DOI:10.3969/j.issn.1673-4785.201311055

时间复杂性和空间复杂性研究

高强, 徐心和
(东北大学 信息科学与工程学院,辽宁 沈阳 110004)

摘 要:计算复杂性是衡量问题求解的难易程度的。研究问题的计算复杂性,可以明确该问题是否存在有效的求解算法。介绍并分析了计算理论的一些基本概念,论述了时间复杂性(包括 P、NP、NP-hard、NP-complete 和 EXPTIME)和空间复杂性(包括 PSPACE、NPSPACE、PSPACE-hard 和 PSAPCE-complete)中的各个主要分类。最后分析了各个复杂性类之间的关系。

关键词:计算复杂性;图灵机; 确定型多项式时间复杂性;非确定型多项式时间复杂性;非确定型多项式时间复杂性的完全问题;确定型多项式空间复杂性;确定型多项式空间复杂性的完全问题;可归约性
中图分类号: TP301.5 **文献标志码**:A **文章编号**:1673-4785(2014)05-0529-07

中文引用格式:高强, 徐心和. 时间复杂性和空间复杂性研究[J]. 智能系统学报, 2014, 9(5): 529-535.
英文引用格式:GAO Qiang, XU Xinhe. Research on time complexity and space complexity[J]. CAAI Transactions on Intelligent Systems, 2014, 9(5): 529-535.

Research on time complexity and space complexity

GAO Qiang, XU Xinhe
(College of Information Science and Engineering, Northeastern University, Shenyang 110004, China)

Abstract: Computational complexity is used to measure the level of difficulty of a problem being solved. The research on computational complexity of the problem can make it explicit. This paper introduces and analyzes some fundamental concepts of the computation theory and discusses some main classes of time complexity (including P, NP, NP-hard, NP-complete and EXPTIME) and space complexity (including PSPACE, NPSPACE, PSPACE-hard and PSAPCE-complete) by examples. Finally the relations among complexity classes are analyzed.

Keywords: computational complexity; Turing machine; P; NP; NP-complete; PSAPCE; PSAPCE-complete; reducibility

理论与实践是密切相关的,计算理论为实际工作者提供了在计算机工程中使用的理性工具[1]。计算复杂性是计算理论的重要分支,它告诉人们问题求解需要多少资源(时间或空间)。它所涉及到的一些典型的难解问题包括旅行商问题[2]、SAT 问题(可满足性问题,它涉及芯片测试、计算机设计、软件工程等应用领域[2])、电路复杂性问题[3]、博弈问题[1]等。通过对计算复杂性的研究,可以更深入地了解计算机的能力及局限性。研究问题的复杂性可以了解该问题求解的难易程度,如果它被证明是难解的,就不必将大量精力花费在寻找有效的求解算法上[1],从而在实际工作中做出理性的抉择。本文介绍了与计算复杂性相关的基本概念,同时综述了目前的研究状况,通过例子分析了各个计算复杂性类的特点及其证明的方法。并对各个计算复杂性类之间的关系作了详细的分析。

收稿日期:2013-11-22.
基金项目:国家自然科学基金资助项目(61370153).
通信作者:高强.E-mail:tommy_06@163.com.

1 计算复杂性

计算复杂性是计算理论的一个分支,它用于衡

量问题被求解所需资源(比如时间或空间)的多少[1],在介绍时间复杂性和空间复杂性之前,先来了解计算理论中的一些基本概念。

1.1 可判定问题

可判定问题是指该问题在算法上是可解的,即该问题在计算模型(如有穷自动机或图灵机)上执行,能够进入接受或拒绝状态而终止计算(即停机状态[4])。不可判定问题是指该问题在算法上是不可解的,即该问题在计算模型上执行不能进入停机状态,即不停机[4]。

1.2 确定型图灵机及非确定型图灵机

1.2.1 确定型图灵机

图灵机(deterministic Turing machine, DTM)(见图 1)是一种通用的计算模型[5],该模型由图灵(Alan Turing)在 1936 年提出[1],能模拟实际计算机的所有计算行为。它用一个无限长的带子作为它的无限存储,有一个读写头,能在带子上读、写符号和左右移动。初始时,带子上只有输入串,其他地方都是空的,如果需要保存信息,它可能将这个信息写在带子上,为了读已经写下的信息,它可将读写头往回移动到这个信息所在的位置。机器持续地计算,直到产生输出为止。机器预置了接受和拒绝 2 种状态,如果进入这 2 种状态,就产生输出接受或拒绝。如果不能进入任何接受或拒绝状态,就继续执行下去,永不停止。

