《算法设计与分析》

第九章 计算复杂性分析

马丙鹏

2023年12月04日

中国科学院大学
University of Chinese Academy of Sciences

第九章 计算复杂性分析

- 9.1 复杂性分类
- 9.2 P 类问题和 NP 类问题
- 9.3 NP 完全问题

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

□证比求易：

- ➢求解一个问题往往比较困难，但验证一个问题相对来说比较容易。

□从是否可以被验证的角度，计算复杂性理论将难解问题进一步划分：

① NP 问题

② 非 NP 问题

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

- 判定问题（decision problem）是要求回答“yes”或“no”的问题。
- 例如，停机问题就是一个判定问题，但是，停机问题不能用任何计算机算法求解，所以，并不是所有的判定问题都可以在计算机上得到求解。
- 在实际应用中，很多问题以求解或计算的形式出现，但是，大多数问题可以很容易转化为相应的判定问题
- 例5:
  - 排序问题：将一个整数序列调整为非降序排列
  - 排序问题的判定形式：给定一个整数序列，是否可以按非降序排列。

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

□例6:

- ➢求图中从结点A到结点B的最短路径。该问题可以转化成如下形式:
  - ✓从A到B是否有长度为1的最短路径? NO
  - ✓从A到B是否有长度为2的最短路径? NO
  - ✓……? NO
  - ✓从A到B是否有长度为k-1的最短路径? NO
  - ✓从A到B是否有长度为k的最短路径? Yes
  - ✓如果问到了k的时候，回答了Yes，则停止发问。我们可以说从结点A到结点B的最短路径长度为k。

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

□例7:

- ➢着色问题:
  - ✓给定无向连通图G=(V, E)，求图 G 的最小色数 k，使得用 k 种颜色对 G 中的所有顶点着色，可使任意两个相邻顶点着色不同。
- ➢图着色问题的判定形式:
  - ✓给定无向连通图 G=(V, E)和一个正整数 k，是否可以用 k 种颜色为图 G 的所有顶点着色，使得任意两个相邻顶点着色不同。

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

□例8:

- ➢哈密顿回路问题:
  - ✓在图 G=(V, E)中，从某个顶点出发，求经过所有顶点一次且仅一次再回到出发点的回路。
- ➢哈密顿回路问题的判定形式:
  - ✓在图 G=(V, E)中，是否存在一个回路经过所有顶点一次且仅一次然后回到出发点。

9.2 P 类问题和 NP 类问题

■ 1. 判定问题

□例9:

- ➢**TSP问题**:
  - ✓在一个带权图G=(V, E)中，求经过所有顶点一次且仅一次再回到出发点，且路径长度最短的回路。
- ➢**TSP问题的判定形式**:
  - ✓给定带权图G=(V, E)和一个正整数 k，是否存在一个回路经过所有顶点一次且仅一次再回到出发点，且路径长度小于等于 k。

9.2 P 类问题和 NP 类问题

■ 2. 确定性算法与P类问题

- 定义9.1 设 A 是求解问题 Π 的一个算法，如果在算法的整个执行过程中，每一步只有一个确定的选择，并且对于同一输入实例运行算法，所得的结果严格一致，则称算法 A 是确定性（determinism）算法。
- 定义9.2 如果对于某个判定问题 Π，存在一个非负整数 k，对于输入规模为 n的实例，能够以$O(n^k)$的时间运行一个确定性算法，得到 yes或 no的答案，则该判定问题 Π 是 P 类问题（polynomial）。
- P 类问题是具有多项式时间的确定性算法来求解的判定问题。事实上，所有易解问题都属于 P 类问题。
- 采用判定问题定义 P 类问题，主要是为了给出 NP 类问题的定义。

9.2 P 类问题和 NP 类问题

■ 2. 确定性算法与P类问题

□例10:

