动态规划

张超

南京外国语学校

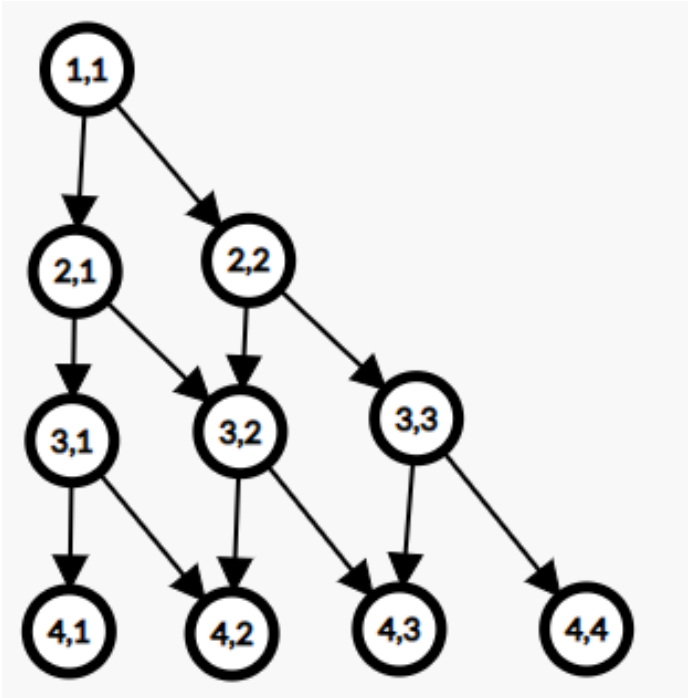
2023 年 11 月 9 日

- 记忆化搜索

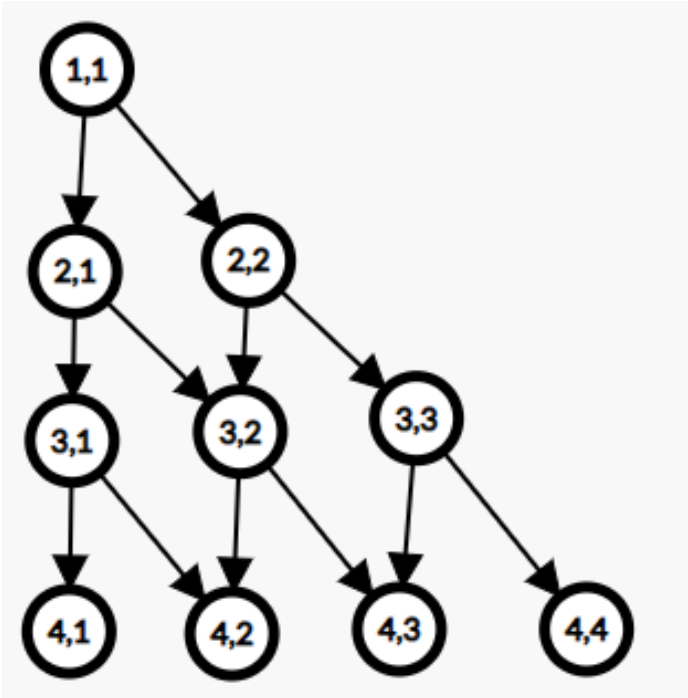
- 记忆化搜索
- 多阶段决策

- 例题 1-数字三角形
  给定 n 行数字，第 i 行有 i 个数字。从 (i,j) 可以走向 (i+1,j) 或者 (i+1,j+1)，求从 (1,1) 走到最后一行，可以获得最大数字和。$n \leq 5000$