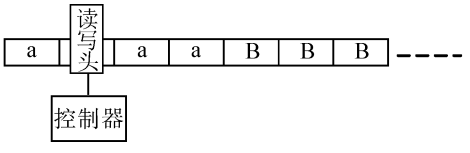

图 1 图灵机示意

Fig.1 The diagram of Turing machine

由于它具有无限的存储带,带上可读也可写,读写头可在带上左右移动等特点,所以相对于另外 2 种计算模型——有穷自动机[1](它的存储资源是有限的)和下推自动机[1](其栈内信息"先进后出"),它具有更强的描述求解某问题的算法的能力。例如:确定型图灵机可以判定语言 A(即,所有由 0 组成、长度为 2 的方幂的字符串),$A = \{0^{2^n} \mid n \geqslant 0\}$。该语言(语言是对问题的形式化描述[5])的算法描述[1] DTM M_2 如下:

对于输入字符串 w:

1)从左往右扫描整个带子,隔一个字符消去一个 0;

2)如果在第 1)步之后,带子上只剩下唯一的一个 0,则接受;

3)如果在第 1)步之后,带子上包含不止一个 0,并且 0 的个数是奇数,则拒绝;

4)读写头返回至带子的最左端;

5)转到第 1)步。

第 1)步每重复一次,就会消去带子上一半的 0。如果执行一次第 1)步后,带子上的 0 的个数是大于 1 的奇数,说明带子上原有的 0 的个数不可能是 2 的方幂,因此拒绝[1]。显然该算法 M_2 判定该语言。对于存储资源有限的有穷自动机是无法描述该算法的,而对于下推自动机,虽然具有无限存储的栈,但由于栈的"先进后出"的特点,无法实现隔一个字符消去一个 0,由此可见图灵机具有更强的描述问题的能力。

1.2.2 非确定型图灵机

确定型图灵机在计算过程中,下一步动作只有一种可能;而非确定型图灵机[5](nondeterministic Turing machine, NTM)是指在计算过程中,机器可以在多种可能性动作中选择一种继续进行,其计算是一棵树,不同分支对应机器不同的可能性[5]。如果计算的某个分支导致接受状态,则机器接受该输入。

非确定型图灵机并不对应任何实际的计算设备,它是一种假想的机器[7]。但它是一个有用的数学定义,有些问题更适于采用非确定型图灵机来描述,如计算机博弈问题,某走棋方在走棋时有多种走法可选择,这与非确定型图灵机的计算过程相同。

每个非确定型图灵机都等价于一个确定型图灵机[1]。也就是说这 2 种图灵机在能力上是等价的。文献[7]中给出如下定理:如果一个确定型图灵机去模拟一个 $t(n)$ 时间的非确定型图灵机,则所需时间为 $2^{O(t(n))}$。

1.3 映射可归约性

概念 问题 A 是映射可归约到问题 B 的,如果存在可计算函数 f,使得对每个 w, $w \in A \Leftrightarrow f(w) \in B$,记做 $A \leqslant_m B$。称 f 为 A 到 B 的归约[1]。

作用 它旨在将一个问题转化为另一个问题,且使得可以用第 2 个问题的解来解第 1 个问题。如果问题 A 可归约到问题 B,就可用 B 的解来解决问题 A[3]。例如:在一个城市认路,可以借助于城市地图,这样就将城市认路问题归约到城市地图问题。

2 时间复杂性

时间复杂性是指求解一个问题所需多少时间。对于固定规模的问题,其所需时间是一个常量;对于任意规模的问题,一般采用渐近记法[1],衡量其求解的复杂程度。如:函数 $f(n) = n^3 + n^2 + 1$,利用渐近记法,则该函数的时间复杂性为 $O(n^3)$(其中 O

