CS2045M: 高级数据结构与算法分析 **2025-2026 学年秋冬学期**

Lecture 6: 回溯法

编写人：吴一航 yhwu_is@zju.edu.cn

从本讲起开始介绍算法设计策略，将主要介绍回溯、分治、动态规划、贪心、NP 问题、近似算法、局部搜索、随机算法、并行算法以及外部排序，其中从 NP 问题到随机算法这四讲将是这门课最具理论色彩的部分，其中有无数闪烁着人类智慧的光芒伟大结论，也有留待未来的天才解决的迷人猜想。当然这也意味着这几讲的内容颇具挑战性，因为事实上在具有强大的理论计算机底蕴的学校，这些内容每一章都可以用整整一门课来讲解，上千页的教材都无法完全展现其魅力，而我们只能在有限的课时中领略一些思想。

在接下来对算法设计的讨论中，有两个要求将是讨论的核心：

1. 有效性（effectiveness）：能保证算法的结果是正确的，或者之后近似算法保证结果在一定近似比内，或者随机算法保证一定概率内是合理的；
2. 有效率（efficiency）：算法的运行速度应当是快速的。

在之后所有的算法中都会思考这两个问题（当然对于分治、动态规划等问题，有效性的保证看起来比较 trivial）。实际上，在学习数据结构的时候也需要考虑操作保持数据结构的结构和序性质，以及希望操作是比较快速的，因此这些目标都是有共通性的。

本讲从思想最简单、看起来不那么应该出现在高级算法设计课程中的回溯法开始。需要注意的是，接下来的讲义很多算法都不会像数据结构操作那样具体分析了，因为理解算法过程非常简单，更重要的是基本思想的传递以及思想的更丰富的应用，以及在理论章节中会补充更多的理论知识，供希望深入理解这些有趣但具有挑战性的专题的同学阅读。

6.1 回溯的基本思想与应用

6.1.1 基本思想

有些问题要求我们给出符合条件的解，而所有的可能性是有限的，这时可以通过所谓的暴力搜索（exhausting search），或称蛮力法（Brute Force）来对所有的可能性逐一检查得到正确的解。这样的方法显然可以保证算法有效性，因为所有可能都被检查了，然而在很多问题中，所有可能性的个数（搜索空间）可能是相当巨大的，这样的算法显然不是很有效率。

然而在很多情况下，我们可以很轻松地通过一些判断将很大一部分可能的结果排除在正确答案之外，这样的排除过程我们称为“剪枝”（pruning），至于为什么叫这个名字，是因为经常会将我搜索正确答案的过程用一棵树表示，我们在八皇后问题中就能看清这一点。

6.1.2 八皇后问题

这是一个非常简单且经典的例子，因为算法太简单这里就不再赘述，这里只强调一个建树的思想。事实上树的建立过程就是模拟搜索决策过程：每一步都有很多种构建最终答案的选择的可能，在选择了其中一步后对于下一步构建答案又有新的可能，依此类推得到所有可能的决策路径。然后在这棵树上做剪枝即可，在八皇后问题中，剪枝的条件判别非常容易（见 PPT 第四页），因此不再赘述。

当然八皇后问题可以直接推广为 n 皇后问题，对具体实现感兴趣的读者可以自行阅读相关资料，例如这个网页。

当然说到复杂度分析不得不提到一些需要注意的结论。在回溯中经常遇到阶乘如 $n!$ 的复杂度，也会碰到指数次方的复杂度。我相信读者只要简单回忆微积分或数学分析中学过的求极限的方法就能回忆起如下结论：

$$\lim_{n\to\infty}\frac{n!}{n^n}=\lim_{n\to\infty}\frac{a^n}{n!}=\lim_{n\to\infty}\frac{n^b}{a^n}=0,$$

当然还有斯特林公式等更精确的估计，相信很多读者之前也有所了解。

6.1.3 收费公路问题

同样是一个很简单的算法，难点可能在于标题英文单词不认识（开个玩笑）。收费公路重建问题 (turnpike reconstruction problem) 是给定一个点集，以及其中点的两两之间距离然后重构出点集中各点位置的问题，这个名称得自于对美国西海岸公路上那些收税公路出口的模拟，它在物理学和分子生物学中都有应用。

想法非常简单，就是首先确定点的个数以及两个端点的位置（当然左端点可以默认为 0），接下来每一步考察目前剩余的最大的距离，这个距离应当是和两个端点之间的距离（为什么呢？例如 PPT 上的例子，第二次考察距离为 7 的点，有的同学可能会想这个距离 7 会不会是与之前选定的 8 之间的距离，即有一个点在 1 处呢？显然这是不可能的，因为 1 和 10 距离为 9，大于第一步的 8. 更抽象而言，每一次都会取出剩下的最大距离，因此当前的最大距离只有可能是和端点的最大距离，否则新加入的点到端点的距离等于它到中间我们取的点的距离加上中间点到端点的距离，这比目前最大距离要大，而我们已经没有剩下这么大的距离了，因此不可能出现这样的情况），于是会有两种选择，从而可以逐步构造出一棵树。决定某个解是否成立的方法就是判断剩余的距离是否能满足这些点之间的距离。

