USA Tutor

Time is Mooney

Analysis by David Yang

Statement Summary

You are given N<=1000 nodes and M<=2000 one-way roads.

Bessie collects m_i dollars every time she visits node i.

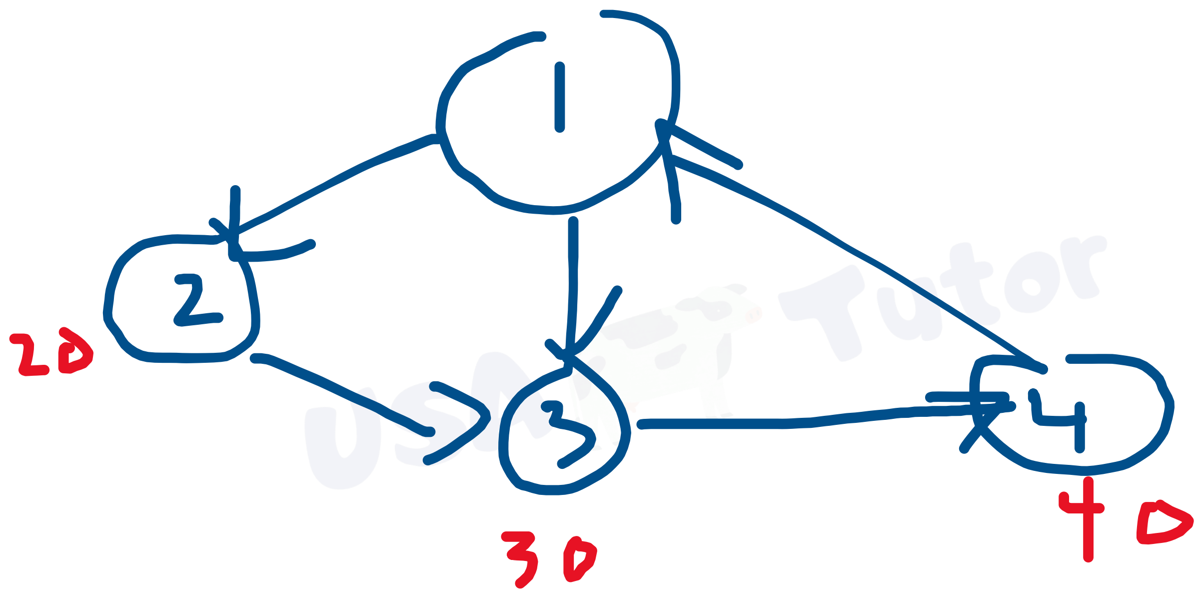
Moving across roads takes 1 day. There is a C<=1000 multiplier.

Your total cost for moving is days *C.

Find the best amount of money Bessie can walk away with.

