ECCV 2006 tutorial on

Graph Cuts vs. Level Sets

part IV

Global vs. local optimisation algorithms

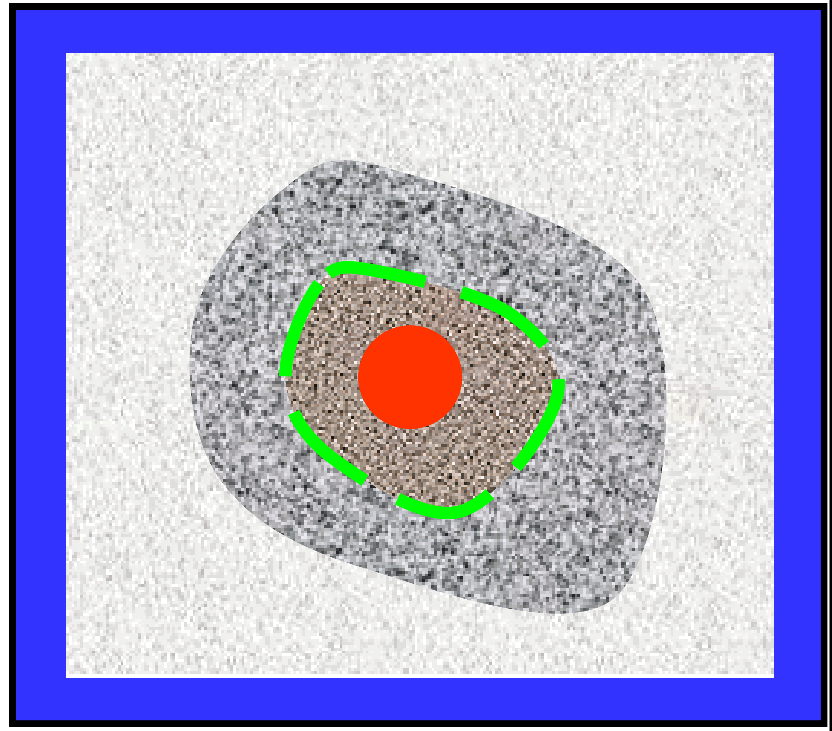
Yuri Boykov — University of Western Ontario

Daniel Cremers — University of Bonn

Vladimir Kolmogorov — University College London

Global vs. local minima (Geodesic active contours)

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation?
  - Graph cuts (e.g. geo-cuts)
    - Same problem, global minimum
    - Convex formulation?

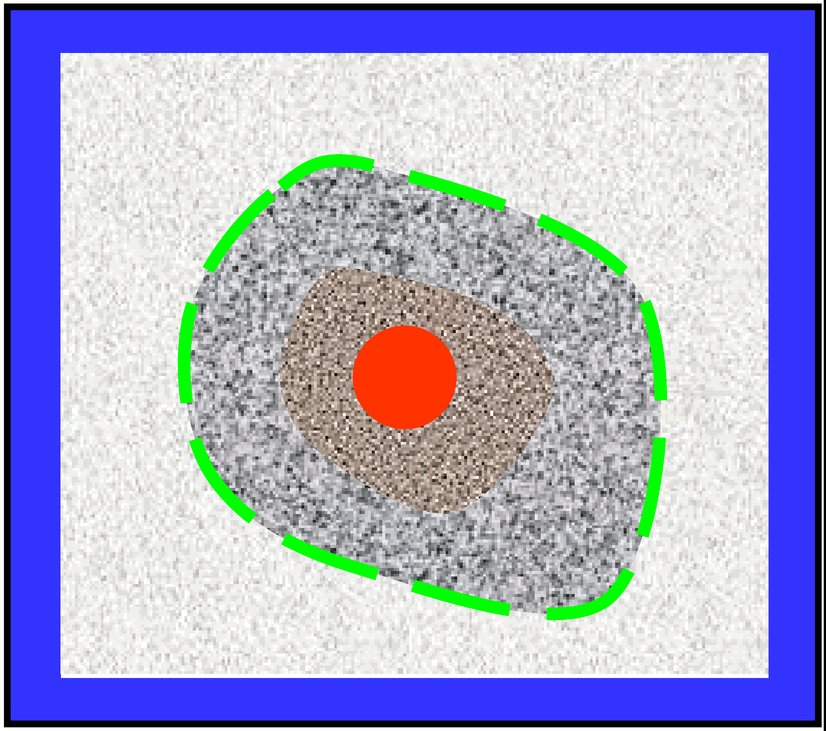


[Anonymous attendee of CVPR'05]:

How is it possible?

Global vs. local minima (Geodesic active contours)

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation?
  - Graph cuts (e.g. geo-cuts)
    - Same problem, global minimum
    - Convex formulation?



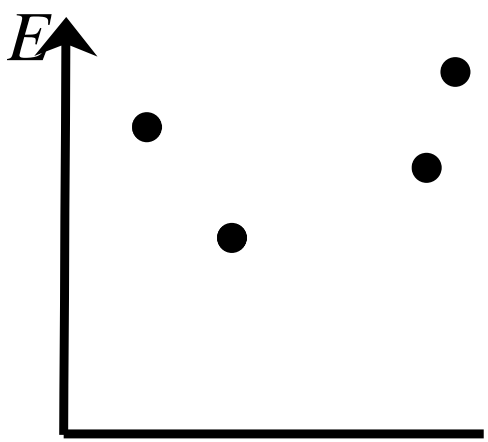
[Anonymous attendee of CVPR'05]:

How is it possible?