- 给出一个有n个整数的表，它们是否按降序排列?
  答:只要检查表中相邻二个数即可，运行时间为O(n)
- 给出二个整数集合，它们的交集是否为空?
  答:先将所有整数排序，然后检查相邻二数是否相等，显然运行时间为$O(n\log_2 n)$。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- 有些计算问题是确定性的，例如“加减乘除”，你只要按照公式推导，按部就班一步步进行，就可以得到结果。
- 但是，有些问题无法按部就班直接进行计算的。
- 例如“找大质数”问题，已知目前最大质数，那么下一个大质数应该是多少呢?有没有一个公式可以一步步推算出来，显然这样的公式是没有的。
- 这种问题的答案，是无法直接计算得到的，只能通过“猜算”来得到结果，这就是非确定性问题。

9.2 P 类问题和 NP 类问题

- 3. 非确定性算法与NP类问题
  - 这些问题通常有个算法，它不能直接告诉你答案是什么，但可以告诉你，某个可能的结果是正确的还是错误的。
  - 这个可以告诉你“猜算”的答案正确与否的算法，称为非确定性算法。
  - 假如“猜算”可以在多项式时间内得到，那么该问题称作“多项式非确定性问题”

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□ 定义9.3 设 A 是求解问题 Π 的一个算法，如果算法 A 采用如下猜测并验证的方式，则称算法 A 是非确定性（nondeterminism）算法：

① 猜测阶段：对问题的输入实例产生一个任意字符串 ω，对于算法的每一次运行，串 ω 的值可能不同，因此，猜测以一种非确定的形式工作。

这个字符串，它可能对应输入实例的个解，也可以不对应解。

事实上，它甚至可能不是所求解的合适形式，它可能在非确定算法的不同次运行中不同。

它仅要求在多项式步数内产生这个串，即在$O(n^i)$时间内，这里$n=|x|$，i是非负整数。对于许多问题，这一阶段可以在线性时间内完成。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□定义9.3 设 A 是求解问题 Π 的一个算法，如果算法 A 采用如下猜测并验证的方式，则称算法 A 是非确定性（nondeterminism）算法：

② 验证阶段：用一个确定性算法验证两件事：

- ✓ 首先，检查在猜测阶段产生的串 ω 是否是合理的形式，如果不是，则算法停下来并得到 no；
- ✓ 另一方面，如果串 ω 是合理的形式，再验证 ω 是否是问题的解，如果是问题的解，则算法停下来并得到 yes，否则，算法停下来并得到 no
- ✓ 我们也要求这个阶段在多项式步数内完成，即在 $O(n^j)$ 时间内，这里j是一个非负整数。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- 非确定性算法的运行时间是由猜测阶段和验证阶段二部分耗费时间组成，因此它是$O(n^k)=O(n^i)+O(n^j)$，k是某个非负整数。
- 非确定性算法不是一个实际可行的算法。
- 引入非确定性算法的目的在于给出 NP 类问题的定义，从而将验证过程为多项式时间的问题归为一类进行研究。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□ 例11:求大整数n的一个真因数(即1和n本身以外的一个因数，并且该因数是素数)。

- 这是一个至今未能找到有效算法的难解问题。对于难解问题，人们除了使用传统型计算方法外，又想出了另一种类型的计算方法，该方法称为“非确定性算法”
- 传说从前有位年轻的国王，想求出整数190 334 261 410 902 619的一个真因素。
  他用2、3、5、7、11、13、.....这些素数逐一去试，化了九牛二虎之力也无法算出，于是他把这个问题交给了宰相。
  国王用的计算方法称为“穷举法”，穷举法属“确定性算法”

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- 例11:求大整数n的一个真因数(即1和n本身以外的一个因数，并且该因数是素数)。
  - 宰相猜想这个数可能是9位整数，于是宰相把全国成年百姓编成十个军，每个军有十个师，每个师有十个旅每个旅有十个团，每个团有十个营，每个营有十个连每个连有十个排，每个排有十个班，每个班有十个组每个组有十个人，于是每个成年百姓都具有一个9位的番号。
  - 然后把题目发下去，让每个成年百姓用自己的番号去除“19033426141090261 9”这个数，若除尽了就把番号报上来。