表示一个常数),因为最高次项的影响占主导地位。另外,也可以通过证明,给一些问题的时间复杂性分类来说明该问题求解的难易程度。

2.1　确定型多项式时间复杂性

确定型多项式时间复杂性(deterministic polynomial time,P)是指可以在确定型图灵机上以多项式时间解决的问题的集合[6]。一般认为多项式时间算法已经足够快了,而指数时间的算法很少采用[1]。属于 P 复杂性类的问题,被认为是容易求解的[8]。属于 P 类的例子:2 个数是否是互素的问题,即 RELPRIME = $\{\langle a, b\rangle \mid a$ 与 b 互素$\}$[1];如果 1 是同时整除 2 个数的最大整数,则称这 2 个数是互素的。如 9 和 10 就是互素的。一种快速地判断 2 个数是否互素的方法是采用欧几里德算法(Euclidean algorithm)[9]来计算 2 个自然数 a 和 b 的最大公因子,记为 $\gcd(x,y)$[10],它是能够整除 2 个自然数的最大整数,如:$\gcd(12,15) = 3$。显然若 $\gcd(x,y) = 1$,说明 x 和 y 是互素的。欧几里德算法用伪代码描述[9]如下:

```
function gcd(a,b)
    while b ≠ 0
        t ← b("←"表示赋值)
        b ← a mod b("mod"表示求余运算)
        a ← t
    return a
```

该算法输出的值就是 a 和 b 的最大公因子。

算法 R 以 $\gcd(a,b)$ 为子程序求解 RELPRIME[1]。

R="对输入$\langle a, b\rangle$,a 和 b 是自然数:

1)运行子程序 $\gcd(a,b)$。

2)若结果为 1,就接受;否则,就拒绝。"

算法 R 的复杂性分析:显然,算法 R 的复杂性由欧几里德算法的复杂性决定。欧几里德算法的复杂性问题已经被彻底研究过了,其复杂性为 h^2(h 为 2 个自然数中最小的那个数的位数)[11],因此算法 R 的复杂性是 h^2,属于 P 类,即多项式时间算法。

2.2　非确定型多项式时间复杂性

非确定型多项式时间复杂性(nondeterministic polynomial time,NP)是指可以在非确定型图灵机上以多项式时间解决的问题的集合[6]。而前面在图灵机的阐述中提到过,一个确定型图灵机去模拟一个 $t(n)$ 时间的非确定型图灵机,所需时间为 $2^{O(t(n))}$,所以属于 P 类的问题与属于 NP 类的问题在求解的时间上存在指数级的差异。可见 NP 类的问题被怀疑是难解的[8]。到目前为止,还不清楚 NP 类问题是否存在有效的多项式时间算法[8]。但是去验证某个问题是否成立是可以在多项式时间完成的[1],比如:合数问题(当一个自然数是 2 个大于 1 的整数的乘积,称该自然数为合数),虽然还没有找到一个有效的多项式时间算法,但可以比较容易地去验证一个数是否是合数,这只需找到该数的一个因子[1],比如 24,它可以是 4 和 6 的乘积,因此它是合数。所以 NP 是具有多项式时间验证机(验证算法)的问题的集合[1]。

NP 类问题的举例　判定一个无向图是否包含指定大小的团的问题属于 NP[12],令 CLIQUE = $\{\langle G, k\rangle \mid G$ 是包含 k 团的无向图$\}$。

团的概念　如果 C 是无向图的顶点集 V 的子集,而且任意 2 个 C 中的顶点都有边连接,则称顶点集 C 为该无向图的团[2]。文献[1]给出一种判定团的非确定型多项式时间算法,如下:

N="对输入$\langle G, k\rangle$,其中 G 是一个无向图:

1)非确定性地选择 G 中 k 个结点的子集;

2)检查 G 是否包含链接 C 中结点的所有边;

3)若是,则接受;否则,拒绝。"

算法分析　该算法是运行在非确定型图灵机上,当然,现实中并没有这种设备,这是一种假想,假设存在这种机器,该算法在非确定型图灵机上以多项式时间运行,而确定型图灵机才是现实中计算机的雏形,所以该算法在计算机上将以指数时间运行,也就是说该算法是不可取的,说明团问题是难解的。

2.3　NP 的完全问题(NP-complete)

一个语言 B 属于 NP-complete,如果它满足 2 个条件:

1)B 属于 NP;

