试题讲评

$n+e$

Tsinghua University

2017 年 7 月 19 日

1. 走出迷宫
   - 题目大意
   - 算法分析
2. 因数游戏
3. 十五数码

1. 走出迷宫
   - 题目大意
   - 算法分析
2. 因数游戏
3. 十五数码

- 给出一个 $n \times m$ 的网格图，有若干个障碍，求出给定两点之间的最短路径长度。
- $1 \le n, m \le 100$

1 走出迷宫
- 题目大意
- 算法分析

2 因数游戏

3 十五数码

- 我早上对着 std 给你萌分析了一遍，还给了代码了
- 没 AC 这题的童鞋是不是要打屁股呀

1. 走出迷宫
2. 因数游戏
   - 题目大意
   - 算法分析
3. 十五数码

- 有 T 组询问，每组询问给定 2 个数 a，b，问从 a 变到 b 最少需要多少步，每次我们可以把 a 加上它的一个因数或者减去它的一个因数，比如，6 可以变成 5,7,4,8,3,9,12。特别地，如果步数 >6 的话，输出 CalcFailed
- 原题的 MaxAns≤ 4
- $1 \leq T \leq 10$ $1 \leq a, b \leq 10^8$

1 走出迷宫

2 因数游戏

- 题目大意
- 算法分析

3 十五数码

Q: 这题该怎么做？

A: 这题我们可以双搜。

Q: 为什么双搜呢？

A 因为答案 ≤ 4 我们如果暴力枚举是 $O(A^4)$ 的，就是 100 台天河一号一起跑还是会 T 的。

- 我们似乎可以将枚举因数降到根号的，假设 n 是 a 的因数，因为 $n \times \frac{a}{n}) = a$，所以 $\frac{a}{n}$ 也是 a 的因数，复杂度就降到 $O(A^2)$ 了，呵呵，A 是 10^8 级别的，所以用天河一号跑还是会 T 的。

- 由于我们知道搜索有个很重要的优化是双搜，它能将复杂度指数降一半，于是乎，我们可以尝试用双搜。

Q: 那该怎么双搜呢？双搜需要满足前半层对后半层的贡献没有直接影响，但是这题里 B 是要通过 A 推过来的，所以该怎么办？

A: 没错，但是 B 推到 A 和 A 推到 B 是一样的，为什么这么说呢，假设由数 N 可以变成 M，设 $N+x=M$，由于 x 是 N 的因数，就可以设 $N=kx(k\neq 0)$，则原式为 $kx+x=M$ 即 $(k+1)x=M$，则 M 是 $k+1$ 和 x 的积，那么每次加或者减的数就必须是两个数的公因数，所以倒着搜也是一样的，我们就可以愉快的写双搜了，复杂度就会将成 $O(A)$ 的线性复杂度了。

- 显然出题人为了把大佬区分出来，以上的方法是不能直接通过本题的。
- 把分解因数的 $O(\sqrt{N})$ 的暴力做法修改为 Miller-Rabin+Rho 即可，复杂度为 $O(N^{1/4})$
- 链接 1 链接 2
- 并且注意一下，中间步骤有可能超过 2^{31}，因此要开 long long

1. 走出迷宫
2. 因数游戏
3. 十五数码
   - 题目大意
   - 算法一
   - 算法二
   - 算法三
   - 无解怎么判断

- 在一个四连通的 4×4 网格中，有 $0 - 15$ 数字的某个排列。现在可以把 0 和周围的某个数字交换，求交换到初始局面的最小交换次数。

- 本题是一道已知始末状态的状态空间搜索题，并且要我们最小化交换次数。实际上就是求最小的转移次数。
- 求最小的转移次数最常见的方法就是广度优先搜索了，第一个搜到的就是最短的，因为它在搜索树中深度是最浅的。

- 本题是一道已知始末状态的状态空间搜索题，并且要我们最小化交换次数。实际上就是求最小的转移次数。
- 求最小的转移次数最常见的方法就是广度优先搜索了，第一个搜到的就是最短的，因为它在搜索树中深度是最浅的。
- 算法步骤
  1. 读入初始状态，标记其深度为 0
  2. 广搜：把每一个合法且没有出现过的数据加入队列。

实现

1. 把 4×4 的表直接封装在一个 struct Table4x4 里，并且整合其深度。
2. queue<Table4x4>q，把状态存到 FIFO 队列里。(最好是循环队列)
3. 把整个表当成一个 16 位数，写一个最简单的 hash 函数。

注意不要重复操作，比如：上一步把 0 往上，下一步把 0 往下……(这样的话除了第一步有四种可能以外，其余每个状态都只能拓展出三个新状态，并且还有可能会与之前已有状态重复)

- 使用双向广搜即可优化算法一。
- 来算算分数：双搜 50+ 剩下输出 No= 60

1. 走出迷宫
2. 因数游戏
3. 十五数码
   - 题目大意
   - 算法一
   - 算法二
   - **算法三**
   - 无解怎么判断

- 使用 IDA* 算法即可通过。
- 估价函数为所有非 0 元素当前位置与目标位置的曼哈顿距离之和
- 容易看出这个估价函数满足 IDA* 的 h 函数性质

- 使用 IDA* 算法即可通过。
- 估价函数为所有非 0 元素当前位置与目标位置的曼哈顿距离之和
- 容易看出这个估价函数满足 IDA* 的 h 函数性质
- 套模板，包括估价，发现 T 了

几个优化技巧

1. 位运算. $i \gg 2$, $i \,\&\, 3$
2. `#define abs(x) (x>=0?x:-(x))`
   一些经常调用的小函数拿去 define 掉，因为声明函数栈空间花时间。平时玩玩可以，考试时非常**不推荐**使用 (注意风险，要加括号!!!)
3. 特判无解情况
4. 二维数组改成一维，$a[4][4] \rightarrow a[16]$
5. * 减少估价函数的计算量. 本题中的 $h(n)$ 满足加法运算
6. 调整 udlr 的顺序，改变搜索序
7. * 把起始局面和目标局面对掉

几个优化技巧

1. 位运算. $i \gg 2$, $i \,\&\, 3$
2. `#define abs(x) (x>=0?x:-(x))`
   一些经常调用的小函数拿去 define 掉，因为声明函数栈空间花时间。平时玩玩可以，考试时非常**不推荐**使用 (注意风险，要加括号!!!)
3. 特判无解情况
4. 二维数组改成一维，$a[4][4] \rightarrow a[16]$
5. * 减少估价函数的计算量. 本题中的 $h(n)$ 满足加法运算
6. 调整 udlr 的顺序，改变搜索序
7. * 把起始局面和目标局面对掉

- 总结：都是脑洞，针对题目特点进行的优化最有效
- 大家还可以试试别的花样 = =

- 十五数码和八数码判断是否有解的方法不同，八数码 0 的移动不影响其余 7 个数字逆序数的奇偶性，而十五数码 0 的左右移动不影响其余 15 个数逆序数的奇偶性（顺序不变），但上下移动改变奇偶（移动三次），加上 0 的话 16 个数逆序数左右改变（移动一次），上下也改变（移动 7 次）
- 需要注意每次移动 0 的距离奇偶性也改变（0 到目标位置的曼哈顿距离不是加 1 就是减一），所以 16 个数逆序数与 0 的距离之和 s 的奇偶性不因 0 的滑动而改变，初始时 s 是奇数，所以只有 s 是奇数的状态才是可到达的