Algorithms

Solve any problem to achieve a rank
View Leaderboard

Topics: Dynamic Programming and Bit Masking

Dynamic Programming and Bit Masking

TUTORIAL PROBLEMS

First thing to make sure before using bitmasks for solving a problem is that it must be having small constraints, as solutions which use bitmasking generally take up exponential time and memory.

Let's first try to understand what Bitmask means. Mask in Bitmask means hiding something. Bitmask is nothing but a binary number that represents something. Let's take an example. Consider the set $A = \{1, 2, 3, 4, 5\}$. We can represent any subset of A using a bitmask of length 5 , with an assumption that if $i^{th}(0 \leq i \leq 4)$ bit is set then it means i^{th} element is present in subset. So the bitmask 01010 represents the subset $\{2, 4\}$

Now the benefit of using bitmask. We can set the i^{th} bit, unset the i^{th} bit, check if i^{th} bit is set in just one step each. Let's say the bitmask, $b = 01010$.

Set the i^{th} bit: $b|(1 << i)$. Let $i = 0$, so,

$$(1 << i) = 00001$$

$$01010|00001 = 01011$$

So now the subset includes the 0^{th} element also, so the subset is $\{1, 2, 4\}$.

Unset the i^{th} bit: $b\&!(1 << i)$. Let $i = 1$, so,

$$(1 << i) = 00010$$

$$!(1 << i) = 11101$$

$$01010\&11101 = 01000$$

Now the subset does not include the 1^{st} element, so the subset is $\{4\}$.

Check if i^{th} bit is set: $b\&(1 << i)$, doing this operation, if i^{th} bit is set, we get a non zero integer otherwise, we get zero. Let $i = 3$

$$(1 << i) = 01000$$

$$01010\&01000 = 01000$$

Clearly the result is non-zero, so that means 3^{rd} element is present in the subset.

Let's take a problem, given a set, count how many subsets have sum of elements greater than or equal to a given value.

Algorithm is simple:

```
solve(set, set_size, val)
    count = 0
    for x = 0 to power(2, set_size)
        sum = 0
        for k = 0 to set_size
            if kth bit is set in x
                sum = sum + set[k]
        if sum >= val
             count = count + 1
    return count
```

To iterate over all the subsets we are going to each number from 0 to 2^{set_size}-1
The above problem simply uses bitmask and complexity is O(2^nn).

Now, let's take another problem that uses dynamic programming along with bitmasks.

Assignment Problem:
There are N persons and N tasks, each task is to be alloted to a single person. We are also given a matrix $cost$ of size $N \times N$, where $cost[i][j]$ denotes, how much person i is going to charge for task j. Now we need to assign each task to a person in such a way that the total cost is minimum. Note that each task is to be alloted to a single person, and each person will be alloted only one task.

The brute force approach here is to try every possible assignment. Algorithm is given below:

```
assign(N,  cost)
    for i = 0 to N
        assignment[i] = i             //assigning task i to person i
    res = INFINITY
    for j = 0 to factorial(N)
        total_cost = 0
        for i = 0 to N
            total_cost = total_cost + cost[i][assignment[i]]
        res = min(res, total_cost)
        generate_next_greater_permutation(assignment)
    return res
```

The complexity of above algorithm is $O(N!)$, well that's clearly not good.

Let's try to improve it using dynamic programming. Suppose the state of dp is $(k, mask)$, where k represents that person 0 to $k-1$ have been assigned a task, and $mask$ is a binary number, whose i^{th} bit represents if the i^{th} task has been assigned or not.
Now, suppose, we have $answer(k, mask)$, we can assign a task i to person k, iff i^{th} task is not yet assigned to any peron i.e. $mask \& (1 << i) = 0$ then, $answer(k+1, mask|(1 << i))$ will be given as:

$$answer(k+1, mask|(1 << i)) = min(answer(k+1, mask|(1 << i)), answer(k, mask) + cost[k][i])$$

One thing to note here is k is always equal to the number set bits in $mask$, so we can remove that. So the dp state now is just $(mask)$, and if we have $answer(mask)$, then

$$answer(mask|(1 << i)) = min(answer(mask|(1 << i)), answer(mask) + cost[x][i])$$

here x =number of set bits in $mask$.
Complete algorithm is given below:

```
assign(N, cost)
    for i = 0 to power(2,N)
        dp[i] = INFINITY
    dp[0] = 0
    for mask = 0 to power(2, N)
        x = count_set_bits(mask)
        for j = 0 to N
            if jth bit is not set in i
                dp[mask|(1<<j)] = min(dp[mask|(1<<j)], dp[mask]+cost[x][j])
    return dp[power(2,N)-1]
```