2)每个属于 NP 的问题在多项式时间内可以归约到 B[1]。

NP-complete 问题被认为是 NP 类中最难的[2],Cook[12]找到并证明了第一个属于 NP-complete 的问题[13],即 SAT[12](satisfiability,可满足性问题[14]),可以根据映射可归约性,证明某个问题属于 NP-complete。如果某个问题被证明属于 NP-complete,则说明该问题不存在有效的多项式时间算法,因此就可以不用将精力花费在寻找有效算法上了[1]。对于 NP-complete 的定义,如果 B 仅仅满足条件 2,这说明该问题本身并不属于 NP,可以说它是 NP-hard[1]。因而该问题可能更加难解。

NP-complete 例子　顶点覆盖问题[15]。

无向图 G 的顶点覆盖是一个顶点集合 V,使得 G 中的每一条边都接触 V 中的至少一个顶点[16]。

顶点覆盖问题旨在确定图中是否存在指定规模的顶点覆盖：

VERTEX-COVER = $\{\langle G, k\rangle \mid G$ 是具有 k 个结点的顶点覆盖的无向图$\}$[1]，顶点覆盖问题属于 NP-complete 类问题[12]。

对于证明某个问题 A 属于 NP-complete，需要依据 NP-complete 的定义，具体步骤如下：

1）证明 A 属于 NP；

2）需要找到一个已被证明是 NP-complete 的问题 B；

3）证明 B 可归约到 A；

4）证明该归约可在多项式时间内完成。

这里步骤 3）是难点，需要一些别出心裁的巧妙思维去构造一个合理的归约函数，文献[1]中，使用已经被证明属于 NP-complete 的问题 3SAT（3SAT，是一个合取范式，该公式中每一个子句都有 3 个文字，它是 Karp[12] 的 21 个 NP-complete 问题的其中之一），设该 3SAT 公式为 $(x \vee x \vee y) \wedge (\bar{x} \vee \bar{y} \vee \bar{y}) \wedge (\bar{x} \vee y \vee \bar{y})$，并且构造了一个归约，如图 2。

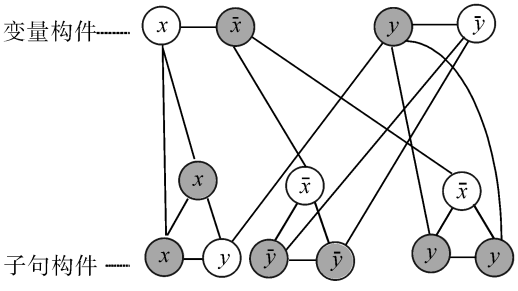

图 2 归约的完整结构图

Fig.2 The complete structure of reducibility

图中包括变量构件和子句构件，变量构件关系对变量的赋值；子句构件对应公式中的 3 个子句。图中共有 $2m + 3l$（m 表示公式中变量的数量，l 表示公式中子句的数量）个顶点，令指定的顶点覆盖规模 $k = m + l$，根据此 3SAT 公式，k 的值为 8，因此要从图中找到满足该 3SAT 公式的规模为 8 的顶点覆盖。

规则[1] 首先为 2 个变量选取满足公式的赋值，即 x 为假、y 为真，选择变量构件中相应的结点作为顶点覆盖中的结点，然后在各个子句构件中选择值不为真的 2 个结点作为顶点覆盖中的结点，其中第 3 个子句的 3 个变量的值都为真，则将不影响顶点覆盖的结点 $\bar{x}$ 去除，选择该子句中的另外 2 个结点，如图中有阴影的结点，这些结点的集合构成该图的一个顶点覆盖。因此说明该图中的顶点覆盖，可以满足该 3SAT 公式（即该公式的值为真），进而说明 3SAT 可归约到顶点覆盖，也就是说 3SAT 问题可以在顶点覆盖问题上求解，根据 NP-complete 定义可知，顶点覆盖问题属于 NP-complete。

2.4 指数时间复杂性（exponential time，EXPTIME）

该复杂性类是一些确定型问题的集合，这些问题可以使用确定型图灵机在 $O(2^{p(n)})$ 的时间内解决，这里的 $p(n)$ 代表的是 n 的某个多项式。属于该复杂性的问题，它的难度不小于 P、NP、NP-complete 以及空间复杂性类（PSPACE 和 PSPACE-complete）[3]。