9.2 P 类问题和 NP 类问题

- 3. 非确定性算法与NP类问题
  - 例11:求大整数n的一个真因数(即1和n本身以外的一个因数，并且该因数是素数)。
    - 很快就有二个人报上了结果，即“436273009”与“436273291”。经国王验证，这二个整数都是素数，并且这二个整数的积就是题目所给的18位整数。

军师旅 团营连 排组人 军师旅 团营连 排组人

436 273 009 * 436 273 291

=

190 334 261 410 902 619

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□这个故事说明算法分析中最基本问题

- ➢求大整数的真因数不能用多项式时间求解，但是验证某数是否是大整数的真因数可以用多项式时间完成。所以，求大整数的真因数要比验证真因数要难得多
- ➢国王用得是确定性计算方法(穷举法)，所以计算很快变得无法进行下去；
- ➢宰相用得是非确定性计算方法，首先猜想，然后验证。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- 定义9.4 如果对于某个判定问题 Π，存在一个非负整数 k，对于输入规模为 n 的实例，能够以$O(n^k)$的时间运行一个非确定性算法，得到 yes 或 no 的答案，则该判定问题 Π 是 NP（nondeterministic polynomial）类问题。
- 对于 NP 类判定问题，关键是该问题存在一个确定性算法，并且能够以多项式时间检查和验证在猜测阶段所产生的答案。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□例如，考虑TSP问题的判定形式，假定算法 A 是求解 TSP 判定问题的非确定性算法，

（1）算法 A 以非确定的形式猜测一个路径是 TSP 判定问题的解；

（2）用确定性算法检查这个路径是否经过所有顶点一次且仅一次并返回出发点，

① 如果答案为 yes，则继续验证这个回路的总长度是否 ≤ k，如果答案仍为 yes，则算法输出 yes；

② 否则算法输出 no。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- NP类问题是难解问题的一个子集。
- 并不是任何一个在常规计算机上需要指数时间的问题（即难解问题）都是 NP 类问题。
- 例如，汉诺塔问题不是 NP类问题，因为对于 n 层汉诺塔需要$O(2^n)$步输出正确的移动过程，一个非确定性算法不能在多项式时间猜测并验证一个答案。
- P类和NP类的区别
  - P类是一个判定问题类，这些问题可以用一个确定性算法，在多项式时间内判定或解出；
  - NP类是一个判定问题类，这些问题可以用一个确定性算法，在多项式时间内检查或验证它们的解

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

- P 类问题存在多项式时间的确定性算法进行判定或求解，显然也可以构造多项式时间的非确定性算法进行判定。因此，**P 类问题属于 NP 类问题**，即

$$P \subseteq NP$$

- NP 类问题存在多项式时间的非确定性算法进行猜测并验证，但是，不一定能够构造一个多项式时间的确定性算法进行判定或求解。因此，人们猜测**P≠NP**。但是，这个不等式是成立还是不成立，至今没有得到证明。

9.2 P 类问题和 NP 类问题

■ 3. 非确定性算法与NP类问题

□ P等于NP的结果

- 一大批耳熟能详的游戏，如扫雷、俄罗斯方块、超级玛丽等，人们将为它们编写出高效的算法，使得电脑玩游戏的水平无人能及
- 整数规划、旅行商问题等许多运筹学中的难题会被高效地解决，这个方向的研究将提升到前所未有的高度
- 蛋白质的折叠问题也是一个NPC 问题，新的算法无疑是生物与医学界的一个福音对人类疾病预防和制药水平将会产生极大的促进
- 现实中用的好多加密算法，核心都是归结到NP 不等于 P 上的。如果我们找到了多项式时间算法，很多密码的破解时间会被大大减少，现在的网银将不再安全

第九章 计算复杂性分析

- 9.1 复杂性分类
- 9.2 P 类问题和 NP 类问题
- 9.3 NP 完全问题

9.3 NP 完全问题

■ 1. 问题变换

□ 定义9.5 假设问题 Π' 存在一个算法 A，对于问题 Π' 的输入实例 I'，算法 A 求解问题 Π' 得到一个输出 O'。另外一个问题 Π 的输入实例是 I，对应于输入 I，问题 Π 有一个输出 O，则问题 Π 变换到问题 Π' 是一个三步的过程：