- 递归搜索



- 递归搜索



- 很多点会被重复访问到，可以采用记忆化搜索，算过的点直接放回最优解。

- 递推

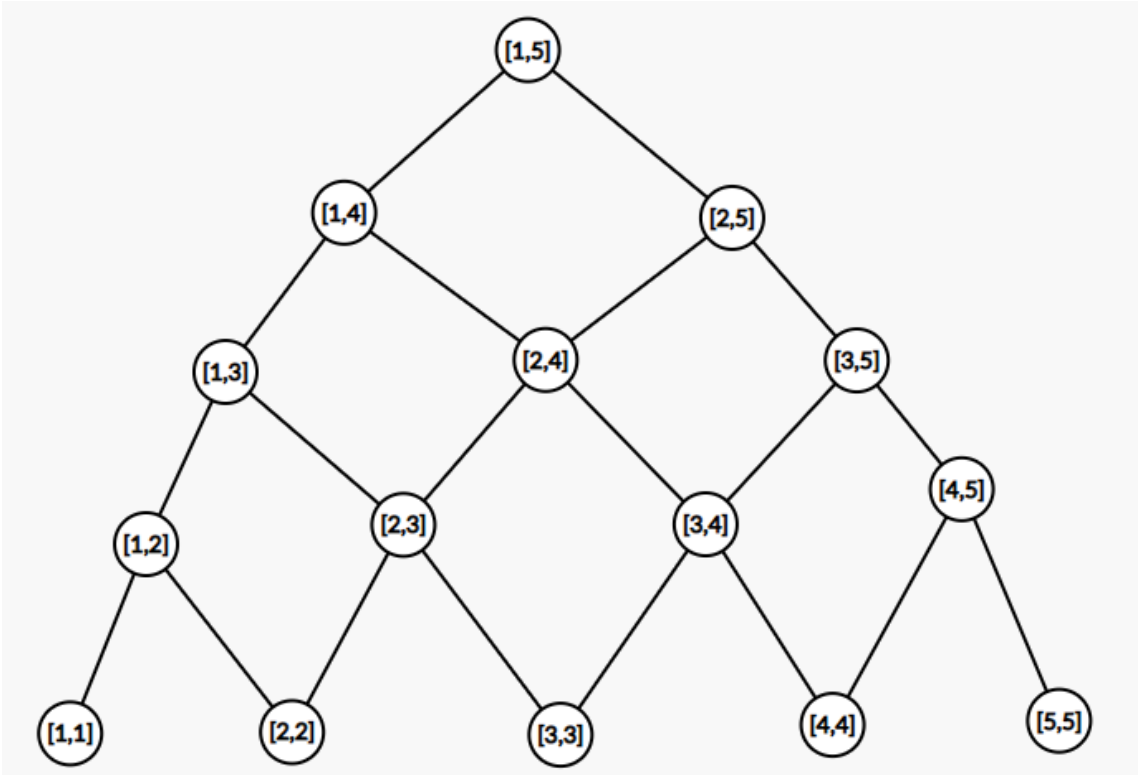
- 递推
- 从最后一层开始，向上更新。

- 递推
- 从最后一层开始，向上更新。
- 填表与刷表

- 例题 2-数字游戏
  给定一个长度为 n 的数组，两个人依次操作，每次可以从队列头部和尾部取出一个数，都想让取出的数总和最大。如果两个人的策略都最优，求先手和后手最后和。$n \leq 5000$

- 搜索：$dfs(L, R)$ 表示当前区间为 $[L, R]$，先手能获得大值

- 搜索：$dfs(L, R)$ 表示当前区间为 $[L, R]$，先手能获得大值
- 会让后手执行 $dfs(L+1, R)$ 或 $dfs(L, R-1)$，选小的让对方去执行。