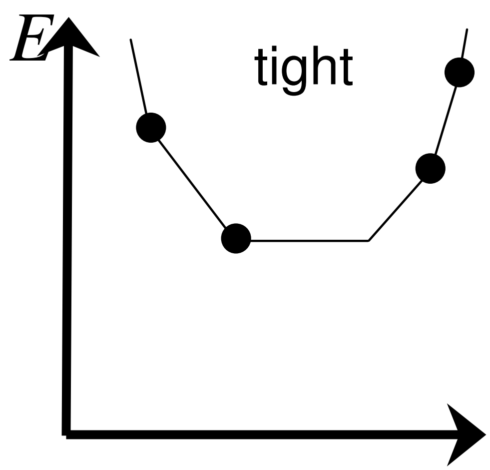
Graph cuts

- Function $E(\mathbf{x})$ of discrete variables: convexity not defined
- Extend the space of solutions: $x_p \in \{0,1\} \;\Rightarrow\; x_p \in [0,1]$
  - Allow *fractional* segmentations
- Extend energy: *linear programming (LP) relaxation*
  - Now convex problem!

submodular function ⇒ integer solution



Energy function
with discrete variables

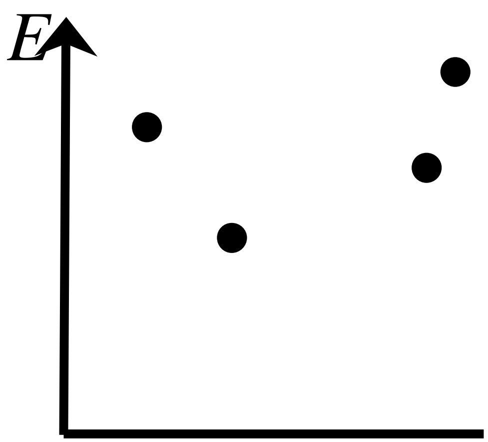


LP relaxation

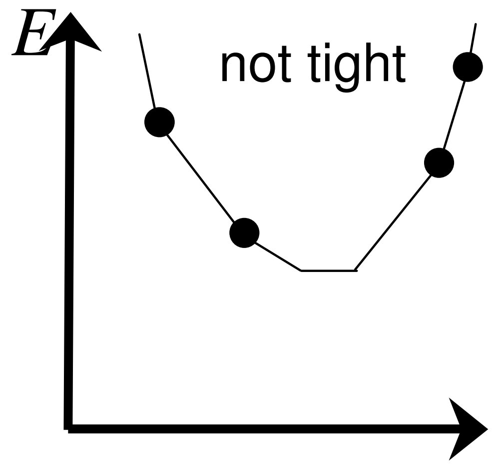
Graph cuts

- Function $E(\mathbf{x})$ of discrete variables: convexity not defined
- Extend the space of solutions: $x_p \in \{0,1\} \Rightarrow x_p \in [0,1]$
  - Allow *fractional* segmentations
- Extend energy: *linear programming (LP) relaxation*
  - Now convex problem!

non-submodular function ⇒ fractional solution (in general)



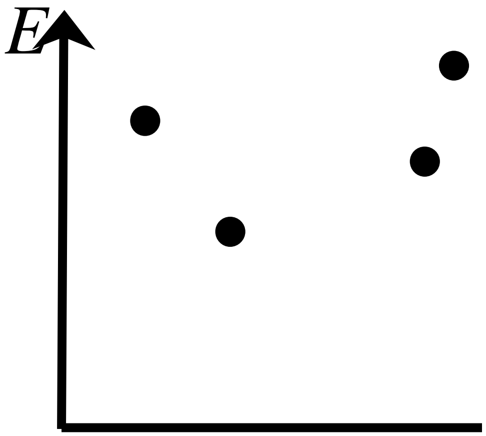
Energy function
with discrete variables



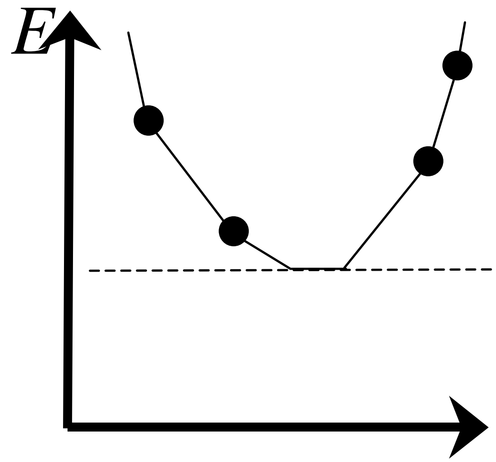
LP relaxation

Solving LP relaxation

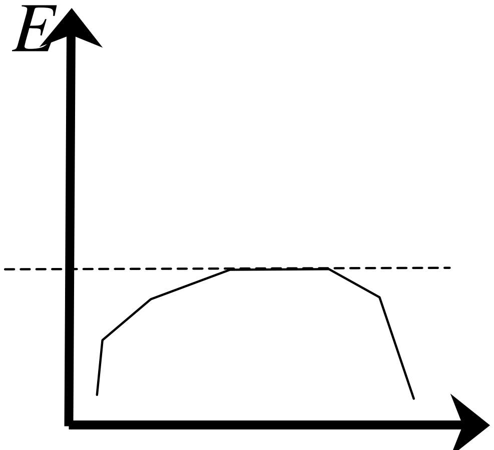
- Too large for general purpose LP solvers (e.g. interior point methods)
- Solve *dual* problem instead of *primal*:
  - Formulate lower bound on the energy
  - Maximize this bound
  - When done, solves primal problem (LP relaxation)
- Two different ways to formulate lower bound
  - Part A: Via *posiforms* => maxflow algorithm (for binary variables)
  - Part B: Via *convex combination of trees* => tree-reweighted message passing