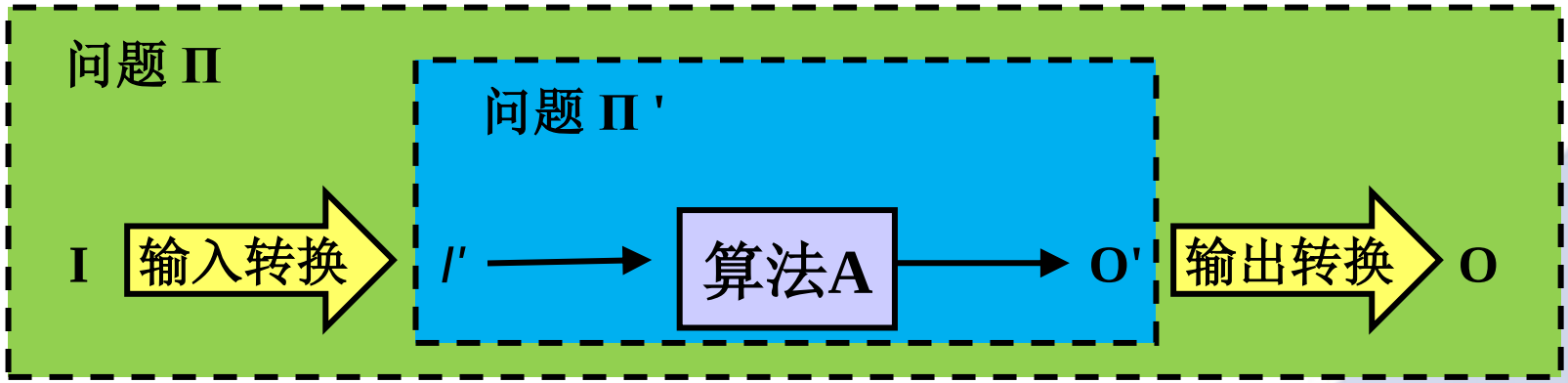
① 输入转换：把问题 Π 的输入 I 转换为问题 Π' 的输入 I'；

② 问题求解：对问题 Π' 应用算法 A 产生一个输出 O'；

③ 输出转换：把问题 Π' 的输出 O' 转换为问题 Π 对应于输入 I 的正确输出。

9.3 NP 完全问题

■ 1. 问题变换



9.3 NP 完全问题

■ 1. 问题变换

- 若在O(τ(n))的时间内完成上述输入和输出转换，则称问题 Π 以 τ(n) 时间变换到问题 Π'，记为 $\Pi \propto_{\tau(n)} \Pi'$，其中，n 为问题规模；
- 若在多项式时间内完成上述输入和输出转换，则称问题 Π 以多项式时间变换到问题 Π'，记为$\Pi \propto_{p} \Pi'$。

9.3 NP 完全问题

■ 1. 问题变换

- 例如，算法 A 可以求解排序问题，输入 I' 是一组整数$X=(x_1, x_2, \ldots, x_n)$，输出 O' 是这组整数的一个排列 $x_{i1} \le x_{i2} \le \ldots \le x_{in}$。
- 考虑配对问题，输入I是两组整数 $X=(x_1, x_2, \ldots, x_n)$ 和 $Y=(y_1, y_2, \ldots, y_n)$，输出 O 是两组整数的元素配对，即 X 的最小值与 Y 的最小值配对，X 的次小值与 Y 的次小值配对，依此类推。
- 经过下述三步，将配对问题变换到排序问题：

