搜索技巧与贪心

北京大学 周尚彦

Oct. 1st

Contents

Depth First Search

深度优先搜索是指从起始状态开始，尽可能深地搜索深的搜索状态的分支的算法。

Depth First Search

深度优先搜索是指从起始状态开始，尽可能深地搜索深的搜索状态的分支的算法。

搜索的过程可以抽象成一棵树，如果在搜索树上描述搜索过程，即从根节点出发，不断地向深处走，当走到叶子节点的时候就开始回溯并搜索其他分支。

Depth First Search

深度优先搜索是指从起始状态开始，尽可能深地搜索深的搜索状态的分支的算法。

搜索的过程可以抽象成一棵树，如果在搜索树上描述搜索过程，即从根节点出发，不断地向深处走，当走到叶子节点的时候就开始回溯并搜索其他分支。

向深处走的过程实际上是子状态入栈的过程，回溯实际上是子状态出栈的过程，搜索过程中对信息的维护实际上是对于这个栈的维护。

影响时间复杂度的因素

影响深度优先搜索时间复杂度的因素主要有二：

- 搜索的最大深度。
- 从每个状态可以转移到的状态数。

对于深度优先搜索的时间复杂度的优化也是从这两个方面下手。

剪枝

剪枝实际上是试图减少从每个状态可以转移到的状态数。

剪枝

剪枝实际上是试图减少从每个状态可以转移到的状态数。

对于计数搜索问题，我们通常采取可行性剪枝，即在可以明确当前状态的后继状态不可能有可行解时及时回溯。

对于最优化搜索问题，我们还可以采取最优化剪枝，即记录当前搜索到的最优解，在可以明确判断当前状态的后继状态不可能由于最优解时及时回溯。

具体的剪枝方法需要结合具体问题具体分析。但要注意控制判断后继状态是否不优和是否可行的判断过程的时间复杂度。

剪枝

剪枝实际上是试图减少从每个状态可以转移到的状态数。

对于计数搜索问题，我们通常采取可行性剪枝，即在可以明确当前状态的后继状态不可能有可行解时及时回溯。

对于最优化搜索问题，我们还可以采取最优化剪枝，即记录当前搜索到的最优解，在可以明确判断当前状态的后继状态不可能由于最优解时及时回溯。

具体的剪枝方法需要结合具体问题具体分析。但要注意控制判断后继状态是否不优和是否可行的判断过程的时间复杂度。

此外在一些启发式搜索中应用到了其他的剪枝方法，由于在 OI 中应用不够广泛，在这里不做赘述。

泰国佛塔

POJ 1190

泰国佛塔形状为一层层的圆柱体，自底向上直径递减。要求制作一个体积为吃 $N\pi$ 的 M 层佛塔。设从下往上数第 $i(1 \le i \le M)$ 层佛塔是半径为 R_i, 高度为 H_i 的圆柱。当 $i < M$ 时，要求 $R_i > R_{i+1}$ 且 $H_i > H_{i+1}$。令外表面面积为 $Q = S\pi$。

请编程，对于给出的 N 和 M，找出佛塔的制作方案（适当的 R_i 和 H_i 的值），使 S 最小。（除 Q 外，以上所有数据皆为正整数）

$N \le 10000, M \le 20$。

状态表示与剪枝方法

状态表示方法：记 search(v,floor,R,H,S) 为当前体积为 v，层数为 $floor$，上一层的 R_i, H_i 为 R, H，已经使用的表面积为 S 的状态。

状态表示与剪枝方法

状态表示方法：记 search(v,floor,R,H,S) 为当前体积为 v，层数为 $floor$，上一层的 R_i, H_i 为 R, H，已经使用的表面积为 S 的状态。

可行性剪枝：

- 剩下的若干层都放最大的圆柱，体积也不够。

最优性剪枝：

- 剩下的若干层都放最小的圆柱，得出的表面积比当前最优解劣。
- 剩下的体积所需的最小表面积加上当前表面积比当前最优解劣。

状态表示与剪枝方法

状态表示方法：记 search(v,floor,R,H,S) 为当前体积为 v，层数为 $floor$，上一层的 R_i, H_i 为 R, H，已经使用的表面积为 S 的状态。

可行性剪枝：

- 剩下的若干层都放最大的圆柱，体积也不够。

最优性剪枝：

- 剩下的若干层都放最小的圆柱，得出的表面积比当前最优解劣。
- 剩下的体积所需的最小表面积加上当前表面积比当前最优解劣。

其中“剩下若干层都放最小/大的圆柱”所用的表面积和“剩下体积所需的最小表面积”都可以预处理。 □

概述

记忆化搜索是指在搜索完某个状态时记录其信息，在再次搜索到该节点时根据记录的信息决定之后的搜索状态。

概述

记忆化搜索是指在搜索完某个状态时记录其信息，在再次搜索到该节点时根据记录的信息决定之后的搜索状态。

在搜索转移图中存在环的情况通常需要记录一个状态是否在搜索栈中以确保不进入死循环。

概述

记忆化搜索是指在搜索完某个状态时记录其信息，在再次搜索到该节点时根据记录的信息决定之后的搜索状态。

在搜索转移图中存在环的情况通常需要记录一个状态是否在搜索栈中以确保不进入死循环。

在需要重复到达某些状态多次的计数搜索问题（无需输出方案）和最优搜索问题中记录当前状态的搜索结果可以节省大量时间。

由于只会从每个状态向后继状态搜索一次，此时搜索的时间复杂度通常为 $O(m+nT)$，其中 n 为状态数，T 为搜索过程在每个状态本身所花费的时间，m 为转移数。