Energy function with discrete variables



LP relaxation

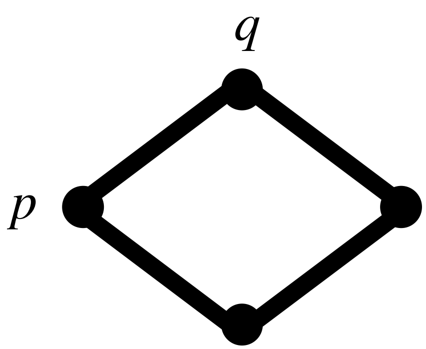


Lower bound on the energy function

Notation and Preliminaries

Energy function

$$E(\mathbf{x} \mid \theta) = \theta_{const} + \underbrace{\sum_{p} \theta_p(x_p)}_{\text{unary terms (data)}} + \underbrace{\sum_{p,q} \theta_{pq}(x_p, x_q)}_{\text{pairwise terms (coherence)}}$$



- x_p are discrete variables (for example, $x_p \in \{0,1\}$)
- $\theta_p(\cdot)$ are unary potentials
- $\theta_{pq}(\cdot,\cdot)$ are pairwise potentials

LP relaxation

- [Schlesinger'76,Koster *et al.*'98,Chekuri *et al.*'00,Wainwright *et al.*'03]

- Introduce indicator variables $x_{p;i}$, $x_{pq;ij}$

$$\sum_{p,i} \theta_p(i) x_{p;i} + \sum_{p,q,i,j} \theta_{pq}(i,j) x_{pq;ij} \rightarrow \min$$

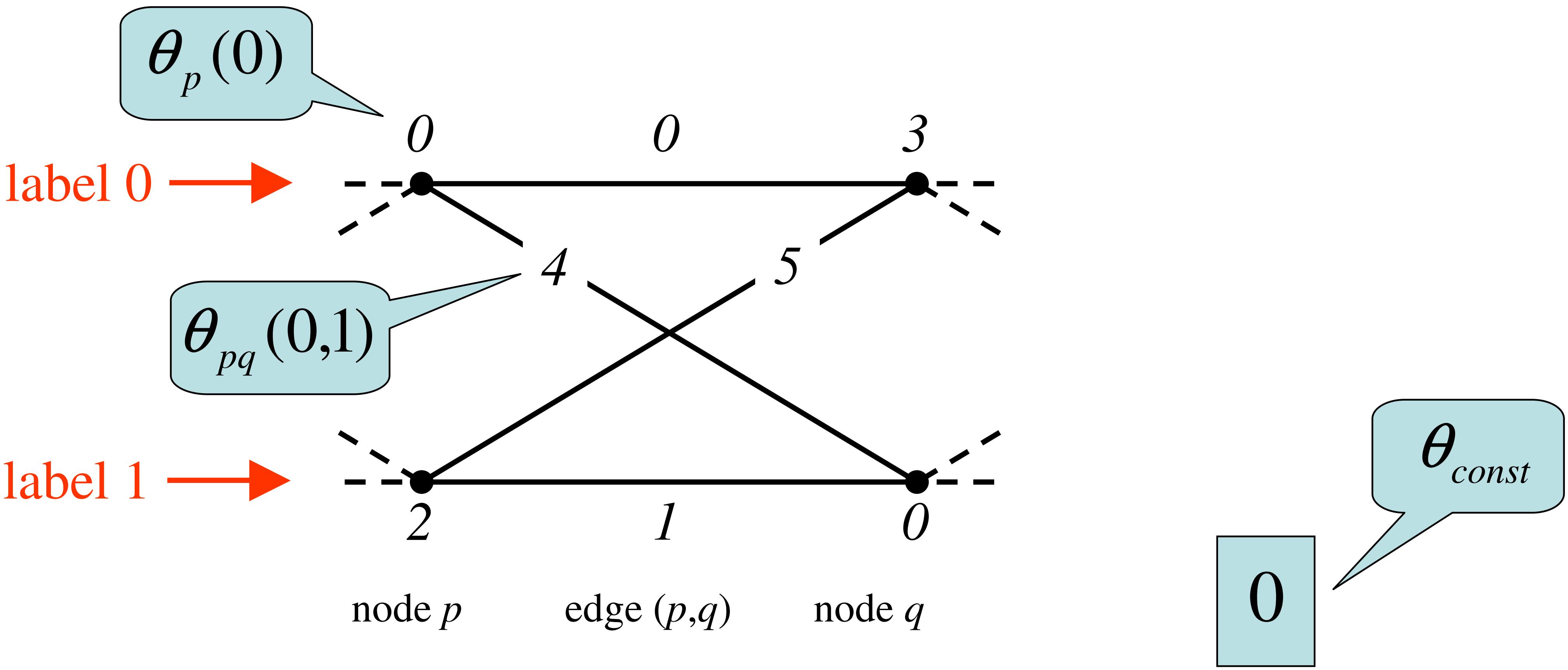
$$\begin{cases} \sum_j x_{pq;ij} = x_{p;i} \\ \sum_i x_{p;i} = 1 \\ x_{pq;ij} \in \{0,1\} \end{cases}$$

