第二章 网络中的若干优化问题

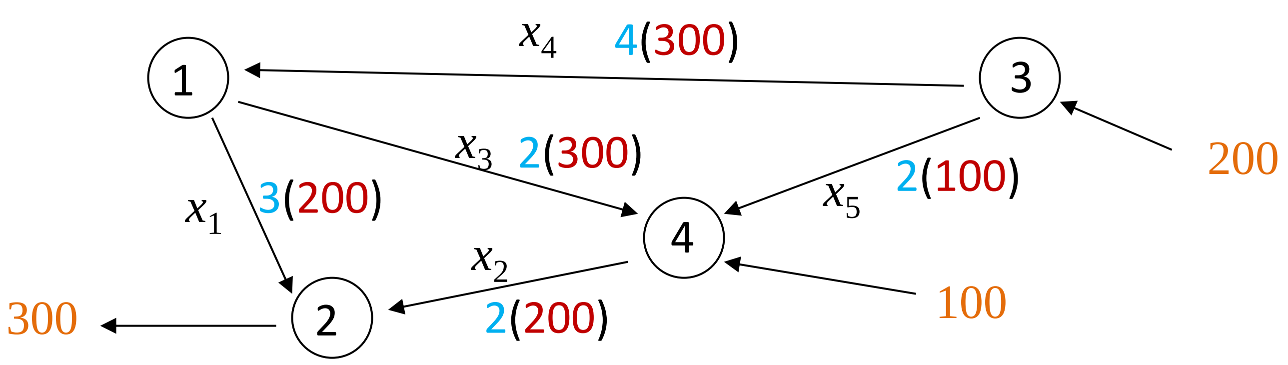
第三节 最小费用流问题

内容

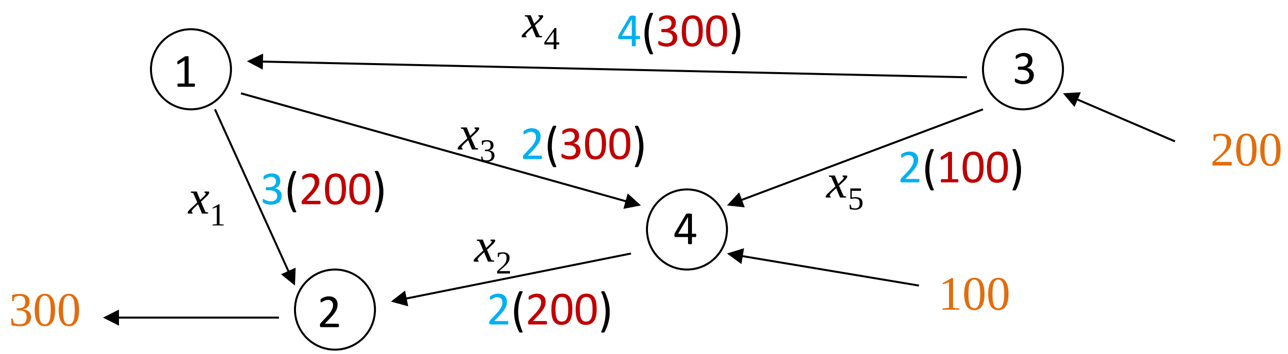
- 最小费用流问题的定义
- 最小费用流问题的最优性条件
- 最小费用流问题的算法
  - Cycle cancelling algorithm
  - Successive shortest path algorithm
  - Primal-Dual algorithm
- 最小费用流问题的应用

最小费用流问题的定义

- 最小费用流问题(minimum cost flow problem)解决的是当网络中的弧有不同容量限制和运输成本时，如何以最小的成本来完成给定的运输任务的问题。



最小费用流问题的定义



- 最小费用流问题可以写成一个线性数学规划问题
- 目标函数：总的运输成本

$$\min \sum_{ij \in A} c_{ij} x_{ij} = 3x_1 + 2x_2 + 2x_3 + 4x_4 + 2x_5$$

- 约束:　1. 节点流量守恒约束　2. 弧上的容量限制

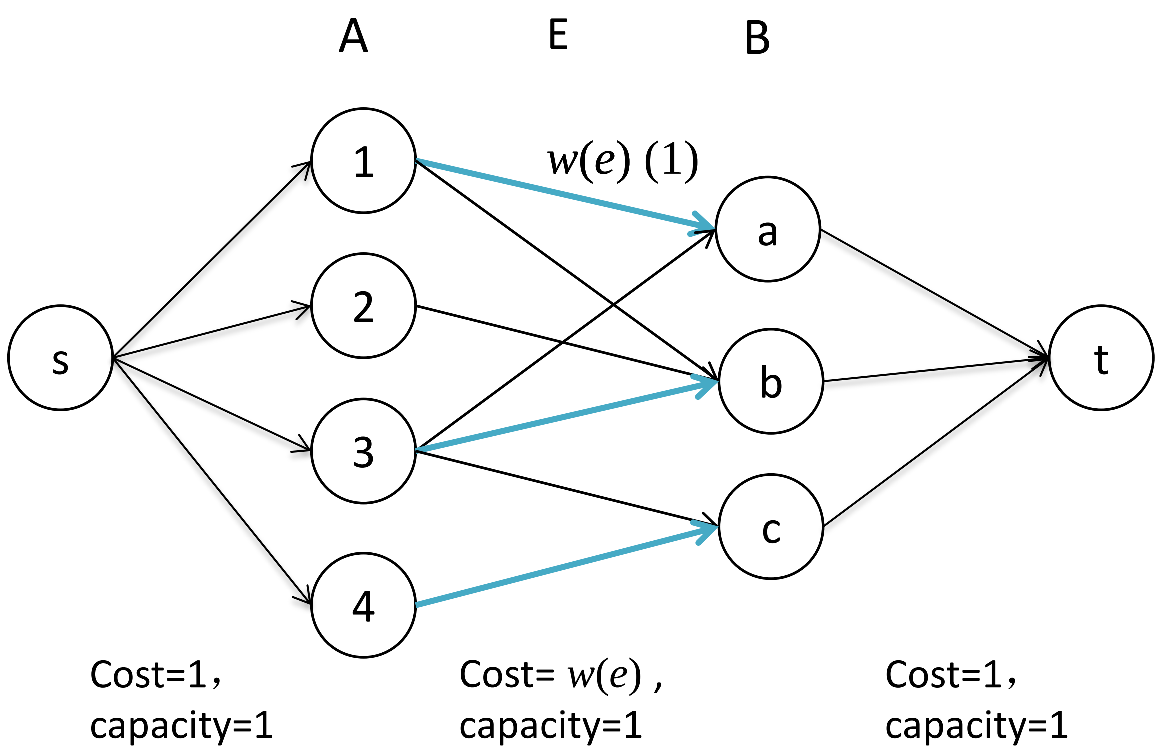
$$\begin{aligned} x_4 - x_1 - x_3 &= 0 \\ x_1 + x_2 &= 300 \\ -x_4 - x_5 &= -200 \\ x_5 - x_2 + x_3 &= -100 \end{aligned}$$

$$\begin{aligned} 0 \le x_1 &\le 200 \\ 0 \le x_2 &\le 200 \\ 0 \le x_3 &\le 300 \\ 0 \le x_4 &\le 300 \\ 0 \le x_5 &\le 100 \end{aligned}$$

