Computing Market Equilibria
Part I — Basics

Lars Engebretsen
Royal Institute of Technology
SWEDEN

Dagstuhl Seminar, 30 August – 3 September 2004

Scenario

A market with set A of agents and set G of goods.

Agent a has

- Amount m_a of money.
- Utility function $u^{(a)}\colon \mathbf{R}^{|G|} \to \mathbf{R}$.

Objective: Buy goods that maximize utilities.

Developed by Walras (1874) and, independently, Fisher (1891).

Mathematical formulation

Suppose that there is a market where goods are sold.

Let $\boldsymbol{p}$ be market prices, i.e., p_g is price of good g.

Agent a wants to

$$\text{maximize } u^{(a)}(\boldsymbol{x}^{(a)}) \text{ given that } \boldsymbol{p} \cdot \boldsymbol{x}^{(a)} \leq m_a.$$

When everybody is doing this, what happens?

Should depend on $\boldsymbol{p}$...

Market clearing prices (Arrow-Debreu theorem)

For concave utility functions, there exists a *price equilibrium*.

For the equilibrium price vector $\hat{\boldsymbol{p}}$, solutions to

$$\text{maximize } u^{(a)}(\boldsymbol{x}^{(a)}) \text{ given that } \hat{\boldsymbol{p}} \cdot \boldsymbol{x}^{(a)} \leq m_a.$$

clear the market exactly. (No surplus, no deficit.)

(Price equilibrium may not exist for indivisible goods.)

Linear markets

This talk considers linear utility functions.

All agents' utilities described by a matrix (u_{ag}).

$u^{(a)}(\boldsymbol{x}) = \sum_{g \in G} u_{ag} x_g$ where x_g is gth coordinate of $\boldsymbol{x}$.

W.l.o.g.: One unit of each good; integral amounts of money.

Optimal strategy in linear markets

For given market prices $\boldsymbol{p}$, there is optimal strategy.

Agent a buys goods g that maximize u_{ag}/p_g.

An agent with several choices may (and will) buy several goods.

So, the prices seem to be the crux.

How do we *find* the market clearing prices?

First answer: It is enough to know roughly what they are.

Second answer: We can find them in finite time.

Third answer: We can find them in polynomial time.

Preliminaries

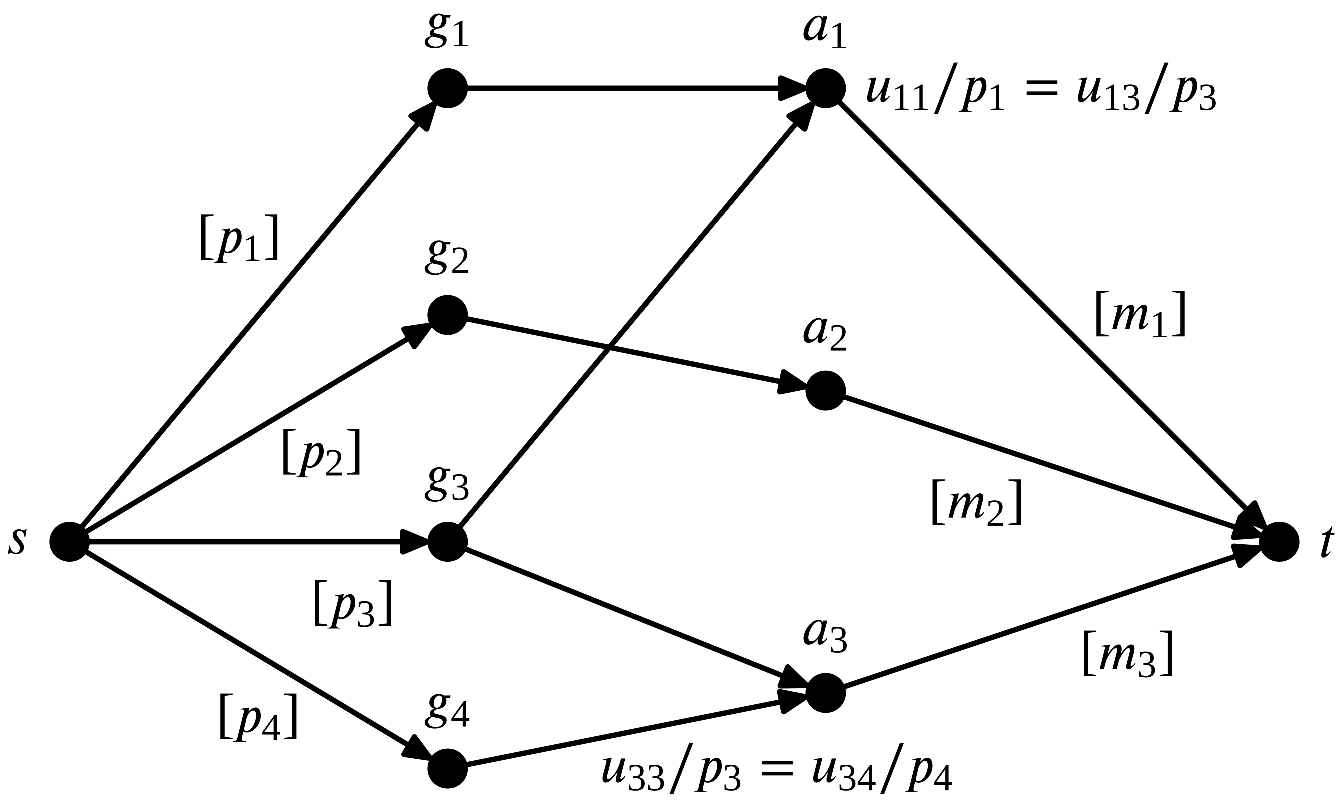
Recall: Given p_g, agent a buys goods with maximum u_{ag}/p_g.

One unit of each good to split between agents.

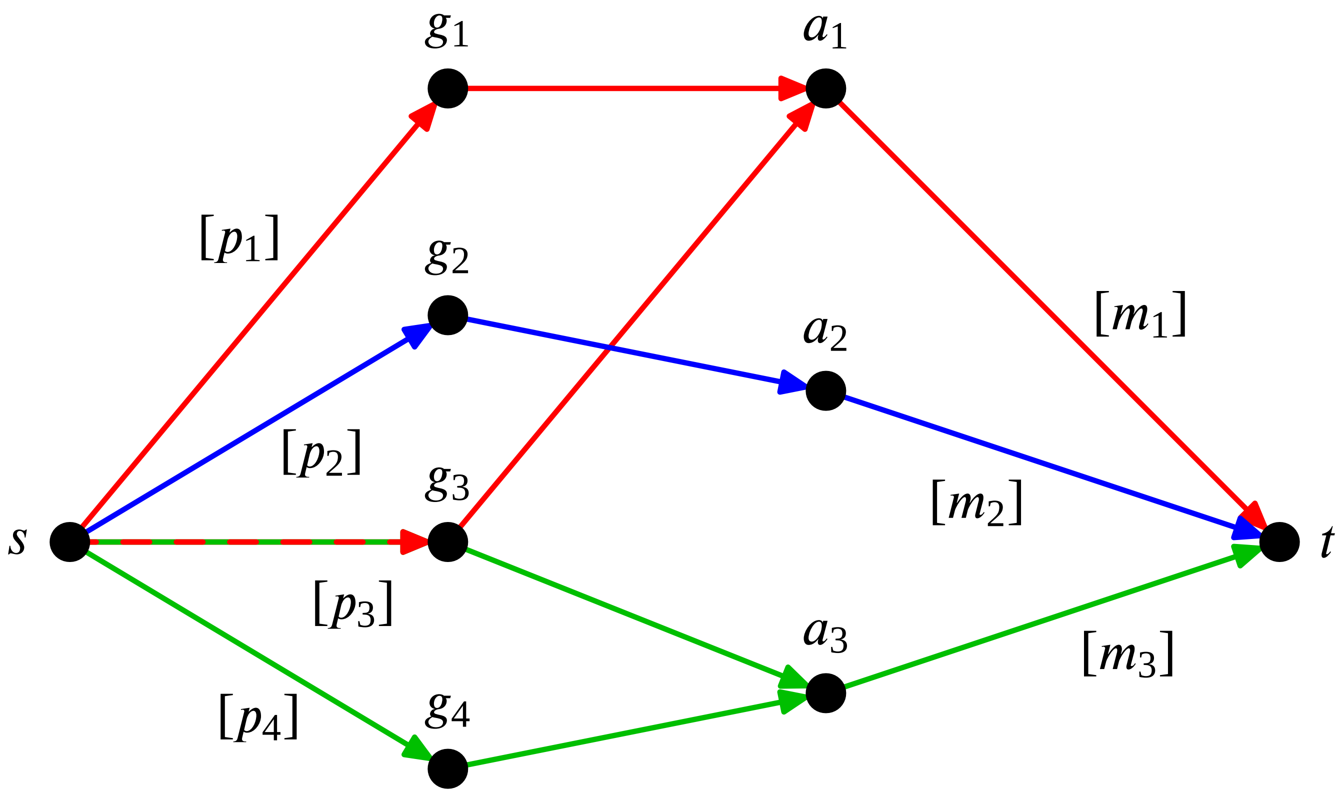
Construct bipartite network that models this fact.

Flow computation gives equilibrium prices.