9.3 NP 完全问题

■ 1. 问题变换

① 把配对问题的输入 I 转化为排序问题的两个输入 I_1' 和 I_2'；

② 排序这两组整数，即应用算法A对两个输入I_1'和I_2'分别排序得到两个有序序列O_1' 和 O_2'；

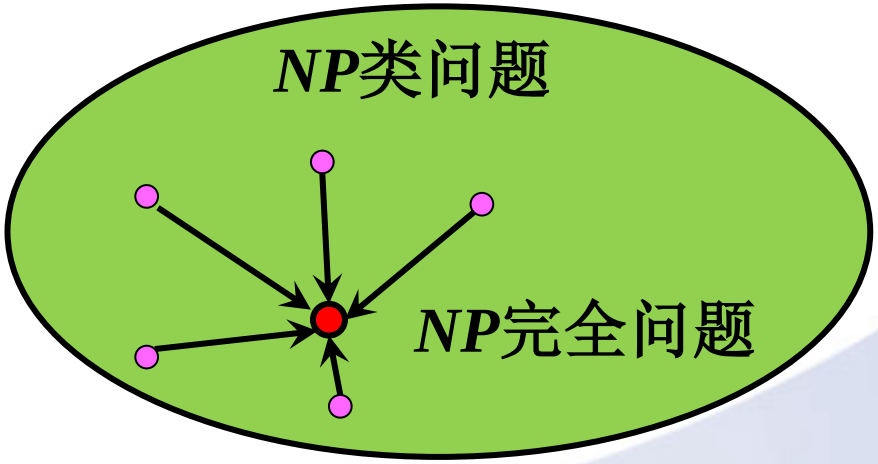
③ 把排序问题的输出 O_1' 和 O_2' 转化为配对问题的输出 O，这可以通过配对每组整数的第一个元素、第二个元素、……来得到。

□ 问题变换的主要目的不是给出解决一个问题的算法，而是给出比较两个问题计算复杂性的一种方式。

9.3 NP 完全问题

■ 2. NP完全问题的定义

□定义9.6 令 Π 是一个判定问题，如果问题 Π 属于 NP 类问题，并且对 NP 类问题中的每一个问题 Π'，都有 $\Pi' \propto_p \Pi$，则称判定问题 Π 是一个NP完全问题（NP complete problem），也称 NPC 问题。

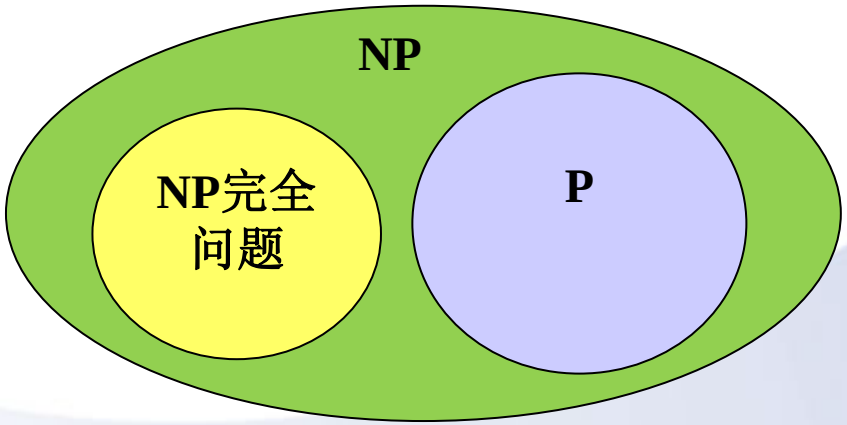


9.3 NP 完全问题

■ 2. NP完全问题的定义

□NP hard 问题

- ➢令 Π 是一个判定问题，对 NP 类问题中的每一个问题 Π'，都有 $\Pi' \propto_p \Pi$，则称判定问题 Π 是一个 NP难问题（NP hard problem）。
- ➢NP 完全问题是 NP 中最难的问题.



9.3 NP 完全问题

■ 2. NP完全问题的定义

- NP 完全问题是 NP 类问题中最有代表性的一类问题，NP 完全问题有一个重要性质：
  - 如果一个 NP 完全问题能够在多项式时间内得到求解，那么 NP 类问题中的每一个问题都可以在多项式时间内得到求解。
- 目前还没有一个 NP 完全问题发现有多项式时间算法。
- 这些问题也许存在多项式时间算法，因为计算机科学是相对新生的科学，肯定还会有新的算法设计技术有待发现；
- 这些问题也许不存在多项式时间算法，但目前缺乏足够的技术来证明这一点。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