My Sample

$C = 1$

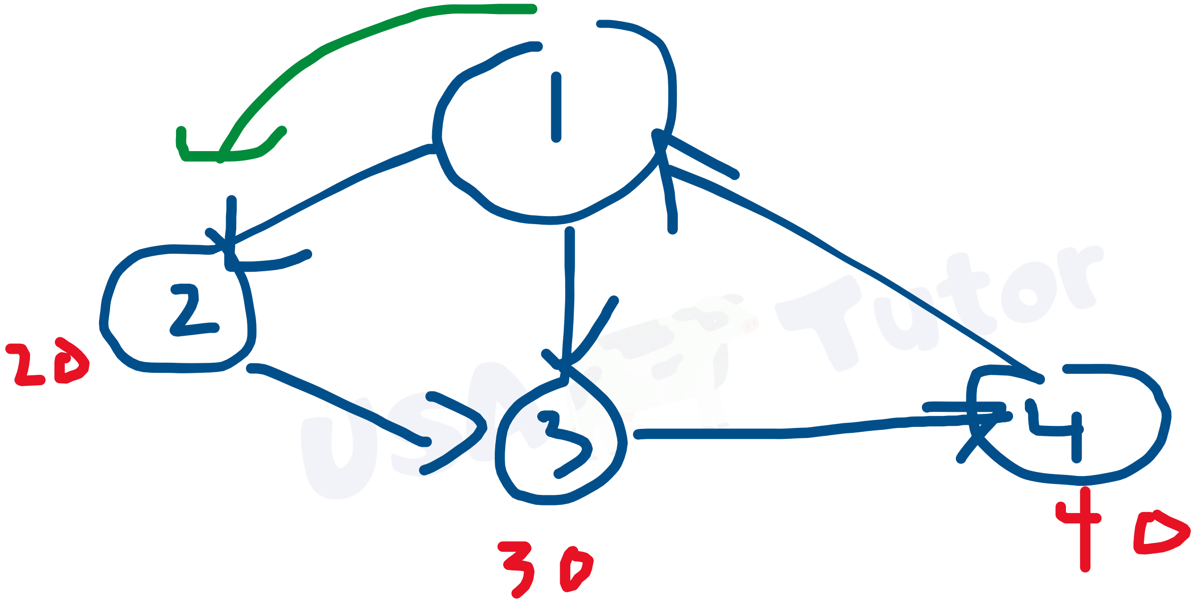


My Sample

$C = 1$

$S = 20$

$m = 1$

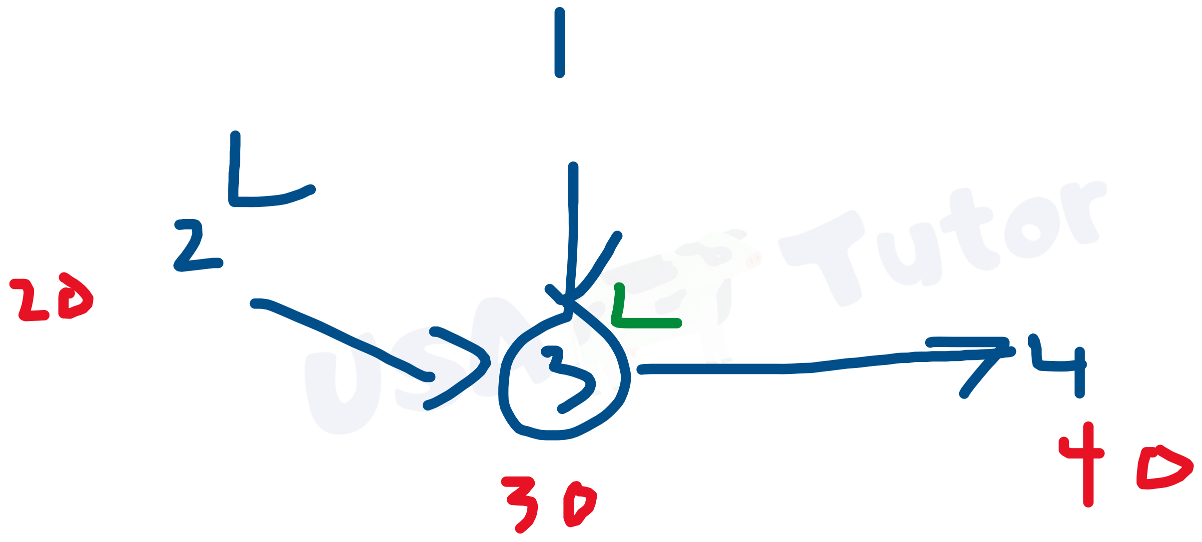


My Sample

$S=30$

$m=1$

$C=1$

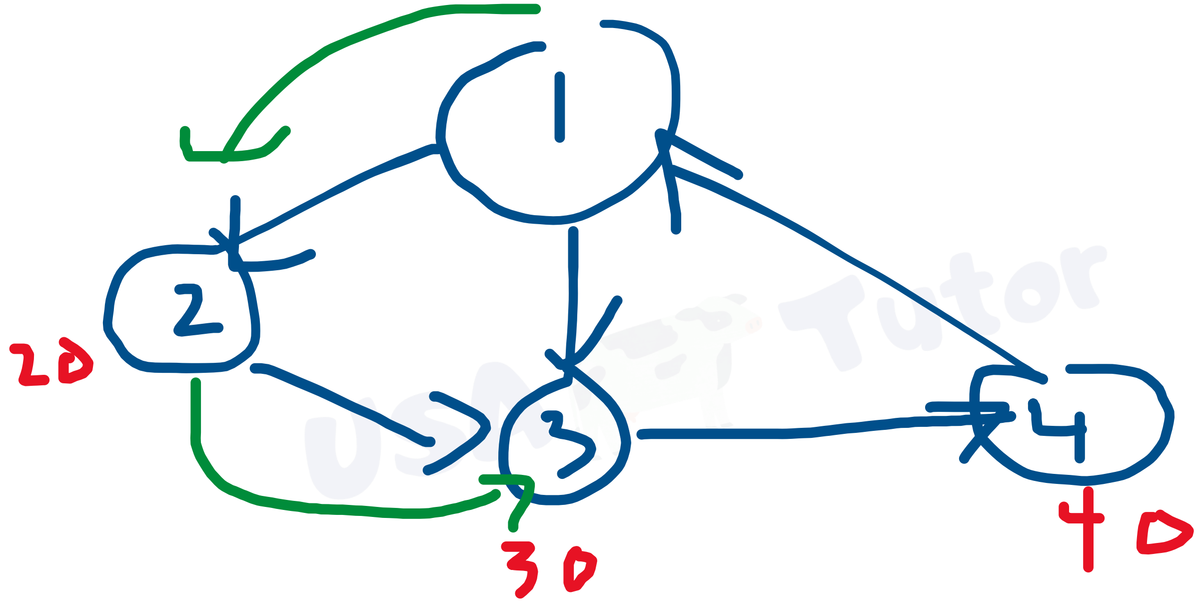


My Sample

$C = 1$

$S = 50$

$m = 2$

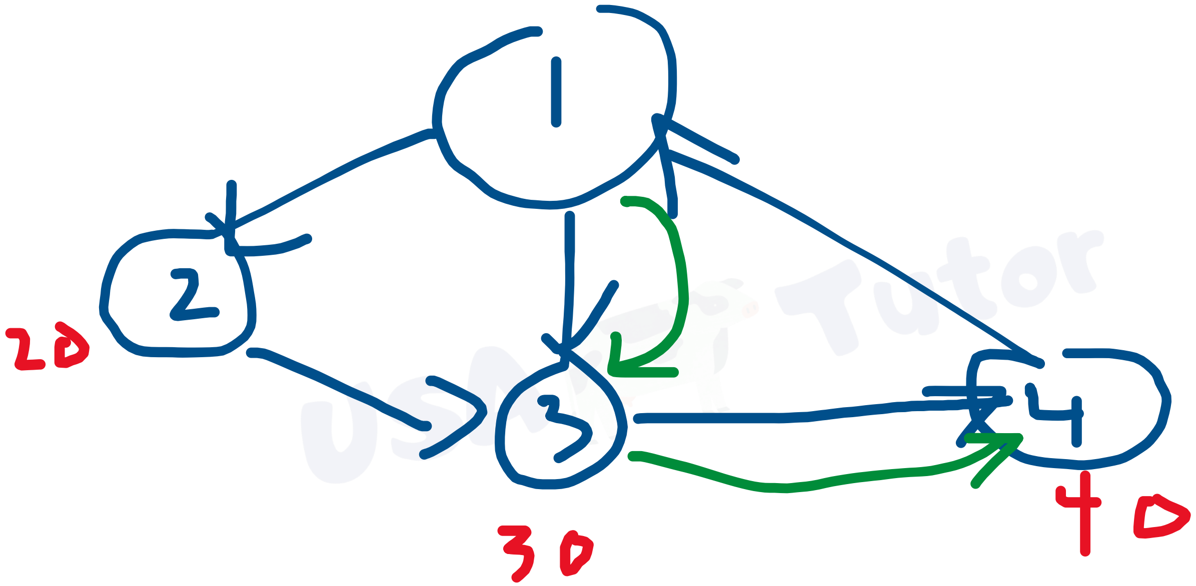


My Sample

$C = 1$

$S = 70$

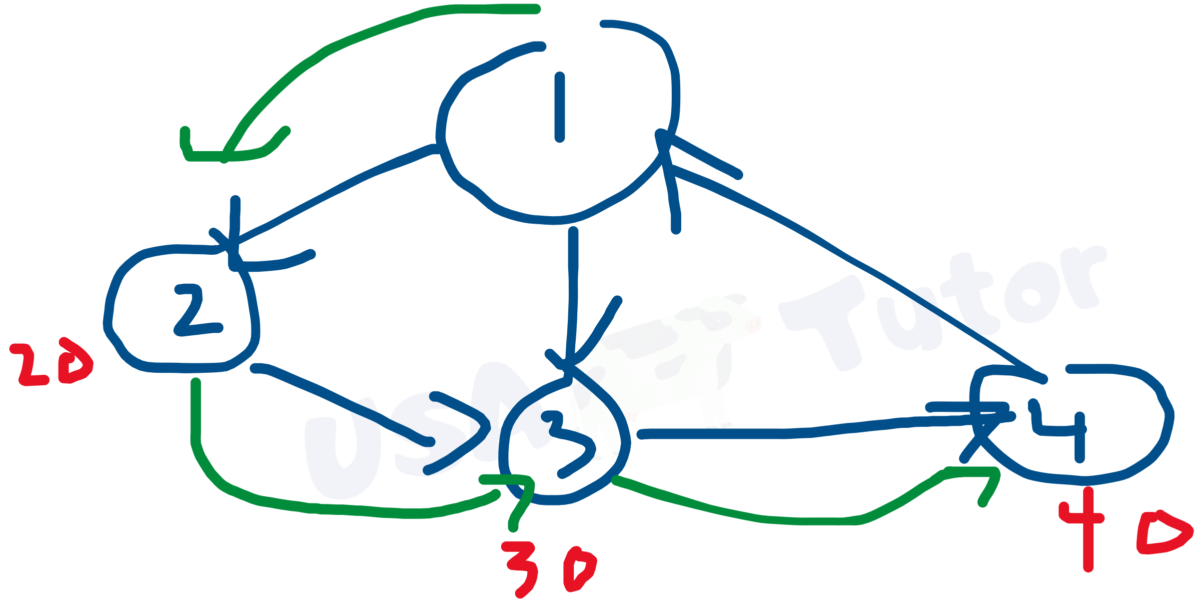
$m = 2$



My Sample

$C = 1$

$S = 90$
$m = 3$

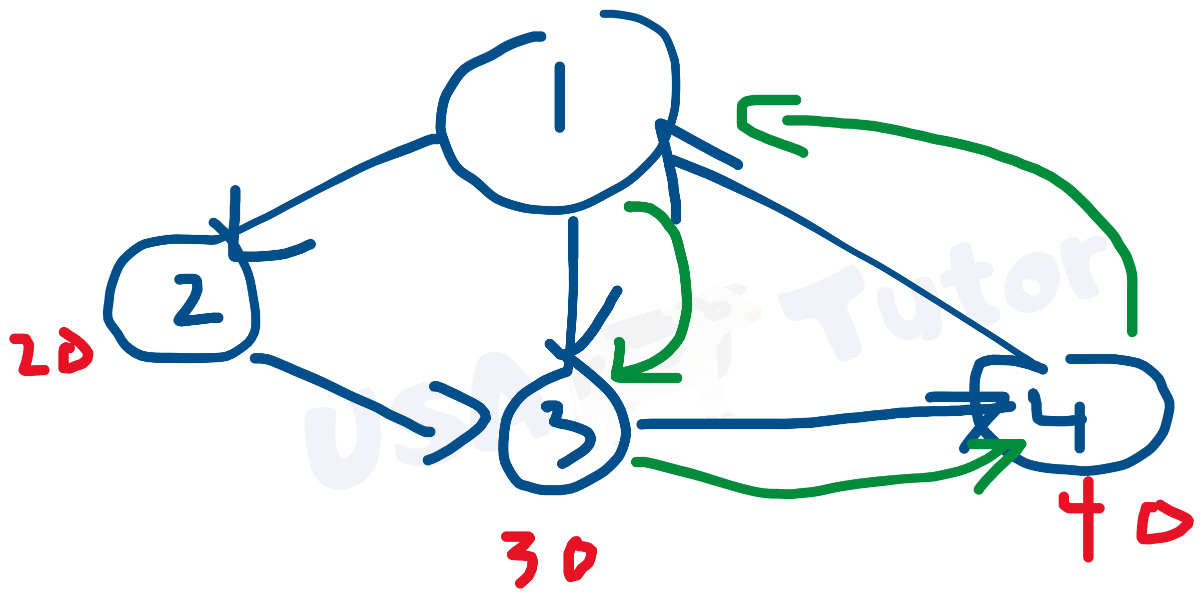


My Sample

$C = 1$

$S = 70$

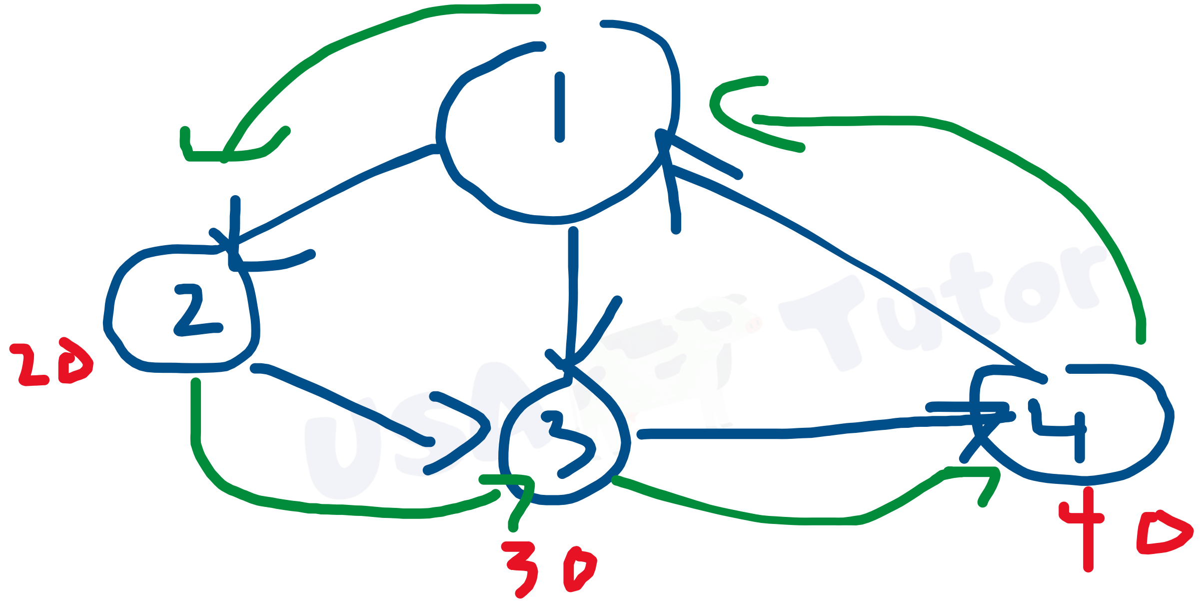
$m = 3$



My Sample

$C = 1$

$S = 90$
$m = 4$



Ideas from our Sample

We found 2 cycles that end at 1:

One with length 3, sum 70

One with length 4, sum 90

If we choose 1 cycle, we can get 90-(4*4)

Choosing 2 Cycles

Let A = sum 70, length 3

Let B = sum 90, length 4

