ALTERNATION

Algorithms & Complexity II
Avarikioti Zeta

March 17, 2014

Alternating Computation

Alternation: generalizes non-determinism, where each state is either "existential" or "universal":
Old: existential states ∃ New: universal states ∀

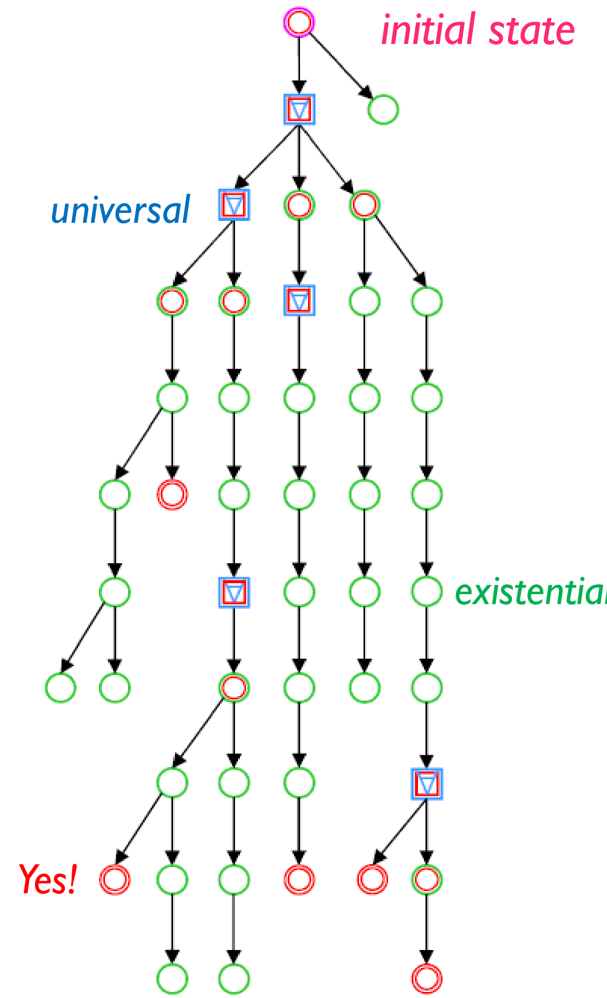
- **It alternates between N… (existential) and coN… (universal)**
- **Existential** state is accepting iff **<u>any</u>** of its child states is accepting **-** OR (without children→ rejects)
- **Universal** state is accepting iff **<u>all</u>** of its child states are accepting – AND (without children→ accepts)
- Alternating computation is a "tree"
- Computation accepts iff its initial state (configuration) is accepting

Alternating Turing Machines (ATMs) & Computation Tree

A nondeterministic TM $N = (K, \Sigma, \Delta, s)$ in which the set of states K is partitioned into two sets, $K = K_{AND} \cup K_{OR}$.

Let x be the input and consider the tree of computations of N on input x.

Each node in this tree is a configuration of the precise machine, and includes the step number of the machine.



Alternating Complexity Classes

ATIME(f(n)): *the class of all languages decided by an ATM, all computations of which on input x halt after at most f(lxl) steps.*

$$AP = \bigcup_{k>1} ATIME(n^k)$$

alternating polynomial time

ASPACE(f(n)): *the class of all languages decided by an ATM that uses no more than f(lxl) space on input x.*

$$AL = \bigcup_{k>1} ASPACE(\log n)$$

alternating logarithmic space

Alternating Complexity Classes

$\text{APSPACE} = \bigcup_{k>1} \text{ASPACE}(n^k)$ alternating polynomial space

$\text{AEXPTIME} = \bigcup_{k>1} \text{ATIME}(2^{n^k})$ alternating exponential time

$\text{AEXPSPACE} = \bigcup_{k>1} \text{ASPACE}(2^{n^k})$ alternating exponential space