- 递推

- 递推
- $dp[L][R] = sum[R] - sum[L-1] - min(dp[L+1][R], dp[L][R-1])$

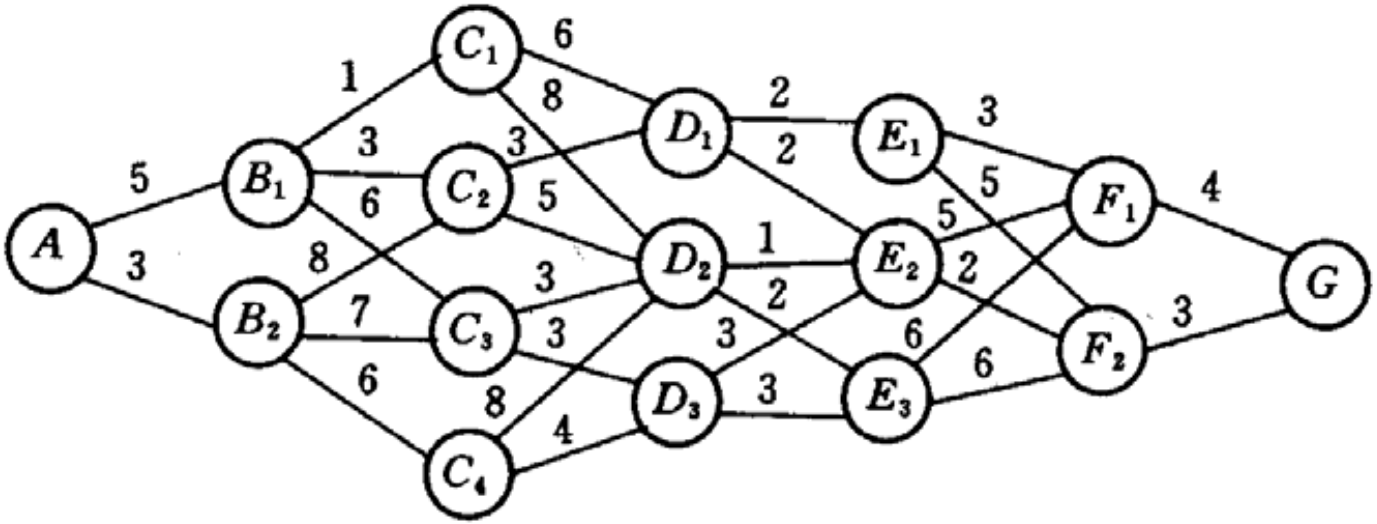
- 递推
- $dp[L][R] = sum[R] - sum[L-1] - min(dp[L+1][R], dp[L][R-1])$
- 空间优化

- 例 3-背包问题
  有 n 个物品，每个物品有体积 c_i 和价值 v_i，背包的体积不超过 V，选择一些物品装入到背包中，求可以获得的最大价值。
  $n \leq 1000, V \leq 20000, 1 \leq c_i, v_i \leq 1000$

- **01** 背包问题：每个物品可以选或者不选

- **01** 背包问题：每个物品可以选或者不选
- 记忆化搜索：把问题转换为第 n 个物品选还是不选，转移到 $n-1$ 个物品的问题。

- **01** 背包问题：每个物品可以选或者不选
- 记忆化搜索：把问题转换为第 n 个物品选还是不选，转移到 $n-1$ 个物品的问题。
- 多阶段决策：每一层的数值表示判断完前 i 个物品能获得最优解



- 多重背包问题：每种物品可以选 $1 \sim t_i$ 个

- 多重背包问题：每种物品可以选 $1 \sim t_i$ 个
- 直接循环，做 t_i 次 01 背包

- 多重背包问题：每种物品可以选 $1 \sim t_i$ 个
- 直接循环，做 t_i 次 01 背包
- 二进制优化：12 个物品，可以用 1 倍，2 倍，4 倍，5 倍四个物品替换。

- 多重背包问题：每种物品可以选 $1 \sim t_i$ 个
- 直接循环，做 t_i 次 01 背包
- 二进制优化：12 个物品，可以用 1 倍，2 倍，4 倍，5 倍四个物品替换。
- 还可以通过单调队列，优化到 $O(nV)$。

- 完全背包：每种物品数量没有限制。

- 完全背包：每种物品数量没有限制。
- 算出每种物品最多能装的数量，用多重背包的二进制优化。

- 完全背包：每种物品数量没有限制。
- 算出每种物品最多能装的数量，用多重背包的二进制优化。
- 可以在本层最优解的基础上，再加入一个物品，或者从上层直接转移过来。

- 完全背包：每种物品数量没有限制。
- 算出每种物品最多能装的数量，用多重背包的二进制优化。
- 可以在本层最优解的基础上，再加入一个物品，或者从上层直接转移过来。
- $dp[i][j] = max(dp[i-1][j], dp[i][j-c[i]] + v[i])$，第一维可以去掉。

- 分组背包：物品分为 m 组，每一组里面的物品最多选一个。

- 分组背包：物品分为 m 组，每一组里面的物品最多选一个。
- 每组物品是一个阶段，$dp[i-1]+(c_j, v_j)->dp[i]$

- 分组背包：物品分为 m 组，每一组里面的物品最多选一个。
- 每组物品是一个阶段，$dp[i-1]+(c_j,v_j)->dp[i]$
- 可以在上层最优解的基础上，加入本组中的每个物品，去更新当前层的最优解。