A+A = 140 – (3+3)^2 = 140 – 36 = 104

A+B = 160 – (3+4) ^2 = 160 – 49 = 111

B+B = 180 – (4+4)^2 = 180 – 64 = 116

Choosing 3 Cycles

Let A = sum 70, length 3

Let B = sum 90, length 4

A+A+A = 210 – (3+3+3)^2 = 210 – 81 = 129

A+A+B = 230 – (3+3+4) ^2 = 230 – 100 = 130

A+B+B = 250 – (3+4+4)^2 = 250 – 121 = 129

B+B+B = 270 – (4+4+4)^2 = 270 – 144 = 126

Choosing 4 Cycles

Let A = sum 70, length 3

Let B = sum 90, length 4

A+A+A+A = 280 – (3+3+3+3)^2 = 280 – 144 = 136

A+A+A+B = 300 – (3+3+3+4) ^2 = 300– 169 = 131

A+A+B+B = 320 – (3+3+4+4)^2 = 320 – 196 = 124

A+B+B+B = 340 – (3+4+4+4)^2 = 340 – 225 = 115

B+B+B+B = 360 – (4+4+4+4)^2 = 360 - 256 = 104

Choosing 5 Cycles

Let A = sum 70, length 3

Let B = sum 90, length 4

$A+A+A+A+A = 350 – (3+3+3+3+3)^2 = 350 – 225 = 125$