relaxation

$$x_{pq;ij} \in [0,1]$$

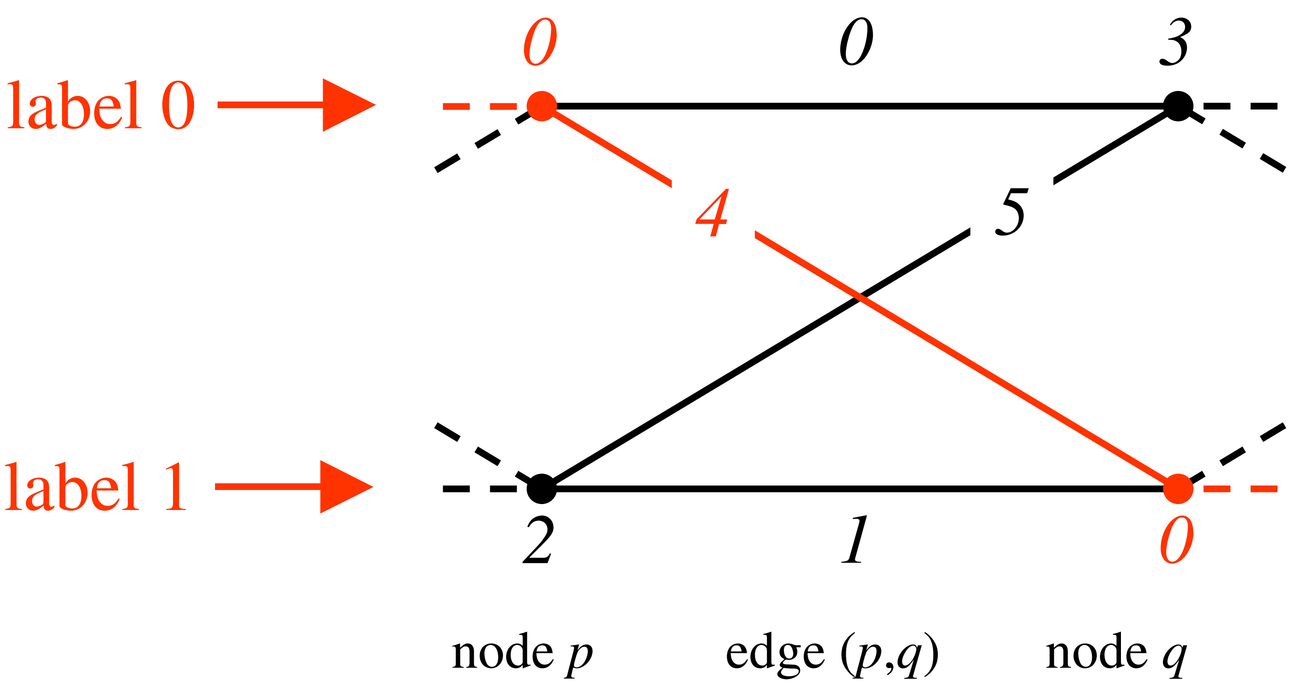
Energy function - visualisation

$$E(\mathbf{x}\,|\,\theta) = \theta_{const} + \sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q)$$



Energy function - visualisation

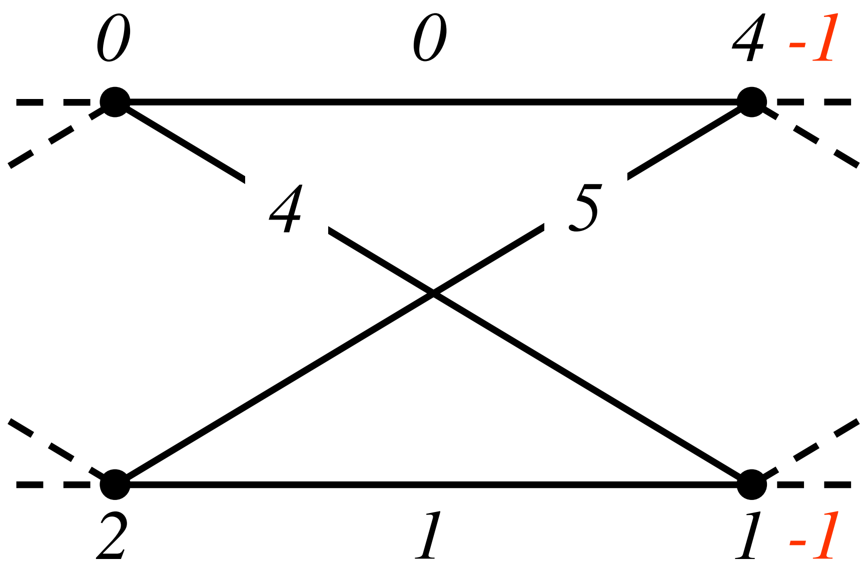
$$E(\mathbf{x} \mid \theta) = \theta_{const} + \sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q)$$



$\theta =$ vector of all parameters

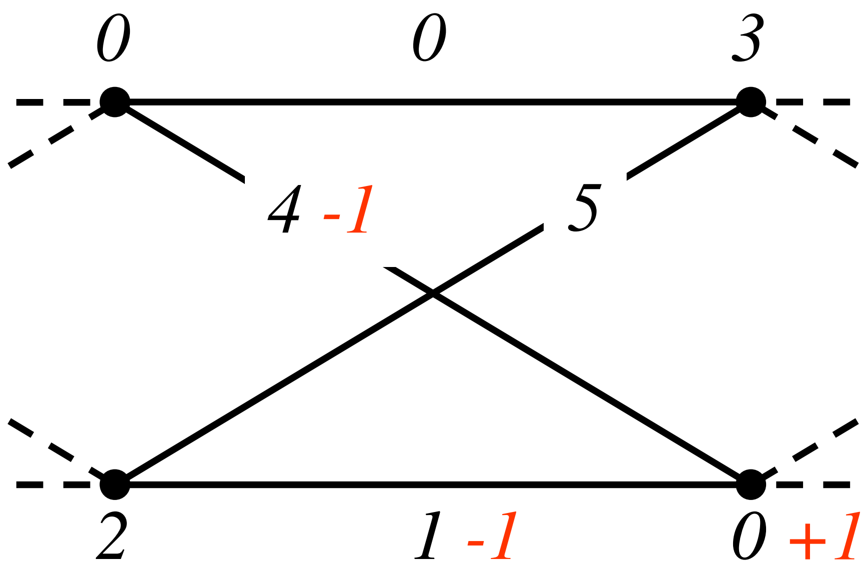
0

Reparameterisation



0 + 1

Reparameterisation



1

- **Definition.** θ' is a reparameterisation of θ $(\theta' \equiv \theta)$ if they define the same energy:

$$E(\mathbf{x} \mid \theta') = E(\mathbf{x} \mid \theta) \quad \text{for any } \mathbf{x}$$

- Maxflow, BP and TRW perform reparameterisations

Part A: Lower bound via posiforms

($\Rightarrow$ maxflow algorithm)

Lower bound via posiforms [Hammer, Hansen, Simeone'84]

$$E(\mathbf{x} \mid \theta) = \theta_{const} + \underbrace{\sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q)}_{\text{non-negative}}$$

maximize (θ_{const})

θ_{const} - lower bound on the energy:

$$E(\mathbf{x} \mid \theta) \geq \theta_{const} \qquad \forall \mathbf{x}$$

Outline of part A

- Posiform maximisation: algorithm?