- 分组背包：物品分为 m 组，每一组里面的物品最多选一个。
- 每组物品是一个阶段，$dp[i-1]+(c_j,v_j)->dp[i]$
- 可以在上层最优解的基础上，加入本组中的每个物品，去更新当前层的最优解。
- 也可以优化第一维

- 例题 4-红警
  要攻打敌方的大本营，它的生命值为 m，有 n 种坦克，每种塔克一旦造出后，每一秒钟都能对大本营造成一定伤害。第 i 种坦克需要 t_i 秒完成制造，制作完成后，每秒产生 v_i 的攻击力，求最少用多少时间就能将敌方的大本营摧毁？
  $m \leq 500, 1 \leq n, t_i \leq 50$

- 时间：天然的阶段

- 时间：天然的阶段
- 每个阶段，需要知道当前的攻击力，敌方剩余血量。

- 时间：天然的阶段
- 每个阶段，需要知道当前的攻击力，敌方剩余血量。
- $dp[i][j]$ 表示第 i 秒，攻击力为 j 时，能构成的最大伤害是多少

- 时间：天然的阶段
- 每个阶段，需要知道当前的攻击力，敌方剩余血量。
- $dp[i][j]$ 表示第 i 秒，攻击力为 j 时，能构成的最大伤害是多少
- 枚举当前时间，制作哪种坦克，
  $dp[i + t[k]][j] = max(dp[i + t[k]][j], dp[i][j] + t[k] * j)$

- 判断某个时间内是否可以消灭，可以二分

- 判断某个时间内是否可以消灭，可以二分
- 知道结束时刻，那么每个坦克被造出后的伤害就知道了

- 判断某个时间内是否可以消灭，可以二分
- 知道结束时刻，那么每个坦克被造出后的伤害就知道了
- 费用提前计算：$dp[i+t[j]] = max(dp[i+t[j]], dp[i] + v[j] * (x - i - t[j]))$

- 考虑倒着来，每辆坦克被制造时，都放在最前面。那么它贡献的攻击时间就已知。

- 考虑倒着来，每辆坦克被制造时，都放在最前面。那么它贡献的攻击时间就已知。
- $dp[i + t[j]] = max(dp[i + t[j]], dp[i] + v[j] * i)$

- 例题 5-巨大背包
  有一个巨大的背包，体积是 V，有 n 件物品，每个物品的体积为 a_i，价值为 b_i，可以使用无限个。求背包能装下的最大价值。
  $n, a_i, b_i \leq 500, V \leq 10^9$

- 因为体积太大了，贪心考虑，肯定性价比高的那件物品一定会放很多。

- 因为体积太大了，贪心考虑，肯定性价比高的那件物品一定会放很多。
- 设性价比最大的物品编号为 x，体积为 a[x]，则其他物品 i 的使用数量一定不会超过 a[x]，否则可以用物品 x，替换掉 a[x] 个 i 物品，结果更优。

- 因为体积太大了，贪心考虑，肯定性价比高的那件物品一定会放很多。
- 设性价比最大的物品编号为 x，体积为 a[x]，则其他物品 i 的使用数量一定不会超过 a[x]，否则可以用物品 x，替换掉 a[x] 个 i 物品，结果更优。
- 进一步证明，其他物品的总数量不会超过 a[x]

- 微观 DP: 设 $m = n * max(a_i)$, 把 $[1, m]$ 这部分体积做背包

- 微观 DP: 设 $m = n * max(a_i)$, 把 $[1, m]$ 这部分体积做背包
- 宏观贪心: 枚举 $i(i \leq m)$，这部分 dp 完成，$V - i$ 这部分贪心，用性价比最大的物品

- 微观 DP: 设 $m = n * max(a_i)$, 把 $[1, m]$ 这部分体积做背包
- 宏观贪心: 枚举 $i(i \leq m)$，这部分 dp 完成，$V - i$ 这部分贪心，用性价比最大的物品
- 总的复杂度是 $O(m * n^2)$