EXPTIME 例子 G_3[17] 游戏，该游戏中的每个局面（position）是一个四元组（τ，W-LOSE(X,Y)，B-LOSE(X,Y)，α'），其中 $\tau \in \{W,B\}$，它表示当前走棋方，W - LOSE = $C_{11} \vee C_{12} \vee C_{13} \vee \cdots \vee C_{1p}$ 和 B-LOSE = $C_{21} \vee C_{22} \vee C_{23} \vee \cdots \vee C_{2q}$ 是属于 12DNF[18] 的布尔公式，其中每个 $C_{1i}(1 \leqslant i \leqslant p)$ 和 $C_{2j}(1 \leqslant j \leqslant q)$ 是最多 12 个文字之间的与运算，每个文字是集合 $X(Y)$ 中的一个变量，如：z 或 $\bar{z}$（变量 z 的非运算）；α' 是对公式 $X \cup Y$ 的一个赋值，如：$X = \{x_1, x_2, x_3\}$，$Y = \{y_1, y_2, y_3\}$，则 $X \cup Y = \{x1, x_2, x_3, y_1, y_2, y_3\}$，对 $X \cup Y$ 赋值，就是对该集合中的各个元素赋值为 0 或 1。

比赛双方交替进行，W 方或 B 方通过改变集合 X 和 Y 中的一个变量的值进行走棋。更精确地，W 方能够从局面（W，W-LOSE(X,Y)，B-LOSE(X,Y)，α'）下棋到局面（B，W-LOSE(X,Y)，B-LOSE(X,Y)，α'），当且仅当 α' 与 α'' 不同，并且在 α' 的赋值下，W-LOSE(X,Y) 的布尔值为 false。如果在 $W(B)$ 走了若干步后，公式 W-LOSE（B-LOSE）为 true，那么 $W(B)$ 失败。

在文献[17]中，给出了四元组中的公式 F（即 W-LOSE(X,Y) 或 B-LOSE(X,Y)）的所有可能的表示形式共有：$\mathrm{card}(V(F)) \leqslant \mathrm{e} \times |F| / \log(|F|)$，其中，e 为常量，$|F|$ 表示公式 F 被编码后的长度，$V(F)$ 表示公式 F 所有可能的表示形式所构成的集合，$\mathrm{card}(V(F))$ 表示集合中的元素总数。而对于表示 G_3 游戏局面的四元组，对于 player I（player II），通过对集合 $X(Y)$ 中的一个变量的赋值来完成下棋，而该变量的值有 2 个，即 0 或 1。所以，某个走棋方在下棋时，可走的局面有 2 个，即对于一个 F 的表示形式，由于四元组有 2 种可能，所以，从第一步开始一直到分出胜负，共有 $\mathrm{e} \times |F| / \log(|F|)$ 个 2 相乘，即 $2^{\mathrm{e} \times |F| / \log(|F|)}$。又用 τ 来识别走棋方，即第一步谁先走有 2 种可能，并且该四元组中有 2 个公式 F（即 W-LOSE(X,Y) 和 B-LOSE(X,Y)），因此，对一个输入 w，其中 n 表示 w 的长度，则 G_3 从开始

走棋到分出胜负,共产生的局面的数量为 $2\times 2^{2en/\log(n)}$,用渐近记法表示,为 $2^{O(n/\log(n))}$,是指数级的。即该游戏属于 EXPTIME 问题。

3　空间复杂性

空间复杂性是指求解一个问题所需多少空间。属于某空间复杂性类的问题,它的求解难度要大于属于时间复杂性类的问题,因为运行快的算法不可能消耗大量的空间,也就是说在每个计算步上最多能访问一个新单元,所以,任何在时间 $t(n)$ 内运行的机器最多能消耗 $t(n)$ 的空间[1]。与时间复杂性一样,也是采用渐近记法来计算求解问题的空间复杂性。另外,也可以通过证明,给一些问题的空间复杂性分类来说明该问题求解的困难程度。