- Binary variables, *submodular* functions
  - Reduction to maxflow
  - Global minimum of the energy

- Binary variables, *non-submodular* functions
  - Reduction to maxflow
    - More complicated graph
  - *Part* of optimal solution

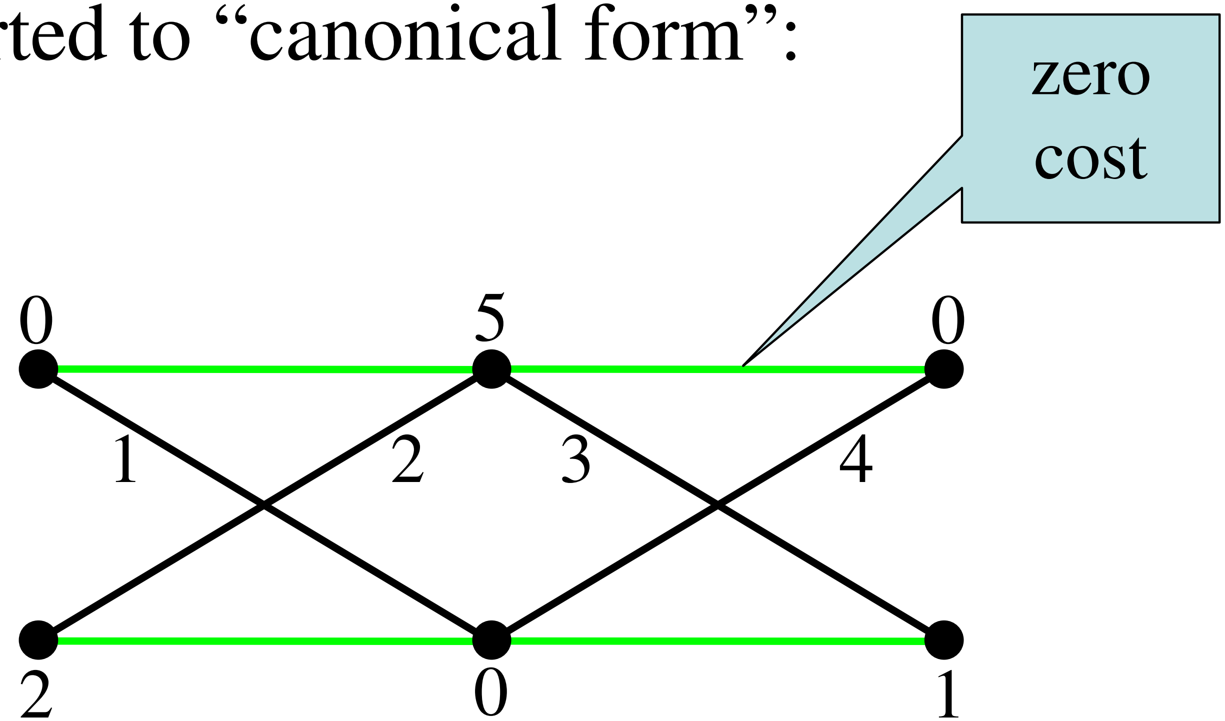
Binary variables, submodular functions

Submodularity and canonical form

- Definition: *E* is *submodular* if every pairwise term satisfies

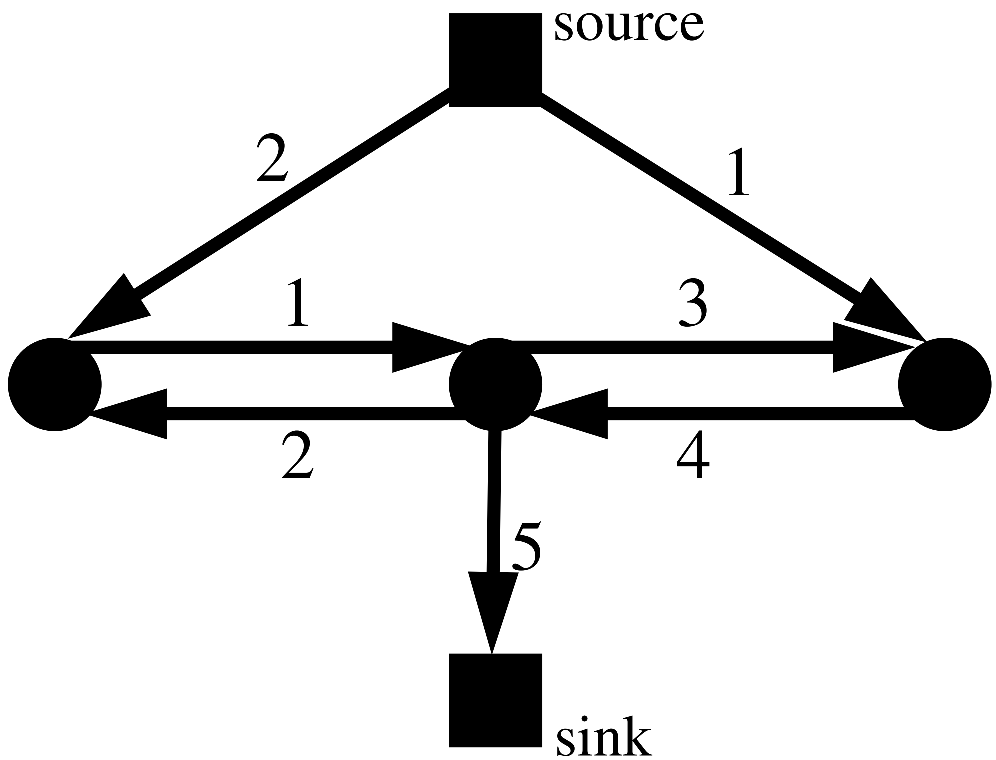
$$\theta_{pq}(0,0) + \theta_{pq}(1,1) \leq \theta_{pq}(0,1) + \theta_{pq}(1,0)$$

