School of Mathematical and Computational Sciences

Indian Association for the Cultivation of Science

Master's/Integrated Master's-PhD Program/ Integrated Bachelor's-Master's Program/PhD Course

Theory of Computation II: COM 5108

Tutorial VII (19 October 2023)

Instructor: Goutam Biswas *Autumn Semester 2023*

1. Justify that $\mathbf{NL} \subseteq \mathbf{P}$.

   Ans. Let $A \in \mathbf{NL}$. As PATH is **NL**-complete, the log-space reduction of A to PATH takes $2^{O(\log n)} = O(n^k)$ steps. It is also known that $PATH \in \mathbf{P}$. So A can be decided in polynomial time.

2. We defined SAT_H as follows:

$$SAT_H = \{\phi 0 1^{n^{H(n)}} : \phi \in SAT, |\phi| = n\}.$$

   The function $H(n)$ is defined as follows:

$$H(n) = \begin{cases} i, & i = \min\{j : j \text{ satisfies } C\} \\ \log\log n, & \text{otherwise.} \end{cases}$$

   C: j is a natural number, $1 \leq j < \log\log n$, such that the TM M_j decides the membership of all $x \in \{0,1\}^*$, $|x| \leq \log n$ in SAT_H, within $j \times |x|^j$ steps,

   (a) Is $H(n)$ non-decreasing?
   (b) What is the maximum number of TMs to simulate to compute $H(n)$?
   (c) On how many input each machine runs?
   (d) What is the upper bound of time to check the membership of ϕ in SAT for all input?
   (e) Give an upper bound of computation time of $H(n)$.

   Ans.

   (a) The function $H(n)$ is non-decreasing. Let $a < b$, $H(a) = k$ and $H(b) = l$. By the definition the TM M_k correctly predict the membership of all input of length $\log a$, and the TM M_l correctly predict the membership of all input of length $\log b$. As $a < b$, $\log a < \log b$. If $l < k < \log\log a$, the value of $H(a)$ cannot be k ($\min\{j : j \text{ satisfies } C\}$).
   (b) There are at most $\log\log n$ machines to simulate.
   (c) There are $1 + 2 + 2^2 + \cdots + 2^{\lfloor \log n \rfloor}$ inputs as the length of the string is at most $\log n$. So the number of input is $2n - 1$.
   (d) Let $x = \phi 0 1^{n^{H(n)}}$. $|\phi| < |x| = k$ The time to construct the truth table of ϕ is less than 2^k. For an input of length length k there are 2^k input strings. Each one takes less than or equal to 2^k steps. So the upper bound of running time of a TM on all input is $\sum_{k=0}^{\lfloor \log n \rfloor} 2^{2k} = \frac{4^{\log n + 1} - 1}{3} = O(n^2)$.

(e) Considering $\log\log n$ number of TMs, the running time is $n^2 \log\log n = O(n^3)$.

3. Let $PAL = \{x \in \{0,1\}^* : x = x^R\}$. Show that $PAL \in \mathbf{L}$.

   Ans. Following is a logspace algorithm for PAL.

   $PAL(x)$
   1 $l \leftarrow 0$
   2 $r \rightarrow n-1$ /* $x = x_0 \cdots x_{n-1}$ */
   3 **do** 4 to 6 **while** $l < r$
   4 **if** $x_l \neq x_r$ *reject*
   5 $l \leftarrow l + 1$
   6 $r \leftarrow r - 1$
   7 *accept*

   Both l and r takes $\log |x|$ space. Index of head positions will also take $\log |x|$ space.
   It looks like a random access machine, but it will work in logspace on an **L** machine.

4. Let x be the binary representation of a positive integer (without leading zeros). The function f computes $f(x) = x + x^R$, where x^R is reverse of x. Give a logspace algorithm to compute $f(x)$.

   Ans. We cannot compute and remember x^R in logspace. Our logspace algorithm is as follows.

   $F(x)$
   1 $i \leftarrow 0$
   2 $j \leftarrow n-1$ /* $x = x_0 \cdots x_{n-1}$ */
   3 $c \leftarrow 0$ /* carry */
   4 **do** 5 to 8 **while** $i < n$
   5 output($(x_i + x_j + c) \bmod 2$)
   6 $c \leftarrow (x_i + x_j + c)/2$
   7 $i \leftarrow i + 1$
   8 $j \leftarrow j - 1$
   9 if $c = 1$ output(c)

   The space used by i, j on the work-tape are $\log n$ each. The carry c is a single bit. The index of head positions will also use
   It looks like a random access machine, but it will work in logspace on a logspace bounded work-tape.

5. The language $A_{NFA} = \{< N, x >: N \text{ is an NFA that accepts } x\}$.

   (a) Show that $A_{NFA} \in \mathbf{NL}$.

   (b) Show that A_{NFA} is **NL**-complete.

   Ans.

(a) The work-tape maintains a pointer to *current state* of N (logspace). The nondeterminism of the **NL** machine helps where there is a non-deterministic transition of the NFA. It nondeterministically chooses the correct transition. The input $< N, x >$ is accepted when x is consumed and the NFA is at a final state.

(b) The reduction from PATH to A_{NFA} is trivial as the graph of $< G, s, d >$ may be viewed as the state transition diagram of an $N =< V(G), \{a\}, \delta, s, \{d\} >$ where $\delta(u, \varepsilon) = \{v \in V(G) : (u, v) \in E(G)\}$, for all $u \in V(G)$.
If d is reachable from s in G, then $< G, s, d > \mapsto < N, \varepsilon >$. Even the length of the path can be constructed in logspace. So all transitions may be labeled with 'a' and $< G, s, d > \mapsto < N, a^c >$, where c is the path length.