证明P类等于NP类的新方法

Malay Dutta

前教授

印度理工学院古瓦哈蒂分校和Anjana K. Mahanta

高哈蒂大学计算机科学系

摘要

本文提出一种证明 P=NP 的新方法。我们建议将一个 NP 完全问题多项式时间归约到线性规划。以往此类尝试采用多项式时间变换，这是一种特殊形式的归约，只调用一次求解较易的线性规划问题的子程序。我们则多次调用该子程序，大幅提高归约的有效性。此外，我们选择的 NP 完全问题也不同寻常。我们定义一种特殊的有向无环图，称为时间图。我们定义此类图中的哈密顿时间路径，并提出哈密顿时间路径问题（HTPATH），证明该问题是 NP 完全的。随后，我们提出一个猜想，若能证明该猜想，便可立即得到这一问题的多项式时间算法，从而证明 P=NP。

1. 引言：P=NP 问题是当今理论计算机科学中最重要的未解问题之一，也与实际应用领域中的许多重要问题密切相关。该问题涉及判定问题，即只有“是”或“否”两种答案的计算问题。P类是具有多项式时间算法的判定问题的类别；NP 是指其输入若被认为答案为“是”，便可在多项式时间内验证的判定问题类别。容易证明，P 包含于 NP。但每个 NP 中的问题是否都属于 P，至今尚无定论。人们发现了一类称为 NP 完全问题的问题，它们是 NP 中最难的问题，因为只要发现任何一个 NP 完全问题属于 P，NP 就将等于 P。应用领域中的许多问题已被证明是 NP 完全的；由于这些问题十分重要，人们开发了各种近似方法来处理它们。

为困难问题设计算法的一项重要技术，是将其多项式时间归约为一个较容易的问题。长期以来，人们并不知道线性规划问题是否存在多项式时间算法，但它在为许多实际应用问题设计高效算法时非常有用。因此，当人们发现线性规划的多项式时间算法后，便投入了大量精力，试图将

某些 NP 完全问题在多项式时间内归约为线性规划（更准确地说，是归约为线性规划可行性问题，这是一种判定问题）。然而，这些尝试大多采用了一种称为多项式时间变换的归约形式（[2] 的定义15.3），这种形式只调用一次包含较容易问题算法的子程序。对于 P类中的问题，这样的子程序就像一把小锤子，我们想用它来拆毁一座大楼（即 NP 完全问题）的结构。直观上显而易见，我们应该多次调用这样的程序（当然，调用次数应受某个多项式限制）。每次调用都能完成一个相对较小的目标。本文采用的正是这种方法。

我们还特意选择了一种 NP 完全问题，将其归约为一个更简单的问题。有向图上的哈密顿回路问题是 NP 完全问题，但在尝试将其归约为线性规划时，不相交回路的存在会带来问题。我们从有向图上的哈密顿回路问题出发，在顶点中引入时间维度，并将顶点表示为 (i,t)，其中 i 表示原图中的顶点编号，t 表示时间层。我们还将其中一个顶点（比如顶点 0）拆分为两个顶点——(0,0) 作为源顶点，(0,n+1) 作为目标顶点。现在，边从一层指向下一层，而

图变成了无环图。原图中的哈密顿回路会映射为我们称作哈密顿时间路径的路径；如果原始有向图有 n+1 个顶点（计入被拆分为两个顶点的顶点 0），那么该图就称为 n 阶时间图。

因此，我们将 n 阶完全时间图 K_n^T 定义为一个分层有向图，它有一个源顶点 (0,0)、一个目标顶点 (0,n+1)，以及 n^2 个内部顶点 (i,t)（1≤i≤n；1≤t≤n）。边 e_{0j0} 从 (0,0) 指向 (j,1)，其中 1≤j≤n；边 e_{ijt}从 (i,t) 指向 (j,t+1)，其中 (1≤i,j≤n（i≠j），1≤t≤n-1）；最后，边 e_{i0n} 从 (i,n) 指向 (0,n+1)，其中 1≤i≤n。K_n^T 的这一定义基于时变旅行商问题 [1]的概念。我们在此关注游客在 n + 1 座城市 0、1、2、...、n 中可能采取的路径。假设游客在第 0 日从城市 0 出发，在第 i 日经过城市 p_i （$1 \le i \le n$；$p_i \ne p_{i+1}$），$1 \le p_i \le n$，最后在第 n 日返回城市 0 + 1。我们可以将游客所走的路径表述为特殊有向图 K_n^T中的一条路径。需要注意的是，对于顶点 (i,t)，i 称为城市编号，t 称为日编号。对于边 e_{ijt}，(i,t) 是源顶点，(j,t+1) 是目标顶点，而 t 是时间层，包含第 t 日与第 t+1日之间的时间区间。对于边 e_{ijt}，若 t=0 ，则 i 必须为 0；若 t=n，则 j 必须为 0。时间路径是从 (0,0) 到 (0,n+1) 的一条路径。它必须经过顶点 (p(t), t)，其中 1≤t≤n，p() 是一个函数