- Can be converted to "canonical form":



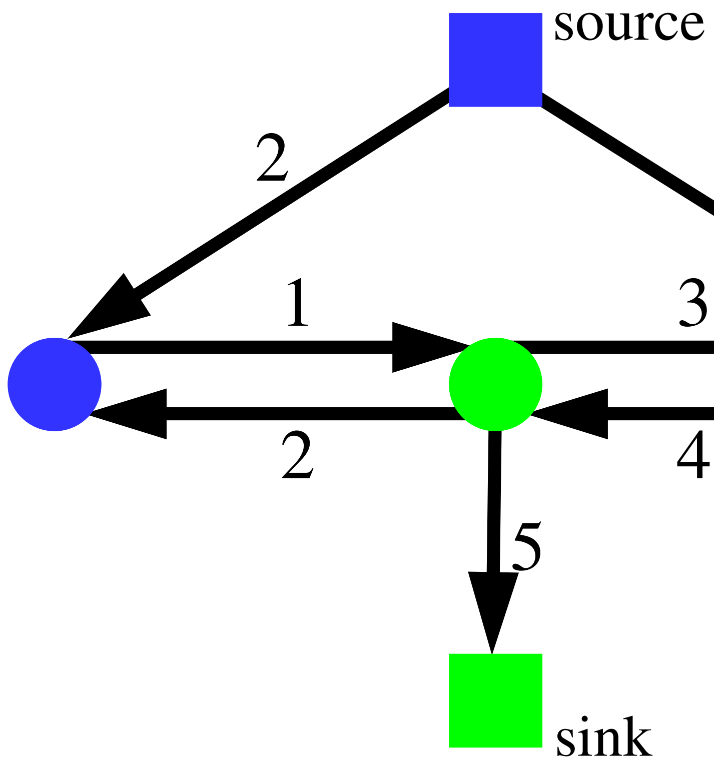
Overview of min cut/max flow

Min Cut problem



Directed weighted graph

Min Cut problem

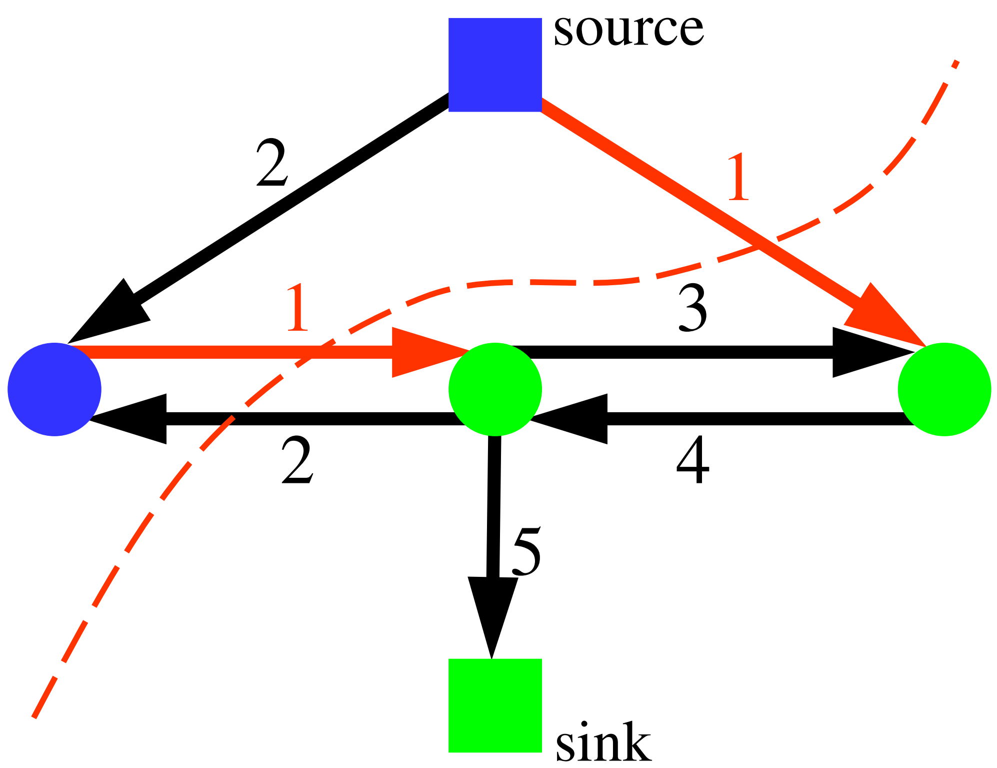


Cut:

S = {source, node 1}

T = {sink, node 2, node 3}

Min Cut problem



Cut:

