第八章作业

袁晨圃

6.1

习题 1 (8.1)：证明对于任意函数 $f:\mathbb{N}\to\mathbb{R}^+$，其中 $f(n)\geq n$，不论用单带图灵机模型还是用双带图灵机模型，所定义的空间复杂性类 SPACE($f(n)$) 总是相同的

解：

两个方向：

a. $\mathrm{SPACE}_{单}(f(n)) \subseteq \mathrm{SPACE}_{双}(f(n))$

显然，用双带图灵机模拟单带图灵机。

b. $\mathrm{SPACE}_{双}(f(n)) \subseteq \mathrm{SPACE}_{单}(f(n))$

使用单带图灵机模拟双带图灵机，不考虑时间情况下，只需要将两条读写带拼接，$O(2f(n)) = O(f(n))$

所以 $\mathrm{SPACE}_{单(f(n))} = \mathrm{SPACE}_{双(f(n))}$

6.2

习题 2 (8.11)：
a. 令 $\mathrm{ADD} = \{\langle x,y,z\rangle \mid x,y,z>0\ 且为二进制整数, x+y=z\}$，证明 $\mathrm{ADD}\in \mathrm{L}$

解：
a. 判断 x,y,z 是否相等只需要一位一位的比较，记录现在的位置 和当前进位 c，只需要 $\log(n)$ 空间，其中 $n=|x|$

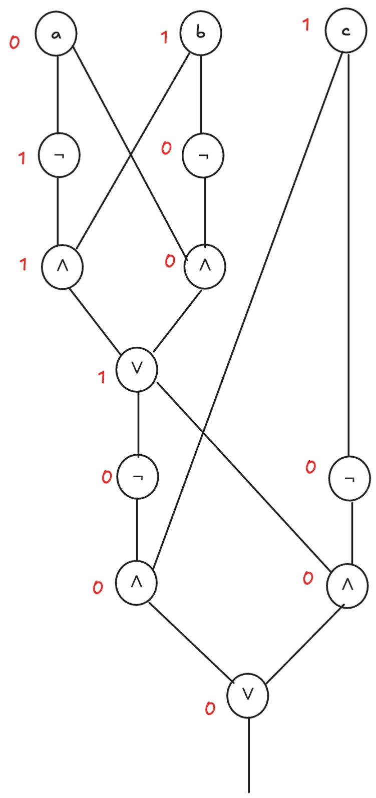
习题 3 (°8.11b)：令 $\mathrm{PAL\text{-}ADD} = \{\langle x,y\rangle \mid x,y>0\ 且为二进制整数, x+y是整数且二进制表示是回文\}$ 证明 $\mathrm{PAL\text{-}ADD}\in \mathrm{L}$

解：可以在 $O(\log n)$ 空间内计算 $x+y$ 的位数（维护当前位置和进位信息），然后枚举每一位，比较是否与对称位置相同

$M=$对于输入$\langle x,y\rangle$
a. 计算 $x+y$ 的位数 k $\quad O(\log n)$
b. 对于每一个位数 i，计算 $(x+y)_{[i]}$ 和 $(x+y)_{[n-i]}$，比较是否相同，不同则拒绝 $\quad O(\log n)$
c. 接受

习题 4 (9.5)：给出三个输入变量上计算奇偶函数的电路，并说明它在输入 011 上的计算历史

解：思路：$(a\oplus b)\oplus c$



习题 5 (°9.13a)：定义函数 $\text{majority}_n : \{0,1\}^n \to \{0,1\}$ 为：

$$\text{majority}_n(x_1, x_2, ..., x_n) = \begin{cases} 0 & \sum x_i < n/2 \\ 1 & \sum x_i \geq n/2 \end{cases}$$