Alternating Space/Time Relationships

Theorem: P $\subseteq$ NP $\subseteq$ AP

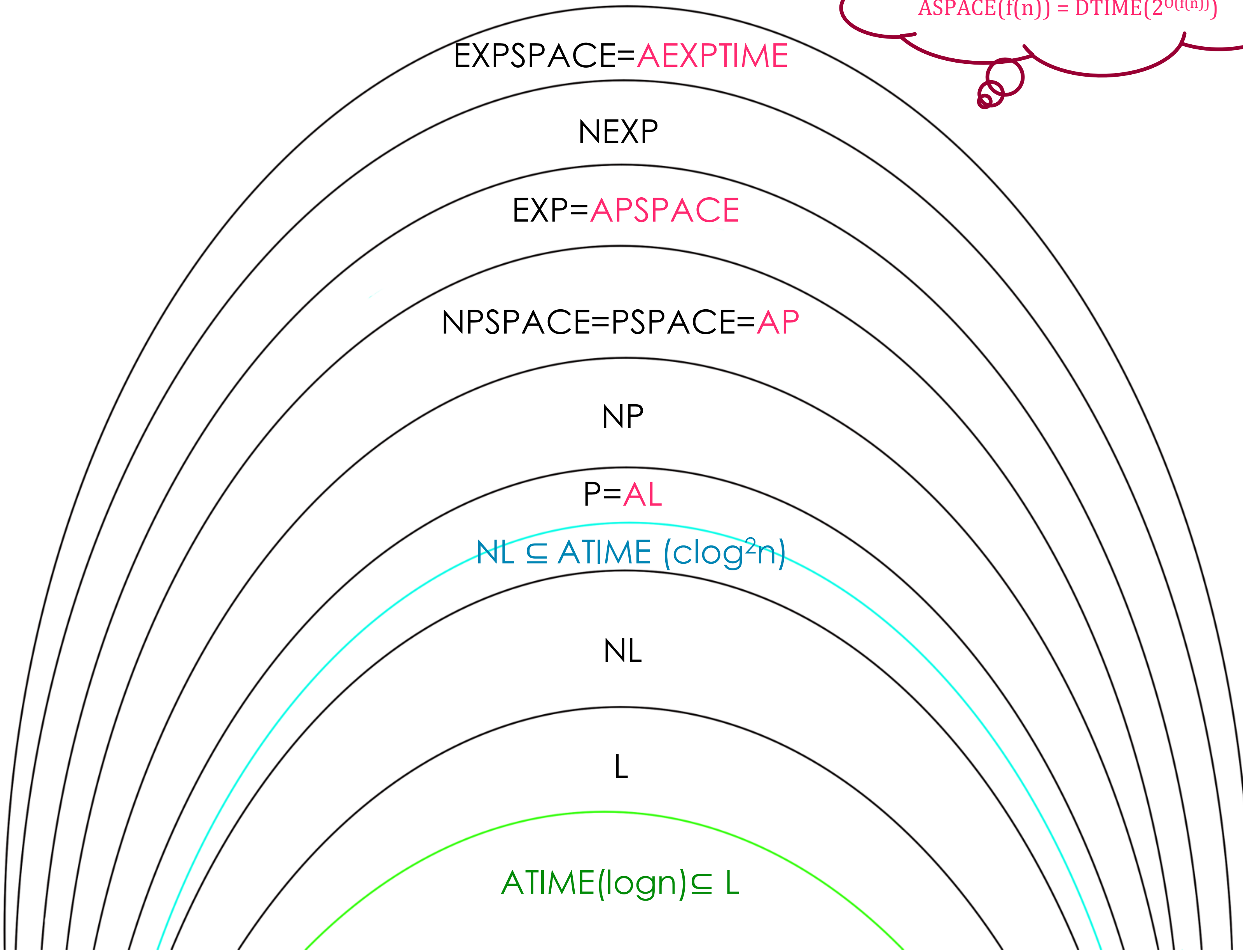
- ATIME(f(n)) $\subseteq$ DSPACE(f(n)) $\subseteq$ ATIME(f^2(n))
- PSPACE = NPSPACE $\subseteq$ APSPACE
- ASPACE(f(n)) = DTIME($2^{O(f(n))}$)
- **AL = P**
- **AP = PSPACE**
- APSPACE = EXPTIME
- AEXPTIME = EXPSPACE

