易守拙 2024300001103
计算理论导引
2025 年 11 月 12 日

计算理论导引第三周作业

Question 1. 利用分配律 $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ 和 $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$，是否可在多项式时间内将合（析）取范式转换成析（合）取范式?

不行. 考虑

$$\varphi = (x_{11} \vee x_{12}) \wedge (x_{21} \vee x_{22}) \wedge \cdots \wedge (x_{n1} \vee x_{n2})$$

我们期望把 φ 化成如下的情形:

$$(x_{11} \wedge x_{21} \wedge \cdots \wedge x_{n1}) \vee (x_{12} \wedge x_{21} \wedge \cdots \wedge x_{n1}) \vee \cdots \vee (x_{12} \wedge x_{22} \wedge \cdots \wedge x_{n2})$$

但这个操作只能通过分配律进行. 时间复杂度是 $O(n \cdot 2^n)$, 它不在 $\mathbf{P}$ 内. 反过来也是一样的.

Question 2. 证明 $\mathbf{EXP}^{\mathbf{EXP}} = \mathbf{2\text{-}EXP}$.

我们来分别证明, $\mathbf{EXP}^{\mathbf{EXP}} \subseteq \mathbf{2\text{-}EXP}$ 和,$\mathbf{EXP}^{\mathbf{EXP}} \supseteq \mathbf{2\text{-}EXP}$.

$\forall x \in \mathbf{EXP}, \exists c \in \mathbb{N}, x \in \mathbf{TIME}(2^{n^c})$. 那么,$\forall x \in \mathbf{EXP}^{\mathbf{EXP}}, \exists y, z \in \mathbf{EXP}, x = y^z$. 我们只考察 $\mathbf{TIME}$ 后对应的正整数 c, 即 y, z 对应的 y_c, z_c; 此时,

$$x \in \mathbf{TIME}((2^{n^{c_y}})^{(2^{n^{c_z}})}) = \mathbf{TIME}(2^{n^{c_y} \cdot 2^{n^{c_z}}})\mathbf{TIME}(2^{2^{\log(n)c_y + n^{c_z}}})$$

. 由于 $\log(n)c_y + n^{c_z} = O(n^C)$, 我们知道 $x \in \mathbf{2\text{-}EXP}$. 反过来的情况类似, 把过程倒过来思考即可.

Question 3. 证明 $\mathbf{2SAT} \in \mathbf{P}$.

由于 $(a \vee b) \iff (\neg a \longrightarrow b) \wedge (\neg b \longrightarrow a)$, 我们把一个 $\mathbf{2SAT}$ 公式和这样的图对应起来:

- 一个变元 x_i 对应图里的两个点, 一个代表 x_i, 另一个代表 $\neg x_i$;
- 对公式中 $(a \vee b)$, 添加两条有向边: $\neg a$ 指向 b, $\neg b$ 指向 a;
- 该公式是不可满足的, 当且仅当在蕴含图中, 存在某个变量 x_i 使得顶点 x_i 和顶点 $\neg x_i$ 位于同一个强连通分量中.

我们可以借助 Tarjan 等搜索算法来爆搜这个图中的强连通分量, 时间复杂度在 $O(n)$. 所以我们证明了命题.

Question 4. 写一段对数空间程序, 解决 `MULP`. 它定义为

$$\texttt{MULP} \stackrel{\text{def}}{=} \{(a, b, c) \mid a, b, c \text{ 为二进制数，且} a \cdot b = c\}$$

这个程序来模拟 a 与 b 的每一位（b_i）相乘然后将这些中间结果相加的过程.

- 在一个带子上存一个进位值 $carry$, 和当前计算位数 j;
- 让 j 从 0 循环到 a 和 b 的长度之和. 在每次循环中, 再循环 b 的每一位, 如果 i 位是 1, 就读取 a 的 $j - i$ 位把他们的乘积结果加到结果中, 从而计算列总和. 如果这个结果 $result\%2 \neq c_j$，则 $a \cdot b \neq c$, 立即停机并拒绝. 如果相等, 更新进位, 进入下一个循环.
- 如果循环正常结束, 检查最终的 carry 是否为 0. 是 0 就接受, 反之拒绝.

  输入 (a, b, c) 存储在只读的输入带上. j 的最大值是输入长度，需要 $O(\log n)$ 空间;carry 在最坏情况下, 进位的值不会超过 b 的长度. 需要 $O(\log n)$ 空间。

  总工作空间为 $O(\log n)$，因此这是一个对数空间算法.

。

Question 5. 证明空间压缩定理: 设图灵机 M 在 $S(n)$ 空间判定 L。对任意 $\epsilon > 0$，存在图灵机 M'，M' 能在 $\epsilon S(n)+1$ 空间内判定 L。

我们的核心思想是构造一个新的图灵机 $\mathbb{M}'$，它使用一个更大的带字母表来“压缩”原图灵机 $\mathbb{M}$ 的工作带。

设 $\mathbb{M} = (Q, \Sigma, \Gamma)$ 是在 $S(n)$ 空间内判定 L 的图灵机, 我们构造这样的 $\mathbb{M}'$: $\mathbb{M}'$ 的工作带字母表 $\Gamma' = \Gamma^k$, 状态是一个元组 (q, j), 其中 $q \in Q$ 是 $\mathbb{M}$ 的当前状态,$j \in \{1, 2, \ldots, k\}$ 是 $\mathbb{M}$ 的读写头在其所处的 k-符号块中的相对位置. 类似于时间的线性加速定理, 我们知道 $\mathbb{M}'$ 可以模拟操作, 并且空间复杂度是

$$S'(n) = \lceil S(n)/k \rceil \leq \frac{S(n)}{k} + 1$$

取 $k \geq 1/\epsilon$ 即证.