $n = |w|$. CompGraph($A(w)$) has size $\leq 2^{O(s(n))}$

Marking algorithm evaluates this in $\text{DTIME}[2^{O(s(n))}]$. 

![Diagram](image-url)
DTIME[$2^{O(s(n))}$] ⊆ ASPACE[$s(n)$]:

Let $M$ be 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.

$$C(t + 1, p, b) \equiv \bigvee_{(a_{−1}, a_0, a_1) \not\rightarrow_b} \bigwedge_{i \in \{−1,0,1\}} C(t, p + i, a_i)$$

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

Note that $M$ accepts $w$ iff $C(2^{k(s(n))}, 1, \langle q_f, 1 \rangle)$

<table>
<thead>
<tr>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{t}$</td>
<td>$\langle q_f, 1 \rangle$ $\square$ $\ldots$ $\square$ $\square$ $\square$ $\square$ $\square$</td>
</tr>
<tr>
<td>$\bar{t} + 1$</td>
<td>$a_{−1}$ $a_0$ $a_1$ $\square$ $\ldots$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$</td>
</tr>
</tbody>
</table>

$$C(t + 1, p, b) \equiv \bigvee_{(a_{−1}, a_0, a_1) \not\rightarrow_b} \bigwedge_{i \in \{−1,0,1\}} C(t, p + i, a_i)$$

This completes the proof of the Alternation Thm. □
<table>
<thead>
<tr>
<th>Class</th>
<th>FO ([n^{O(1)}])</th>
<th>FO(LFP)</th>
<th>SO(Horn)</th>
<th>FO((\log n)^{O(1)})</th>
<th>FO[\log n]</th>
<th>FO(CFL)</th>
<th>FO(TC)</th>
<th>FO(DTC)</th>
<th>FO(REGULAR)</th>
<th>FO(COUNT)</th>
<th>FO</th>
<th>LOGTIME Hierarchy</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXPTIME</td>
<td>SQ(n^{O(1)})</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSPACE</td>
<td>FO(2^{n^{O(1)}})</td>
<td>FO(PFP)</td>
<td>SO(TC)</td>
<td>SO(n^{O(1)})</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>FO(n^{O(1)})</td>
<td>FO(LFP)</td>
<td>SO(Horn)</td>
<td>FO((\log n)^{O(1)})</td>
<td>FO[\log n]</td>
<td>FO(CFL)</td>
<td>FO(TC)</td>
<td>FO(DTC)</td>
<td>FO(REGULAR)</td>
<td>FO(COUNT)</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>SAT</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>AC^1</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>sAC^1</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>NL</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>NC^1</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>ThC^0</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
<tr>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>AC^0</td>
<td>FO</td>
<td>LOGTIME Hierarchy</td>
</tr>
</tbody>
</table>