从 {1,2,...,n} 到 {1,2,...,n}。如果 p() 是一个置换，则该路径称为哈密顿时间路径（htp）。哈密顿时间路径的特征函数称为哈密顿时间流（记作“htf”），它是定义在 K_n^T的边集 $E(K_n^T)$ 上的有理值函数空间中的一个向量；该空间在逐点运算下构成向量空间。对于定义在 $E(K_n^T)$ 上的函数 f，support(f) 表示 f 非零处的边集。当且仅当 f = 0时，support(f) 为空集。

一个阶数为 n 的时间图 G 是 K_n^T 的一个子图，其顶点与 K_n^T 相同，且其边集 E(G) 是 $E(K_n^T)$ 的子集。G 中的 htf 指支撑为 E(G) 子集的 htf。由 G 中的 htf 生成的函数向量空间记为 H(G)。对于 H(G) 中的 f，f 在任意时间层的边上的取值之和在每一层都相同，这个量称为 f 的流。如果 G 中存在 htf，则称时间图 G 为哈密顿图。哈密顿时间路径问题（HAMTPATH）是这样一个问题：给定一个阶数为 n 的时间图 G，G 是否为哈密顿图?

在第2节中，我们将证明 HAMTPATH 是 NP 完全的。在第3节中，给定一个时间图 G 以及 G 中的一条边 e，我们定义一个线性规划问题 LP(G,e)；如果 G 中不存在一条经过 e 的哈密顿时间路径，则该问题不可行。基于此定义，我们将 G 的一些边归类为无用边。在

第4节中，我们陈述我们的猜想。在最后的第5节中，我们证明，如果该猜想成立，就能得到一个用于 HAMTPATH 的多项式时间算法，从而证明 P=NP。在该算法中，HAMTPATH 被归约为线性规划可行性问题，并通过多次（但次数为多项式级）调用一个用于判定线性规划问题是否可行的子程序来完成。这构成了根据 [2] 中定义15.2 所述意义，将 HAMTPATH 多项式时间归约到线性规划可行性问题（而非 [2] 中定义15.3 所述的多项式时间变换）。

2. HAMTPATH 问题的 NP完全性： 在 [3],第7.3节中，HAMPATH 问题被定义为判定一个有向图是否包含一条连接两个指定的源节点和终端节点 s 与 t 的哈密顿路径。定理 7.46 在 [3] 中证明了 HAMPATH 是 NP 完全的。我们将在此证明，HAMPATH 可在 $O(n^3)$ 时间内归约到 HAMTPATH，从而证明 HAMTPATH 是 NP 完全的。

定理2.1：HAMPATH 可在 $O(n^3)$ 时间内归约到 HAMTPATH。

证明：给定一个有向图 D，其顶点集合为 {s,t,1,2,...,n}，其中 s 是指定的源顶点，t 是

指定的终端顶点，边集合为 E，我们按如下方式构造一个 n 阶时间图 G：

对于每个 t=1,2,...n-1，若且唯若 (i,j) 属于 E，e_{ijt} 才是 G 中的一条边。对于每个 j=1,2,...,n，若且唯若 (s,j) 属于 E，e_{0j0} 才是 G 中的一条边。对于每个 i=1,2,...,n，若且唯若 (i,t) 属于 E，e_{i0n} 才是 G 中的一条边。很容易证明，D 存在一条从 s 到 t 的哈密顿路径，当且仅当 G 存在一条哈密顿时间路径。因此，根据 D 构造 G 的方法是从 HAMPATH 到 HAMTPATH 的归约。此外，给定 D 后，可以在 $O(n^3)$ 时间内完成 G 的构造。这就证明了该定理，并可得出结论：HAMTPATH 是 NP 完全问题。

