2024 暑假集训——背包 DP

xuyan1

2024-08-16

符号说明

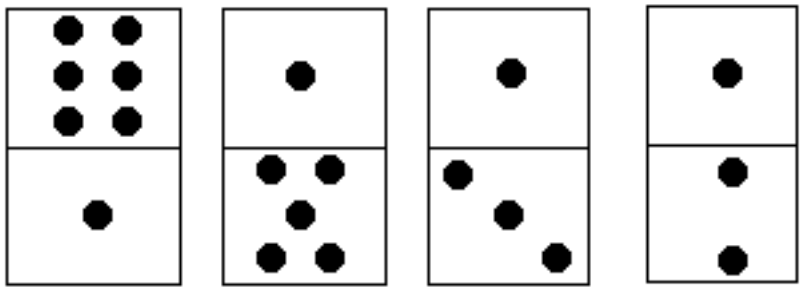
- n：物品个数
- W, m：背包容量上限
- $w[i]$：第i个（或第i种）物品的重量
- $v[i]$：第i个（或第i种物品的价值

01 背包

- 基础二维 DP：$f[i][j]=\begin{cases}\max(f[i-1][j],f[i-1][j-w[i]]+v[i])\\ f[i-1][j]\end{cases}$
- 一维 DP：$f[j]=\max(f[j],f[j-w[i]]+v[i])$，先枚举$i$再枚举$j$
  - 倒序枚举j
  - 原地滚动数组，如果想不清楚，建议用$i\%2$法滚动数组
  - 背包合并：$f[j]$是（考虑前$i-1$个物品时）重量上限与最大价值的离散函数，考虑分配给i多少空间（0或$w[i]$两种情况）
- 方案数模型：P1164 小 A 点菜
- bool-DP：P1877 音量调节
- 状态设计技巧：P1282 多米诺骨牌、P1156 垃圾陷阱

01 背包——多米诺骨牌

- 给定一行 n 个多米诺骨牌，每个由上下2个方块组成，用最少的旋转次数使多米诺骨牌上下2行点数之差达到最小($1 \leq n \leq 1000$)



- 考虑最后一张旋转还是不旋转
- 方法 1：记前i个骨牌，上方块点数之和j，下方块点数之和k，获得的最优解$dp[i][j][k]$，$O\left(n(6n)^2\right) = O(36n^3)$

01 背包——多米诺骨牌

- 方法 2：记前i个骨牌，上方块点数之和-下方块点数之和的差为j时，最优解$dp[i][j]$
- $dp[i][j] = \min(dp[i][j], dp[i-1][j-(b[i]-a[i])]+1)$
  - $j-(b[i]-a[i]) \in [-5000, 5000]$
- $dp[i][j] = \min(dp[i][j], dp[i-1][j-(a[i]-b[i])])$
  - $j-(a[i]-b[i]) \in [-5000, 5000]$
- $dp[1][a[1]-b[1]] = 0$，$dp[1][b[1]-a[1]] = 1$

01 背包——垃圾陷阱

- 奶牛落在了一个深井中，井深为D，有T个时间，每个时间会落下来一个物品，物品有高度$h[i]$和可以加的血量$f[i]$，奶牛初始血量为10，如果它可以爬出深井，求最早时间；否则，求最长存活时间（$n \le 100, h[i] \le 25$）
- 方法 1：记$dp[i][j][k]$为前i个时间，达到血量为j，高度为k的最短时间，记$d = a[i].t - a[i-1].t$
- $dp[i][j][k] = dp[i-1][j+d-a[i].f][k] + d$
  - $j - a[i].f \ge 0, j + d - a[i].f \le 2500$
- $dp[i][j][k] = \min(dp[i][j][k], dp[i-1][j+d][k-a[i].h] + d)$
  - $k - a[i].h \ge 00, j + d \le 2500$
- $dp[0][10][0] = 0, dp[i][j][k] = \infty$

01 背包——垃圾陷阱

- 奶牛落在了一个深井中，井深为D，有T个时间，每个时间会落下来一个物品，物品有高度$h[i]$和可以加的血量$f[i]$，奶牛初始血量为10，如果它可以爬出深井，求最早时间；否则，求最长存活时间（$n \le 100, h[i] \le 25$）
- 记$dp[i][j]$为前i个时间，达到高度为j时的最大生命值，记$d = a[i].t - a[i-1].t$
- $dp[i][j] = ?$
- 第一问答案：$\min_{j \ge D, dp[i][j] \ge 0} a[i].t$
- 第二问答案：$\max_{dp[i][0] \ge 0} a[i].t + dp[i][0]$