概述

记忆化搜索是指在搜索完某个状态时记录其信息，在再次搜索到该节点时根据记录的信息决定之后的搜索状态。

在搜索转移图中存在环的情况通常需要记录一个状态是否在搜索栈中以确保不进入死循环。

在需要重复到达某些状态多次的计数搜索问题（无需输出方案）和最优搜索问题中记录当前状态的搜索结果可以节省大量时间。

由于只会从每个状态向后继状态搜索一次，此时搜索的时间复杂度通常为 $O(m+nT)$，其中 n 为状态数，T 为搜索过程在每个状态本身所花费的时间，m 为转移数。

实际上这样的记忆化搜索常常可以被看作是按 DFS 序进行的 DP/递推过程。 □

着色方案

SCOI 2008

有 n 个木块排成一行，从左到右依次编号为 $1 \sim n$。你有 k 种颜色的油漆，其中第 i 种颜色的油漆足够涂 c_i 个木块。所有油漆刚好足够涂满所有木块，即 $c_1 + c_2 + \ldots + c_k = n$。相邻两个木块涂相同色显得很难看，所以你希望统计任意两个相邻木块颜色不同的着色方案。

$n \leq 15, k \leq 5$。 □

sol.

search(a, b, c, d, e, l) 表示当前有能涂 1 次的油漆 a 个，能涂 2 次的 b 个...前一个颜色为 l 的方案数即可。

□

概述

迭代加深搜索对深度优先搜索进行了一定改进，对搜索树的深度进行控制，即有界深度优先搜索。

概述

迭代加深搜索对深度优先搜索进行了一定改进，对搜索树的深度进行控制，即有界深度优先搜索。

在程序找到目标之前，通过迭代不断增大搜索最大深度 d 以保证完备性和最优性。虽然会有不少重复搜索，但是鉴于每增加一次 d，则搜索的时间复杂度会以指数级别增加，所以重复搜索的时间可以忽略。

概述

迭代加深搜索对深度优先搜索进行了一定改进，对搜索树的深度进行控制，即有界深度优先搜索。

在程序找到目标之前，通过迭代不断增大搜索最大深度 d 以保证完备性和最优性。虽然会有不少重复搜索，但是鉴于每增加一次 d，则搜索的时间复杂度会以指数级别增加，所以重复搜索的时间可以忽略。

迭代加深搜索通常用于那种搜索树又深又宽、但是解并不是很深的情况。 □

棋盘染色

codevs 1049

有一个 5×5 的棋盘，上面有一些格子被染成了黑色，其他的格子都是白色，你的任务的对棋盘一些格子进行染色，使得所有的黑色格子能连成一块，并且你染色的格子数目要最少。

读入一个初始棋盘的状态，输出最少需要对多少个格子进行染色，才能使得所有的黑色格子都连成一块（四联通）。 □

sol.

枚举答案，search(x, y, k) 表示 (x, y) 之前的格子已经染色完毕，已经染了 k 个格子的状态。 □

概述

将搜索问题分为两个子问题，通过合并子问题的答案获取最终的答案。

通常用于集合搜索问题。 □

最大独立集

求最大独立集，点数 $n \leq 50$

最大独立集

给一张图，求最大独立集，点数 $n \leq 40$。 □

将 n 分为两个部分，大小分别为 $e = n/2$。

先搜索第一个部分，搜索完毕后统计 $g[st]$ 表示在第一个部分中选取状态为 st 的子集的最大独立集大小。

再搜索第二个部分，搜索到叶子状态时通过 ban（第二个部分中选取叶子状态所表示的点后在第一部分中不能选择的点的集合）来统计答案。

将 n 分为两个部分，大小分别为 $e = n/2$。

先搜索第一个部分，搜索完毕后统计 $g[st]$ 表示在第一个部分中选取状态为 st 的子集的最大独立集大小。

再搜索第二个部分，搜索到叶子状态时通过 ban（第二个部分中选取叶子状态所表示的点后在第一部分中不能选择的点的集合）来统计答案。

总复杂度为 $O(2^e e)$。 □

Breadth First Search

宽度优先搜索又被称为广度优先搜索，它是先将起始状态加入队列，然后每次从队列中取出一个状态，将其后继状态加入队列，直到所有状态均被访问为止。

它并不考虑结果的可能位址，而是彻底地搜索所有状态，所以很少有基于 BFS 的启发式算法，也很少对 BFS 进行剪枝。

相对于 DFS，BFS 更加难于保存当前节点的状态，所以 BFS 在爆搜中的应用较少。

BFS 常常用于需要保证第一次访问节点时走的是最短路径的问题。

单源最短路

给定一张边权为 0 或 1 的图，求从起点出发到达每个点的最短路径。

□

sol.