3. **一些初步定义和结果：**设 G 是一个阶数为 n 的时间图，e 是 G 中的一条边。我们定义线性规划问题 LP(G,e)：对于定义在 E(G) 上且满足以下条件的函数 f：

$$\sum_j f(e_{0j0}) = 1 \quad (1)$$

$$\sum_j f(e_{ijt}) - \sum_{j'} f(e_{j'it-1}) = 0 \quad \text{for } 1 \le i,t \le n \quad (2)$$

$$\sum_{jt} f(e_{ijt}) = 1 \quad \text{for } 1 \le i \le n \quad (3)$$

$$\sum_{我} f(e_{i0n}) = 1 \quad \text{for } 1 \le i \le n \quad (4)$$

$$f(e) = 1 \quad (5)$$

$$f(e') \ge 0 \quad \text{for every } e' \text{ in } E(G \quad (6)$$

最小化 <0,f>)

G 中任何满足 f(e) = 1 的 htf f 都是 LP(G,e) 的可行解。因此，如果 LP(G,e) 不可行，那么 G 中就不存在经过 e 的哈密顿时间路径。这样的边称为 G 中的无用边，而

从 G 中删除无用边不会影响 G 中哈密顿时间路径的集合。

4. 我们的猜想： 现在陈述我们的猜想：

猜想 4.1：一个边集非空且没有无用边的时间图是哈密顿的。

假设这个猜想不成立，且 G 是一个边集非空、没有无用边的非哈密顿时间图。我们从 G 中任取一条边 e。由于 e 不是无用边，因此 LP(G,e) 有一个可行解 f。这个解无法表示一个 htf，因为 G 不是哈密顿图。因此，G 中存在边 e1 和 e2，使得 f(e1) 和 f(e2) < 1。我们称 e1 和 e2 为 e 的子节点。但 e1 和 e2 也都不是无用边，因此它们各自都有一对子节点。如果我们能以某种方式证明这个过程不可能无限持续下去（因为边的数量有限），那么这个猜想就得以证明。我们竭尽全力寻找这个猜想的反例，但未能找到。这让我们隐约觉得这个猜想或许是正确的。如果这个猜想成立，我们将在下一节证明 P= NP。顺便一提，这个猜想提出了一个新问题，读者可以将其作为挑战尝试解答。

5. 这一猜想成立的后果：假设猜想 4.1 成立。那么，判断给定的时间图 G 是否为哈密顿图的显然方法，就是移除 G 的所有无用边。如果所有边都被移除，那么 G

不是哈密顿图。如果最终剩下一个不含无用边的非空边集，那么根据猜想 4.1，所得的图 G′ 是哈密顿图。由于 G′ 的边集包含于 G 的边集，因此 G 是哈密顿图。

不过，我们必须谨慎确定删除边的顺序。例如，对于边为 e_{010}, e_{121}, e_{131}, e_{242}, e_{352}, e_{453}, e_{543}, e_{524}, e_{434}, e_{205},e_{305}, 、阶为 5 的时间图，如果我们试图先删除 e_{010} ，它将不会被删除。之后，如果我们尝试删除其他边，那么它们都会被删除，因为这些边都是无用边。结果，我们只剩下一条边 e_{010} ，显然这会违反猜想，并判定该图为哈密顿图。检查边是否可以删除的正确方法是：一旦发现某条边无用，就从头重复这一过程。因此，利用猜想 4.1，我们可以得到以下用于 HAMTPATH 问题的算法。

算法 5.1：

给定一个时间图 G：1) G′ = G 2) 如果 G′ 没有边，则返回“非哈密顿” 3) 对于 G′ 中的每条边 e 4) 如果 LP(G′,e) 不可行，则从 G′ 中移除 e 并转到 2) 5) 结束循环 3 6) 如果 G′ 中还剩有边，则返回“哈密顿”

G 中有 $O(n^3)$ 条边。对于 G 中的每条边，LP(G’,e) 至多调用一次。LP(G’,e) 的规模以 n 的多项式为上界，而判定 LP(G’, e) 可行性的复杂度以 LP(G’,e) 的规模的多项式为上界。因此，算法 5.1 的运行时间以 n 的多项式为上界。由于算法 5.1 能解决 HAMTPATH 问题，而该问题是 NP 完全的，因此，如果猜想 5.1 成立，我们可以得出 P = NP。

6. 参考文献

[1] Fox, K.R., Gavish, B., Greaves, S.C., (1980). 关于（时间相关）旅行商问题的一种 n 约束表述。运筹学研究。28:1018-1021。

[2] Papadimitriou, C. H., Steiglitz, K. (1998). 组合优化：算法与复杂性。 纽约州米尼奥拉：Dover Publications Inc.

[3] Sipser, M., (2013). 计算理论导论。 3rd 版：Cengage Learning。