1 Notes on 16th Feb.

1.1 Main Topic

Theorem 1

\[ NSPACE(s(n)) \subseteq DSPACE(s^2(n)) \]  

(1)

Before proving the above theorem, we will introduce one new computation model: **Alternating Turing Machine.** The possible computations of an alternating Turing Machine M on an input word x can be represented by a tree \( T_x \) in which the root is initial configuration, and the children of a nonterminal node \( C \) are the configurations reachable from \( C \) by one step of \( M \). For a word \( x \) in \( L(M) \), define an **accepting subtree** \( S \) of \( T_x \) as follows:

1) \( S \) is finite.
2) The root of \( S \) is the initial configuration with input word \( x \).
3) If \( S \) has an existential configuration \( C \), then \( S \) has exactly one child of \( C \) in \( T_x \); if \( S \) has a universal configuration \( C \), then \( S \) has all children of \( C \) in \( T_x \).
4) Every leaf is a configuration whose state is the accepting state \( q_A \).

Observe that Non-deterministic machine is one special case of **ATM**.

Theorem 2

\[ \begin{align*}
ASPACEx(s(n)) &= DTIME(2^{O(s(n))}) \\
DSPACE(x(n)) &= ATIME(s(n)^{O(1)})
\end{align*} \]  

(2)

Theorem 3

\[ \begin{align*}
AL &= P \\
ASPACEx(n) &= E \\
ATIME(n^{O(1)}) &= PSPACE \\
ASPACEn^{O(1)} &= EXP
\end{align*} \]  

(3)

Theorem 4 **Either**

\[ DSPACE(log n) \subset ASPACE(log n) \]  

(4)

or

\[ DTIME(n^{O(1)}) \subset ATIME(n^{O(1)}) \]  

(5)

Thus, alternation is more powerful than deterministic computation in either the time-bounded or the space-bounded setting.

The first theorem will be the direct corollary of the following theorem:

Theorem 5

\[ \begin{align*}
1)\ NSPACE(s(n)) &= ATIME(S^2(n)) \\
2)\ ATIME(T(n)) &= DSPACE(T(n)) \\
3)\ ASPACEx(s(n)) &= DTIME(2^{O(s(n))}) \\
4)\ DTIME(T(n)) &= ASPACEx(log T(n))
\end{align*} \]  

(6)
proof: 1) Let $A \in \text{NSPACE}(s(n))(A$ is accepted by some NTM $M$ in $\text{SPACE}(s(n))$ and time $t^{k+s(n)})$

construct one new algorithm:

Begin
  on input $x$
    compute $s(|x|)$
    compute $T = 2^{s(|x|)}$
    call PATH( $C_{int}, C_{acc}$ )
    ( $PATH \text{ return } TRUE \text{ if } C_{int} \rightarrow C_{acc}$ in $t$ step )
End

PATH( $C,D,t$ )
Begin
  if $t \leq 1$, then
    check if $C = D$ or $C \rightarrow D$
  else
    Guess config $E$
    check
      PATH( $C,E,$ ) PATH( $E,D,$ )
End

Because the recursive depth is $O(s(n))$ and each call requires $O(s(n))$ (This is the time required to guess $E$), ATM’s accepting tree’s depth can’t be longer than $O(s^2(n))$. End (of proof of theorem 5.1).

2) let $A \in \text{ATIME}(t(n))$

construct a Deterministic algorithm to compute $A$

BEGIN
  on input $x$
    let $C = \text{initial config of } M$
    call Eval( $C$ )
END

Eval( $C$ )
if $C$ is existential node
  let $C_1$ be the first child of $C$
  if Eval( $C_1$ ) then return TRUE
  else return Eval( $C_2$ )
if $C$ is universal node
  if Eval( $C_1$ ) is false, return FALSE
  else return Eval( $C_2$ )

counting: recursive depth is $t(n)$, each call use less than $t(n)^2$ space(In the next lecture, we’ll improve this method to get the claimed conclusion.) End (of theorem 5.2.1).