Time complexity of above algorithm is $O(2^n n)$ and space complexity is $O(2^n)$.
This is just one problem that can be solved using DP+bitmasking. There's a whole lot.
Let's go to another problem, suppose we are given a graph and we want to find out if it contains a Hamiltonian Path. This problem can also be solved using DP+bitmasking in $O(2^n n^2)$ time complexity and $O(2^n n)$ space complexity. Here's a link to it's solution.

Contributed by: Vaibhav Jaimini

TEST YOUR UNDERSTANDING

Micro and Graph

Micro is having a graph containing N vertices and M edges, each edge is having an associated weight. He knows for sure that the graph contains several Hamiltonian Paths (path that visits all the vertices exactly once). He challenged his friend to find out the minimum element of the array which stores the strength of all Hamiltonian Paths in the graph. Strength of a Hamiltonian Path is the sum of weight of edges in that path. But Micro himself does not know the answer to it. Help Micro in solving this problem.

Input:
First line consists of two space separated integers denoting N and M.
Each of the following M lines consists of three space separated integers X, Y and Z denoting that there is an edge between vertices X and Y having weight Z.

Output:
Print the answer in a new line.

Constraints:
$1 \leq N \leq 10$
$1 \leq M \leq 20$
$1 \leq X, Y \leq N$
$1 \leq Z \leq 100$

SAMPLE INPUT

```
4 4
1 2 4
2 3 5
2 4 2
3 4 1
```

SAMPLE OUTPUT

```
7
```

Enter your code or Upload your code as file. Save C (gcc 5.4.0)

```
/*
// Sample code to perform I/O:

scanf("%s", name);              // Reading input from STDIN
printf("Hi, %s.\n", name);      // Writing output to STDOUT

// Warning: Printing unwanted or ill-formatted data to output will cause the test cases to fail
*/

// Write your code here

```

1:1

Provide custom input

Press Ctrl-space for autocomplete suggestions.

COMPILE & TEST SUBMIT

COMMENTS (18) SORT BY: Relevance

Login/Signup to Comment

Georgy Chebanov 9 months ago
test #3 is incorrect (n=6, but exist edges connected to 7)
▲ 6 votes ● Reply ● Message ● Permalink

Nikhil Kumar Arya 10 months ago
code of assignment problem :
#include<bits/stdc++.h>
using namespace std;

```
int main()
{
int n;
cin>>n;
int cost[n][n];

for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
cin>>cost[i][j];
}
}
int powi = pow(2,n);
int dp[powi];
memset(dp,INT_MAX,sizeof(dp));

dp[0] = 0;
int mask;
for(mask =0;mask<powi;mask++)
{
int x = __builtin_popcount(mask);
for(int j=0;j<n;j++)
{
if(mask&(1<<j)!=0)
{
dp[mask|(1<<j)] = min(dp[mask|(1<<j)],dp[mask]+cost[x][j]);
}
}

}

cout<<dp[powi-1];
return 0;
}
```

▲ 1 vote ● Reply ● Message ● Permalink

Ankit Beniwal 6 months ago

can you please explain the little bit more above solution ?

▲ 0 votes ● Reply ● Message ● Permalink

kaushik raj Edited 2 months ago

your code gives wrong output for
3
1 1 1
1 1 1
1 1 1

▲ 2 votes ● Reply ● Message ● Permalink

Sachin Kumar 15 days ago

does this code really work?

▲ 0 votes ● Reply ● Message ● Permalink

Anurag Sharma Edited 5 months ago

In assign(N,cost)
What does the 8th line
if jth bit is not set in i
mean ? In above line i is referring to which i in code ?

▲ 3 votes ● Reply ● Message ● Permalink

S Prakash 5 months ago

In the tutorial, there seems to be a mistake in Assignment problem.
"if jth bit is not set in i" in pseudo code ??
There is no i there

▲ 0 votes ● Reply ● Message ● Permalink

Priyanshu Varshney 5 months ago

I think it is "mask" instead of i.