Chandra, Stockmeyer & Kozen, 1981

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

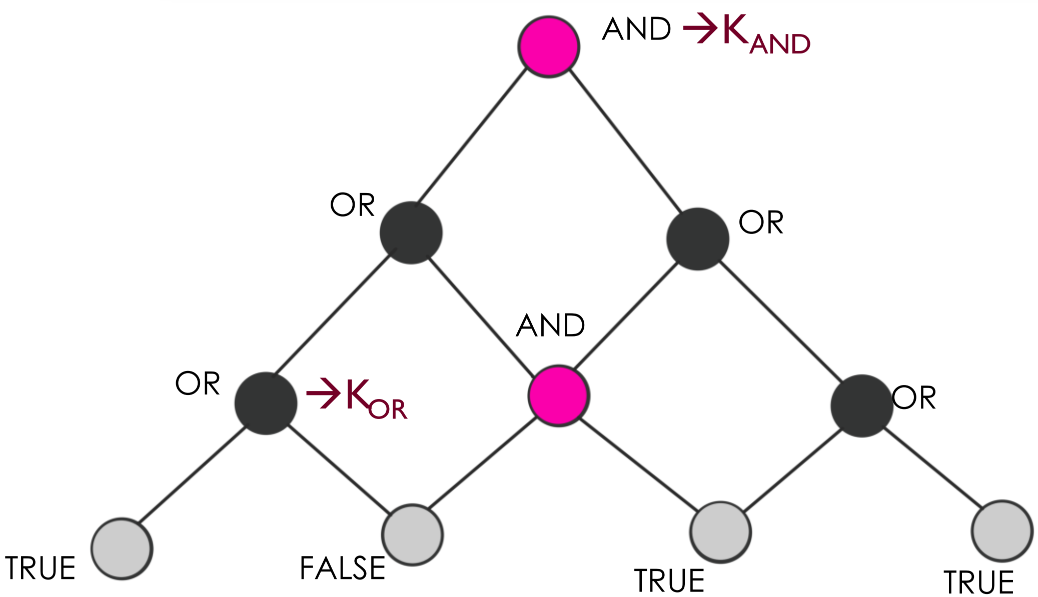
$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{ATIME}(f^2(n))$

$\mathrm{ASPACE}(f(n)) = \mathrm{DTIME}(2^{O(f(n))})$



AL=P

- MCVP is P-Complete (Ch.8)
- MCVP is AL-Complete
- Both classes are closed under reductions and they have the same complete problem



The monotone circuit value problem is composed of a set of gates $g_1, \ldots, g_n$ where each is:

- an AND gate, $g_i = g_j \wedge g_k$,
- an OR gate, $g_i = g_j \vee g_k$
- a constant value, gi = true or false.

We wish to compute the value of g_n.

AL=P (MCVP is AL-complete)

MCVP ∈ AL

- The input of ATM is a circuit
- The machine examines the output gate of the circuit:
  - ✓ If it is an AND gate, then the machine enters an AND state;
  - ✓ if the output gate is an OR gate, then it enters an OR state.
- The machine determines the two gates that are predecessors of the output and it nondeterministically chooses one.
- The same process is repeated at the new gate, till the input gate where the machine accepts if it is a true gate, and rejects if it is a false gate.

Only logarithmic space is needed.

AL=P (MCVP is AL-complete)

We will now show that any language in AL is reducible to MCVP.

Consider such a language, L, the corresponding Turing Machine, M, and an input, x. We shall construct a circuit such that it evaluates to True if and only if M accepts x.

- The gates of the circuit are all pairs of the form (C,i), where C is a configuration of M on input x, and i stands for the step number, an integer 0 and $|x|^k$
- There is an arc from gate (C1,i) to (C2,j) if and only if C2 yields in one step C1 and j = i + 1
- Gate type depends on the state:
  - ✓ If $C \in K_{OR}$ → OR gate
  - ✓ If $C \in K_{AND}$ → AND gate
  - ✓ If $C \in F$ (yes) → TRUE gate
  - ✓ If $C \in F$ (no) → FALSE gate
  - ✓ If C is s (initial state) → output gate

AP=PSPACE

Let φ be a Boolean expression with n variables then the expression $\exists x_1 \forall x_2 \ldots Q_n x_n$, where the quantifiers alternate is a QSAT expression.