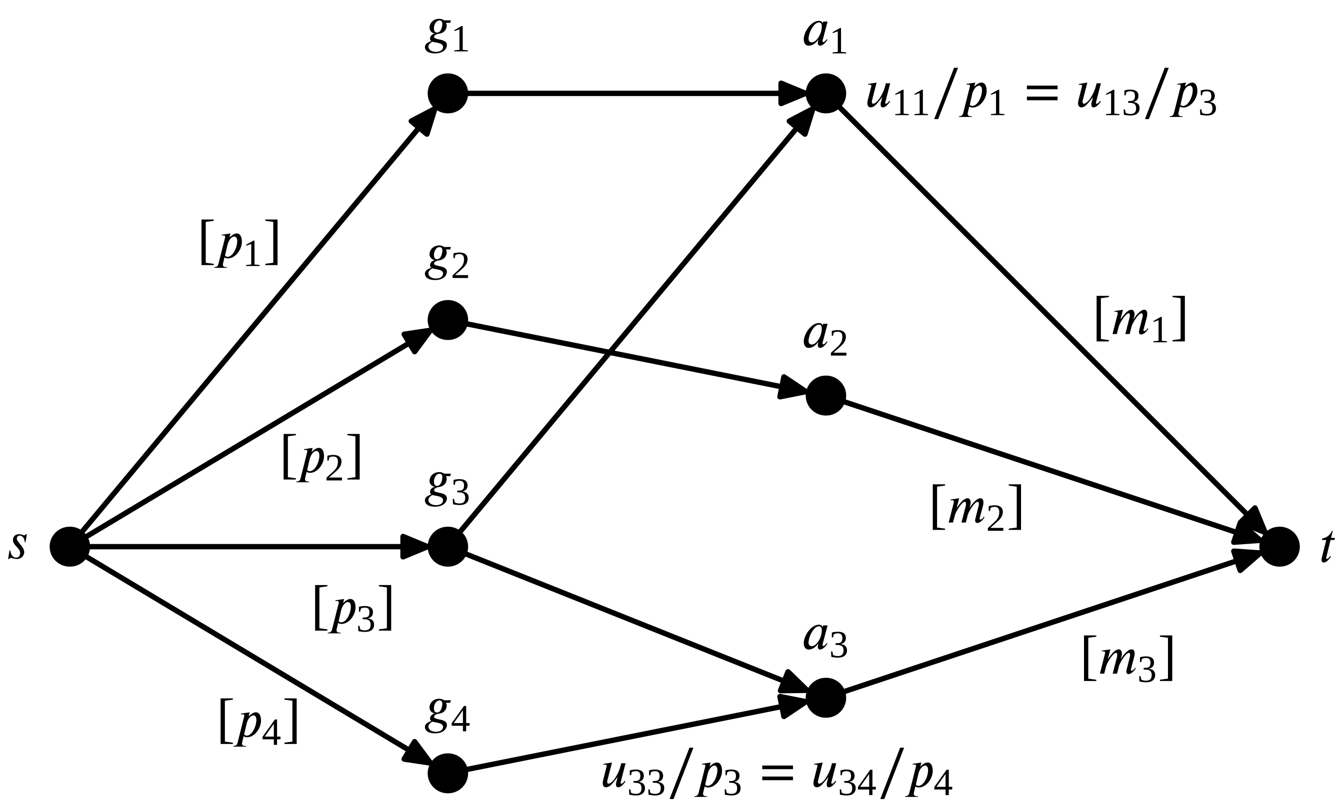
Construction of the bipartite network



Flow in the bipartite network



Flow conservation gives equilibrium



Computing the prices

Ansatz: Very low prices.

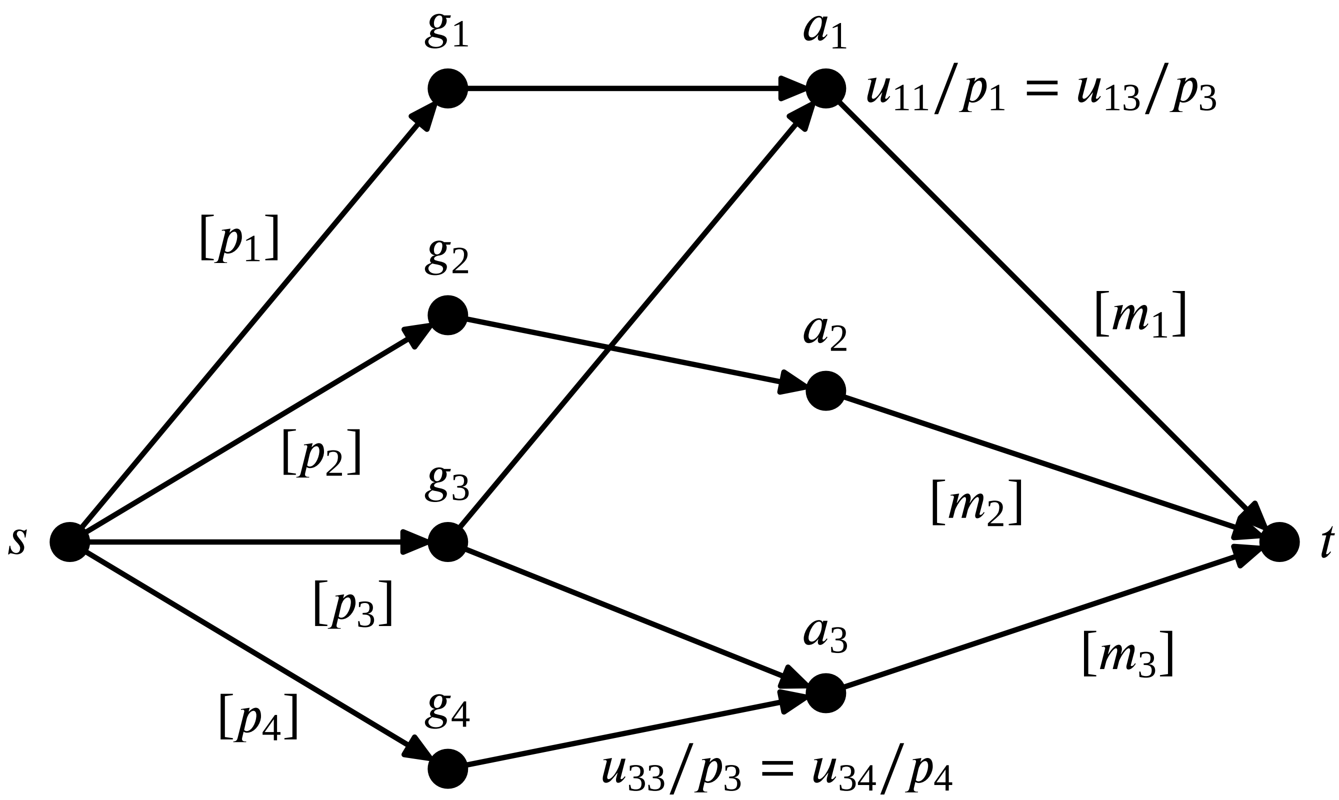
Compute flow in induced bipartite network.

All goods are sold, some money left.

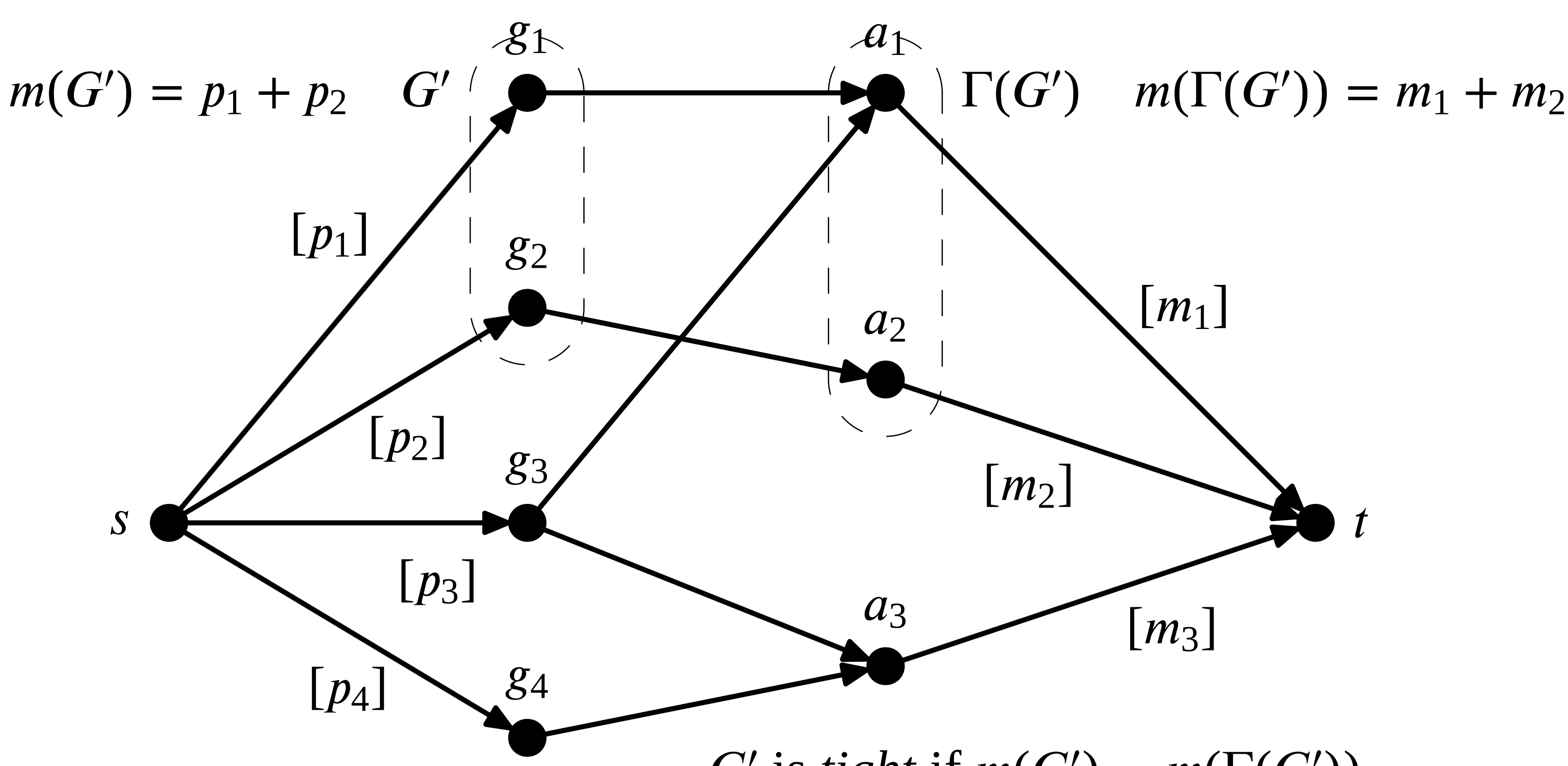
Gradually increase prices until market clears.