最小费用流问题的应用

1. 二分图的最小权重匹配

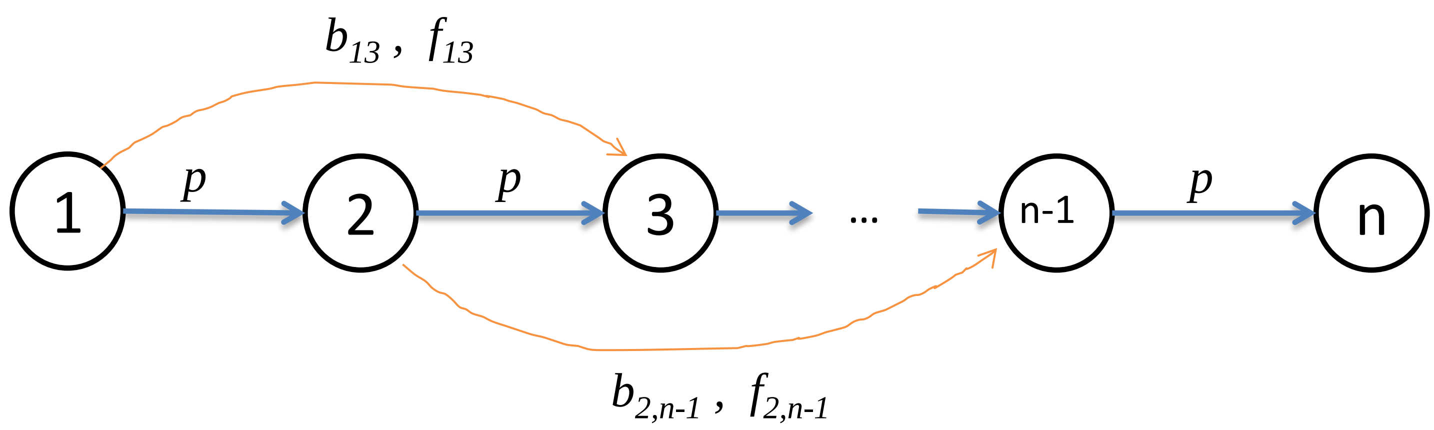
给定一个二分图 $G = (A \cup B, E)$, 命 $w: E \to R$ 为任意匹配边的权重. 找到一个该二分图的匹配方案$M \subseteq E$使得该匹配下的总权重最小。



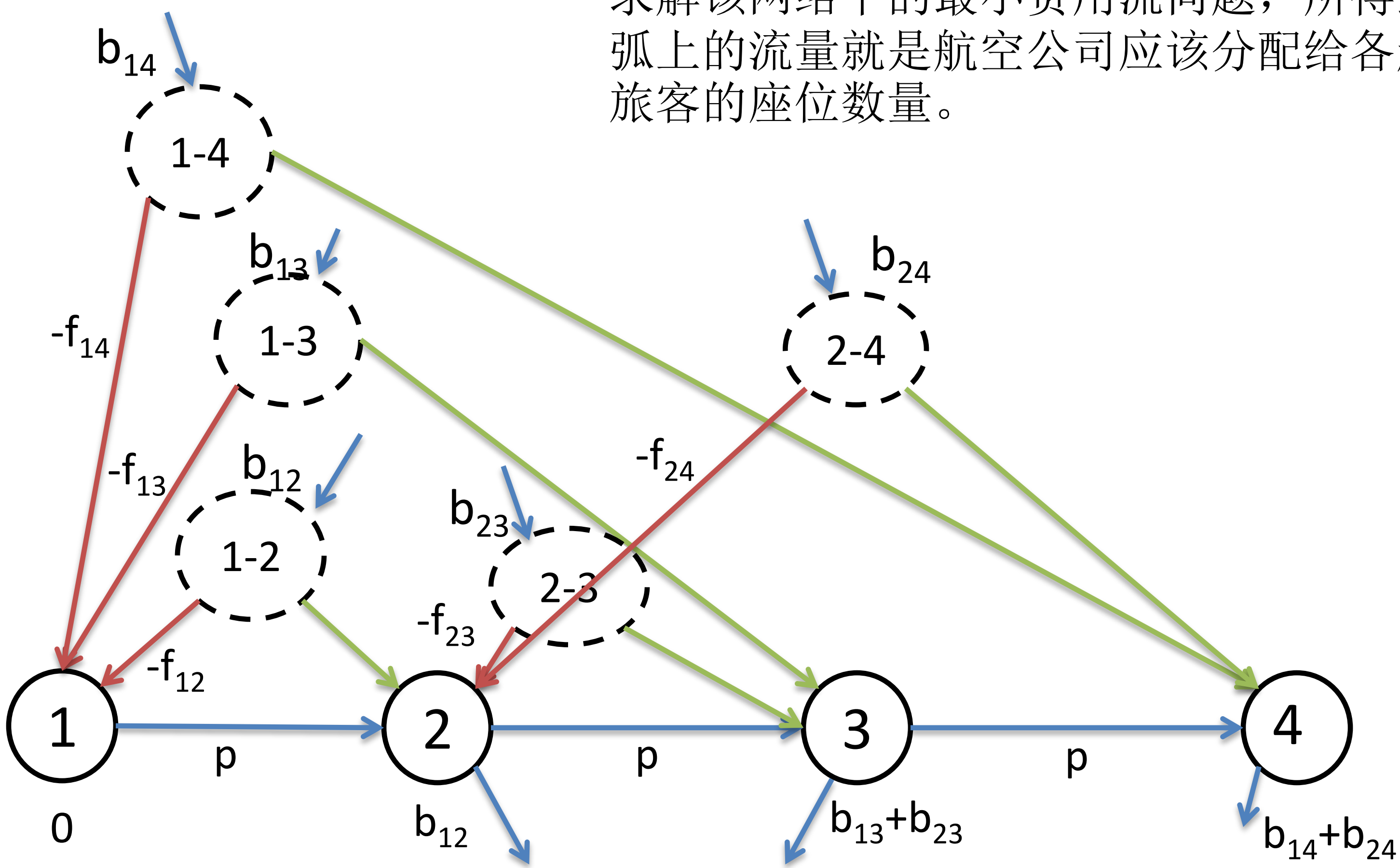
最小费用流问题的应用

2. 联程航班的最优装载量分配

考虑某航空公司的一架飞机，它以固定的顺序访问一系列的城市 1,2,3…,n，如下图。飞机的容量为p。命 b_{ij} 表示往返于i和j点的乘客数量，f_{ij} 表示该航空公司出售的i和j点的机票价格。航空公司希望确定它分配给不同起止点间旅客需求的座位数量，使得在不超过飞机容量限制的情况下最大化其收益。