3.1　确定型多项式空间复杂性

确定型多项式空间复杂性(deterministic polynomial space,PSPACE)是指能够被确定型图灵机在多项式空间内解决的问题的集合[19]。属于 PSPACE 问题举例:全量词化的布尔公式(true qualified boolean formula,TQBF)[6],形如:$\forall x\exists y(x\wedge y)$,该公式的含义是:对于任意的 x,存在 y,使得 $x\wedge y$ 为真。全量词化的布尔公式是指公式中出现的所有变量,都有量词对其约束[6]。TQBF 问题就是判定一个全量词化的布尔公式是真或假的问题,它属于 PSPACE。其多项式空间算法[1]如下:

T = 对输入的全量词化的布尔公式:

1)若该公式不含量词,则它是一个只有常数的表达式。计算公式的值,若为真,则接受;否则,则拒绝。

2)若公式等于 $\exists x\varphi$,在 φ 上递归地调用 T,首先用 0 替换 x,然后用 1 替换 x。只要有一个结果是接受,则接受;否则,拒绝。

3)若 φ 等于 $\forall x\varphi$,在 φ 上递归调用 T,首先用 0 替换 x,然后用 1 替换 x。若 2 个结果都是接受,则接受;否则,拒绝。

算法分析　该算法模拟的是量词化布尔公式的计算过程,每一次递归就是对公式中最左侧的一个量词所约束的变量赋值(假设该公式为前束范式[18])。递归的深度就是该公式 φ 中包含的变量的个数,设为 m,因此每层只需存储一个变量,所以该算法的空间消耗是 $O(m)$。因此该算法属于多项式空间算法。

3.2　NPSPACE

它是指能够被非确定型图灵机在多项式空间内解决的问题的集合。空间与时间不同,它可以被重复使用,根据萨维奇定理[20],如果一台非确定型图灵机能够利用 $f(n)$ 空间解决某个问题,那么一台确定型图灵机能够在至多 $f^2(n)$ 空间解决相同的问题。由该定理可得推论:NPSPACE = PSPACE[1]。

3.3　PSPACE 的完全问题

一个语言 B 属于 PSPACE 的完全问题[1](PSPACE-complete),如果它满足 2 个条件:

1)B 属于 PSPACE;

2)每个属于 PSPACE 的问题在多项式时间内归约到 B。

PSPACE-complete 属于 PSPACE 类中最困难的问题,和 NP-complete 的定义类似,如果 B 仅仅满足条件 2,说明 B 不属于 PSPACE,可以说它是 PSPACE-hard,那么 B 可能更加难解。常见的属于 PSPACE-complete 的问题有 QBF[6](量词的布尔公式)、formula game[1](公式博弈)、棋类游戏[1](chess game)等。

举例　广义地理学游戏[1](generalized geography game)是一种地理学游戏,选手们轮流给出世界各地的城市名称。每一座选中的城市的首字母必须与前一座城市的尾字母相同,城市的名称不允许重复,游戏从某个指定的城市开始,直到某一方不能延续为止。对于证明某问题属于 PSPACE-complete 的过程与 NP-complete 的证明类似,首先要证明该问题属于 PSPACE,文献[1]中给出了一个多项式空间的算法,证明了该问题可在多项式空间求解。然后选择了一个已被证明为 PSPACE-complete 类的问题如 formula game(实际上,它就是 TQBF,只是假设有 2 个选手交替为每个变量赋值)并独出心裁地构造了一个归约,如图 3[1]。