由边权为 0 的边转移过去的状态放到队首，边权为 1 的边转移过去的状态放到队尾即可。

sol.

由边权为 0 的边转移过去的状态放到队首，边权为 1 的边转移过去的状态放到队尾即可。

复杂度 $O(n+m)$。 □

拓扑排序

给定一张拓扑图，确定每个点的拓扑序。

sol.

将入度为 0 的点加入队列，在将每个点取出队列时，将其出点的入度减 1，同时把新的入度为 0 的点加入队列。

复杂度 $O(n+m)$。

□

概述

贪心即在每一步选择中都采取在当前状态下最好或最优（即最有利）的选择，从而希望导致结果是最好或最优的算法。

需要满足：子问题的局部最优解一定是全局最优解的子集。

最短路

给定一张 n 个点的图，求单源最短路。

对于每个点有在 t_i 时间之前不能到达的限制.

$n \leq 100000$。

最短路

给定一张 n 个点的图，求单源最短路。

对于每个点有在 t_i 时间之前不能到达的限制.

$n \leq 100000$。

dijkstra

合并果子

NOIP 2004

n 堆果子，每次选择两堆合并，代价为两堆果子的重量和，求把所有果子合并到一堆的最小费用。

$n \leq 100000$

合并果子

NOIP 2004

n 堆果子，每次选择两堆合并，代价为两堆果子的重量和，求把所有果子合并到一堆的最小费用。

$n \leq 100000$

经典题，哈夫曼树。

荷马史诗

NOI 2015

Allison 最近迷上了文学。她喜欢在一个慵懒的午后，细细地品上一杯卡布奇诺，静静地阅读她爱不释手的《荷马史诗》。但是由《奥德赛》和《伊利亚特》组成的鸿篇巨制《荷马史诗》实在是太长了，Allison 想通过一种编码方式使得它变得短一些。

一部《荷马史诗》中有 n 种不同的单词，从 1 到 n 进行编号。其中第 i 种单词出现的总次数为 w_i。Allison 想要用 k 进制串 s_i 来替换第 i 种单词，使得其满足如下要求: 对于任意的 $1\ i, j\ n\ i\ j$ 都有：s_i 不是 s_j 的前缀。

现在 Allison 想要知道，如何选择 s_i，才能使替换以后得到的新的《荷马史诗》长度最小。在确保总长度最小的情况下，Allison 还想知道最长的 s_i 的最短长度是多少？一个字符串被称为 k 进制字符串，当且仅当它的每个字符是 0 到 $k-1$ 之间（包括 0 和 $k-1$）的整数

sol.

k 叉哈夫曼树。

□

建筑抢修

JSOI 2007

小刚在玩 JSOI 提供的一个称之为“建筑抢修”的电脑游戏：经过了一场激烈的战斗，T 部落消灭了所有 z 部落的入侵者。但是 T 部落的基地里已经有 N 个建筑设施受到了严重的损伤，如果不尽快修复的话，这些建筑设施将会完全毁坏。现在的情况是：T 部落基地里只有一个修理工人，虽然他能瞬间到达任何一个建筑，但是修复每个建筑都需要一定的时间 t_i。同时，修理工人修理完一个建筑才能修理下一个建筑，不能同时修理多个建筑。如果某个建筑在一段时间之内没有完全修理完毕 d_i，这个建筑就报废了。你的任务是帮小刚合理的制订一个修理序，以抢修尽可能多的建筑。

$N \leq 150000$

sol.

按 d 排序，倒序贪心。

□

Muzeum

PA 2014

二维平面上有 N 个警卫，M 个物品。

警卫的目光全都看向 y 轴正方向，并且有相同大小的视角（视角的正切值给定）。

贿赂一个警卫的花费为 a_i，偷走一个物品的收益为 b_i，求最大收益。

$N, M \leq 10^5$

不考虑数据范围就是裸的最大权闭合子图，然而这个数据范围显然跑不了网络流。

不考虑数据范围就是裸的最大权闭合子图，然而这个数据范围显然跑不了网络流。

考虑坐标变换，将所有警卫的视角变成直角且让视角两边平行与坐标轴。

最小割 $\Longleftrightarrow$ 最大流。目的是要流出尽量多的流量。

将所有 element 按照 x 坐标排序，每碰到一个物品时就将其加入平衡树，碰到一个警卫就尽量往 y 坐标大于等于它且最小的物品流流量。

□

随机化加贪心

有时间就讲讲