完全背包

- 基础二维 DP：$f[i][j]=\begin{cases}\max(f[i-1][j],f[i][j-w[i]]+v[i])\\ f[i-1][j]\end{cases}$
- 一维 DP：$f[j]=\max(f[j],f[j-w[i]]+v[i])$，先枚举$i$再枚举$j$
  - 正序枚举j
  - 原地滚动数组
- 找零钱模型：P1474 Money System、P5020 [NOIP2018 提高组]货币系统
- 投资模型：P1853 投资的最大效益、P5662[CSP-J2019]纪念品

多重背包

- $c[i]$为第i种物品的个数
- $f[i][j]=\max\limits_{0\le k\le \min(c[i],j/w[i])} f[i-1][j-k*w[i]]+k*v[i]$
- 复杂度：$O\left(W\sum_i c[i]\right)$、$O\left(\sum_i\left(1+\frac{W}{W[i]}\right)\frac{W}{w[i]}*w[i]\right)$
- 一维 DP：$f[j]=\max\limits_{0\le k\le \min(c[i],j/w[i])}(f[j-k*w[i]]+k*v[i],f[j])$
  - 滚动数组优化
  - 背包合并：$f[j]$是（考虑前$i-1$种物品时）重量上限与最大价值的离散函数，考虑分配给i多少空间（$k*w[i]$）
- 混合背包
- P1833 樱花、P2347 砝码称重
- P1941 [NOIP2014 提高组] 飞扬的小鸟

多重背包优化——转为 01 背包问题

- 暴力优化：将第i种物品表示$c[i]$个价值为$v[i]$的“单位物品”，n种物品的多重背包问题即转化为了$\sum_i c[i]$个物品的 01 背包问题
- 二进制物品优化：将第i种物品表示$\log_2 c[i]$量级的物品，n种物品的多重背包问题即转化为了$\sum_i \log_2 c[i]$个物品的 01 背包问题
  - 例：$c[i] = 7$时，可以表示为3个物品，分别为$1, 2, 4$倍的“单位物品”
  - $c[i] = 10$呢？
- P1776 宝物筛选
- P6567[NOI Online #3 入门组]买表
- P1782 旅行商的背包

多重背包优化——转为 01 背包问题

```
for(int i=1;i<=n;i++){
    for(int k=1;k<=c[i];k*=2){ // 枚举 k 倍“单位物品”
        c[i]-=k;
        for(int j=W;j>=k*w[i];j--)
          f[j]=max(f[j],f[j-k*w[i]]+k*v[i]);
    }
    int k=c[i]; // 最后剩下的
    for(int j=W;j>=k*w[i];j--)
      f[j]=max(f[j],f[j-k*w[i]]+k*v[i]);
}
```

多重背包优化——单调队列

- $f[i][j]=\max\limits_{0\le k\le \min\left(c[i],\frac{j}{w[i]}\right)} f[i-1][j-k*w[i]]+k*v[i]$
- $j=\frac{j}{w[i]}*w[i]+r$，$r=j\%w[i]$，记$j'=\frac{j}{w[i]}$，当r确定时，j'可以确定j
- $f[i][j']=\max\limits_{\max(j'-c[i],0)\le k'\le j'} f[i-1][k']+(j'-k')*v[i]$
- 维护计算$f[i][j']$时的$\arg\max\limits_{\max(j'-c[i],0)\le k'\le j'} f[i-1][k']+(j'-k')*v[i]$
- $j'++$时，目标是$\arg\max\limits_{\max(j'+1-c[i],0)\le k'\le j'+1} f[i-1][k']+(j'+1-k')v[i]$，队头出队后，单调队列可以复用，再让$j'+1$入队，再求$f[i][j'+1]$。
- 对每个余数r都求一遍即可，复杂度$O(nW)$

多重背包优化——单调队列

- $\arg\max\limits_{\max(j'-c[i],0)\le k'\le j'} f[i-1][k']+(j'-k')*v[i]$

```
#define cal(i,j,r,k) f[i-1][k*w[i]+r]+(j-k)*v[i]
for(int r=0;r<w[i];r++){ head=1;tail=0;
  for(int j=0;j*w[i]+r<=W;j++){
      while(head<=tail&&q[head]<max(j-c[i],0)) head++;
      while(head<=tail&&cal(i,j,r,q[tail])<cal(i,j,r,j))
tail--;
      q[++tail]=j;
      f[i][j*w[i]+r]=cal(i,j,r,q[head]);
  }
} // define 仅为展示代码，实现时不建议写
```

背包撤消

- 计数类 01 背包中，把第i个物品“拿出来”
- $f[j]=f[j]-f[j-w[i]]$
- 正序枚举j，$f[j-w[i]]$为不含第i个物品的方案数
- P4141 消失之物

作业

- 团队题单
  - 背包 DP-基本操作
  - 背包 DP