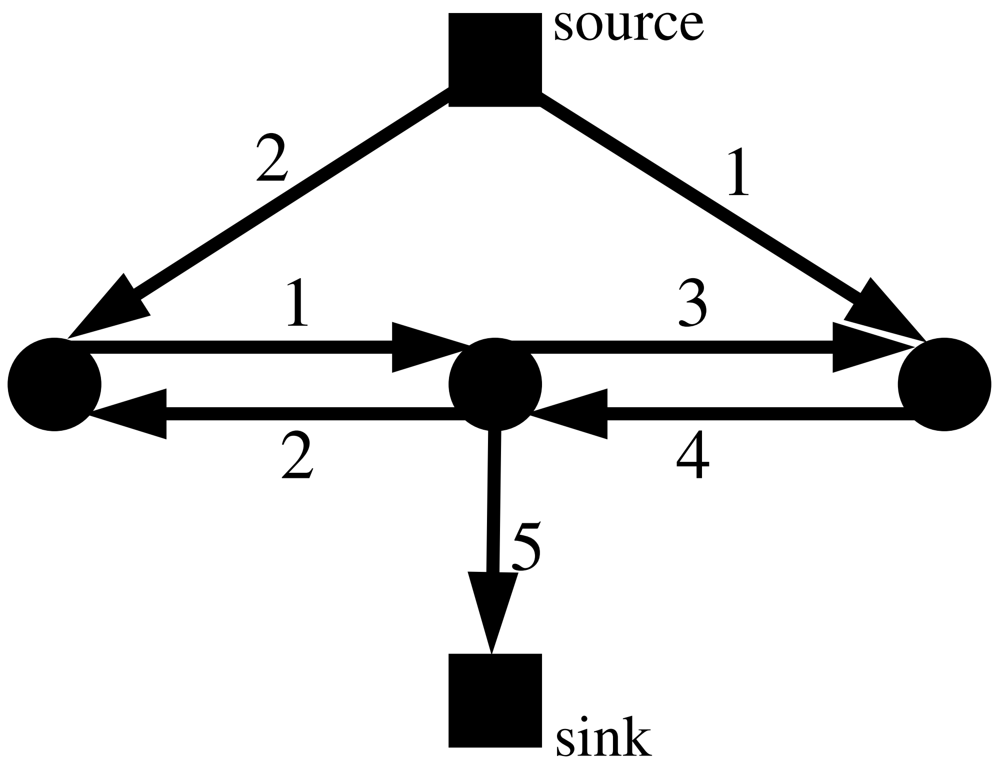
S = {source, node 1}

T = {sink, node 2, node 3}

Cost(S,T) = 1 + 1 = 2

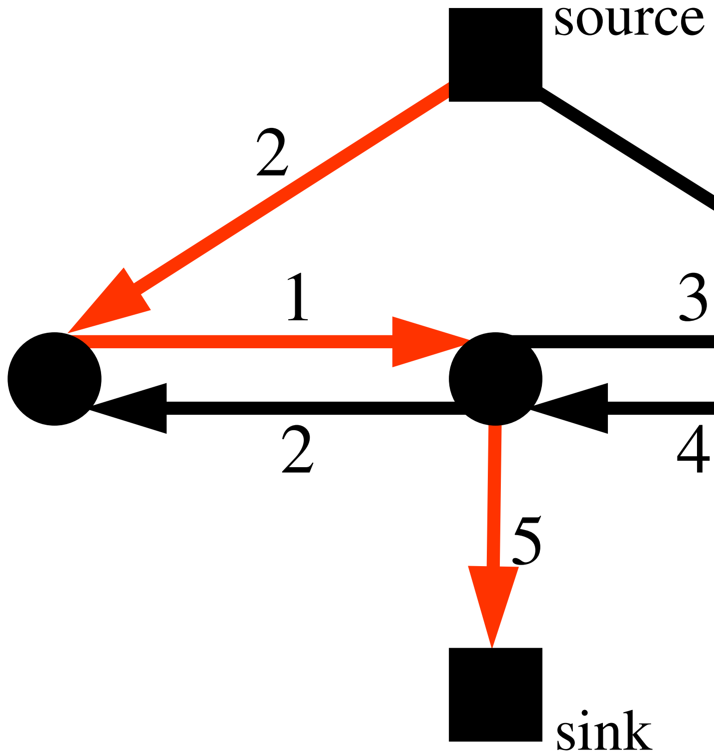
- Task: Compute cut with minimum cost

Maxflow algorithm