Complication: As prices increase, the network changes!

The bipartite network again



Subsets, neighbours and money



G' is *tight* if $m(G') = m(\Gamma(G'))$

Increasing the prices

Increase prices uniformly, scale by some factor.

This does not change the network!

Maintain invariant: For all $G' \subseteq G$: $m(G') \leq m(\Gamma(G'))$.

Recall: G' tight if $m(G') = m(\Gamma(G'))$.

At some point, some set $\hat{G} \subseteq G$ becomes tight.

Finding a tight set

Task: Find x such that prices $x\boldsymbol{p}$ gives first tight set.

Try $x = m(A)/m(G)$. At least not too small...

Correct value $\hat{x}$ satisfies $\hat{x}m(\hat{G}) = m(\Gamma(\hat{G}))$.

If we are lucky, $x = m(A)/m(G)$ is correct!

Then entire G is tight.

What if $x = m(A)/m(G)$ is bad choice?

Assume that $x = m(A)/m(G)$ is too large.

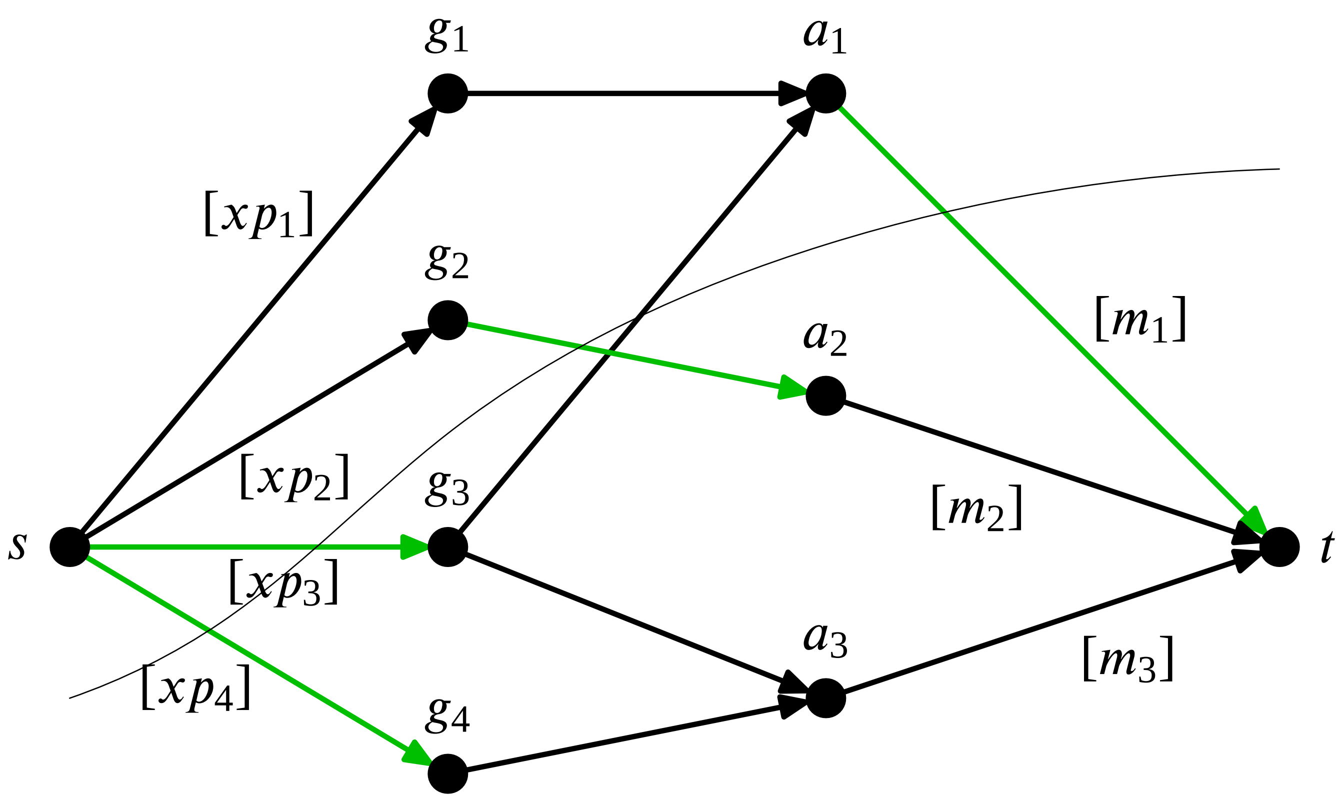
Consider minimum cut in network with prices $x\boldsymbol{p}$.

Cut is between $\{s\} \cup G_S \cup A_S$ and $G_T \cup A_T \cup \{t\}$.

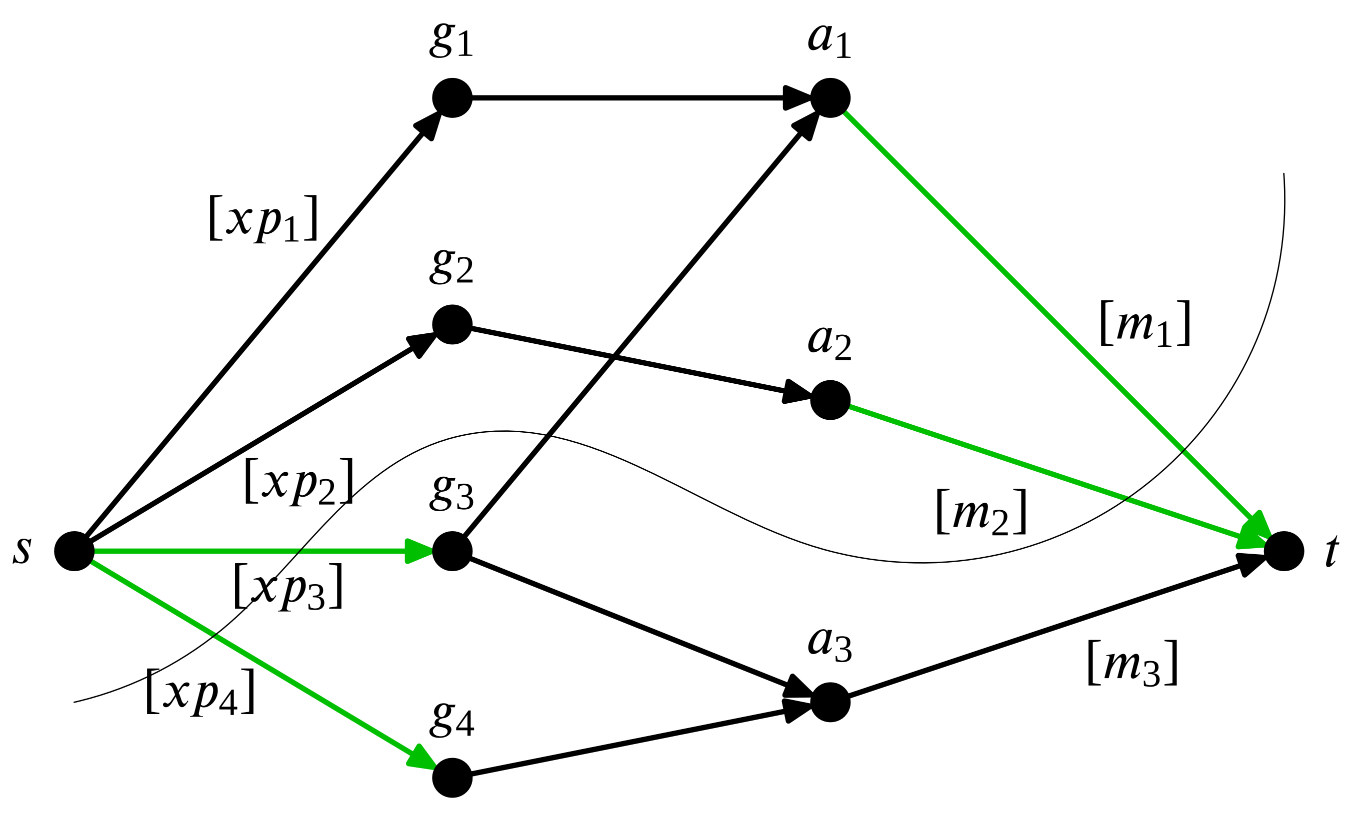
Let $\hat{G}$ be tight set; $\quad \hat{x}m(\hat{G}) = m(\Gamma(\hat{G}))$; $\quad \hat{x} < x$.

Zoom in on $\hat{G}$.