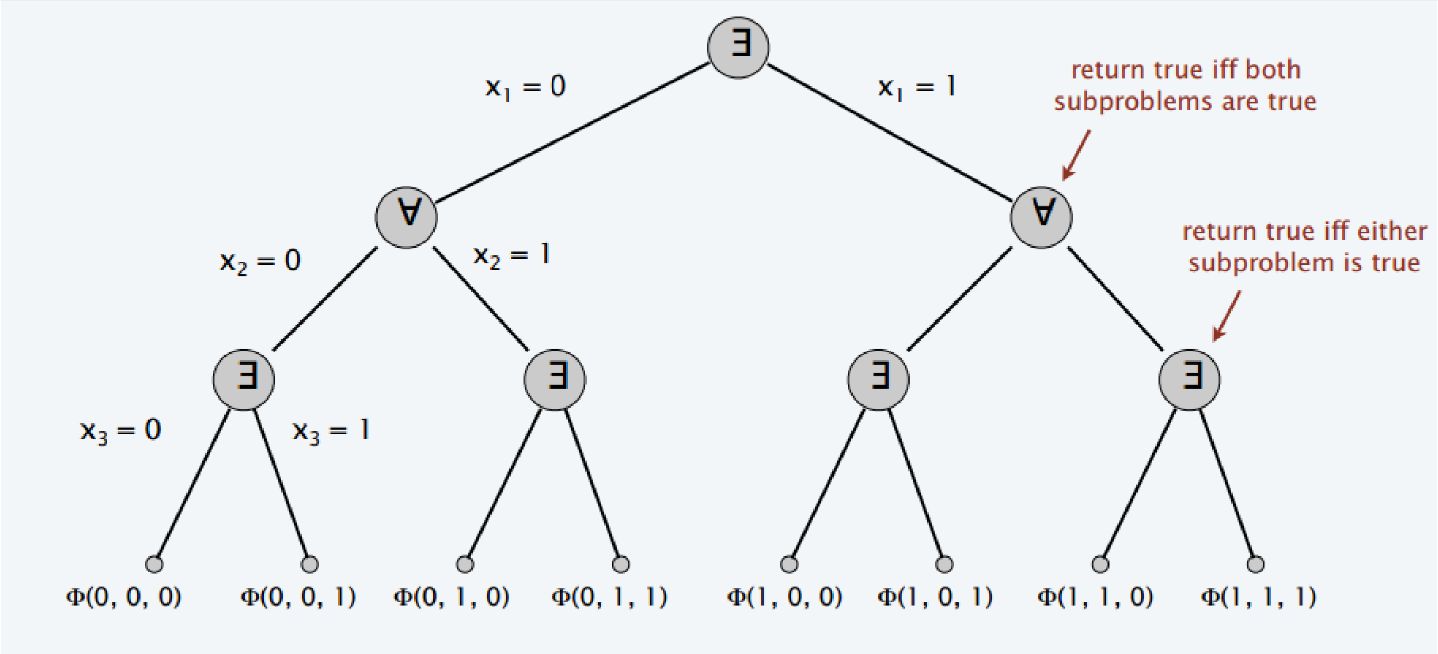
- QSAT is PSPACE-Complete
- QSAT is AP-Complete
- Both classes are closed under reductions and they have the same complete problem

AP=PSPACE
(QSAT is PSPACE-complete)

QSAT ∈ PSPACE

- All possible truth assignments of the variables can be arranged as the leaves of a full binary tree of depth n
- We turn this tree into a Boolean circuit, where all gates at the i-th level are AND if i is even and OR gates if I is odd.
- The input gate is true iff the truth assignment satisfies φ.

We can evaluate the circuit in O(n) space.



AP=PSPACE
(QSAT is PSPACE-complete)

We will now show that any language in PSPACE is reducible to QSAT

- For input x consider the configuration graph of M→ 2^m configurations, m= $O(n^k)$
- **Reachability method**:

  ψ_i (X,Y) is true ⇔ configuration Y can be reached from configuration X in ≤ 2^i steps, for i= 0,1,…,m
- **QSAT**: $x \in L \rightarrow \psi_m (A,B)$
- ψ_0 (A,B) can be written in DNF
- Bad idea: $\psi_{i+1} = \exists Z [\psi_i (A,Z) \wedge \psi_i (Z,B)]$
- **Savitch's trick:** $\psi_{i+1} = \exists Z \forall X \forall Y [((X=A \wedge Y=Z) \vee (X=Z \wedge Y=B)) \rightarrow \psi_i (X,Y)]$
- Convert to prenex **DNF**
- L $\leq_{log}$ co**QSAT**
- **PSPACE**=co**PSPACE**

AP=PSPACE
(QSAT is AP-complete)

QSAT ∈ AP

- The computation will guess the truth values of the variables X_1, X_2, . .. one- by-one, where existentially quantified variables are guessed at states in K_{OR} , while universally quantified ones at states in K_{AND}.
- A final state is accepting if the guessed truth assignment satisfies the expression, and rejecting otherwise.
- It follows from the definition of acceptance for alternating machines that a quantified expression is accepted iff it is true; the time needed is polynomial.