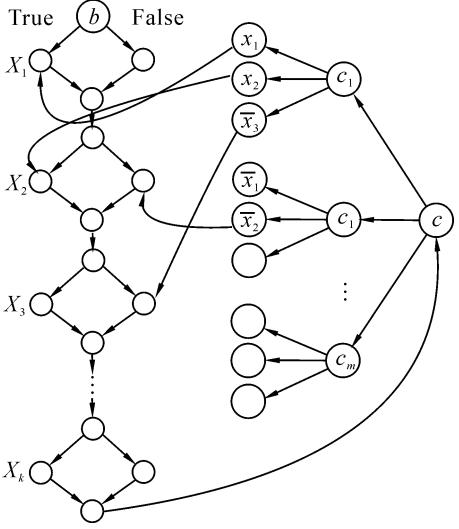

图 3　归约的全部结构

Fig.3　The complete structure of reducibility

这是一个广义地理学的有向图,在该图上模拟进行 formula game,设布尔公式为 $\exists x_1 \forall x_2 \exists x_3 \forall x_4 \cdots \exists x_k [(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3} \vee \cdots) \wedge \cdots \wedge ()]$,从图中顶端的 b 点开始,2 个选手 I 和 II 分别给变量选择赋值(设 x_1, x_2, …, x_k 处的左侧结点表示 1,右侧表示 0),最终某一方因无法选择结点(根据广义地理学游戏的规则,图中每个结点只能选择一次)而输掉比赛,即比赛结束,该归约确保:如果公式为 true,则选手 I 获胜;否则选手 II 获胜。这说明在广义地理学游戏上模拟进行 Formula Game,并最终能够求解。因此证明了 Formula Game 可归约到广义地理学游戏。结合 PSPACE-complete 的定义,可知广义地理学问题属于 PSPACE-complete。

4 各种复杂性类之间的关系

前文介绍了常见的时间复杂性类和空间复杂性类,下面来讨论这些复杂性类中哪个相对简单,哪个相对更加难解。

4.1 P 与 NP 的关系

P 是在确定型图灵机上以多项式时间求解的问题的集合;NP 是在非确定型图灵机上以多项式时间求解的问题的集合;NP 问题若在确定型图灵机上求解,就需要用确定型图灵机去模拟非确定型图灵机的运行过程,这种模拟所需要的时间是指数级的,因此,普遍认为 NP 问题要难于 P 类问题,即 P ⊆ NP[3]。但是否 P = NP ,有许多学者都在致力于这方面的研究工作,如果 P = NP ,那么所有 NP 问题都能够找到有效的多项式时间算法,也就是说许多难解的问题都可以迎刃而解。但是还没有成功[1],大多数学者还是接受 P ≠ NP [2],COOK[13] 证明了第一个属于 NP-complete 的问题——可满足性问题(satisfiability, SAT),同时根据映射可归约性,在某种意义上,是将所有 NP 问题归为了同一个问题[2]。这为研究 P 和 NP 是否相等的学者们提供了一个方便,即只要证明一个 NP-complete 问题存在有效的多项式时间算法,那么所有的 NP 问题都存在多项式时间算法[2]。

4.2 时间复杂性与空间复杂性的关系

前面已经提到过,运行快的算法不可能消耗大量的空间,也就是说在每个计算步上最多能访问一个新单元,所以,任何在时间 $t(n)$ 内运行的机器最多能消耗 $t(n)$ 的空间[1]。因此空间复杂性类(PSPACE 和 NPSPACE)的问题要难于时间复杂性(P 和 NP)类的问题。

4.3 空间复杂性与 EXPTIME 的关系

根据文献[1]中的引理:设 M 是有 q 个状态和 g 个带符号的 LBA(一种受限制的图灵机)。对于长度为 n 的带子,M 恰有 qng^n 个不同的格局。可推出,需要空间为 $f(n)$ 的 PSPACE 问题,根据渐近记法,最多有 $f(n) \times 2^{O(f(n))}$ 个不同的格局,假设处理每个格局需要一个时间单元,在最差的情况下(遍历到第 $f(n) \times 2^{O(f(n))}$ 个格局,图灵机输出接受或拒绝,即处于停机状态),求解该问题所用时间等于 $f(n) \times 2^{O(f(n))}$。即该 PSPACE 问题必定在时间 $f(n) \times 2^{O(f(n))}$ 内运行,因此 PSPACE ⊆ EXPTIME[3]。所以,各个计算复杂性类之间的关系为 P ⊆ NP ⊆ PSPACE = NPSPACE ⊆ EXPTIME[1],如图 4。