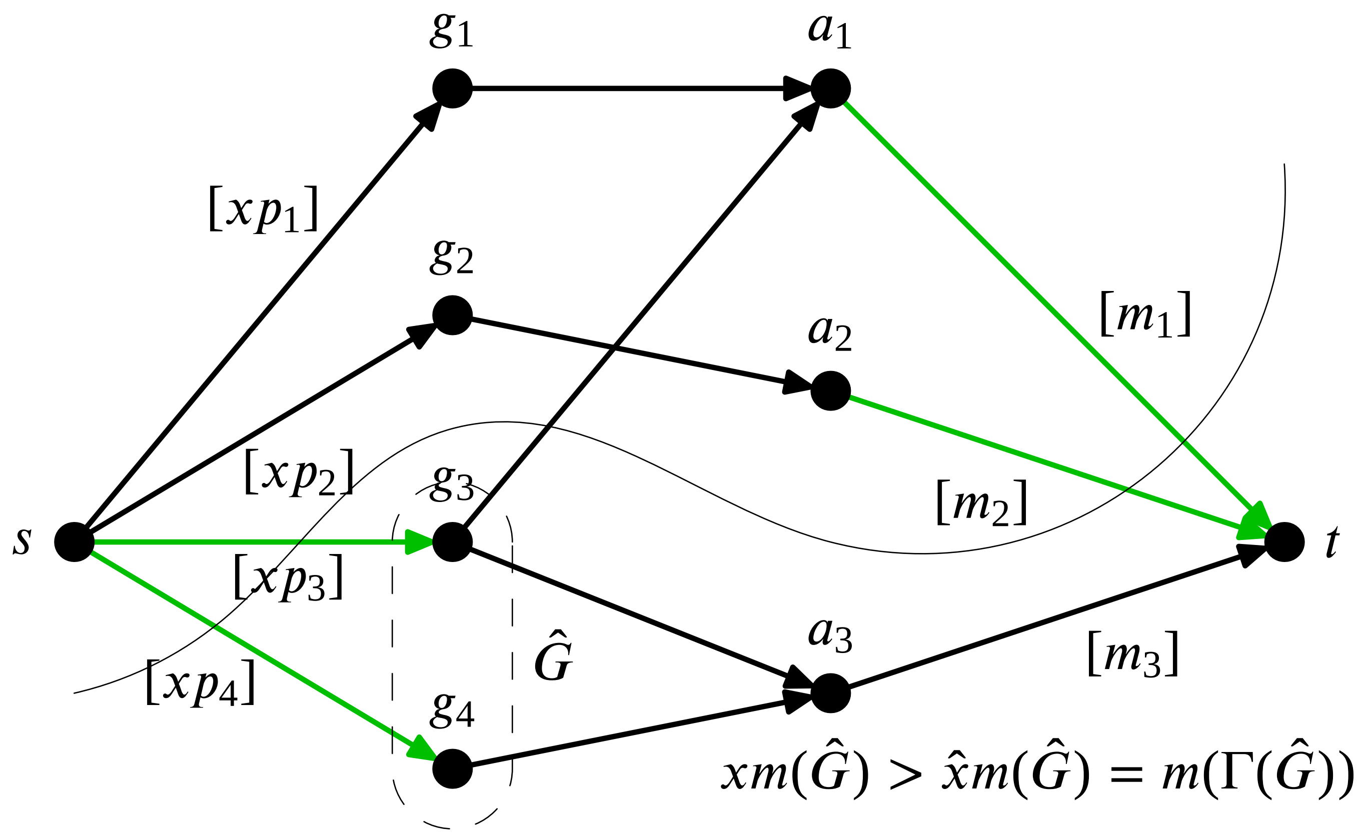
Possible minimum cuts



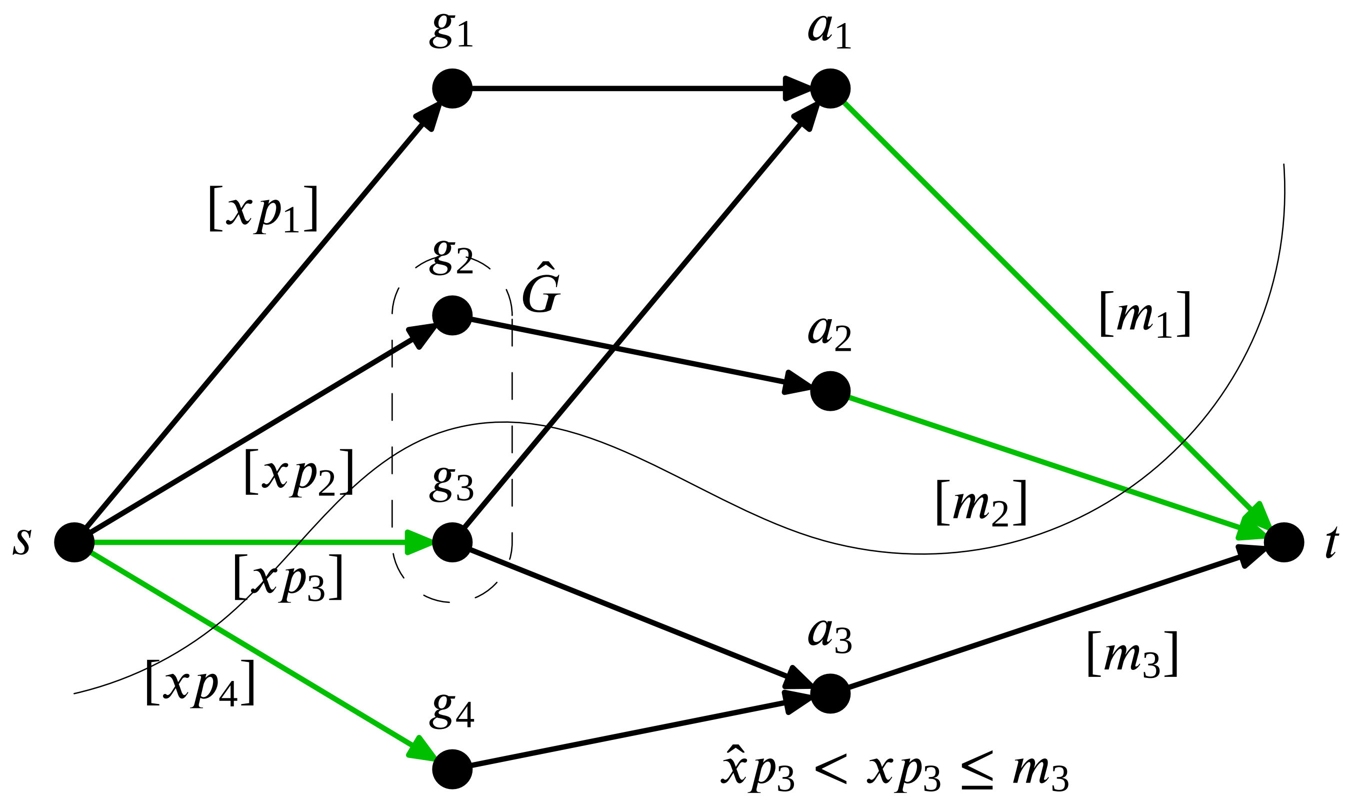
Where is tight set located?