- 证明一个判定问题 Π 是 NP 完全问题需要经过两个步骤：

① 证明问题 Π 属于 NP 类问题，也就是说，可以在多项式时间以非确定性算法实现验证；

② 证明一个已知的 NP 完全问题能够在多项式时间变换为问题 Π。

- 1971年，Cook在Cook定理中证明了SAT可满足问题是NP完全的。
- 1972年，Karp证明了十几个问题都是NP完全的。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

□SAT问题（boolean satisfiability problem）。

- ➢也称为合取范式的可满足问题，来源于许多实际的逻辑推理及应用。对于合取范式 $A = A_1 \wedge A_2 \wedge \ldots \wedge A_n$，子句 $A_i = a_1 \vee a_2 \vee \ldots \vee a_k$（$1 \le i \le n$），$a_i$ 为某一布尔变量或布尔变量的非，称为文字。一个子句是文字的析取。
- ➢SAT 问题是指：是否存在一组对所有布尔变量的赋值（true或false），使得合取范式 A 的值为真。
- ➢例如

$f = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_3 \vee x_4 \vee \bar{x}_5) \wedge (x_1 \vee \bar{x}_3 \vee x_4)$

当x_1和x_3都为真、其余文字任意赋值时，f值为真。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

□最大团问题（maximum clique problem）

- ➢图 G=(V, E)的团是图 G 的一个完全子图，该子图中任意两个互异的顶点都有一条边相连。团问题是对于给定的无向图 G=(V, E)和正整数 k，是否存在具有 k 个顶点的团。

□图着色问题（graph coloring problem）。

- ➢给定无向连通图 G=(V, E) 和正整数k，是否可以用 k 种颜色对 G 中的顶点着色，使得任意两个相邻顶点的着色不同。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

- □哈密顿回路问题（hamiltonian cycle problem）。
  - ➢在图 G=(V, E)中，是否存在经过所有顶点一次且仅一次并回到出发顶点的回路。
- □TSP问题（traveling salsman problem）。
  - ➢给定带权图 G=(V, E) 和正整数 k，是否存在一条哈密顿回路，其路径长度小于等于 k。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

□顶点覆盖问题（vertex cover problem）。

- ➢设图 G=(V, E)，V' 是顶点 V 的子集，若图 G 的任一条边至少有一个顶点属于 V'，则称 V' 为图 G 的顶点覆盖。顶点覆盖问题是对于图 G=(V, E) 和正整数 k，是否存在顶点 V 的一个子集 V'，使得图 G 的任一条边至少有一个顶点属于 V' 且 |V'| ≤ k。

□子集和问题（sum of subset problem）。

- ➢给定一个整数集合 S 和一个正整数 k，判定是否存在 S 的一个子集 S'，使得 S' 中的整数之和等于 k。

9.3 NP 完全问题

- 3. 基本的NP完全问题
  - 库克在1971年找到了第一个NP完全问题，即可满足性问题。此后，人们又陆续发现很多NP完全问题
  - 到目前为止已经找到了大约4000个NP完全问题，

9.3 NP 完全问题

■ 3. 基本的NP完全问题

- 人们发现，所有的NP完全问题都可以在多项式时间内转换到可满足性问题。只要它们中的一个，如果存在“多项式时间确定性算法”的话，那么NPC问题中的所有问题，都可以用“多项式时间确定性算法”来求解
- 人们于是就猜想，既然NPC问题的所有可能解，都可以在多项式时间内验证，对于此类问题是否存在一个确定性算法，可以在多项式时间内直接给出解呢?这就是著名的NP=P?的猜想。这是21世纪计算机科学家向数学家提出的世界难题

9.3 NP 完全问题

■ 3. 基本的NP完全问题

- 解决NP=P?的猜想无非两种可能。一种是找到一个这样的算法，只要针对某个特定NP完全问题找到一个算法所有这类问题都可以迎刃而解了，因为它们可以转化为同一个问题。
- 另外的一种可能，就是这样的算法是不存在的。那么就要从数学理论上证明它为什么不存在。
- 前段时间轰动世界的一个数学成果，是几个印度人提出了一个新算法。该算法可以在多项式时间内，证明某个数是或者不是质数。而在这之前，人们认为质数的证明是个非多项式问题。可见，有些看来好象是非多项式问题，其实是多项式问题，只是人们目前还不知道它的多项式解而已。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