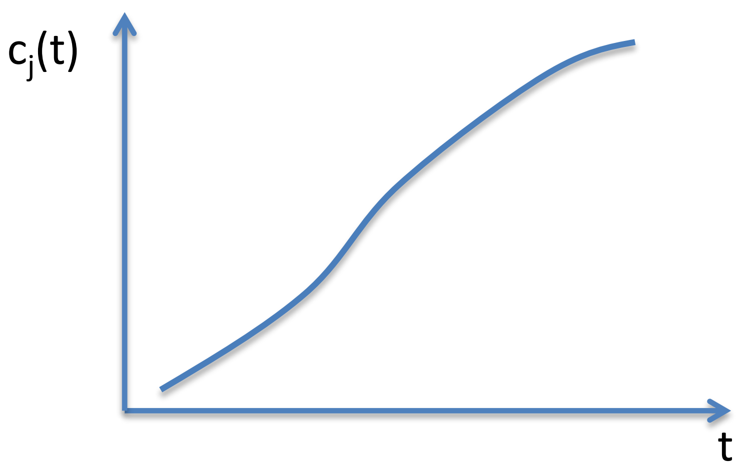
求解该网络下的最小费用流问题，所得到的红色弧上的流量就是航空公司应该分配给各起止点间旅客的座位数量。



最小费用流问题的应用

3. 考虑延迟成本的排程问题

考虑一个有q台机器的平行机排程问题，共有p个任务需要执行。每个任务的作业时间都是$\boldsymbol{\alpha}$. 每个任务的延迟成本是其完成时间的单调增函数：$c_j(t)$ $j=1,\ldots p$. 请找到这p个任务的排程方案，使得总延期成本 $\sum_{j=1}^{N} c_j(t)$ 最小.

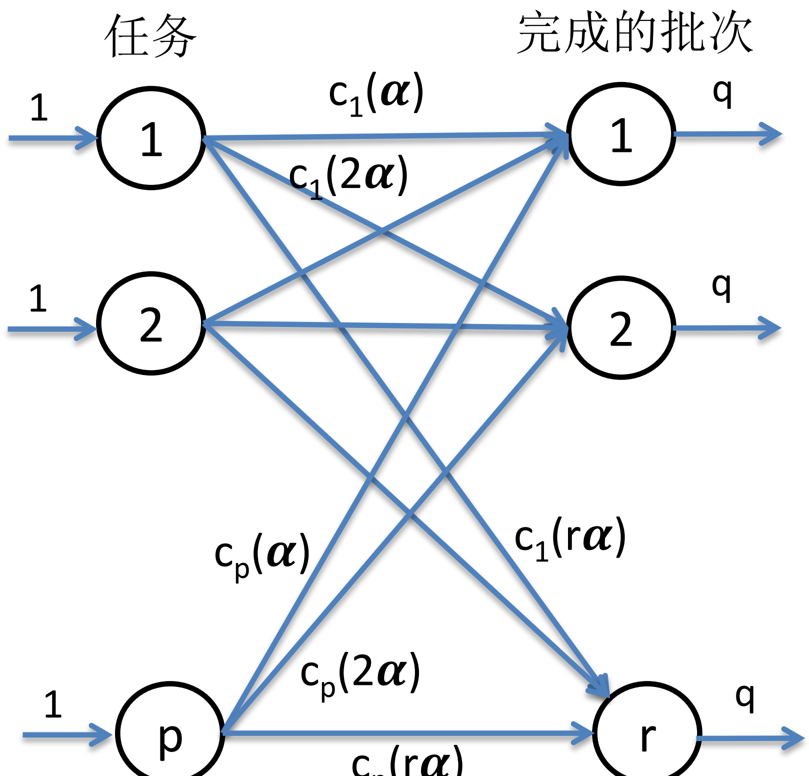


机器	0	α	2α	3α	4α
1	1	3	2	7	
2	8	4	15	13	
…					
q	6	11			

最小费用流问题的应用

3. 考虑延迟成本的排程问题

考虑一个有q台机器的平行机排程问题，共有p个任务需要执行。每个任务的作业时间都是$\boldsymbol{\alpha}$. 每个任务的延迟成本是其完成时间的单调增函数：$c_j(t)$ j=1,…p. 请找到这p个任务的排程方案，使得总延期成本 $\sum_{j=1}^{N} c_j(t)$ 最小.



$r=\lceil p/q \rceil$：每台机器的平均任务处理数

最小费用流问题的求解

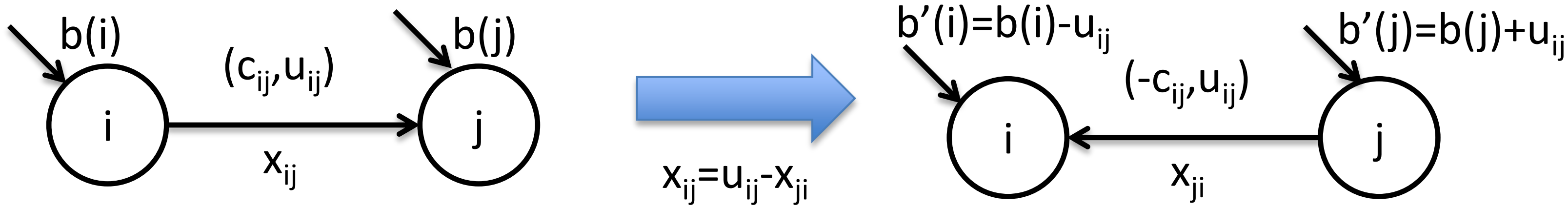
- The algorithm for min-cost network flow problem uses the ideas coming from both the shortest path problem and the maximum flow problem
- Shortest path
  - Optimality conditions: $d(i)+c_{ij} \geq d(j)$ for any (i,j)
  - Update $d(j)$ if the condition is violated
- Maximum flow
  - Working on residual network

Basic Assumptions

- All data are integral
  - Cost, supply/demand, capacity
- The problem is feasible
- There is a direct path between any pair of nodes
  - If not, we can add a dummy arc with a very large cost
- All arc costs are nonnegative
- All lower bounds are zero
  - Non-zero lower bound can be removed (c.f. the maximum flow problem with positive lower bounds)

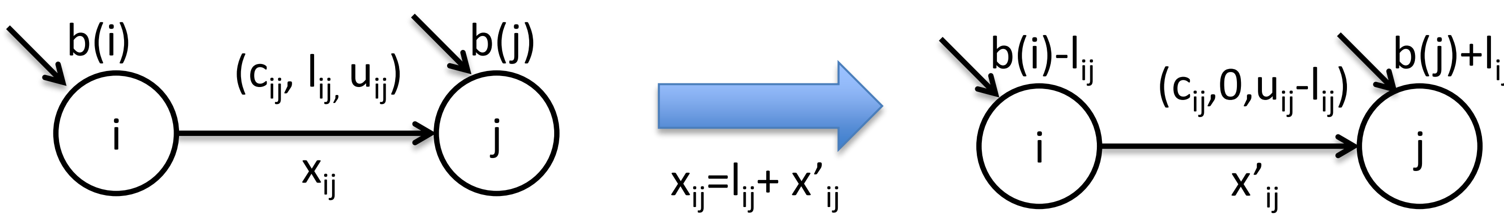
Network transformations

- **Arc reversal:** typically used to remove arcs with negative costs



Send u_{ij} units of flow on the arc (which decreases b(i) by u_{ij} and increases b(j) by u_{ij} and then replace arc (i,j) by arc (j,i) with cost $-c_{ij}$. The new flow x_{ji} measures the amount of flow we 'remove' from the 'full capacity' flow of u_{ij} .