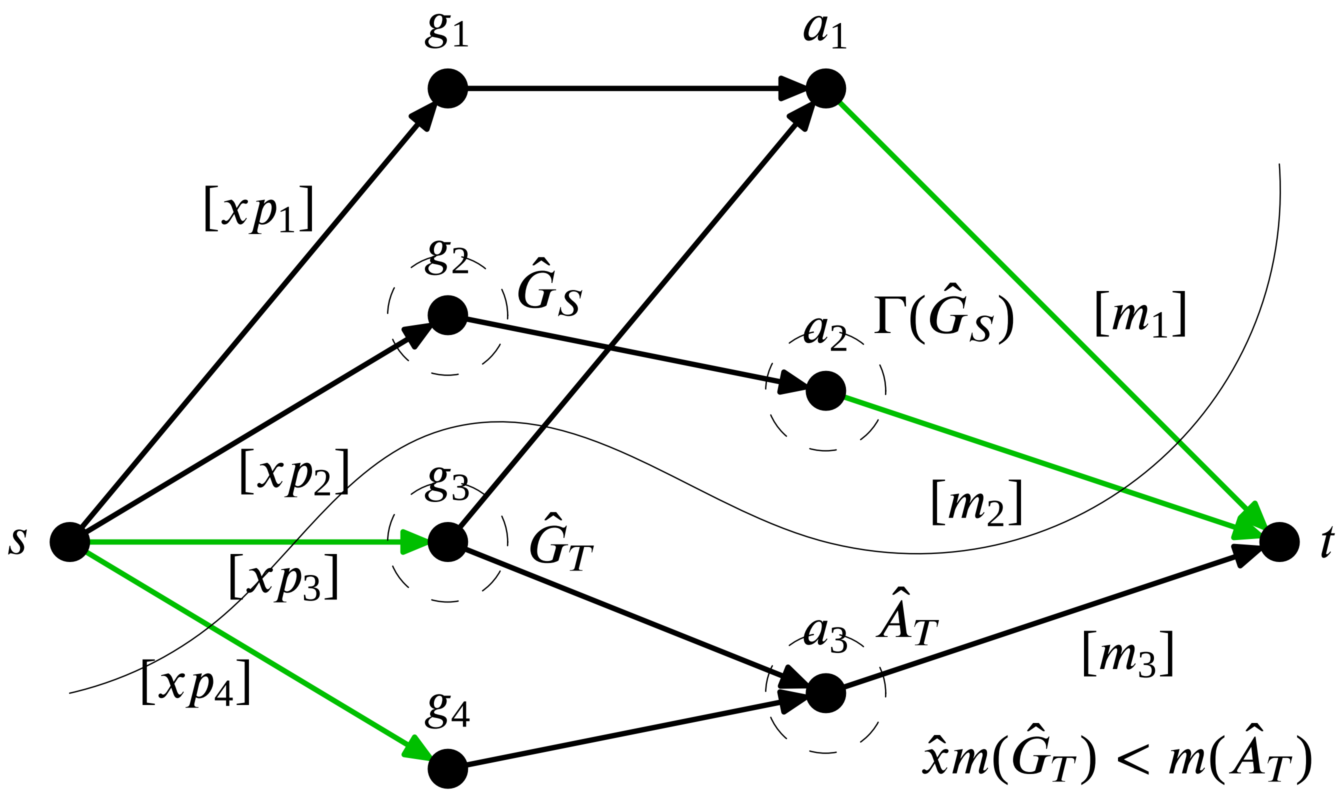
Cannot get rid of incoming flow



Other properties of min-cut…

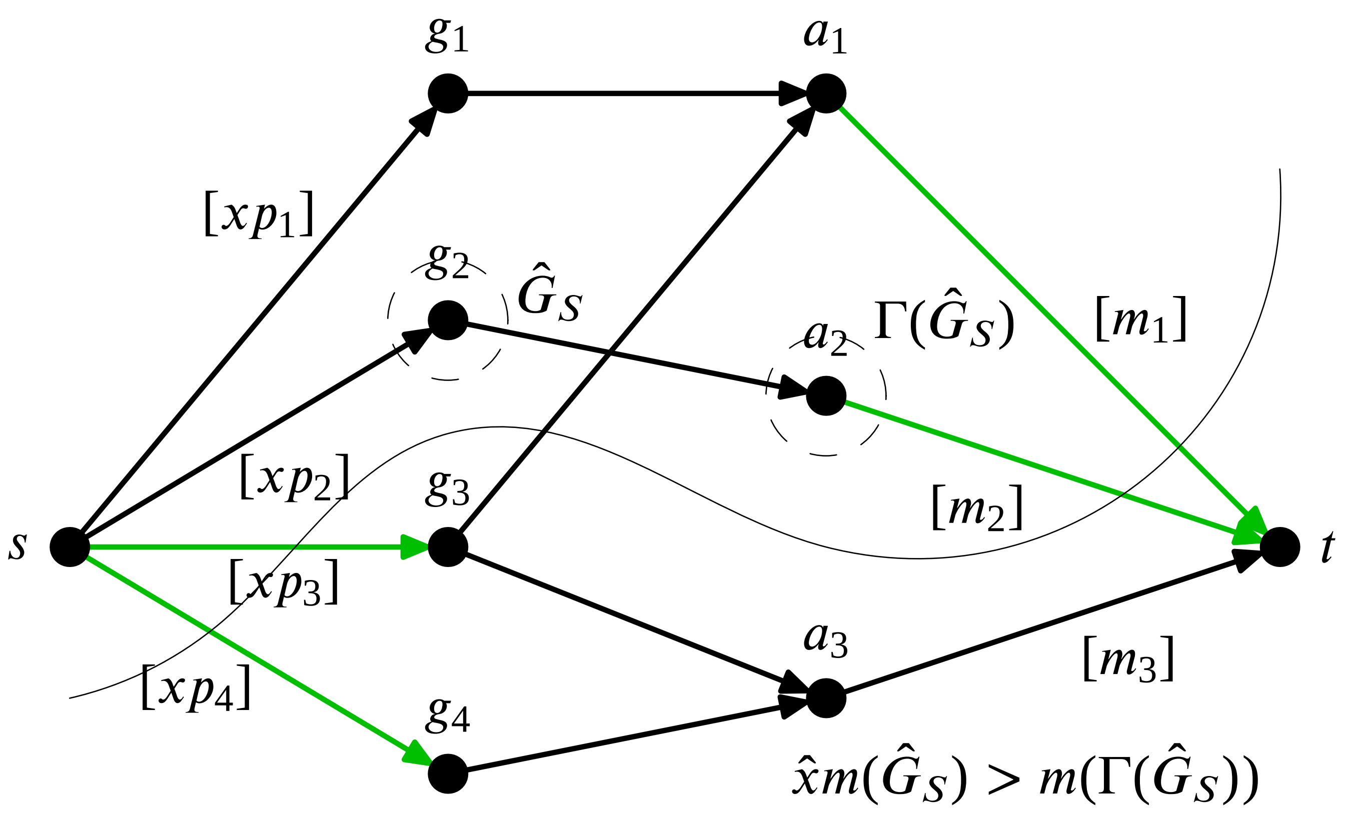


. . . combined with definition of tight set



$$\hat{x}m(\hat{G}_S) + \hat{x}m(\hat{G}_T) = \hat{x}m(\hat{G}) = m(\Gamma(\hat{G})) \geq m(\Gamma(\hat{G}_S)) + m(A_T)$$

Contradiction: there is a "tighter" set



Finding a tight set

Either entire G is a tight set.

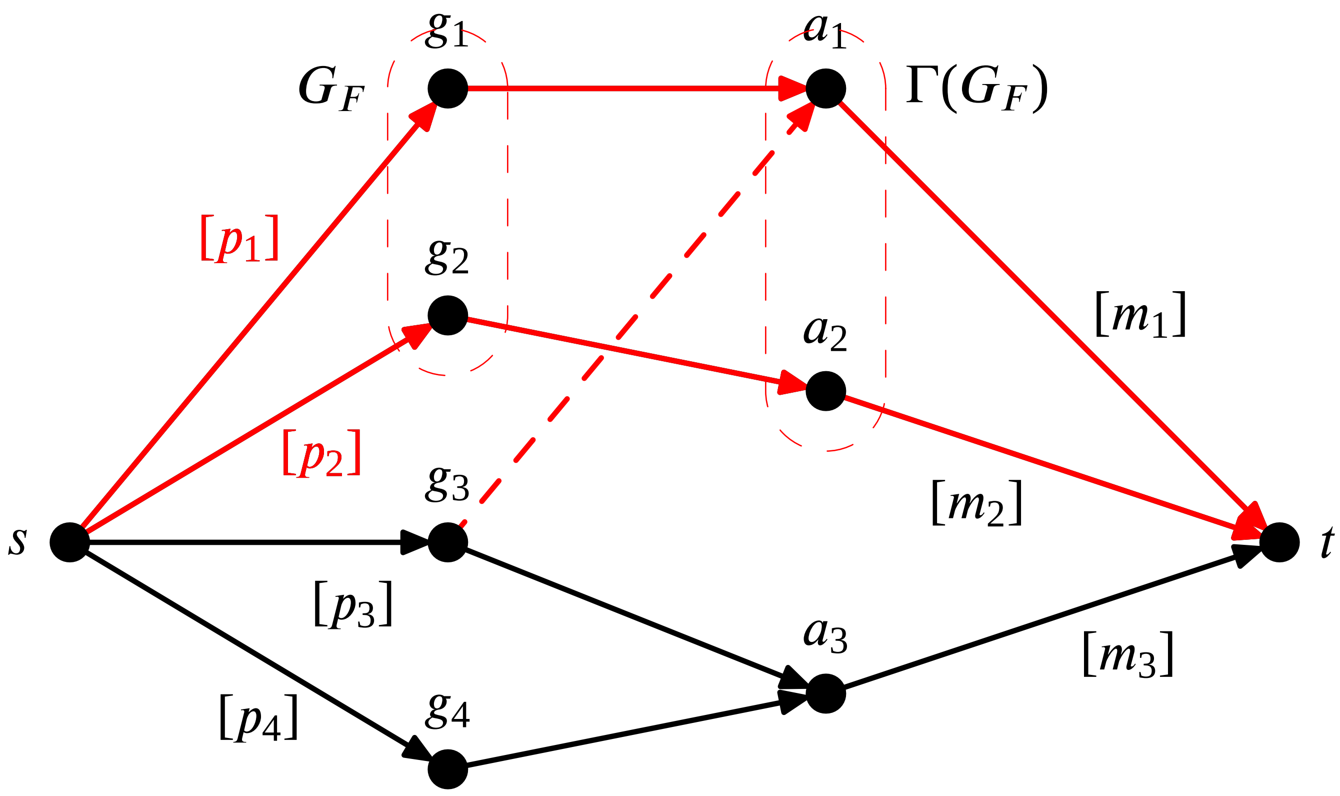
Or, we can recurse and search on smaller graph.

Requires in total $|G|$ max-flow computation.

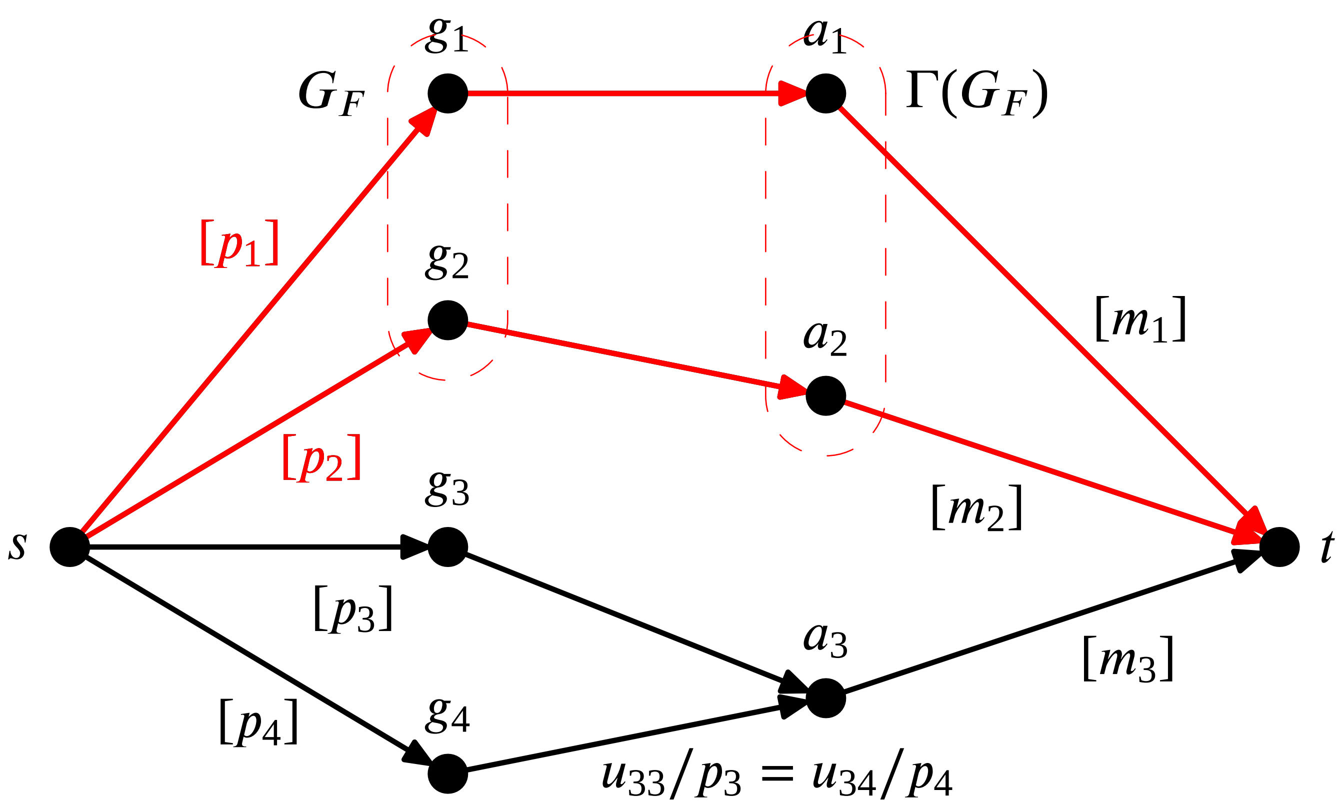
Prices in tight set cannot be increased more.