AP=PSPACE
(QSAT is AP-complete)

We will now show that any language in AP is reducible to QSAT

- The computation of a polynomial-time ATM can be captured by a table, with extra nondeterministic choices.
- The quantifiers are universal if the current state is in K_{AND} and existential if the current state in K_{OR}.
- The variables standing for nondeterministic choices at even levels are existentially quantified, and at odd levels universally.
- The ATM accepts the input iff the resulting quantified expression is true.

ATMs restricted to a fixed number of alternations

For every $i \in N$, we define $\Sigma_i TIME(T(n))$ to be the set of languages accepted by a $T(n)$-time ATM M whose initial state is labeled "∃" and on which every input and on every (directed) path from the starting configuration in the configuration graph, M can alternate at most i−1 times from states with one label to states with the other label.

For every $i \in \mathbb{N}$, $\Sigma_i^p = \cup_c \Sigma_i \mathbf{TIME}(n^c)$

For every $i \in N$, we define $\Pi_i TIME(T(n))$ to be the set of languages accepted by a $T(n)$-time ATM M whose initial state is labeled "∀" and on which every input and on every (directed) path from the starting configuration in the configuration graph, M can alternate at most i−1 times from states with one label to states with the other label.

For every $i \in \mathbb{N}$, $\Pi_i^p = \cup_c \Pi_i \mathbf{TIME}(n^c)$

The class TISP

For every two functions $S, T : \mathrm{N} \to \mathrm{N}$, define ***TISP(T(n), S(n))*** *to be the set of languages decided by a TM M that on every input x takes at most $O(T(|x|))$ steps and uses at most $O(S(|x|))$ cells of its read-write tapes.*

Note: $\mathrm{TISP}(T(n), S(n)) \neq \mathrm{DTIME}(T(n)) \cap \mathrm{SPACE}(S(n))$

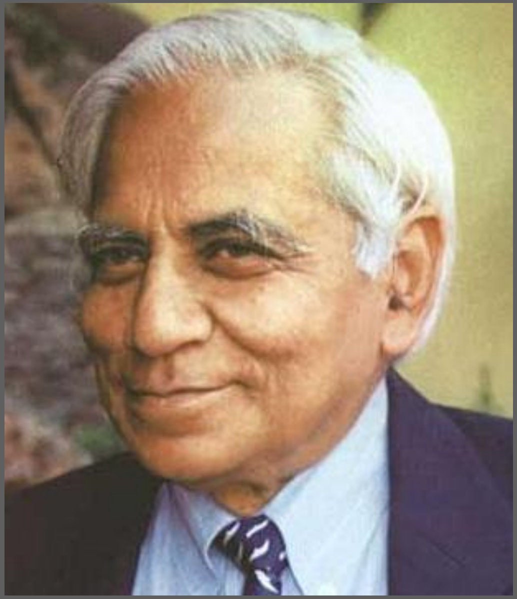
- $\mathrm{SAT} \notin \mathbf{TISP}(n^{1.1}, n^{0.1})$
- $\mathbf{NTIME}\,(n) \not\subseteq \mathbf{TISP}(n^{1.2}, n^{0.2})$
- $\mathbf{TISP}(n^{12}, n^{2}) \subseteq \Sigma_2\mathbf{TIME}(n^{8})$
- If $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^{1.2})$, then $\Sigma_2\mathbf{TIME}(n^{8}) \subseteq \mathbf{NTIME}(n^{9.6})$

BIBLIOGRAPHY

- CHRISTOS PAPADIMITRIOU, COMPUTATIONAL COMPLEXITY, ADDISON WESLEY, 1994
- SANJEEV ARORA AND BOAZ BARAK, COMPUTATIONAL COMPLEXITY: A MODERN APPROACH, CAMBRIDGE UNIVERSITY PRESS, 2009
- MICHAEL SIPSER , INTRODUCTION TO THE THEORY OF COMPUTATION, THOMSON COURSE TECHNOLOGY, 2006

DEXTER KOZEN

LARRY STOCKMEYER



ASHOK K. CHANDRA