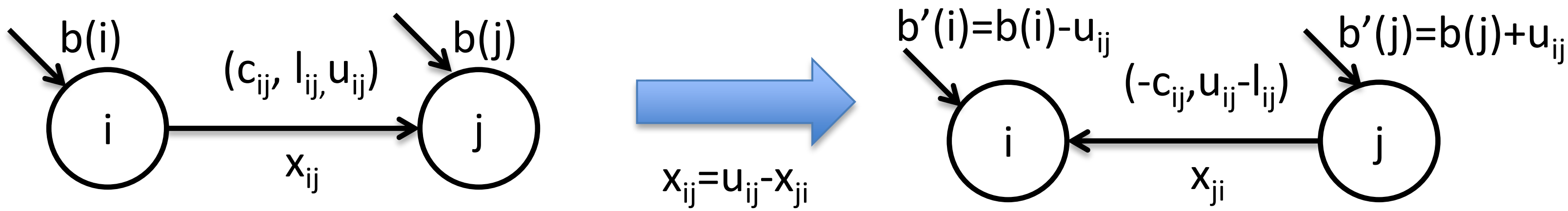
- **Removing non-zero lower bounds** l_{ij}



Send l_{ij} units of flow on the arc (which decreases b(i) by u_{ij} and increases b(j) by u_{ij} and measures the incremental flow x'_{ij} on the arc beyond the flow value l_{ij} .

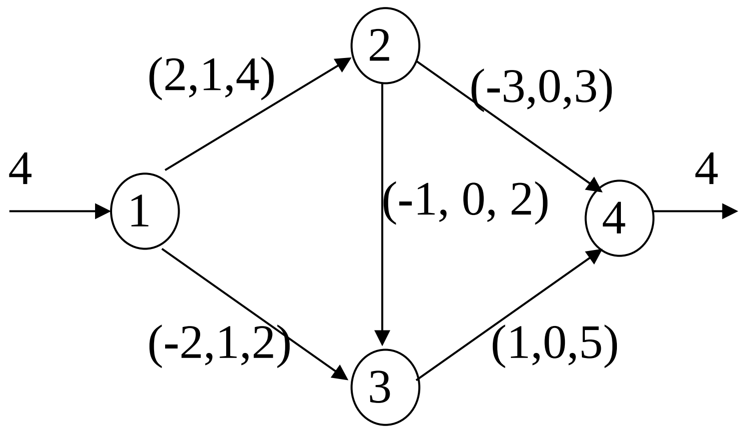
Network transformations

- **How to transform arcs with both non-zero lower bounds and negative costs?**



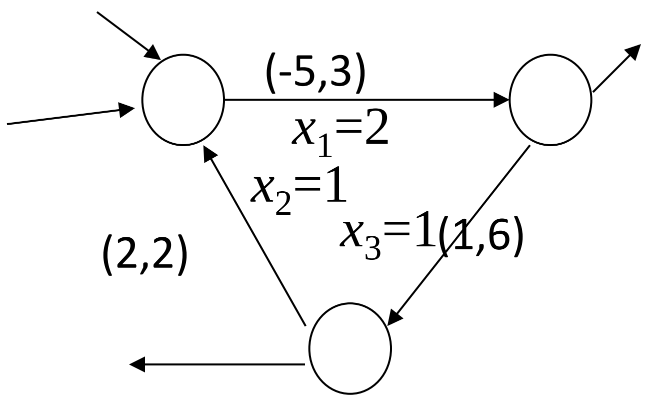
Example

- Transform the following network to a one with no negative arc and no non-zero lower bound

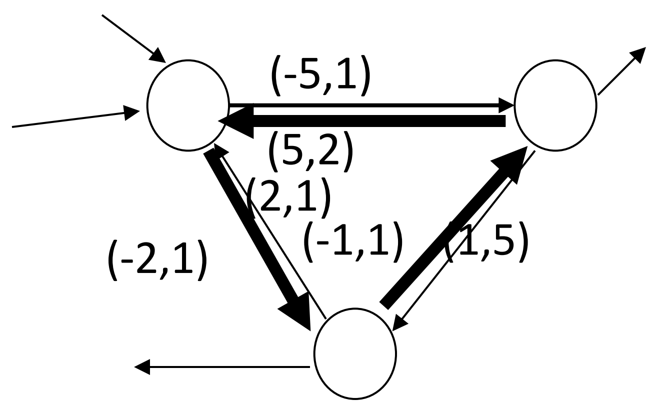


Residual Network for a Feasible Flow

- For a given flow x on network G, we can define a residual network $G(x)$
  - For x_{ij} on each arc (i,j) with (c_{ij},u_{ij}) in G, we have two arcs in $G(x)$:
    - (i,j) with $(c_{ij}, u_{ij}-x_{ij})$: reduce capacity on the forward arc
    - (j,i) with $(-c_{ij}, x_{ij})$: increase capacity on the backward arc which is with a negative cost



A network with a flow x



The residual network $G(x)$

Algorithms

- Cycle canceling algorithm

- Successive shortest path algorithm

- Primal-Dual algorithm

Cycle canceling algorithm

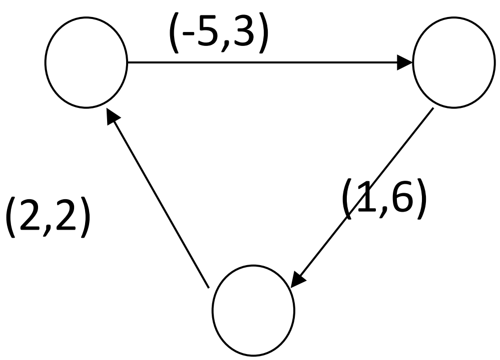
最小费用流问题的最优性条件1

Negative cycle optimality conditions

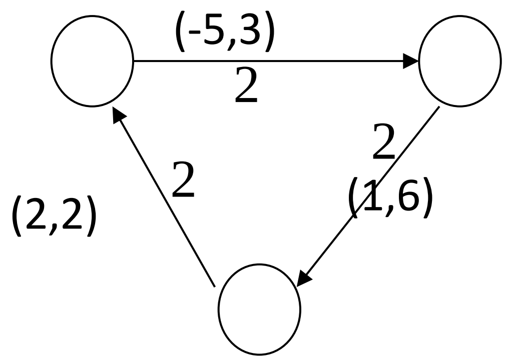
A feasible solution *x* is optimal if and only if the residual network *G*(*x*) contains no negative cost directed cycles.