所以 majority_n 返回输入中的多数派。证明 majority_n 可以用下面的电路计算:

a. $O(n^2)$ 规模
b. $O(n \log n)$ 规模

解：

n 位加法器 $\{0,1\}^n \to \{0,1\}^{n+1}$ 的电路复杂度是 $O(n)$，只需要每一位计算$c_i = a_i \oplus b_i \oplus \text{Carry}_{i-1}$ 和 $\text{Carry}_i = (a_i \wedge b_i) \vee (\text{Carry}_{i-1} \wedge (a_i \vee b_i))$，特别地 $\text{Carry}_{-1} = 0$

n 位比较器 $\{0,1\}^n \to \{0,1\}^3$ 的电路复杂度是 $O(n)$。从高到低依次计算，每一位有三个门 $\text{lt}_i, \text{eq}_i, \text{gt}_i$，连接：$\text{lt}_i = \text{lt}_{i+1} \vee (\text{eq}_{i+1} \wedge (\neg a_i \wedge b_i)), \text{eq}_i = \text{eq}_{i+1} \wedge ((a_i \wedge b_i) \vee (\neg a_i \wedge \neg b_i)), \text{gt}_i = \text{gt}_{i+1} \vee (\text{eq}_{i+1} \wedge (a_i \wedge \neg b_i))$，初始 $\text{lt}_{n+1}, \text{eq}_{n+1}, \text{gt}_{n+1} = 0, 1, 0$，输出 $\{\text{lt}_0, \text{eq}_0, \text{gt}_0\}$

a. 实例化 $2n$ 个 n 位加法器，分为两组，分别计算 0 的个数和 1 的个数。 每一组内部，每一个加法器的输入是当前的 count(上一个加法器输出) 和当前位是否是 0/1 的一个输入

最终将两组加法器的结果 $\text{sum}_1, \text{sum}_0$ 送入 n 位比较器，$\text{majority} = \text{eq} \vee \text{gt}$

复杂度 $2nO(n) = O(n^2)$

b. 其实不需要 n 位加法器，操作数位数不超过 $\lceil \log_2(n) \rceil$

递归处理，每次序列长度分成一半。

对于段数量为 2^k 的层，每段长度不超过 $\lceil n/2^k \rceil$，需要实例化 2^{k-1} 个 $\lceil n/2^k \rceil$ 位加法器，复杂度 $2^{k-1} \cdot O\big(n/2^k\big) = O(n)$

层数是 $O(\log n)$ 的

所以总复杂度：$O(n \log n)$

6.3

习题 6 (8.6)：证明 PSPACE 难的语言也是 NP 难的

解：设语言 A 是 PSPACE 难的。

对任意语言 $L \in \text{NP}$，因为 $\text{NP} \subseteq \text{NPSPACE} = \text{PSPACE}$，所以 $L \in \text{PSPACE}$，所以 L 能多项式时间规约到 A，所以 A 是 NP 难的

习题 7 (8.16)：$\text{STRONGLY-CONNECTED} = \{\langle G \rangle \mid G \text{ 是强连通图}\}$

证明 STRONGLY-CONNECTED 是 NL 完全的

证明：

a. STRONGLY-CONNECTED 是 NL 的

枚举节点对 (u, v)，调用 PATH 的 NL 算法，若不存在 $u \longrightarrow v$或$v \longrightarrow u$则 拒绝，反之枚举结束后 接受

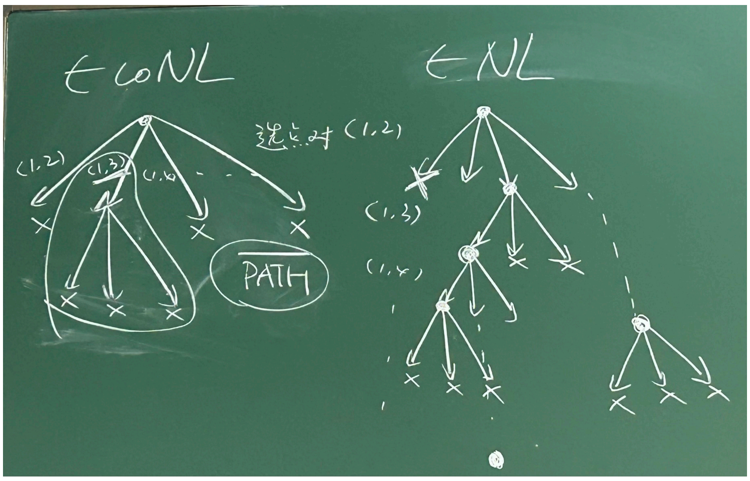
多使用的空间是 u, v 编号，$O(\log n)$

由于 *NL=coNL*，所以可以证明 $\overline{\text{STRONGLY-CONNECTED}} \in \text{coNL}$

非确定性地猜不存在路径的点对

因为 PATH 的机器有很多路径，直接取反会导致在任意输入上都接受

所以应该调用 $\overline{\text{PATH}}$ 的机器



b. STRONGLY-CONNECTED 是 NL 难的

将 PATH 规约到 STRONGLY-CONNECTED。

对于 $\langle G, s, t\rangle$，构造一个图 G' 使得 $\langle G, s, t\rangle \in \text{PATH} \iff G'$ 是强连通图