AKS算法（Agrawal-Kayal-Saxena算法）是一种用于确定一个数是否是素数的算法。它是由Manindra Agrawal，Neeraj Kayal和Nitin Saxena于2002年发明的，是第一个在多项式时间内证明素性的算法。这个算法是基于一个数学定理，即对于任意一个素数p，它的莫比乌斯函数（也称作欧拉函数）在p处的值是-1。 AKS算法首先检查这个性质是否成立，然后使用数学归纳法证明数字是素数。

输入：整数$n > 1$。

1. 检查n是否为完美幂：如果$n = a^b$对于整数$a > 1$和$b > 1$，则输出*composite*。
2. 找到满足$\mathrm{ord}_r(n) > (\log_2 n)^2$的最小$r$。（如果$r$和$n$不是互质的，则跳过此$r$）
3. 对于所有 $2 \leq a \leq \min(r, n-1)$，检查$a$不整除$n$：如果$a \mid n$对于某些 $2 \leq a \leq \min(r, n-1)$，输出*composite*。
4. 如果$n \leq r$，输出*素数*。
5. 对于$a = 1$至$\left\lfloor \sqrt{\varphi(r)} \log_2(n) \right\rfloor$做

   如果 $(X + a)^n \neq X^n + a \pmod{X^r - 1, n}$, 输出*复合*;
6. 输出*质数*。

这里$\mathrm{ord}_r(n)$是 n 模 r 的乘法*阶数*，$\log_2$是二进制对数，并且 $\varphi(r)$是r的 欧拉总函数。

9.3 NP 完全问题

■ 3. 基本的NP完全问题

□学习NP完全理论的意义

- ➢今天，NP-complete一词已经成为算法设计者在求解规模大而又复杂困难的问题时所面临的某种难以逾越的深渊的象征。
- ➢2000年初，美国克雷数学研究所的科学顾问委员会选定了七个“千年大奖问题”，该研究所的董事会决定建立七百万美元的大奖基金，每个“千年大奖问题”的解决都可获得百万美元的奖励。NP完全问题排在百万美元大奖的首位，足见它的显赫地位和无穷魅力。
- ➢在科学和很多工程技术领域里，常常遇到的许多有重要意义而又没有得到很好解决的难题是NP完全问题，这类问题的数目不断增加。

9.3 NP 完全问题

■ 4. NP类问题的计算机处理

- NP 类问题是计算机难以处理的，但在实际应用中却经常会遇到，因此，人们提出了解决 NP 类问题的各种方法。

（1）采用先进的算法设计技术。

- 当实际应用中问题规模不是很大时，采用动态规划法、回溯法、界限剪枝法等算法设计技术还是能够解决问题的。

9.3 NP 完全问题

■ 4. NP类问题的计算机处理

（2）充分利用限制条件。

- 许多问题，虽然理论上归结为一个 NP 类问题，但实际应用中可能包含某些限制条件，有些问题增加了限制条件后，可能会改变性质。例如，
  - ✓0/1背包问题中，限定物品的重量和价值均为正整数；
  - ✓图着色问题中，限定图为可平面图；
  - ✓TSP问题中，限定边的代价满足三角不等式，等等。

9.3 NP 完全问题

■ 4. NP类问题的计算机处理

（3）近似算法。

- 很多问题允许最终解有一定程度的误差。近似算法是求解NP类问题的一个可行的方法。

（4）概率算法。

- 将随机性的操作加入到算法运行中，同时允许结果以较小的概率出现错误，并以此为代价，获得算法运行时间大幅度减少。

9.3 NP 完全问题

■ 4. NP类问题的计算机处理

（5）并行计算。

- 虽然从原理上讲增加处理机的个数不能根本解决NP类问题，但并行计算是解决计算密集型问题的必经之路。

（6）智能算法。

- 遗传算法、蚁群算法等来源于自然界的优化思想，称为智能算法，是解决最优化问题的有力手段。

End