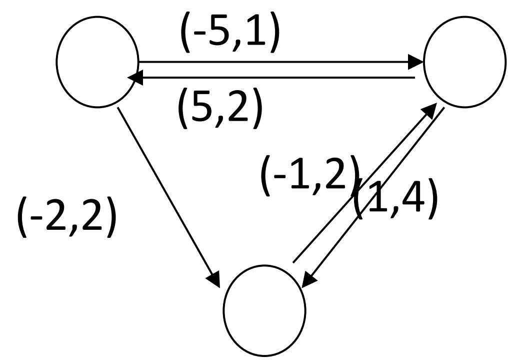
Increase 2 units of flow along the negative cycle

Cost change is (-5+2+1)*2 = -4 < 0



A residual network with a negative cycle





Residual network after flow increase: The negative cycle is cancelled

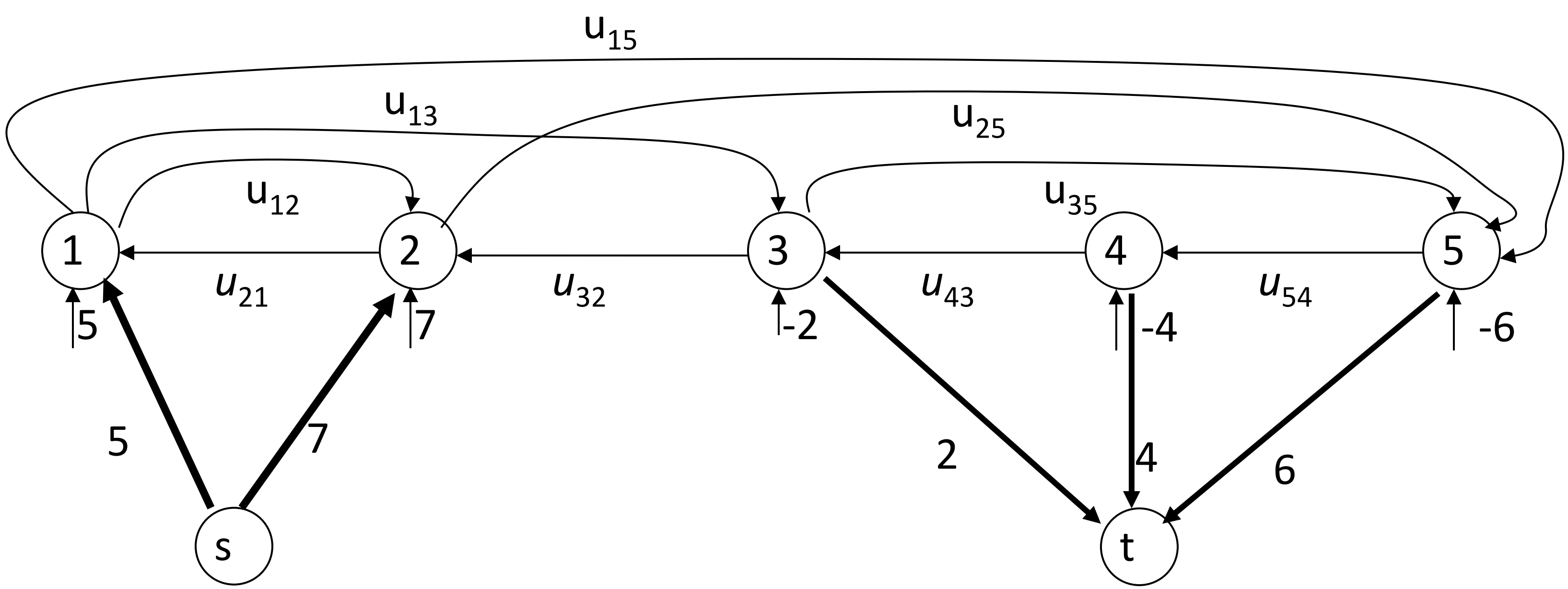
Cycle Canceling Algorithm

- Find a feasible flow x
- **While** $G(x)$ contains a negative cycle W **do**
  - **Begin**
    - Let δ=min{ r_{ij} | (i,j) on W}
    - Augment δ units of flow along W and update $G(x)$
  - **End**

How to find an initial feasible flow?

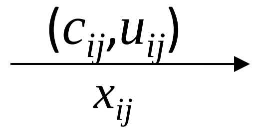
How to detect a negative cycle?

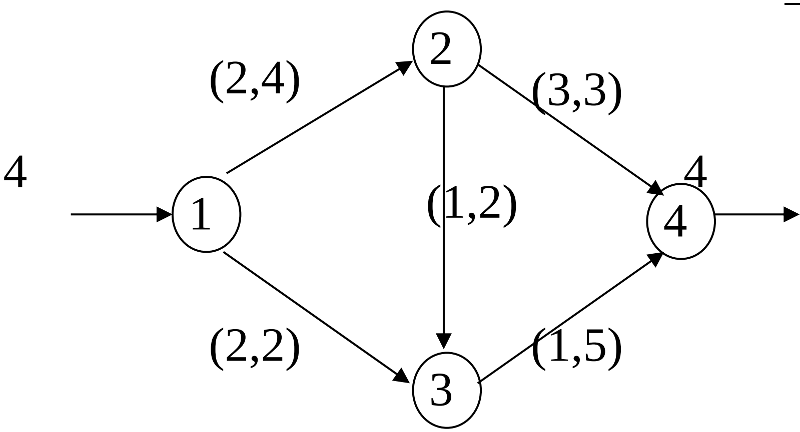
Initial Feasible Solution



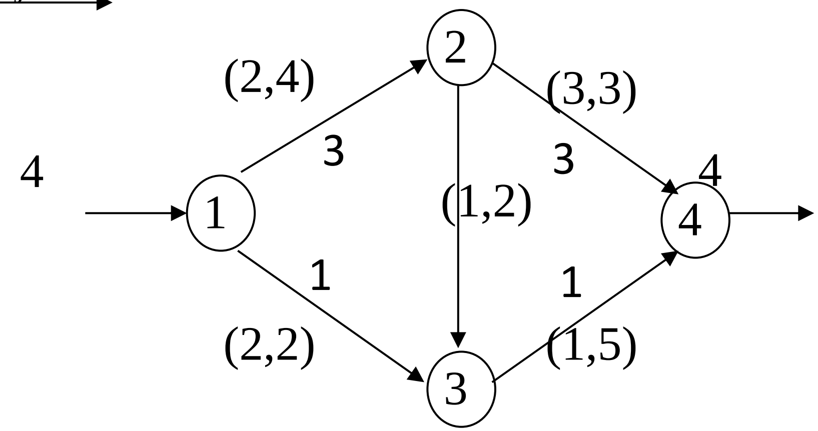
- By solving a maximum flow problem
  - A virtual source node *s* with arcs to each node with a supply
    - Let the arc capacity be the supply quantity
  - A virtual destination node *t* with arcs from each node with a demand (negative supply)
    - Let the arc capacity be the demand quantity
  - Is the maximum flow from *s* to *t* equal to 12 (why 12)?