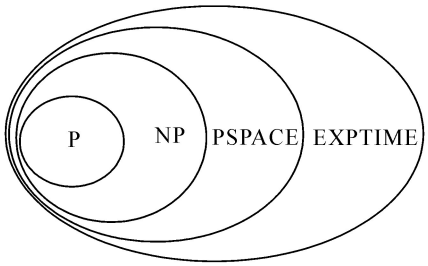

图 4 各复杂性类之间的关系

Fig.4 The relations of some complexities

5 结束语

计算复杂性是计算理论的一个分支,它是一个理论性很强的计算机科学问题,极具挑战性。通过研究问题的计算复杂性,可以了解该问题求解的难易程度,如果是难解的(如 NP-complete、PSPACE-complete),即该问题不存在有效的多项式时间求解算法,则不必将大量的精力放在寻找该问题的有效算法上,从而提高了工作效率。论述了计算理论的一些基本概念,并通过例子论述了各个计算复杂性类的证明方法和当前计算复杂性理论的研究现状。最后详细分析了各个复杂性类之间的关系。在该领域中,还存在一些尚待解决的难题,比如:P 是否等于 NP,许多学者都在致力于这方面的研究工作,如果 P = NP ,那么所有 NP 问题都能够找到有效的多项式时间算法,也就是说许多难解的问题都可以迎刃而解。但是这个等式还没有被成功证明;各个复杂性类之间的包含关系是否正确也有疑问,这些包含关系是否存在着真包含,这些都是国内外学者们今后重点研究的课题。

参考文献:

[1]SIPSER M. Introduction to the theory of computation[M]. 2nd ed. Beijing: China Machine Press, 2006: 265-323.

[2]DASGUPTA S, PAPADIMITRIOU C H, VAZI-RANI U V. Algorithm[M]. New York: McGraw-Hill Science/Engineering/Math, 2006: 257-264.

[3]PAPADIMITRIOU C. Computational complexity[M]. Reading, Massachusetts: Addison-Wesley, 1994: 491-493.

[4]朱一清.可计算性与计算复杂性[M].北京:国防工业出版社,2006: 44-45.

[5]堵丁柱.计算复杂性导论[M].北京:高等教育出版社, 2000: 21-22.

[6]ARORA S, BARAK B. Computational complexity[M]. New York: Cambridge University Press, 2007: 24-85.

[7]陈志平,徐宗本.计算机数学——计算复杂性理论与 NPC、NP 难问题的求解[M].北京:科学出版社,2001: 81-83.

[8]NP [DB/OL]. [2013-10-17]. http://zh.wikipedia.or-g/wiki/NP.

[9]COPRIME[DB/OL].[2013-10-17].http://zh.wikiped-ia.org/wiki/COPRIME.

[10]GRAHAM R L, KNUTH D E, PATASHNIK O. Concrete mathematics[M]. Massachusetts: Addison-Wesley, 1989: 107-110.

[11]HONSBERGER R. Mathematical gems II[M]. Washington, DC: The Mathematical Association of America, 1976: 54-57.

[12]KARP R M. Reducibility among combinatorial problems [Z]. New York: Plenum Press, 1972: 85-103.

[13]COOK S A. The complexity of theorem proving procedures [C]//Proceedings of the Third Annual ACM Symposium on Theory of Computing. New York, USA, 1971: 151-158.

[14]ZHANG Lintao. Searching for truth: techniques for satisfiability of boolean formulas[D]. Princeton: Princeton University, 2003: 10-14.

[15]CORMEN T H, LEISERSON C H, RIVEST R L, et al. Introduction to algorithms[M]. 2nd ed. Cambridge, Massachusetts:MIT Press, 2009: 1108-1110.

[16]VERTEX COVER [DB/OL]. [2013-10-19]. http://zh.wikipedia.org/wiki/VERTEX COVER.

[17]STOCKMEYER L J, CHANDRA A K. Provably difficult combinatorial games[J]. SIAM J Compuf, 1979 8(2): 151-174.

[18]罗森.离散数学及其应用[M].北京:机械工业出版社, 2011: 21-30.

[19]PSPACE[DB/OL].[2013-10-20].http://zh.wikipedia.org/wiki/PSPACE.

[20]SAVITCH W J. Relationships between nondeterministic and deterministic tape complexities[J]. Journal of Computer and System Sciences, 1970 4(2): 177-192.

作者简介:

高强,男,1980 年生,讲师,博士研究生,主要研究方向为机器博弈、计算复杂性理论。

徐心和,男,1940 年生,教授,博士生导师,中国人工智能学会常务理事,主要研究方向为控制理论与应用、系统仿真、智能机器人、机器博弈等,主持完成国家自然科学基金项目、863 基金项目、国家“八五”、“九五”攻关课题 13 项,其中 8 项通过省、部级鉴定,获科技进步奖国家三等 1 项,省部级科技进步奖多项。发表学术论文 300 余篇。