Freeze them and continue by raising remaining prices.

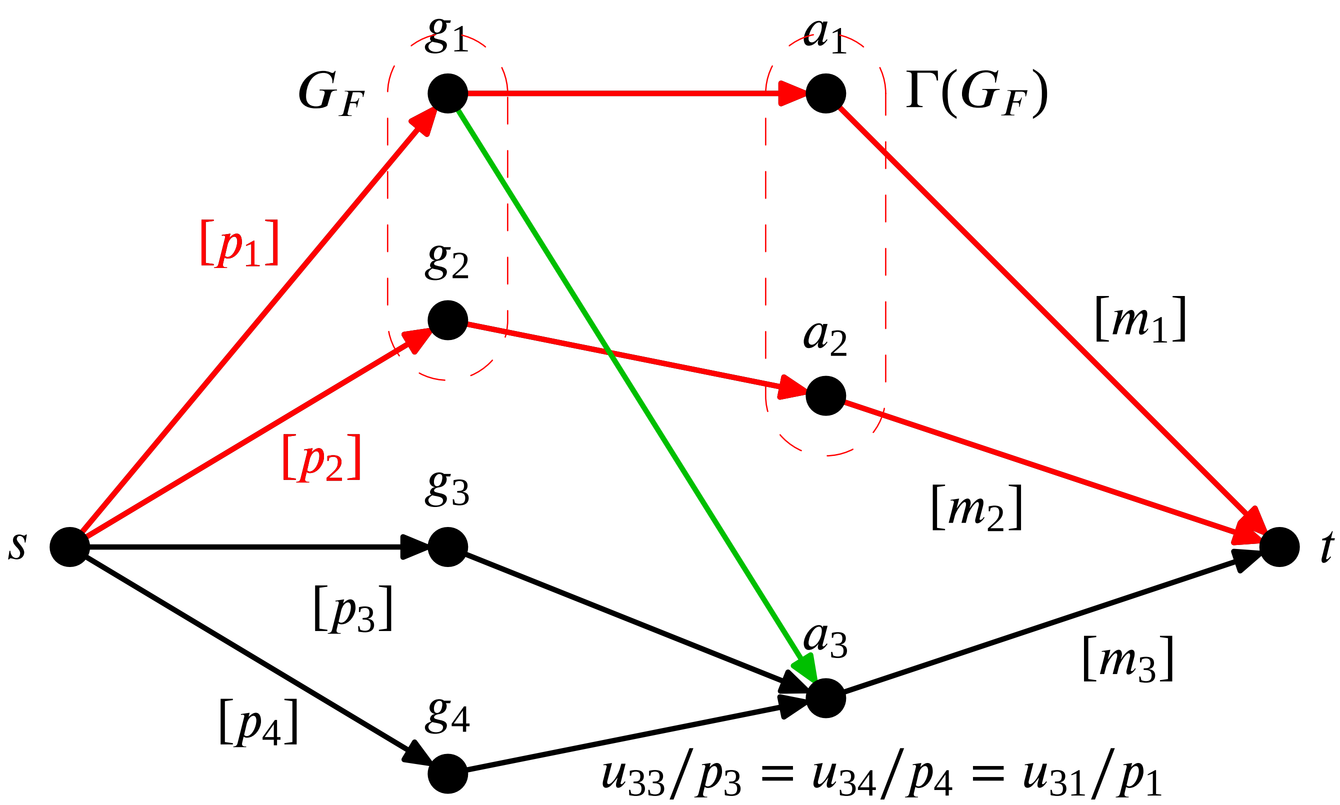
Freezing the variables



Increasing active prices



A new edge appears



Phases in the algorithm

Either new set becomes tight: Freeze it and continue.

Or a new edge appears: Take back frozen component.

Does this terminate?

How soon does it terminate?

Proof of termination

Running time will be function of

$$U = \max_{a,g} u_{ag} \quad \text{and} \quad M = \sum_a m_a$$

Size of input: Polynomial in $\log U$ and $\log M$.

Lower bound the increase in flow as set becomes tight.

When total increase is M, algorithm finishes.

Summing up

Precision: Denominators in prices are at most $|G|U^{|G|}$.

Increase $1/(|G|^2 U^{2|G|})$ of flow in every phase.

Algorithm terminates in $M|G|^2 U^{2|G|}$ phases.

Far from polynomial!

Polynomial running time

Want polynomial running time; modify previous algorithm.

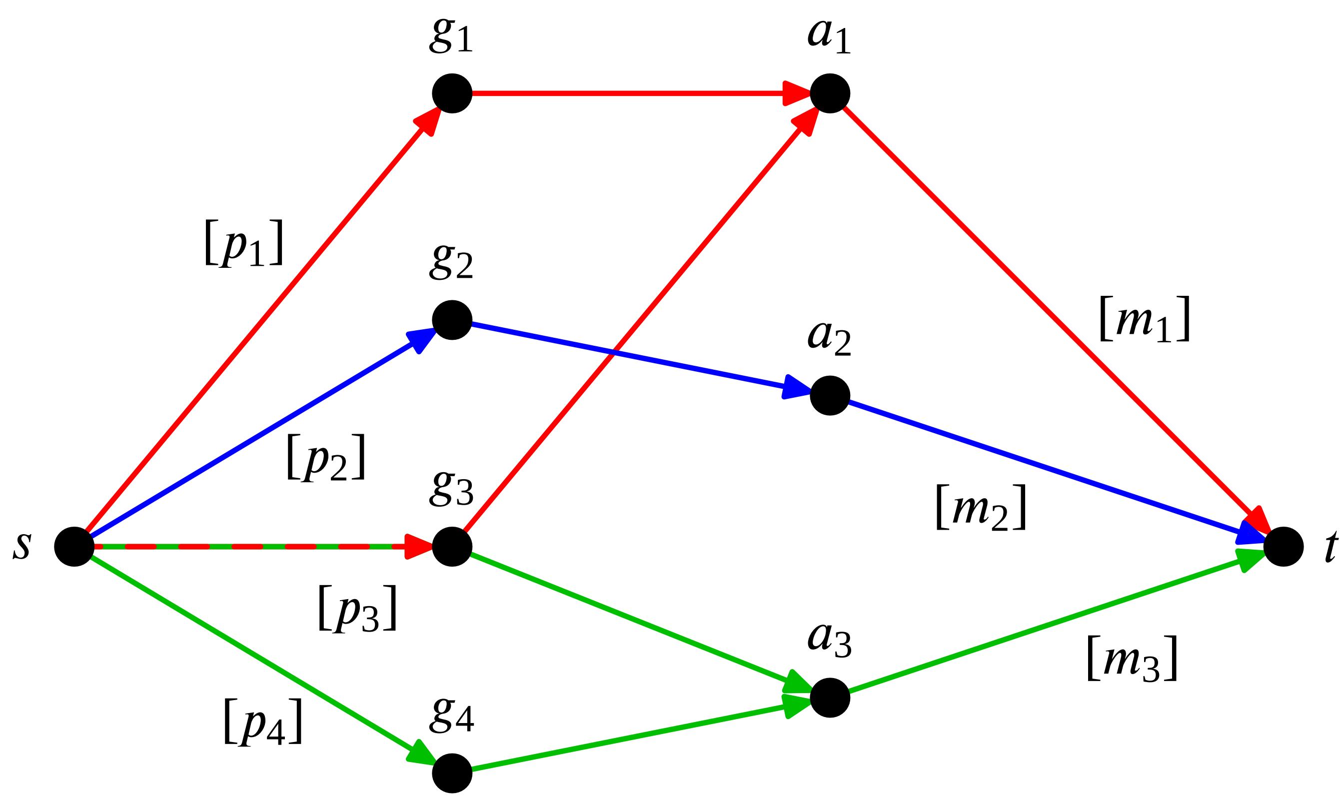
"The one that cries most, gets most."

Only buyers with lots of extra money may buy goods.

Apart from that, same algorithm as before.

Tight sets, appearing edges, iterations, phases...

Maximum flow in the bipartite network



Balanced flow

Let $\gamma_a(\boldsymbol{p}, f)$ be surplus money of agent a.