Examples for Cycle Canceling

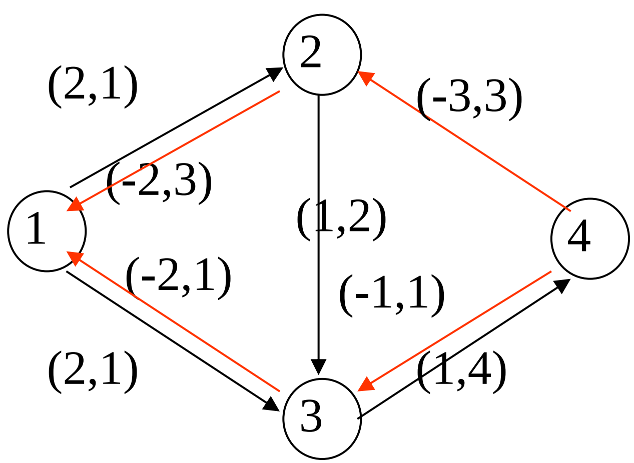




Original network



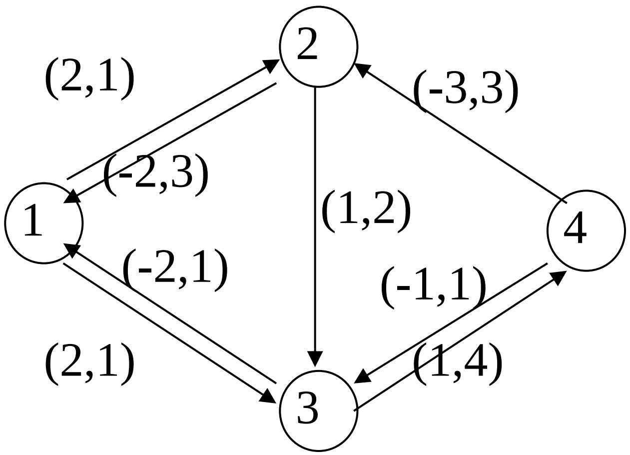
A feasible solution on the original network

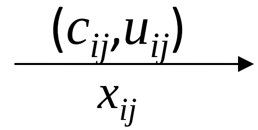


Residual network for the feasible solution

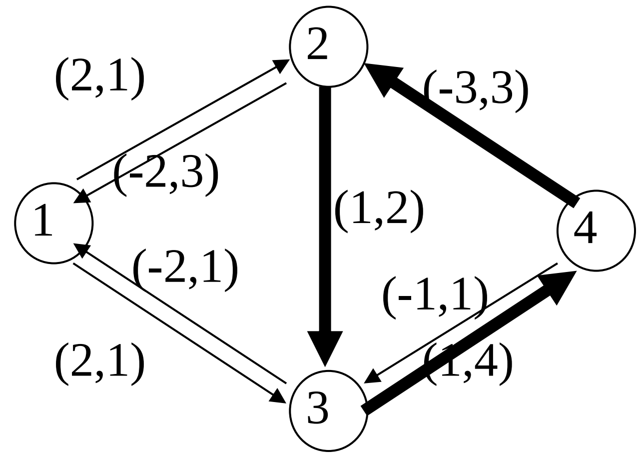
Consider: Given G and $G(x)$, how can we know what is x?

Examples for Cycle Canceling



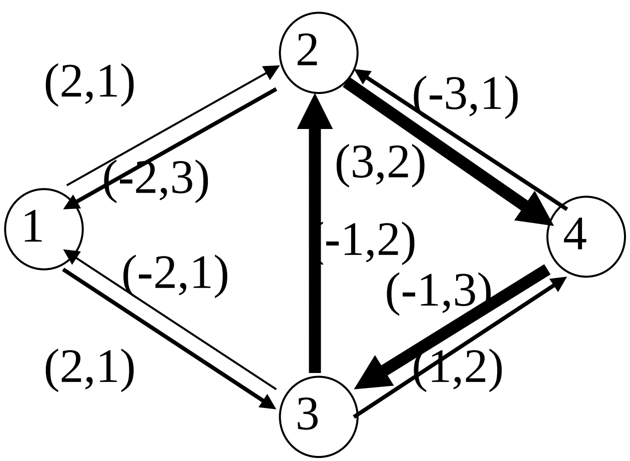


Residual network for the feasible solution



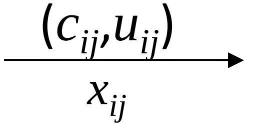
A negative cycle on the residual network