- 微观 DP: 设 $m = n * max(a_i)$, 把 $[1, m]$ 这部分体积做背包
- 宏观贪心: 枚举 $i(i \leq m)$，这部分 dp 完成，$V - i$ 这部分贪心，用性价比最大的物品
- 总的复杂度是 $O(m * n^2)$
- 还可以通过同余最短路解决

- 例题 6-删除背包
  给定 n 个物品，每个物品的体积为 a_i
  求删除第 i 件物品之后，构成 $1 \sim m$ 各个体积的方案数模 10。
  $n, m \leq 2000$

- 考虑普通的 01 背包运算，加入第 i 件物品时：
  for(int j=m;j>=a[i];j−)dp[j]+=dp[j-a[i]];

- 考虑普通的 01 背包运算，加入第 i 件物品时：
  for(int j=m;j>=a[i];j−)dp[j]+=dp[j-a[i]];
- 逆运算，删除第 i 件物品时：
  for(int j=a[i];j<=m;j++)dp[j]-=dp[j-a[i]];

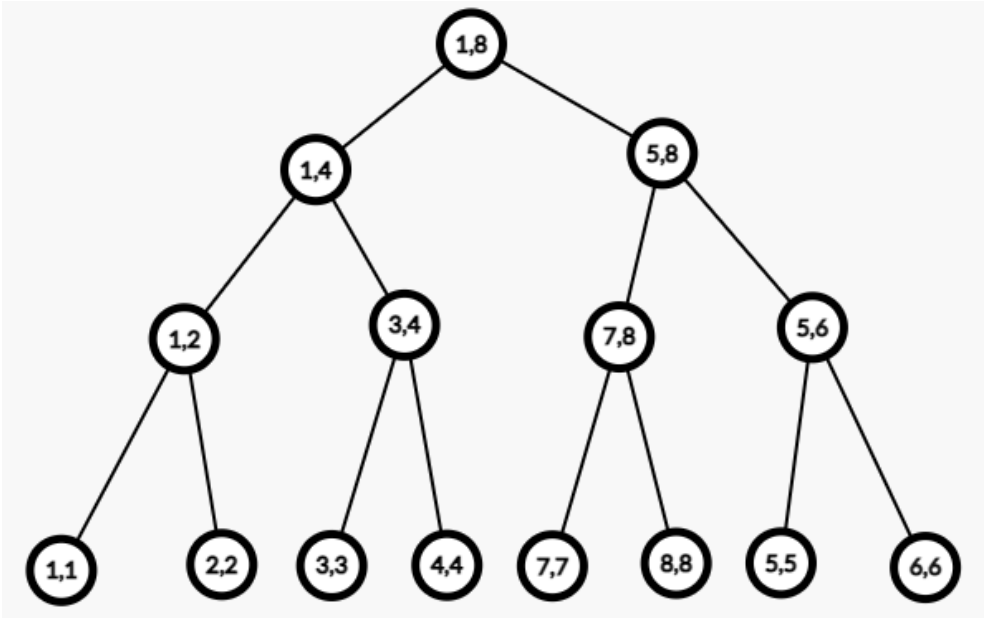
- 考虑普通的 01 背包运算，加入第 i 件物品时：
  for(int j=m;j>=a[i];j−)dp[j]+=dp[j-a[i]];
- 逆运算，删除第 i 件物品时：
  for(int j=a[i];j<=m;j++)dp[j]-=dp[j-a[i]];
- 复杂度是 $O(nm)$

- 考虑处理删除第 2 件物品和第 3 件物品时，其他物品很多都是一样的。第 1 和 4 ～ n 件物品都还是一样的。这两个问题有很多重叠部分。

- 考虑处理删除第 2 件物品和第 3 件物品时，其他物品很多都是一样的。第 1 和 4 ∼ n 件物品都还是一样的。这两个问题有很多重叠部分。
- 考虑使用分治，利用公共部分。总的复杂度是 $O(m * n \log n)$

- 考虑处理删除第 2 件物品和第 3 件物品时，其他物品很多都是一样的。第 1 和 4 ∼ n 件物品都还是一样的。这两个问题有很多重叠部分。
- 考虑使用分治，利用公共部分。总的复杂度是 $O(m * n \log n)$
- 递归到本层时，先把上一层的 dp 结果转移过来，往下递归某个儿子时，把另一个儿子的所有物品都加入到其中再递归下去。