Let $\boldsymbol{\gamma}(\boldsymbol{p}, f) = (\gamma_a(\boldsymbol{p}, f))_{a \in A}$.

Previous analysis bounded $\|\boldsymbol{\gamma}\|_1$, we now bound $\|\boldsymbol{\gamma}\|_2$.

There can be several maximum flows in corresponding network.

A *balanced flow* is a maximum flow that minimizes $\|\boldsymbol{\gamma}(\boldsymbol{p}, f)\|_2$.
(Can be found with $|G|$ max-flow computations.)

Understanding the distribution of surplus

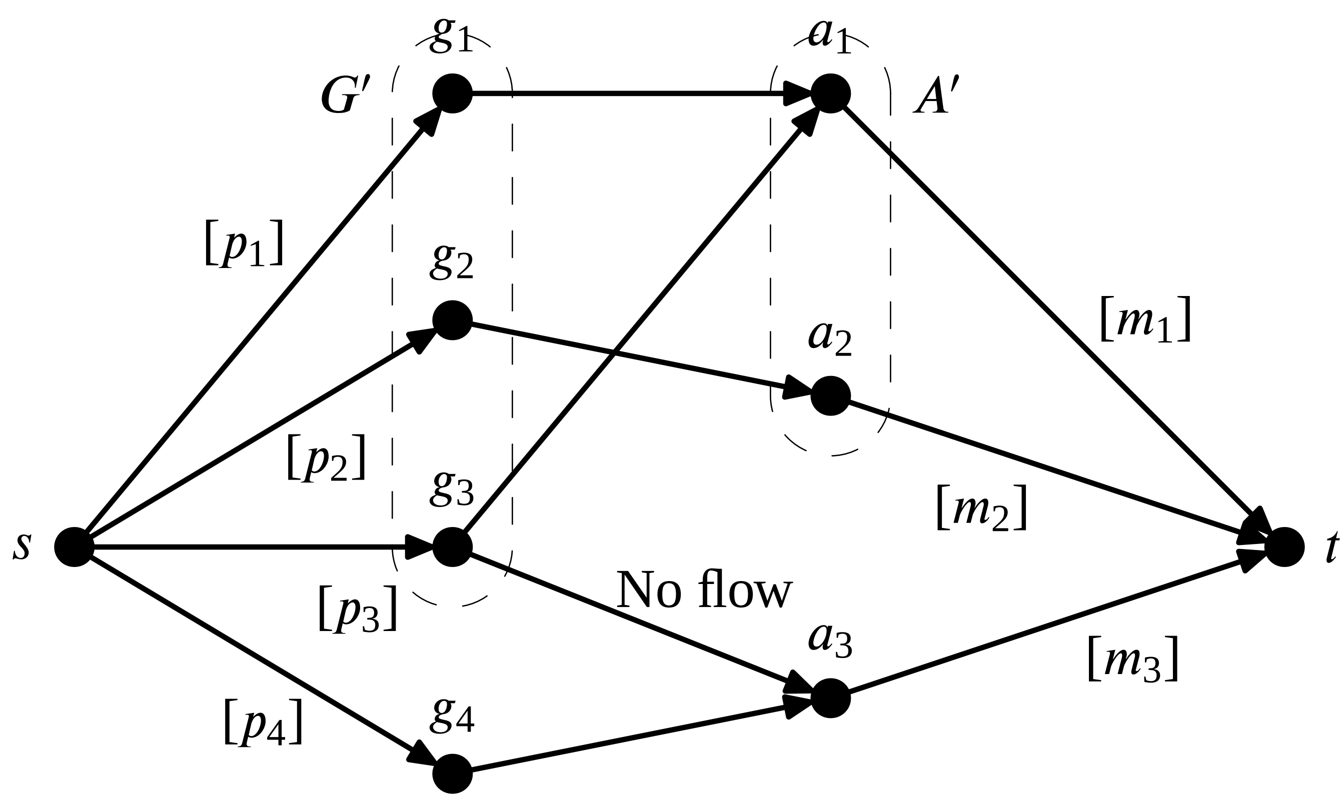
Start with price vector $\boldsymbol{p}$. Compute balanced flow.

Let A' be agents with highest surplus.

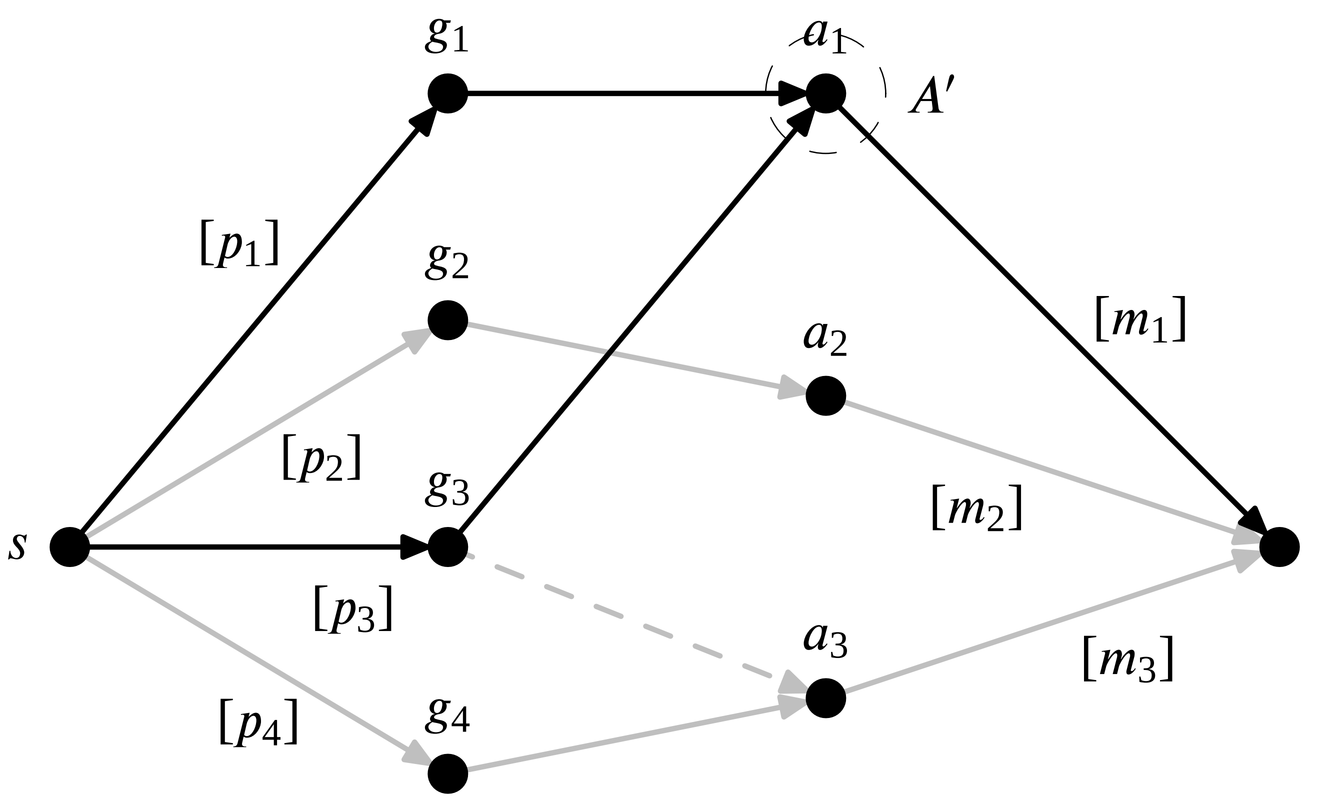
Let G' be goods that can be sold to A'.

Claim: No flow in edges (g, a) where $g \in G'$ and $a \in A \setminus A'$.

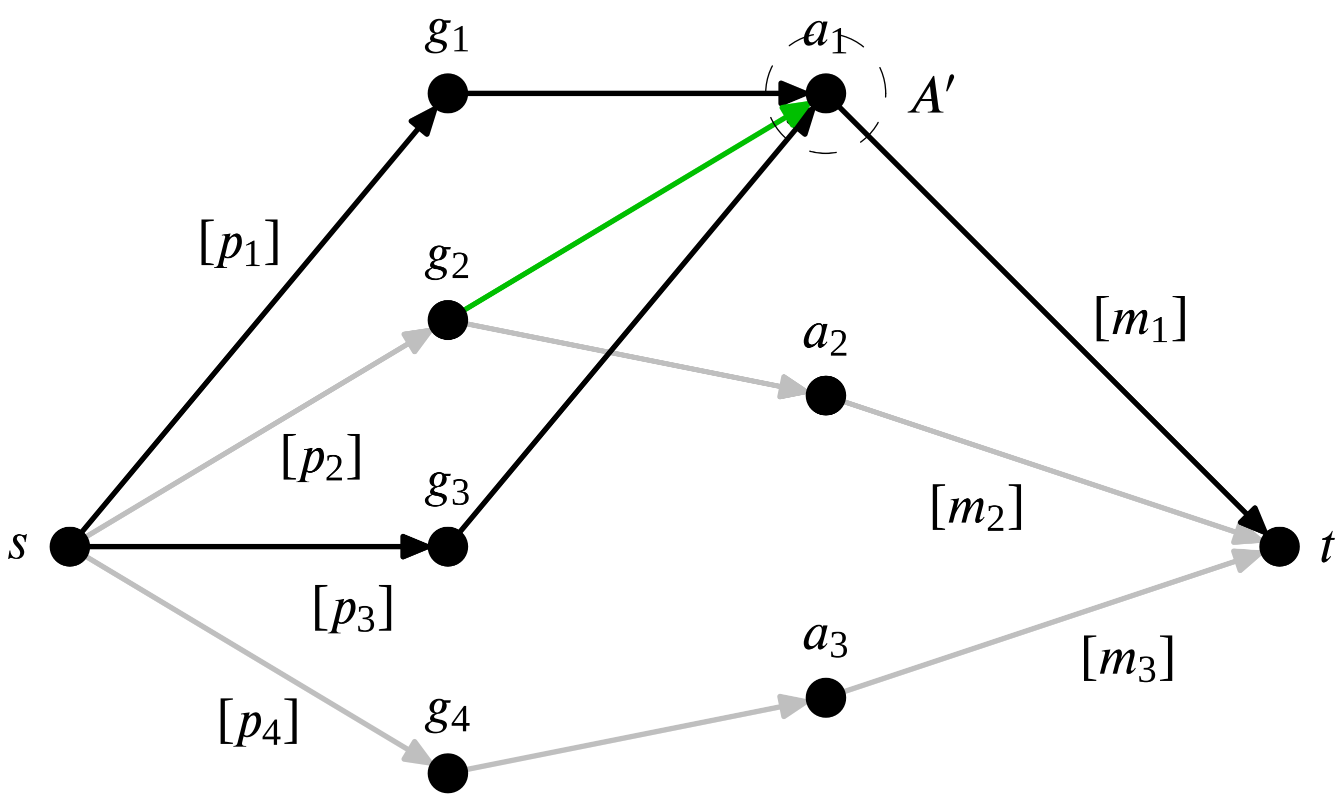
Understanding the distribution of surplus



Increasing the prices



A new edge appears



Handling the new edge

Add new edge to network; recompute balanced flow.