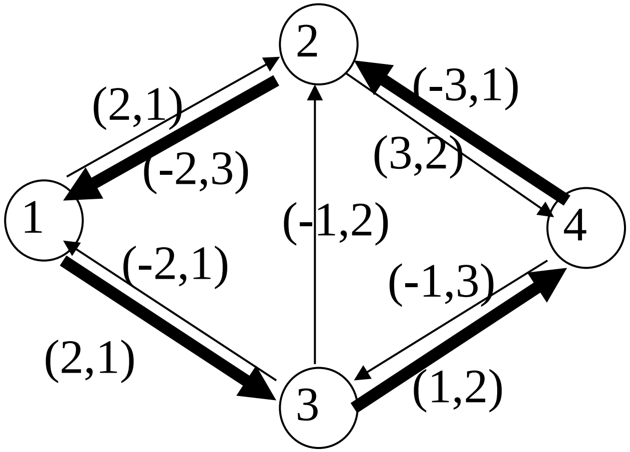
The flow to be sent is min{3,2,4} = 2



Residual network after sending 2 units of flow

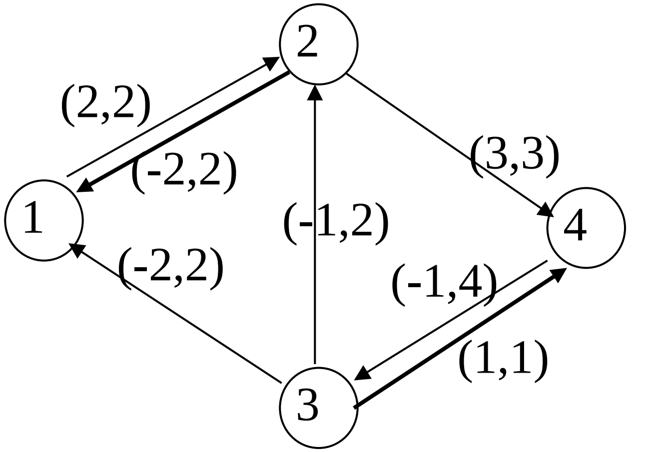
Examples for Cycle Canceling





Another negative cycle on residual network

The capacity for the negative cycle is 1

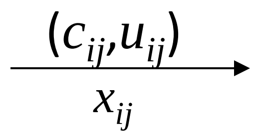


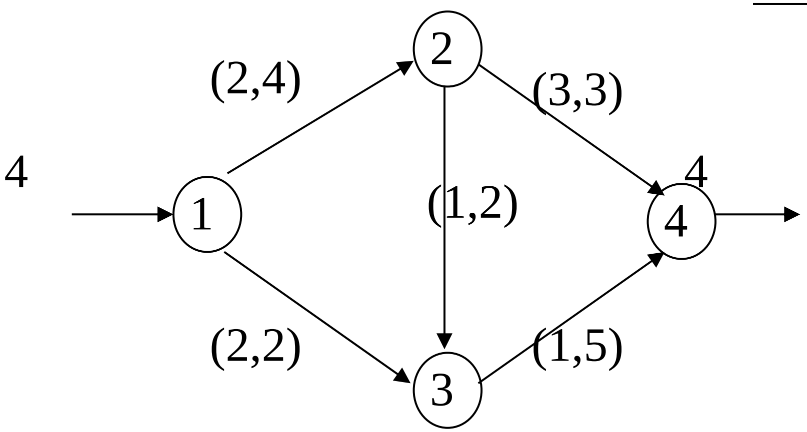
New residual network after sending 1 unit of flow along the negative cycle

No other negative cycles

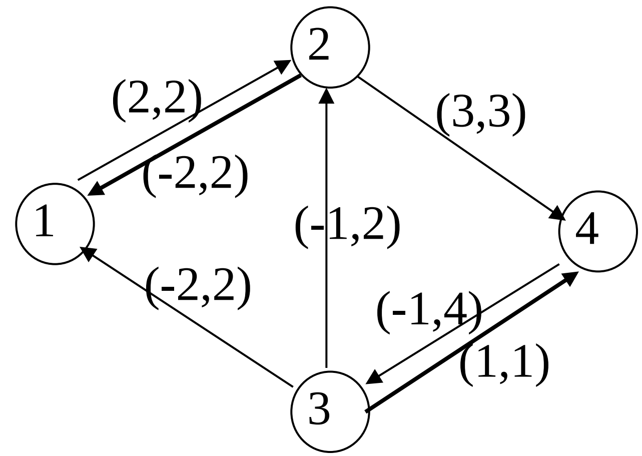
The solution cannot be improved

Examples for Cycle Canceling

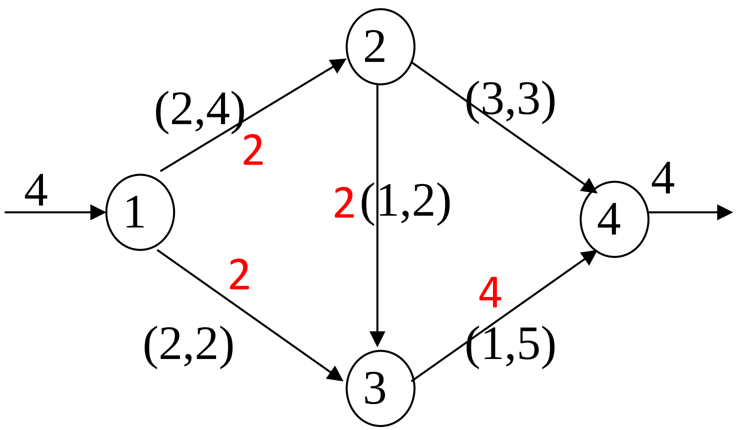




Original network



Final residual network

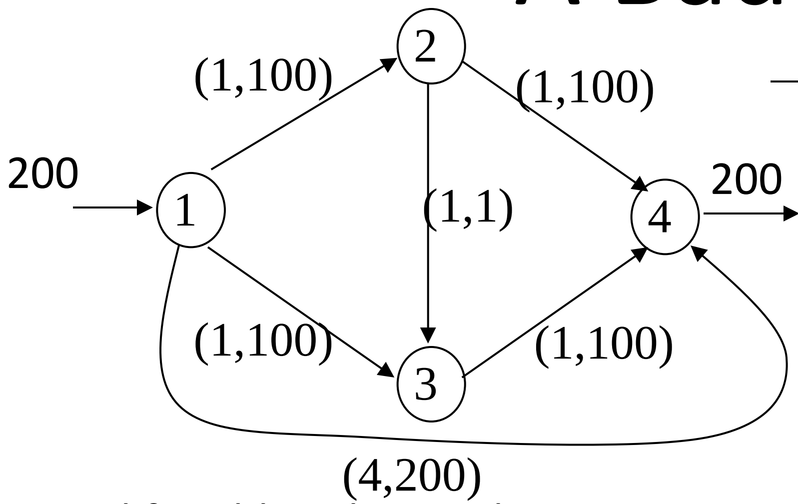


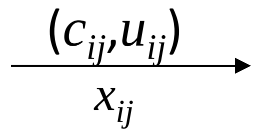
Optimal solution

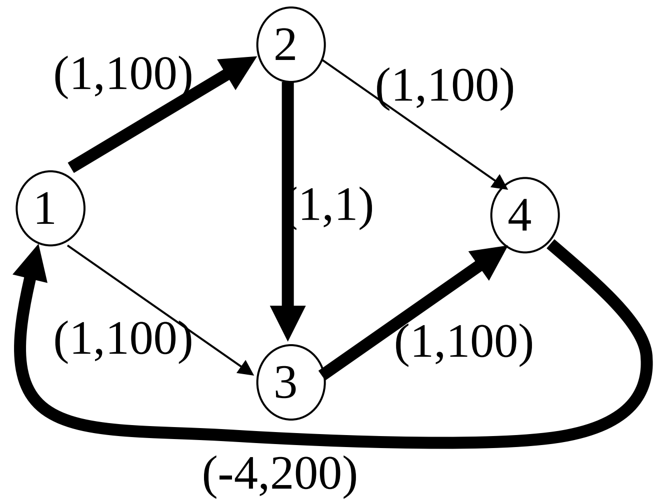
Cycle Canceling Algorithm: Complexity

- Number of iterations is in $O(mCU)$
  - C: maximum absolute value of the cost
  - U: maximum capacity
  - mCU is the maximum possible cost for a feasible solution
  - In each iteration the cost is reduced by at least one
- Within each iteration, we need to find a negative cycle, which is in $O(nm)$
  - For example, the FIFO label correcting algorithm
- Overall time complexity is $O(nm^2CU)$
  - Pseudo polynomial time

