计算复杂性 作业 1

李煦阳 DZ21330015 (njulixuy@163.com)

2021 年 9 月 21 日

1. 2.8

(a) 要证明所有的 **NP** 问题（$L \in \mathbf{NP}$）都可以规约至停机问题。

已知 $\forall L \in \mathbf{NP}. \exists M, p. \forall x. x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)}. M(x,u) = 1.$

令规约函数 $f(x) = \langle\langle\alpha\rangle, \langle\beta, \theta, x\rangle\rangle$，其中 $M_\beta = M$, θ 为 p 的编码，M_α 重复遍历 $u \in \{0,1\}^{p|x|}$，在 $M_\beta(x,u) = 1$ 时停机，否则不停机。易知其复杂度为常数，是一个双射函数。

基于 M 构造 `HALT` 问题：输入为 $\langle\langle\alpha\rangle, \langle\beta, \theta, x\rangle\rangle$，其对应的语言为 L'

现证 $x \in L \Leftrightarrow f(x) \in L'$.

(a) $x \in L \Rightarrow f(x) \in L'$

由构造 L' 的方式可知，对于 $x \in L$，$f(x)$ 在 `HALT` 上会停机，所以 $f(x) \in L'$。

(b) $f(x) \in L' \Rightarrow x \in L$

反证，若存在 $f(x) \in L'$ 且 $x \notin L$，则 x 在构造的图灵机上不会停机，所以 `HALT`$(f(x))$ 不为 0，所以 $f(x) \notin L'$

■

(b) `HALT` 问题不是 **NP** 问题。

易知 `HALT` 问题是不可判定问题（并不存在一种算法可以描述 `HALT` 问题）（将 `HALT` 问题带入自身可证）。只需证所有 **NP** 问题都是可判定的（即可以找到一个通用算法），便可说明 `HALT` 不是 **NP**。

对于每个 **NP** 问题，已知 M，**P**，对于一个输入 x，我们可以对解空间进行 $EXP(p(|x|)$ 次枚举寻找 certificate，并在多项式时间内演算每个可能解的真假（总复杂度为 $EXP(p(|x|))$），所以 **NP** 是可判定的。

■

2. 2.15

(a) 证明 `CLIQUE` 问题（某图的 K 个顶点的全连接子图是否存在问题）是 **NPC** 的，只需证独立集问题与 `CLIQUE` 问题可以相互规约。

对于 `INDSET`(V, E, K)，构造 `CLIQUE`$(V, \overline{E}, K)$，其中 $\overline{E} = \{(v_1, v_2) \mid (v_1, v_2) \notin E\}$. 规约函数显然是多项式时间的。正确性在于，连接与独立（不连接）是显然对偶的。

类似地，`CLIQUE` 问题也可规约成 `INDSET` 问题。所以 `CLIQUE` 是 **NPC** 的。■

(b) 证明顶点覆盖问题（某图是否存在大小为 K 的顶点子集，使得图中的每一条边至少有一个顶点落于其中）是 **NPC** 的。只需证明独立集问题可以与顶点覆盖问题相互规约。

首先证明顶点覆盖 `VC`(V, E, K) 问题可以规约至独立集问题 `INDSET`$(V, E, |V| - K)$。令找到的独立集为 S（可以存在多于 $|V| - K$ 大小的独立集，但只取到 $|V| - K$），$|S| = |V| - K$，我们证明 $S' = V - S$ 一定是个顶点覆盖，其大小为 K.

假设 S' 不是顶点覆盖，那么 $\exists v_1, v_2 \in V - S' = S. (v_1, v_2) \in E$，这与 S 是独立集矛盾。

其次证明独立集问题 `INDSET`(V, E, K) 可以规约至顶点覆盖问题 $VC(V, E, |V| - K)$。令找到的顶点覆盖集合为 S，$|S| = |V| - K$，我们证明 $S' = V - S$ 一定是独立集。

假设 S' 不是独立集，则 $\exists v_1, v_2 \in V - S' = S. (v_1, v_2) \in E$，这意味着 S 不是顶点覆盖，矛盾。■

3. 2.23

证明 $\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{coNP}$.

