**程式設計研討專題3**

110403518 林晉宇 12/21

**一、題意:**

給n個硬幣(n<=200)，每個硬幣權重不同(<=100)，有三個小朋友A,B,C，要求拿最多錢的人與拿最少錢的人差的最小值。

**二、方法1 - 原始0/1 dp**

定義二維陣列**dp[i][j]** (boolean型態)，小朋友A拿到的錢是**i**，小朋友B拿到的錢是**j**，而小朋友C拿到的錢即為錢的總量扣掉A及B的錢 **(sum-i-j)**。用01背包方法實作，如果**dp[i][j]=1** 代表該方案可行，反之。最後找出差最小的方案。

**Pseudo Code:**

*初始化-> n, sum=0, ans=INF, coin[200], dp[20000][20000]=0*

*dp[0][0]=1 //代表 A拿0元 B拿0元 C拿全部*

*for(i: 0~n) input <- coin[i], sum+=coin[i]*

*for(k: 0~n)*

*for(i: sum~0)*

*for(j: sum~0)*

*if(dp[i][j])*

*dp[i+coin[k]][j]=1, dp[i][j+coint[k]]=1*

*for(i: 0~sum)*

*for(j: 0~sum)*

*if(dp[i][j])    ans=min(拿最多錢的人-拿最少錢的人)*

1. **方法2 - Bit Operation**

用C++的bitset實作，bit operation好處是”or”比較的bit變少(原本一個bool是1byte=8bit, 現在變成1bit)，理論上可以快8倍。

**Pseudo Code:**

*初始化-> n, sum=0, ans=INF, coin[200]*

*初始化-> bitset<20000> dp[20000]*

*dp[0]=1*

*for(i: 0~n) input <- coin[i], sum+=coin[i]*

*for(k: 0~n)*

*for(i: sum~0)*

*for(j: sum~0)*

*if(dp[i][j])*

*dp[j+coin[i]]=dp[j+coin[i]]|dp[j]*

//代表A拿走coin[i]

*dp[j]=dp[j]|dp[j]<<coin[i]*

//代表B拿走coin[i]

*for(i: 0~sum)*

*for(j: 0~sum)*

*if(dp[i][j])    ans=min(拿最多錢的人-拿最少錢的人)*

1. **時間比較**
2. 原始dp(coin陣列型態int): 9174ms
3. 原始dp(coin陣列型態bool): 6326ms
4. BitOperation: 1194ms
5. **結論**

經過實測發現用bit operation的方法確實可以比原始dp快不少，因為相比之下，or運算不用做那麼多次，同時透過c++的bitset，可以一次比較整個bitset，不用再多一個迴圈。Top down的版本最後沒有實作出來，所以只有拿兩個版本的去做比較。