# On the Construction and Algorithm of Network Flow

**Translation by Kirito Feng**

JIANG PENG

CTSC 2001

## Contents

# §1 Introduction

## §1.1 Understanding Network Flow Algorithms

The network flow algorithm is an efficient and practical algorithm. Compared with other graph theory algorithms, the model is more complex and the programming complexity is higher. However, it integrates other graph theory algorithms (such as the shortest path algorithm), so it is applicable. The scope is also wider, and it can often solve some problems that seem to be NP-hard or NP-complete.

## §1.2 Specific Applications

The most challenging part of the application of network flow for problem solving is the construction of models. The standard model is often useless, and you need to understand the nature of various components of network flow (e.g. point capacity, upper and lower limits of capacity, multiple edges, etc.), and our experiences, as well as our creativity.

# §2 Sample Problems

## §2.1 Project Development Planning

Macrosoft is preparing to develop a future development plan. The development projects proposed by various departments of the company are summarized into a planning table, which contains many projects. For each project, the required investment or expected profit is given in the planning table Since the implementation of certain projects must rely on the development of other projects, the projects on which they depend are essential if they are to be implemented. As the president of Macrosoft, please pick a subset of these projects to maximize your company's net profit.

### Input

The input file includes the number of projects, $N$, and the budget $C_i$ for each project, and the set of projects $P_i$ on which it depends. The format is as follows:

The first line is $N$;

The next $N$ rows each describe an project: the first integer is $C_i$. A positive integer represents profit, and a negative integer represents investment. The remaining numbers are the set of projects that project $i$ is dependant on.

### Output

The first line should contain the company's largest net profit. This should be followed by a project selection plan that maximizes net profit. If there are multiple scenarios, the one with the least number of selected projects should be output. The projects should be separated by a newline, and output in ascending order.

### Constraints

$0 \leq N \leq 1\,000$

$-1\,000\,000 \leq C_i \leq 1\,000\,000$

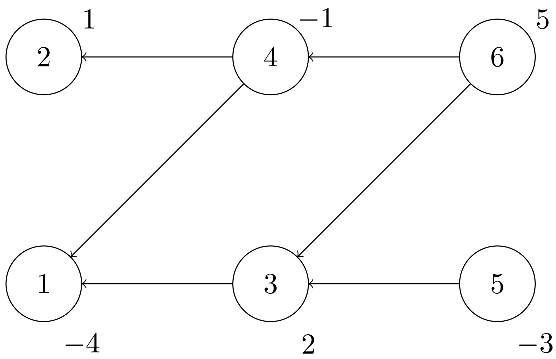
### Sample Run

| Sample Input | Sample Output |
|---|---|
| 6 | 3 |
| -4 | 1 |
| 1 | 2 |
| 2 2 | 3 |
| -1 1 2 | 4 |
| -3 3 | 6 |
| 5 3 4 | |

## §2.2 Solution

We can model this problem with graph theory. Given a directed graph $G = (V, E)$ with $N$ vertices, each vertex represents a project, and the vertices have a weight $C_i$ representing the budge of the item. The directed edges are used to represent dependencies between projects, and the directed edges from $u$ to $v$ indicated the project $u$ depends on project $v$.

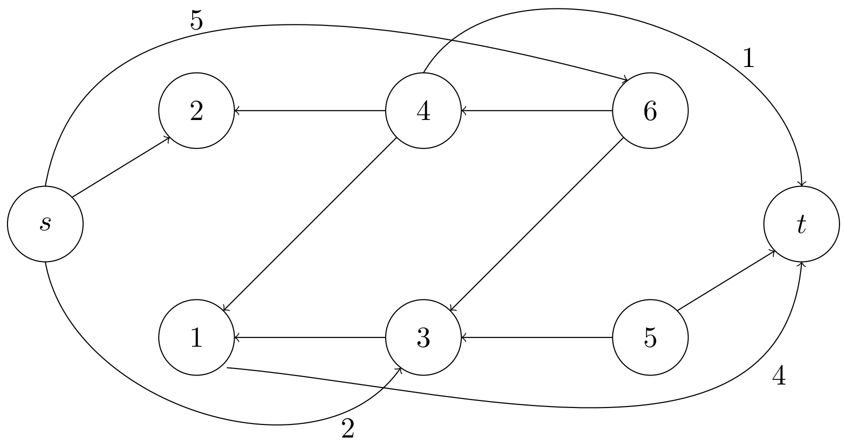
The problem is now equivalent to finding a subset of vertices $V'$ such that for all edges $(u, v) \in E$, if $u \in V'$, then $v \in V'$, and the sum of the weights of all vertices in $V'$ is maximized.




Candidate solutions include complete search, which would take $\mathcal{O}(2^N)$. Another candidate solution is dynamic programming, however, this is a general graph, not a tree or DAG, thus the solution would be similar to the complete search approach and would still have an exponential time complexity.

Thus we use network flow. We construct a node for each project, and add a source $s$ and sink $t$. If a project $i$ depends on project $j$, then an edge with infinite capacity is construct from vertex $i$ to vertex $j$. For each project, we add an edge of with capacity $C_i$ from the source if $C_i \geq 0$, and an edge of capacity $-C_i$ to the sink if $C_i < 0$.

Let the minimum cut of this graph be $F$. Then the maximum profit is $R - F$, where $R$ is the budget of all profitable projects, and the vertices in the cut represent the items selected by the optimal plan.




### §2.2.1 Proof of Correctness of the Algorithm

- Establish a one-to-one correspondence between the project selection scheme and the cut $(S, T)$ of the flow network: Any one of the project selection schemes can correspond to one cut $(S, T)$ in the network, $S = s + \{\text{all selected items}\}$, $T = V - S$. For any project selection scheme that does not satisfy the dependency relationship, the corresponding cut has the following characteristics:

  There is a capacity of $+\infty$ edge $< u, v >$, $u$ belongs to $S$ and $v$ belongs to $T$. At this time, the capacity of the cut is infinite, and obviously it is impossible to be the minimum cut of the network.

- For any cut $(S, T)$, if it corresponds to a qualified scheme, its net profit is $R - C(S, T)$. The reasons for the actual net profit being less than the previous $R$ are:

  1. If the profit item $i$ is not selected, that is, the vertex $i$ is included in $T$, then there is an edge with a capacity $C_i$ from the source $s$ to the vertex $i$.
  2. If the profit item $i$ is not selected, that is, the vertex $i$ is included in $T$, then there is an edge with a capacity $C_i$ from the source $s$ to the vertex $i$.

  In summary, the smaller the cut capacity, the greater the net profit of the plan.

- The minimum cut method: According to the maximum flow minimum cut theorem, the minimum cut of the network can be obtained by the method of maximum flow.

The key to solving this problem lies in the establishment of mathematical models of flow networks. The unique feature of this problem modeling is that the previous network flow problem usually uses the flow representation solution, and this problem uses the cut representation solution and makes full use of the nature of the cut. The flow is only the means to find the minimum cut. This opened up a new way of thinking about constructing network flows to solve problems.

Looking at this problem at first glance, it is quite difficult to link it to the network flow. You must be proficient in the various properties of the flow network, and after repeated analogous attempts, you can discover the commonality between them.

## §3 Associated Thinking

As a derivative of this problem, estimate a completion time for each project and assume that the company can only perform one project at a time. The question now is: how to choose some projects that can be completed in a given time, so that the company gets the most benefit. I have not found an effective algorithm for this problem, and I hope that interested students can study together.