课上已证 $\mathbf{P} \subseteq \mathbf{NP}$（构造 $\mathbf{NP}$ 问题，令 certificate 为空，于是 $\mathbf{NP}$ 的 M 就是 $\mathbf{P}$ 的 M. 显然保证多项式时间），只需证 $\mathbf{P} \subseteq \mathbf{coNP}$。

构造方式类似，只需额外令 $M'(x,u) = M(x)$ 即可，即忽略 certificate。 ■

4. 2.24

证明两个 $\mathbf{coNP}$ 的定义等价，
即证对于 L，$\overline{L} \in \mathbf{NP} \Leftrightarrow \exists M, p.\ \forall u^{p(|x|)}.\ M(x,u) = 1$.

令 L 在 $\mathbf{NP}$ 语言中的图灵机与多项式分别为 M'，p'.

已知 $x \in \overline{L} \Leftrightarrow x \notin L \Leftrightarrow \exists u'.\ M'(x,u') = 1$，取否可得，$x \in L \Leftrightarrow \neg\exists u'.\ M'(x,u') = 1 \Leftrightarrow \forall u'.\ \mathtt{flip}(M'(x,u')) = 1$. 也就是说，由基于 2.19 定义的语言，它构造地满足 2.20 的定义，其中 $M = flip \cdot M'$ 与 $p = p'$.

由于一直在用等价推导，反方向的证明也是类似的。 ■

5. 2.25

即证 $\mathbf{P} = \mathbf{NP} \to \mathbf{NP} = \mathbf{coNP}$. 对于 $L \in \mathbf{P}$, 若可以证明 $\overline{L} \in \mathbf{P}$，根据定义 2.19（补集在 NP 中的语言 $\mathbf{coNP}$ 的），并且由于 $\mathbf{P} = \mathbf{NP}$，那么 $\mathbf{NP} = \mathbf{coNP}$. 下面证明 $L \in \mathbf{P} \to \overline{L} \in \mathbf{P}$.

对于 $L \in \mathbf{P}$,我们有图灵机 M,$M(x) = 1 \Leftrightarrow x \in L$. 也即 $M(x) = 1 \Leftrightarrow x \notin \overline{L}$，取否，得 $\mathtt{flip}(M(x)) = 1 \Leftrightarrow x \in \overline{L}$. $\mathtt{flip}$ 显然不影响时间复杂度。也就是说，对于 $\overline{L}$，可以找到多项时间的 $M' = \mathtt{flip} \cdot M$，使得 $x \in \overline{L} \Leftrightarrow M'(x) = 1$. $\mathbf{P}$ 中语言的补集仍属于 $\mathbf{P}$ 得证，进而，$\mathbf{NP} = \mathbf{coNP}$ 得证。 ■

6. 2.30, Berman's Theorem

证明若 SAT 可以规约至一元问题 L，那么 SAT 可以找到一种多项式时间算法（剪枝算法）。

令 SAT 到 L 的多项式时间规约为 f，公式 φ 有 k 个自由变量，长度为 n。φ 的可满足性等价于两个有 $k-1$ 个自由变量的子公式的可满足性（将一个自由变量分别取 0 与 1），两个子公式的长度也为 n. 若将 k 个变量全部拆解，则 φ 的可满足性变为 2^k 个公式的可满足性（一个树状搜索问题）。但

是借助 f，可以在拆解过程中不断剪枝，不必计算全部 2^k 的公式的值。

由于 f 是多项式时间的（设为 $p(\cdot)$），那么 $f(x)$ 的长度一定在 $p(|x|)$ 内。在前述的问题拆解过程（每一次拆解使公式数增加 1），每当公式集合达到 $p(|x|)+1$，对每个公式 φ 求解 $f(\varphi)$，根据鸽笼原理，至少有一个公式不是一元的或者与已有的一元公式重复，我们可以用线性时间（也就是多项式时间）筛去（剪枝掉）这部分重复的或者非一元的公式。最终，只需在剩余的 $p(|x|)$ 大小的公式集合验算可满足性。

但还需要说明剪枝次数也是多项式时间的。公式集中每一个公式的自由变量个数都小于等于 k，意味着每个公式最多引发 k 次剪枝，即一共最多引发 $k \cdot p(|x|)$ 次剪枝，k 与 n 是多项式关系的，所以总复杂度仍是多项式时间的。

■