We can see that the point of diminishing returns has started. $125<136$.

What was the point of that?

In that example, we had choices. 2 choices.

Pick cycle A

Pick cycle B

Interesting things that relate to DP:

- It didn't matter which order I picked my cycles in
- We have a choice every time
- If I did this recursively we would have 2^n, bad.

Picking better ways to do a thing

Think not in the terms of cycles, but nodes

What dp[][] do I need to restore any given answer?

dp[node][moves] = max sum

dp[node][moves] = max sum

Node is current node

Moves is how many moves

Sum is your money (does not include time cost)

You can always calculate time cost at the END, by doing

sum – C* (moves^2)

Solution

BFS from node 1

Stop when your moves >1000

Add neighbor to queue if:

dp[neighbor][moves+1] < dp[cur][moves] + money[neighbor]

set dp[n][m+1] = dp[c][m] + m[n]

Read in Input

```
int N, M, C;
cin>>N>>M>>C;
vector<int> money;
for(int i=0; i<N; i++){
    int num; cin>>num;
    money.push_back(num);
}
for(int i=0; i<M; i++){
    int a, b; cin>>a>>b;
    a--; b--;
    adj[a].push_back(b);
}
```

BFS

```
dp[0][0]=0;
while(dq.size()>0){
    pii cur = dq.front();
    int curNode = cur.first;
    int curMoves = cur.second;

    dq.pop_front();
    if(curMoves>1000) continue;
    for(int next: adj[curNode]){
        if( dp[next][curMoves+1] < dp[curNode][curMoves] +
money[next]){
            dp[next][curMoves+1] = dp[curNode][curMoves] +
money[next];
            dq.emplace_back(next, curMoves+1);
        }
    }
}
```

Print out Answer

```
int best=0;
for(int i=0; i<1001; i++){
   best = max(best, dp[0][i] - C* (i*i));
}
cout<<best<<endl;
```