搜索及其优化

- Noip貌似搜索题比较常见。
- 但是更高难度的比赛中，以搜索作正解的题很少，搜索主要用来打暴力。

- 不过，在完全只会暴力的情况下，若干个巧妙的合法性剪枝/高效的估价函数，可以拿到与数据湿度成正相关的得分。

折半搜索

- 如果问题能分成两个相对独立的部分，且两部分搜索结果能很方便合并，那么就可以利用折半搜索让搜索的层数变成原来的一半，大幅优化算法。
- 双向bfs就是一种折半搜索。

ZJOI 2005 九数码问题

- 有一个3*3的网格，方格上写有0~8这九个数字。
- 每次可以进行以下两种操作之一：
  - 将网格外围的8个方格按顺时针挪一个位置。
  - 将中间一行向右移动一个位置，最右边的方格被移到最左边。
- 给出一个初始状态，求把网格变成目标状态的最少的操作次数。

ZJOI 2005 九数码问题

- 双向bfs裸题

POJ 1186

- 已知一个n元高次方程：
  - $k_1 * x_1^{p_1} + k_2 * x_2^{p_2} + \cdots + k_n * x_n^{p_n} = 0$
- 假设未知数均为不大于m的正整数
- 求这个方程解的个数。
- n≤6，m≤150

POJ 1186

- 枚举前3个未知数的所有结果，存入哈希表，再去搜后3个未知数。

Rikka with Sequence II

- 一个长度为n的数列，现在可以从中选出若干个数，满足选出的所有数的平均数小于等于中位数。
- 求出满足条件的方案数。
- 数列长度为偶数的时候，中位数为排序后最中间的两个数的平均数。
- $n \leq 40$

Rikka with Sequence II

- 中位数一定是数列中某两个数的平均数，可以$O(n^2)$枚举中位数。
- 假设这两个数是l和r，给所有数减去中位数后，那么这时的方案就相当于从[1,l)和(r,n]中选出同等数量的数，使得它们的和≤0。
- 如果一个数原来的下标<l则权重为1，否则为-1。则问题转化为给出若干数，求选出一些数使得权重和为0，且权值和≤0的方案数。
- Meet in the middle: 将数列分成均等的两部分，分别处理出两边的所有情况，然后按权重分组，每组内排序统计答案即可。
- 将一个有序序列上全部加入一个数，依然有序，所以在排序时直接归并，优化掉一个n的复杂度，$O(2^{\frac{n}{2}} * n)$

迭代加深搜索

- 对于某些问题，状态数无限，而bfs无法承受空间复杂度，那么，可以限制dfs的层数，然后逐步放开限制。

codevs 1288

- 给出一个分数x/y，将其分解成若干个互不相同的形如1/a的分数。
- 求最优方案。
- 分解后的分数数量越少越好，如果数量一样，分母序列字典序越小越好。

codevs 1288

- 显然不能直接dfs，那么迭代加深分数个数。
- 然后限制分母序列不降，可以去掉很多无用状态。
- 限制了分数个数，而分母大小依然没有界限，两个剪枝：
  - 未确定分数的个数 * 当前分数大小 < x/y - 已确定分数之和，退出。
  - 已确定分数之和 > x/y，退出。

ZOJ 1937

- 构造一个数列$\{a_i\}$，要求：
  - $a_1 = 1$
  - $\forall i > 1$，$\exists j, k < i$，$a_i = a_j + a_k$
  - $a_t = n$
  - t尽量小
- n≤400

ZOJ 1937

- 迭代加深序列长度。
- 任意一个非单增的序列，都可以找到一个单增且不更劣的序列，那么我们可以限制序列单调递增。
- 可以从大到小枚举a_j和a_k，增强对后面的元素的限制。

启发式搜索

- 利用问题拥有的启发信息来引导搜索，~~说白了就是玄学~~。
- 估价函数：对每一个状态x引入了一个估价函数f(x)=g(x)+h(x)，其中g(x)是初始状态到当前状态的实际代价，而h(x)是当前状态到目标状态的估计代价。
- 为了保证搜索结果的正确性，h(x)不能劣于当前状态到目标状态的代价的最优值。也就是，h(x)一定是一个乐观的估计。

- 估价函数可以通过忽略部分条件来设计，用预处理或低复杂度的枚举/dp/贪心来计算。

启发式搜索

- A*：BFS的启发式版本。
- 两定点间的K短路：用当前节点x到终点的最短路作为h(x)。

- idA*：迭代加深DFS的启发式版本，f(x)=g(x)+h(x)超过代价限制则退出。
- 用估价函数进行最优性剪枝的普通dfs：有些地方也将其叫idA*。事实上，在OI中，这个比上面那两个用的多多了。

BZOJ 1085

- 在一个5*5的棋盘上有12个白色骑士，12个黑色骑士和1个空位。
- 一个骑士能按照“日”字的走法移动到空位上。
- 给定一个初始棋盘和目标棋盘，求最少的步数。

BZOJ 1085

- idA*：估价函数就是每个骑士到达任意一个合法位置的最短步数和，这个可以用dp预处理出来。

UVA 10605

- 给定一个n*m的网格，其中有k个关键位置。
- 用任意条端点在边界上、互不相交的折线，覆盖所有关键位置。
- 求最小的折线总长度。

- n,m≤11，k≤10

UVA 10605

- 关键位置很少，而难以处理的地方是折线不能相交。
- 忽略不可相交的条件，那么估价函数可以为：
  - h(S)=用若干条端点在边界上的、可相交的链，覆盖点集S，最短总长度。
- 这个用枚举子集/状压dp搞一下。

BZOJ 4622

- 给出平面上n个目标点，m台武器（只能用一次）。
- 使用一台武器，把距离它不超过K、符合要求的目标全部摧毁。
- 摧毁目标必须按编号顺序，而且同一时刻只有一个武器是开启的。
- 求摧毁全部目标至少需要多少武器。
- $n,m \leq 100$

BZOJ 4622

- 每个武器肯定会消灭编号连续的一些目标，可以考虑枚举区间划分，然后每个区间与能完全覆盖它的武器连边，跑二分图匹配。
- 忽略武器仅能使用一次的条件，能$O(n^3)$dp出一个估价函数，表示当前已经摧毁1~i的目标，还需要至少几个武器，最优性剪枝。
- dfs中每次将x方点加入一个新点表示一个编号区间，跑匈牙利，不能增广则退出，可行性剪枝。
- 预处理当前最大右端点，从大到小枚举右端点，降低搜索量，同时可以简化二分图中加删边/增广的回溯过程。

随机化搜索

- 将原问题的最优化条件转化成一个函数，然后通过随机化的过程不断调整当前函数的自变量取值（这个过程往往要考虑原问题的函数图像的一些性质来使用合适的方法），从而找到一定范围和精度要求下的最优答案。
- OI中一般都是爬山/模拟退火，实现起来更像是~~调参~~迭代而不是搜索。对于浮点数自变量的最优化问题/部分提答题极其有效。

剪枝/其他优化

- 剪枝分两类：
  - 可行性剪枝：这个就看具体题目了，比如斗地主之类的，讲的话没什么意义，在其他题上基本用不上。稍微通用点的，比如奇偶性。
  - 最优性剪枝：配合估价函数。
- 后继节点排序：玄学，优先搜更可能出现更优解的、加强未来限制的。
- 精确覆盖问题：DLX，实际是一个高效的常数优化，但在可重覆盖中效果微小。