A Bad Example

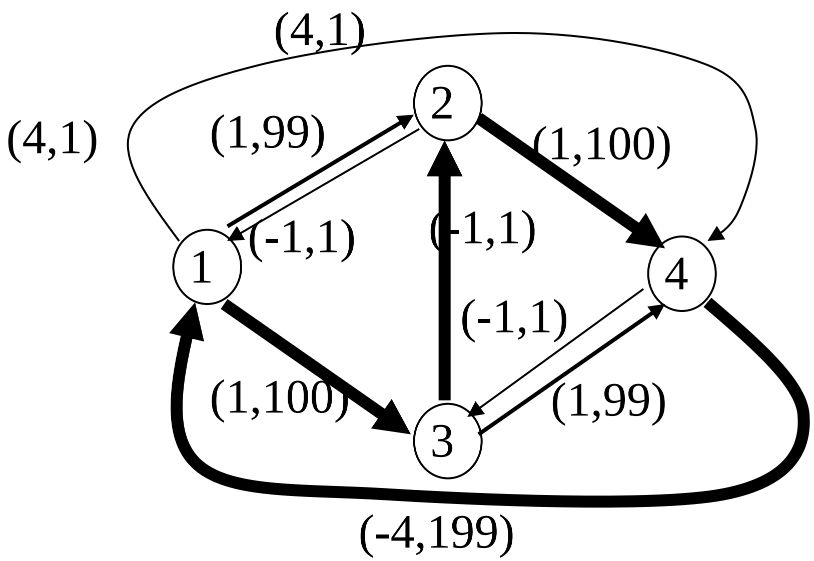


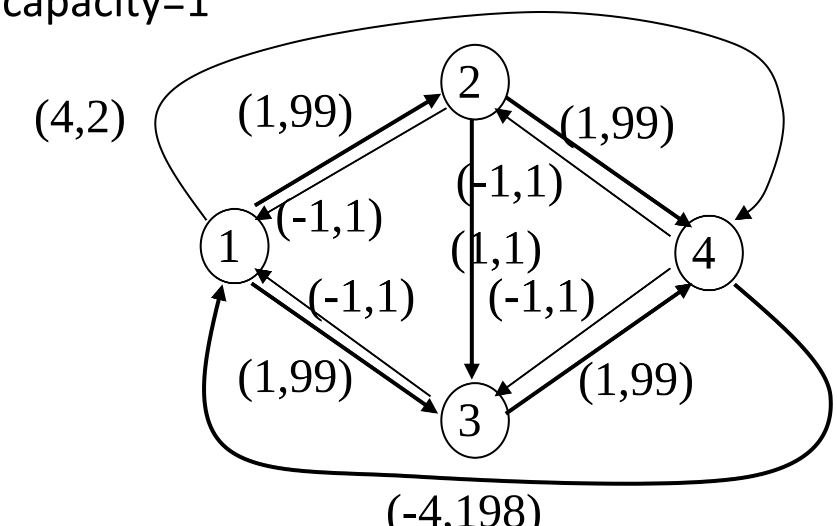




Initial feasible solution chooses sending 200 units flow on arc (1,4)

Residual network & a negative cycle with capacity=1



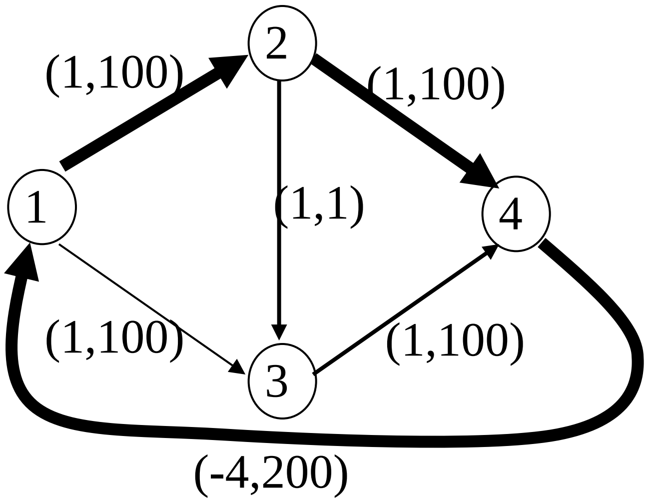


New Residual network & negative cycle

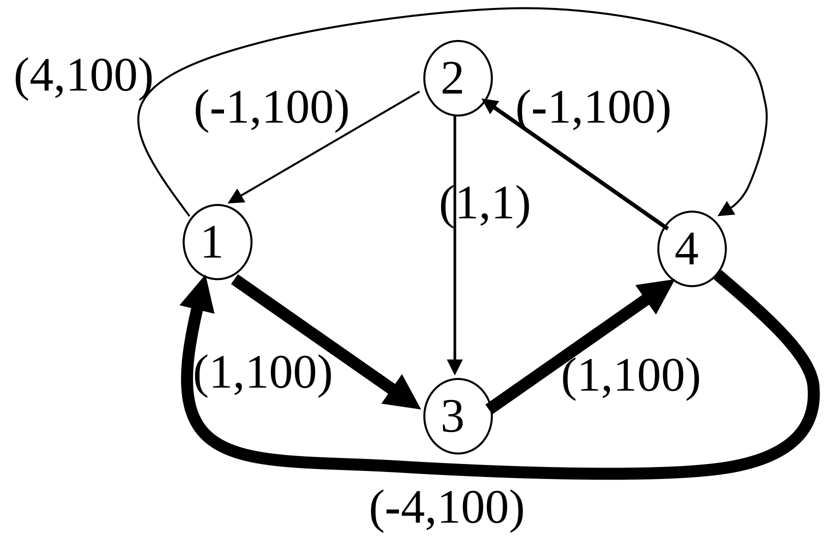
After two iterations

A Better Case Example

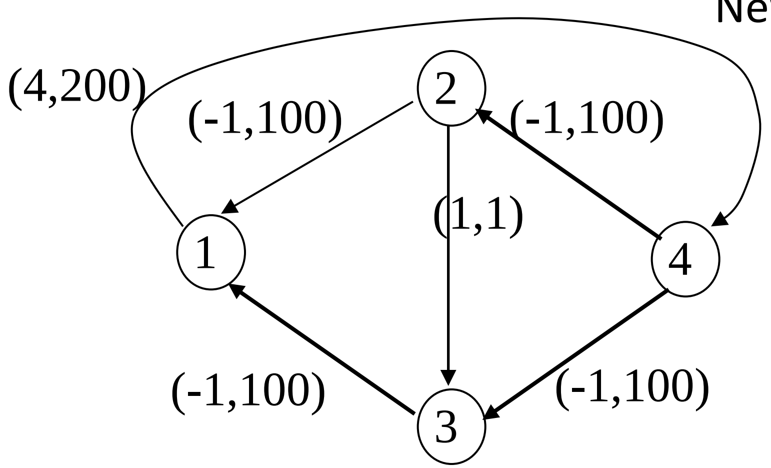
If we repeat the above process, it will take 200 iterations to find the optimal solution. If we choose a different negative cycle, two iterations are enough.



A negative cycle with capacity=100



New Residual network & negative cycle



Final Residual network

Do we have better algorithms?