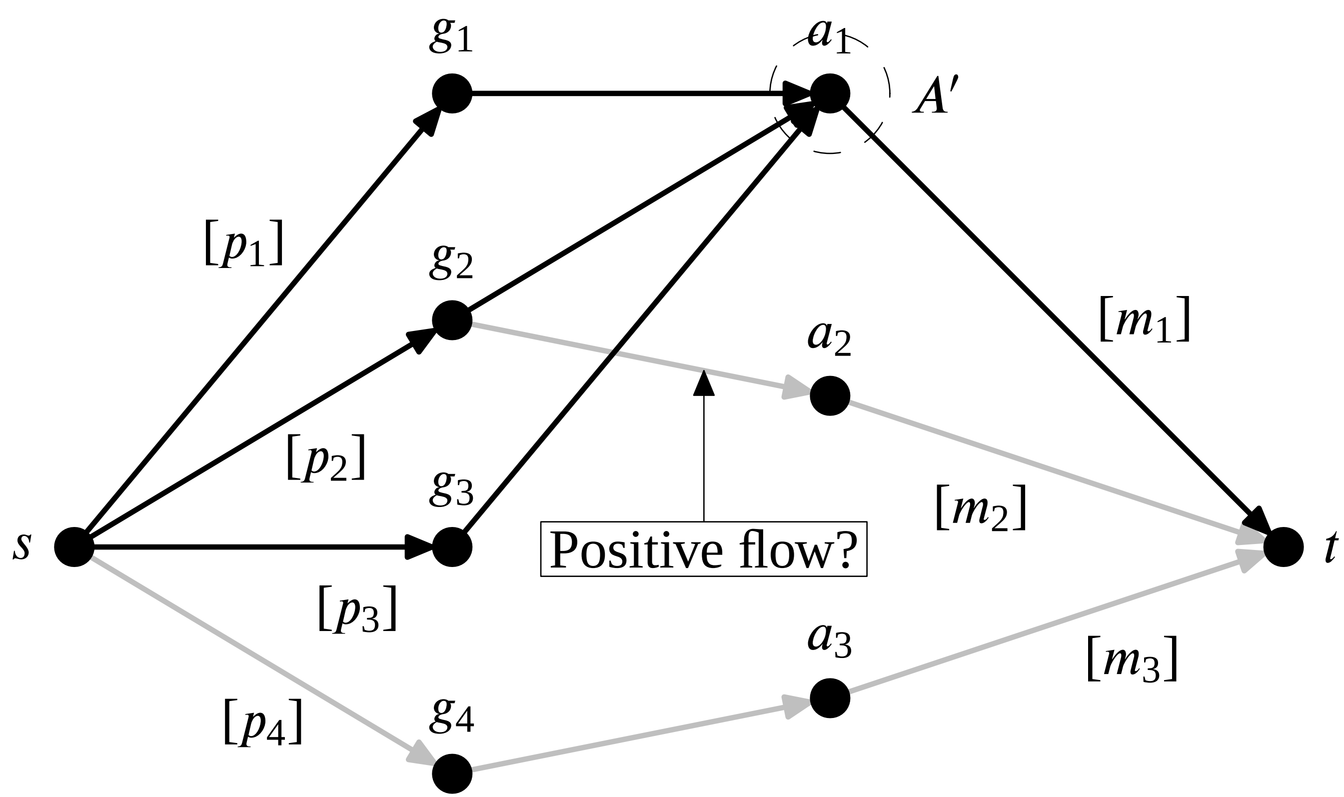
This reduces $\|\boldsymbol{\gamma}(\boldsymbol{p}, f)\|_2$.

Update set of active agents.

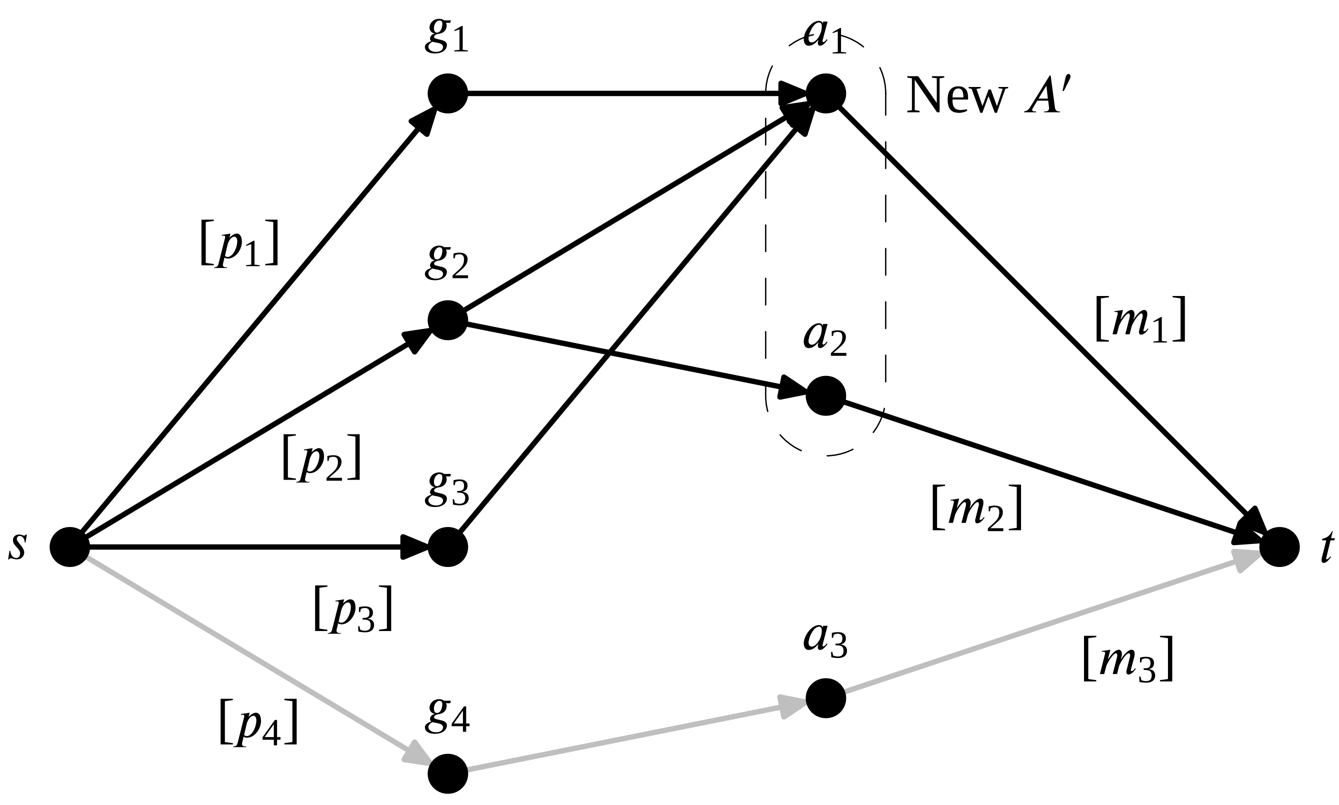
Should be no flow from active goods to non-active agents.

This is needed to maintain the network.

Adding active agents



Continue working with new network



Ending one phase — Starting a new one

A phase ends when surplus of some active buyer is zero.

Start new phase at current prices.

Compute balanced flow; A' is set with maximum surplus…

In each phase $\|\boldsymbol{\gamma}(\boldsymbol{p}, f)\|_2^2$ drops by a factor $\left(1 - \frac{1}{|G|^2}\right)$.

Hence $\|\boldsymbol{\gamma}(\boldsymbol{p}, f)\|_2^2$ drops by a factor of e in $|G|^2$ phases.

This gives termination in polynomial time.

Conclusions and open questions

Algorithm in this talk (Devanur, Papadimitriou, Saberi and Vazirani 2002) gives polynomial time exact solution.

It is a subroutine in algorithms for more general markets.

How fast is the first algorithm presented here?

Conjecture: Runs in strongly polynomial time.

Can algorithms be applied to more general markets?

Other algorithms

Already Fisher (1891) has computational perspective.

Explicit solution given by certain convex program (Eisenberg and Gale 1959).

Ellipsoid algorithm on the convex program gives PTAS.

Ellipsoid algorithm in fact gives exact answer in poly-time.

Interior point algorithm on the convex program gives better running time (Ye 2004).