状态压缩动态规划

动态规划的状态有时候比较难，不容易表示出来，需要用一些编码技术，把状态压缩的用简单的方式表示出来。典型方式：当需要表示一个集合有哪些元素时，往往利用 2 进制用一个整数表示。

*：一般有个数据 n<16 或者 n<32 这个很可能就是状态 DP 的标志，因为我们要用一个 int 的二进制来表示这些状态。要注意好这些数据规模的提示作用。

*：确定了为状态 DP，那么第一步就是预处理，求出每行所有可能的状态了，cnt 记录总的状态数，stk[]记录所有的可能状态。以炮兵阵地为例：

```
int cnt, stk[MAX];


void findStk(int n){    // 求出所有可能的状态。
    for(int i = 0; i < (1<<n); i ++)
      if(ok(i)){                    //  判断这种状态可不可行。
         stk[cnt] = i;
         sum[cnt ++] = getSum(i);     //  计算这种状态包含了几个炮兵。
      }
}


bool ok(int x){          //  判断状态 x 是否符合，即是否会出现两个大炮间隔小于 2。
    if(x & (x<<1)) return false;
    if(x & (x<<2)) return false;
    return true;
}


int getSum(int x){       //  求出状态 x 中安装了多少门大炮，x 的二进制有几个 1。
    int num = 0;
    while(x > 0){
       if(x & 1) num ++;
       x >>= 1;
    }
    return num;
 }
```

*：然后就是 DP 部分了，明确好状态转移方程。先特殊处理第 1 行，然后按状态转移方程求出剩下的值。

经典问题：TSP

一个 n 个点的带权的有向图，求一条路径，使得这条路经过每个点恰好一次，并且路径上边的权值和最小（或者最大）。或者求一条具有这样性质的回路，这是经典的 TSP 问题。

n <= 16 (重要条件,状态压缩的标志)

如何表示一个点集：

由于只有 16 个点，所以我们用一个整数表示一个点集：

例如：

5 ＝ 0000000000000101；（2 进制表示）

它的第 0 位和第 2 位是 1，就表示这个点集里有 2 个点，分别是点 0 和点 2。

31 ＝ 0000000000011111； （2 进制表示）

表示这个点集里有 5 个点，分别是 0，1，2，4，5；

所以一个整数 i 就表示了一个点集；整数 i 可以表示一个点集，也可以表示是第 i 个点。

状态表示：

dp[i][j]表示经过点集 i 中的点恰好一次，不经过其它的点，并且以 j 点为终点的路径,权值和的最小值，如果这个状态不存在，就是无穷大。

状态转移：

单点集：状态存在 dp[i][j] = 0；否则无穷大。非单点集：

1：状态存在 dp[i][j] = min(dp[k][s] + w[s][j])

k 表示 i 集合中去掉了 j 点的集合，s 遍历集合 k 中的点并且 dp[k][s]状态存在，点 s 到点 j 有边存在，w[s][j]表示边的权值。

2.：状态不存在 dp[i][j]为无穷大。

最后的结果是： min(dp[(1<<n) – 1][j]) (0 <= j < n)；

技巧：利用 2 进制，使得一个整数表示一个点集，这样集合的操作可以用位运算来实现。例如从集合 i 中去掉点 j：

k = i & (~(1<<j)) 或者 k = i - (1<<j)