Lecture 16: Nondeterministic Logspace

Date: October 17, 2023.

Problem 1. For any functions S, T, DSPACE$(S(n))$ = co-DSPACE$(S(n))$ and DTIME$(T(n))$ = co-DTIME$(T(n))$

For any $\mathcal{C}$, co-$\mathcal{C} = \{A \mid \bar{A} \in \mathcal{C}\}$. $A \in$ DSPACE$(S(n))$/DTIME$(T(n))$. $\exists$ TM M. $L(M) = A$

Algo $\bar{A}$: Input x
Run M on x
Flip M's answer

M is ~~DSP~~ $S(n)$-space bounded. We can assume that M is $2^{O(S(n))}$-time bounded

Theorem 1 (Immerman-Szelepcsényi Theorem). *For* $S(n) \geq \log n$, NSPACE$(S(n))$ = *co*-NSPACE$(S(n))$.

$A \in$ NSPACE$(S(n))$ $\exists$ NTM N. s.t $L(N) = A$ and N is $S(n)$-space bounded.

Goal: Design a nondeterministic $S(n)$-space bounded algorithm for $\bar{A}$

Configuration graph of N on x: $G_N(x) = (V, E)$

V = Configurations of N on input x. — Since N is $S(n)$-space bounded, configuration is string in $\Delta^{S(n)}$ (Δ is some alphabet)

$E = \{(c_1, c_2) \mid c_1 \vdash^1_N c_2\}$.

$|V| = m \leq |\Delta|^{S(n)} \sim 2^{O(S(n))}$

start = initial configuration of N on x.

$x \in A$ iff x is accepted N iff $\exists$ a path from start to some accepting configuration in $G_N(x)$.

$x \in \bar{A}$ iff x is not accepted by N iff there is no path from start to any accepting configuration in $G_N(x)$.

Define $R_i = \{c \mid \exists$ path of length $\leq i$ from start to $c\}$.

$x \in A$ iff $\exists$ accepting configuration $a \in R_m$

$x \in \bar{A}$ iff $\forall$ accepting configurations $a \notin R_m$.

Claim: $R_i \in$ NSPACE$(S(n))$. — There is a NTM M s.t M is $S(n)$-space bounded and, on any input x, M answers yes iff $x \in R_i$.

M: Input c

Guess $c_1, c_2 \ldots c_k$ $(k \leq i)$
Check $c_1 = $ start,
$c_i \vdash^1_N c_{i+1}$, and $c_k = c$.

→ config = start
for $j = 1$ to i
  if c = config
    return accept.
  else
    guess c'
    If config $\vdash^1_N c'$ then config = c'
    else abort.
return reject

Claim: If $k = |R_i|$ is known then there is NTM M_k s.t. M_k is $S(n)$-space bounded and $L(M_k) = \overline{R_i}$

M_k : Input c.

Guess $c_1, c_2 \ldots c_k$
Check $c_j \in R_i \ \forall j$.
If $c \neq c_j \ \forall j$ then accept.

prev = null
For $j = 1$ to k.
Guess $c' >$ prev
if $c' \in R_i$
if $c = c'$
reject.
else
abort.
prev = c'
Accept.

If $|R_m|$ can be "computed in NSPACE$(S(n))$" then $\overline{A} \in$ NSPACE$(S(n))$

$|R_0| = 1$

~~Step~~ Suppose $|R_i| = k$. Determine $|R_{i+1}|$

ctr = 0
For every configuration c,
prev = null
For ~~every~~ $j = 1$ To k.
Guess $c' >$ prev
if $c' \in R_i$
if $c = c'$ or $c' \vdash_N^1 c$ then
ctr++; break.
else
abort.
prev = c'

$|R_m|$ computed iteratively starting from $|R_0|$.

Logspace Computable Functions: A function f is **computable in logspace** if there is a Turing machine M such that on any input x, M halts with $f(x)$ written on its output tape, and M uses at most $O(\log |x|)$ cells on its work-tape.

Logspace Reductions: A is reducible to B in **logspace** (denoted $A \leq_m^{\log} B$) if there is a logspace computable function f such that for any x, $x \in A$ iff $f(x) \in B$.

(NL, P, NP..) (NL, P, NP...)

Proposition 2. *If* $A \leq_m^{\log} B$ *and* $B \in \mathrm{L}$ *then* $A \in \mathrm{L}$.

Algo A: Input x.
Compute $f(x)$
Run Algo for B on $f(x)$.

Proposition 3. *If* $A \leq_m^{\log} B$ *and* $B \leq_m^{\log} C$ *then* $A \leq_m^{\log} C$.

Hardness and Completeness: Let $\mathcal{C} \in \{\mathrm{NL}, \mathrm{P}, \mathrm{NP}, \ldots\}$. A problem B is **$\mathcal{C}$-hard** if for any $A \in \mathcal{C}$, $A \leq_m^{\log} B$. B is **$\mathcal{C}$-complete** if B is $\mathcal{C}$-hard and $B \in \mathcal{C}$.

Maze Problem: MAZE is the following problem: Given a directed graph $G = (V, E)$ and vertices $s, t \in V$, determine if there is a path from s to t.

Theorem 4. MAZE *is* NL-*complete.*