▲ 0 votes ● Reply ● Message ● Permalink

Vasja Pavlov 6 months ago

Output for case #4 should be 107 not 103...

▲ 2 votes ● Reply ● Message ● Permalink

Ankit Kumar a year ago

Let i=3 (1<<i)=01000 should be there.
It is written wrong the shifting has been done wrong for the third case if ith bit is set

▲ 1 vote ● Reply ● Message ● Permalink

Himanshu Garg 8 months ago

why we are using min() function , i think we are always comparing infinity with some value. please explain my doubt.

▲ 1 vote ● Reply ● Message ● Permalink

Honey Shah 8 months ago

i didnt get assignment problem ? Can someone explain me please ?

▲ 1 vote ● Reply ● Message ● Permalink

coder 6 months ago

Is there any link to find solution with an example for this assignment problem.

▲ 0 votes ● Reply ● Message ● Permalink

Ajay Verma 5 months ago

for brute force solution of assign prob, should not the complexity be n*n! instead of just n! ??

▲ 1 vote ● Reply ● Message ● Permalink

kaay Sii Edited a year ago

why is there an edge numbered '7' in the third input file when it says n = 6?

▲ 0 votes ● Reply ● Message ● Permalink

Ravi Rahul 6 months ago

Input cases seem to be wrong

▲ 0 votes ● Reply ● Message ● Permalink

Nisha Aggarwal Edited 5 months ago

```
/* IMPORTANT: Multiple classes and nested static classes are supported */
/*
* uncomment this if you want to read input.
//imports for BufferedReader
import java.io.BufferedReader;
import java.io.InputStreamReader;
//import for Scanner and other utility classes*/
import java.util.*;
class Graph {
static int v;
static int[][] arr;

Graph(int v)
{
this.v=v;
this.arr=new int[v][v];
for(int i=0;i<v;i++)
{
for(int j=0;j<v;j++)
{
arr[i][j]=0;
}
}
}

public static void addEdge(int a,int b,int w)
{
arr[a][b]=w;
}
public static void main(String args[] ) throws Exception {
Scanner input=new Scanner(System.in);
int n=input.nextInt();
int m=input.nextInt();
Graph graph=new Graph(n);
int[][] arr1=new int[m][m];
int totalWeight=0;
for(int i=0;i<m;i++)
{
int a=input.nextInt();
int b=input.nextInt();
int w=input.nextInt();
graph.addEdge(a-1,b-1,w);
graph.addEdge(b-1,a-1,w);
arr1[i][0]=a-1;
arr1[i][1]=b-1;
totalWeight+=w;

}
//System.out.println("hello");
for(int i=0;i<m;i++)
{
totalWeight-=arr[arr1[i][0]][arr1[i][1]];
//System.out.println("hello"+totalWeight);
graph.BFS(arr1[i][0],arr1[i][1],totalWeight);
totalWeight+=arr[arr1[i][0]][arr1[i][1]];
}

System.out.print(minNumber);

}

public static void BFS(int s,int z,int w)
{
LinkedList<Integer> queue=new LinkedList<Integer>();
boolean[] visited=new boolean[v];
visited[s]=true;
queue.add(s);
int l=arr[s][z];
arr[s][z]=0;
int strengthcal=0;
while(!queue.isEmpty())
{
int temp=queue.poll();
for(int i=0;i<arr[temp].length;i++)
{
if(arr[temp][i]!=0 && !visited[i])
{
visited[i]=true;
strengthcal+=arr[temp][i];
```

```
queue.add(i);
}
}
}
// System.out.println("hello=="+w+"=="+strengthcal);
arr[s][z]=l;
calculateStrength(strengthcal,w);
// System.out.println("hello"+strengthcal+"=="+w);

}
static int minNumber=Integer.MAX_VALUE;
public static void calculateStrength(int a,int b)
{
int max=-99999;
if(a>(b-a))
{
max= a;

}
else
{
max=(b-a);
}

if(minNumber>max)
{
minNumber=max;
}
/* if(a*(b-a)<minNumber)

{
minNumber=a*(b-a);
}*/

}
}
```

0 votes ● Reply ● Message ● Permalink

Nisha Aggarwal 5 months ago

In third case vertex given 6 but 7 is also present .

0 votes ● Reply ● Message ● Permalink