- 保留所有原始边
- 添加一条从 t 到所有顶点（除了t）的边
- 添加一条从所有顶点 （除了 s）到 s 的边

若不存在 $s \longrightarrow t$ 路径，则图显然不是强连通分量，因为新添加的边不影响 s 到 t 的连通性

若存在 $s \longrightarrow t$ 路径，则对任意两点 a, b，都存在路径 $a \to s \longrightarrow t \to b$，所以图是强连通分量

这个规约可以在对数空间内完成（按顺序扫描边输出，并存下节点编号构造新边）

习题 8 (°8.18)：证明 A_{NFA} 是 NL 完全的

解：

a. $A_{\text{NFA}} \in \text{NL}$

对于输入 $\langle N, w\rangle$，模拟 NFA N 的运行，只需要存下当前状态 q 和输入带读取到的位置，空间 $\log(n)$，然后扫描输入带以及非确定地进行转移

b. A_{NFA} 是 NL 难的

将 PATH 规约到 A_{NFA}。

对于 $\langle G, s, t\rangle$，构造一个 NFA N 使得 $\langle G, s, t\rangle \in \text{PATH} \iff N$ 接受字符串 ε。

- 每一个节点就是 N 中的一个状态 $a \leftrightarrow q_a$
- 每一条边 $a \to b$ 是 N 中的一个转移 $q_a \overset{\varepsilon}{\longrightarrow} q_b$
- 起始状态 q_s，接受状态 q_t

这个规约可以在对数空间内完成（按顺序扫描节点和边输出）

习题 9 (8.25)：梯子 是一个字符串序列 $s_1, s_2, ..., s_k$，其中每个字符串与前一个恰好只在一个字母上不同。

$\text{LADDER}_{\text{DFA}} = \{\langle M, s, t\rangle \mid M \text{ 是一个 DFA}, L(M)\text{包含一个以}s\text{开头以}t\text{结束的梯子}\}$

证明 $\text{LADDER}_{\text{DFA}}$ 属于 PSPACE

解：对 $\langle M, s, t\rangle$ 声明一张图 G，其中每个节点对应 $L(M)$ 中的一个长度为 $|s|$ 的字符串。对图 G 惰性求值。

图 G 中两节点有边当且仅当它们之差一个字符

那么 $\text{LADDER}_{\text{DFA}} = \{\langle M, s, t\rangle \mid \langle G_M, s, t\rangle \in \text{PATH}\}$

a. s, t 长度不同则 拒绝

b. 如果 M 不接受 s 则 拒绝

c. 使用 PATH 的 NL 算法，每次猜测下一个点后先模拟 M 运行，如果不接受则 拒绝，否则继续猜测下一个点，如果步数达到 $|\Sigma|^{|s|}$，则拒绝

节点个数是指数级的，所以记录节点编号需要多项式空间

可以描述清楚非确定性判断 PATH 的 NL 算法，因为这里对其有一些修改

所以 $\text{LADDER}_{\text{DFA}} \in \text{PSPACE}$

6.4

习题 10 (°8.31)：考虑 PUZZLE 问题的双人版，每名选手开始时都有一叠排好序的谜卡。他们轮流地按序把卡片放进盒子，并有权选择哪一面上。如果在最终的盒子中所有孔的位置都被堵住了，则选手 I 赢。如果还有孔的位置没被堵住，则选手 II 赢。

证明对于给定的卡片的起始格局，判定哪位选手有必胜策略的问题是 PSPACE 完全的。

解：*将 TQBF 规约到 PUZZLE-GAME*

对于含 n 个变元的 TQBF 公式 $\varphi = \exists x_1 \forall x_2 \exists x_3 ... [\psi]$

使用 SAT 到 PUZZLE 的规约，将 ψ 转化为卡片集合 $\{c_1, c_2, ..., c_n\}$，每张代表一个变量，每种放置方式对应一组变量赋值，这个规约可以在多项式时间内完成。