伪代码实际上就是以上思路的直接体现，首先通过最大距离取点，然后判断这个点合不合要求，然后递归，递归成功则找到一种解，递归失败则撤回操作返回上一层，选择另一种可能。事实上这也引出了一

般回溯问题的框架，在 PPT 第 10 页有介绍。

6.1.4 博弈

如果一个问题涉及双人乃至多人决策，那么就是一个博弈问题，Tic-tac-toe 就是一个典型的博弈问题。无论讨论什么博弈，如下几个条件都是必不可少的：

1. 参与人：在 Tic-tac-toe 中就是两个参与方，计算机与人类；

2. 策略：在 Tic-tac-toe 博弈中，策略就是计算机或人类在看到一个棋局后会作出的行动决策；

3. 效用函数：上过微观经济学课程的同学应该并不陌生，没上过的同学应当也知道所谓经济学的理性人假说。在一般博弈中，理性人追求的是自己的效用最大化，通俗来说就是自己越开心越好，当然在 Tic-tac-toe 中就是赢的可能越大越好。PPT 中将赢的可能用自己与对手的 “number of potential wins at position P” 之差来表达（number of potential wins 就是当前局势下，最大有多少种可能的赢法，也就是最后 8 种赢法中还剩几种是当前局势下能达到的，8 种赢法就是 3 横 3 竖 2 对角线）。

事实上，这上面的策略和效用函数是息息相关的，参与者希望他的行动能让他的效用最大化，这就是参与者决定他的策略的出发点。在 PPT 中，我们为参与者制定的策略是所谓的 “最大最小策略”（或 “最小最大策略”），即已知别人是要最大化效用的，所以我的选择是要最小化别人最大化的效用（或者我知道别人要最小化我的效用，我就要最大化别人最小化的我的效用）。事实上，在两人零和博弈中（Tic-tac-toe 就是，只需看效用函数的定义就可知两个参与者的效用恰为相反数），这一最小最大策略得到的解比纳什均衡具有更好的性质。感兴趣的同学可以进一步了解博弈论相关的基本概念，毕竟同学们基本都是学习计算机专业的，祖师爷冯 · 诺依曼的很多著名工作都是与博弈论相关的（因此他也被尊称为 “博弈论之父”），了解一些博弈论的常识还是有趣且有必要的。

当然我们的重点是回溯以及剪枝，这里重点介绍了 $\alpha - \beta$ 剪枝。所谓 α 剪枝就是当我要最大化别人最小化的我的效用时，如 PPT 第 16 页所示，我已经知道，如果我选左边这条路，最少也能得到 44 的效用，但如果我选择右边这条路，叶子有一个结点使得我只能得到 40 的效用，又因为别人一定是要最小化我的效用的，所以右边这条路不管右下方那个叶子是多少，我能得到的效用也不超过 40 了，所以右下角那个点就被 α 剪枝剪掉了。β 剪枝是对称的情况，这里不再赘述，只强调这里要区分清楚二者的差别，并且自己要梳理清楚逻辑，然后注意被剪掉的是 PPT 图中染黑的点，这一问题在作业、考试中还是很常见的。

当然，在搜索空间太大的时候，$\alpha - \beta$ 剪枝也是很复杂的，所以可能会随机采样几种情况进行模拟，最大化采样到的情况的效用，这也就是所谓的 “蒙特卡罗方法” 的基本想法。

6.2 讨论题

讨论 1：在国际象棋中，设 B 是棋盘的大小，规定在 R 行 C 列上的马可以走到 $1 \leqslant R' \leqslant B$ 行和 $1 \leqslant C' \leqslant B$ 列处，且要么

$$|R - R'| = 2及|C - C| = 1,$$

要么

$$|R - R'| = 1及|C - C'| = 2,$$

（即马只能横跳两格加纵跳一格，或者横跳一格加纵跳两格）马的一次环游是马在棋盘上的一系列跳行，它恰好访问所有的方格一次最后又回到开始的位置。

1. 如果 B 是奇数，证明马的环游不存在（提示：考虑国际象棋棋盘有黑色和白色格子，找到马跳和黑白格之间的关系）；
2. 若存在马的环游，给出一个回溯算法找出马的一次环游（核心是给出剪枝策略，最好能比最容易想到的条件有优化）。

【注:】这一问题被称为 closed knight's tour problem，回溯算法对于这一问题而言显得非常暴力，感兴趣的读者可以进一步参考维基百科等资料。

讨论 2: 证明：Tic-tac-toe 中，下面三种情况有且只有一种情况成立：

1. 计算机（画圈）有一个制胜策略；
2. 人类（画叉）有一个制胜策略；
3. 双方都有一个至少保证平局的策略。

【提示:】可以搜索冯 · 诺依曼从国际象棋博弈出发得到的类似结论，有两种常见的证明方法，第一是从后往前的数学归纳法，第二是使用逻辑表达式。