- 考虑处理删除第 2 件物品和第 3 件物品时，其他物品很多都是一样的。第 1 和 4 ∼ n 件物品都还是一样的。这两个问题有很多重叠部分。
- 考虑使用分治，利用公共部分。总的复杂度是 $O(m * n \log n)$
- 递归到本层时，先把上一层的 dp 结果转移过来，往下递归某个儿子时，把另一个儿子的所有物品都加入到其中再递归下去。
- 递归到物品 3 时，此时 dp 中包含 $[5,8],[1,2],[4,4]$



- 例题 7-鱿鱼游戏 1
  小 Q 被抓来参加鱿鱼游戏了。一个长度为 n 的木板被悬挂在空中，参与游戏的人会被放在距离左边界 x 的位置上，接下来，游戏者会随机向左和向右走，m 步之后，仍然在木板上的就赢得奖金。求小 Q 赢得奖金的概率。
  $1 \leq n \leq 10^3$，$0 \leq m \leq 2 \times 10^4$，$0 \leq x \leq n$

- 考虑状态，需要知道步数和当前坐标。

- 考虑状态，需要知道步数和当前坐标。
- 当前状态最终活下的概率是它后继状态存活概率乘以转移概率之和

- 考虑状态，需要知道步数和当前坐标。
- 当前状态最终活下的概率是它后继状态存活概率乘以转移概率之和
- 用记忆化搜索实现，终态的概率为 1。

- 例题 8-鱿鱼游戏 2
  小 Q 被抓来参加鱿鱼游戏了。一个长度为 n 的木板被悬挂在空中，参与游戏的人会被放在距离左边界 x 的位置上，接下来，游戏者会随机向左和向右走。m 步之后，仍然在木板上的就赢得奖金。每次的指令都是随机的，掉下去或者时间到了，游戏都会结束，求游戏进行的期望秒数。
  $1 \leq n \leq 10^3$，$0 \leq m \leq 2 \times 10^4$，$0 \leq x \leq n$

- 考虑状态，需要知道步数和当前坐标。

- 考虑状态，需要知道步数和当前坐标。
- 当前状态的期望步数是它后继状态步数加 1 乘以转移的概率

- 考虑状态，需要知道步数和当前坐标。
- 当前状态的期望步数是它后继状态步数加 1 乘以转移的概率
- 用记忆化搜索实现，终态的期望为 0。

- 例题 9-图连通
  给出无向图 G(V,E). 已经有一些边连接这些点。每次操作任意加一条非自环的边 (u,v), 每条边的选择是等概率的. 问使得 G 连通的期望操作次数. (|V|<=30,|E|<=1000)

- 倒着考虑，图连通的时候，期望操作步数为 0

- 倒着考虑，图连通的时候，期望操作步数为 0
- 图连通的前一次，应该是两个连通块。再前一次是 3 个连通块。

- 倒着考虑，图连通的时候，期望操作步数为 0
- 图连通的前一次，应该是两个连通块，再前一次是 3 个连通块。
- 只需要知道每个连通块的大小，而不关心连通内的具体编号。

- 倒着考虑，图连通的时候，期望操作步数为 0
- 图连通的前一次，应该是两个连通块。再前一次是 3 个连通块。
- 只需要知道每个连通块的大小，而不关心连通内的具体编号。
- 把连通块按大小排序，当作状态。

- 倒着考虑，图连通的时候，期望操作步数为 0
- 图连通的前一次，应该是两个连通块。再前一次是 3 个连通块。
- 只需要知道每个连通块的大小，而不关心连通内的具体编号。
- 把连通块按大小排序，当作状态。
- 状态总数为 30 的整数拆分。

- 倒着考虑，图连通的时候，期望操作步数为 0
- 图连通的前一次，应该是两个连通块。再前一次是 3 个连通块。
- 只需要知道每个连通块的大小，而不关心连通内的具体编号。
- 把连通块按大小排序，当作状态。
- 状态总数为 30 的整数拆分。
- 记忆化搜索实现。