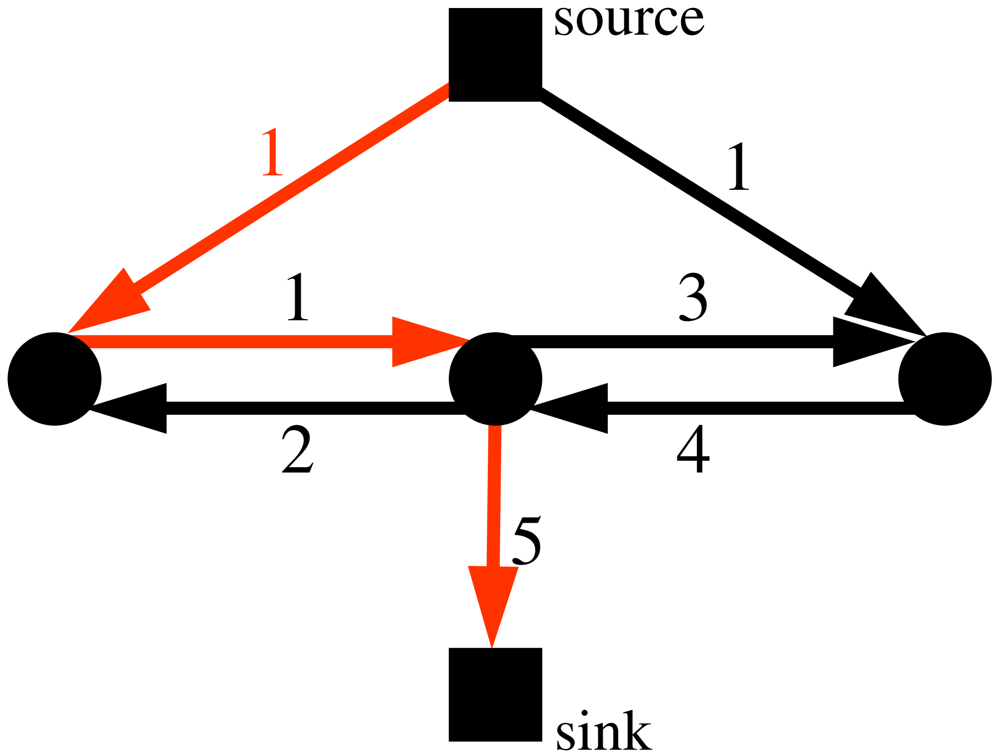
value(flow)=0

Maxflow algorithm



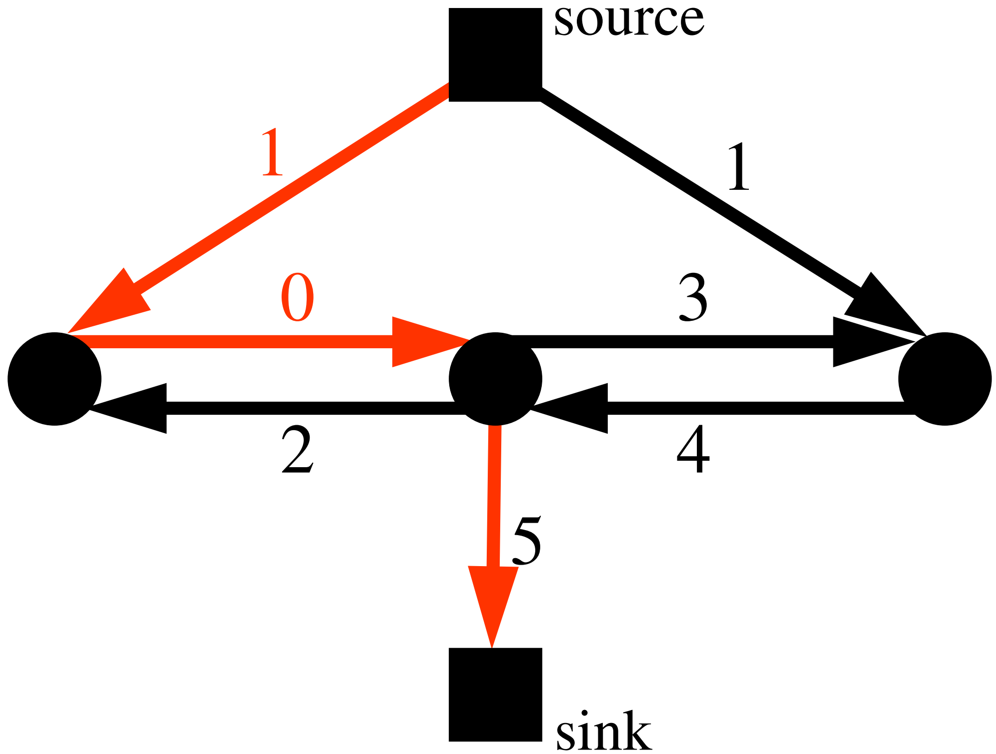
value(flow)=0

Maxflow algorithm



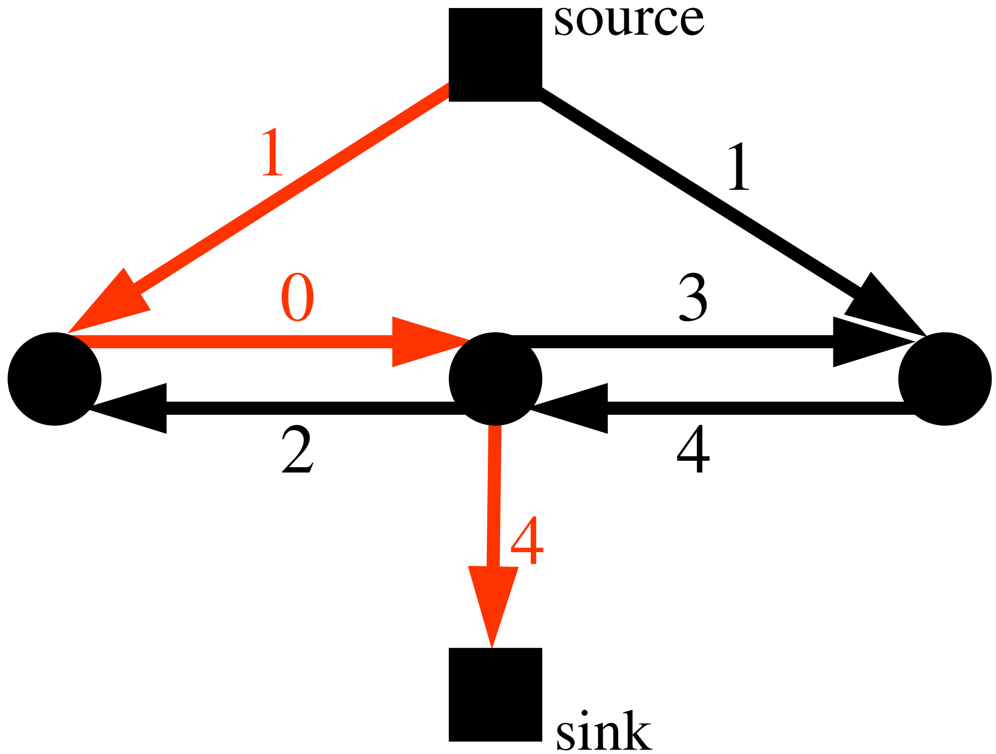
value(flow)=0

Maxflow algorithm



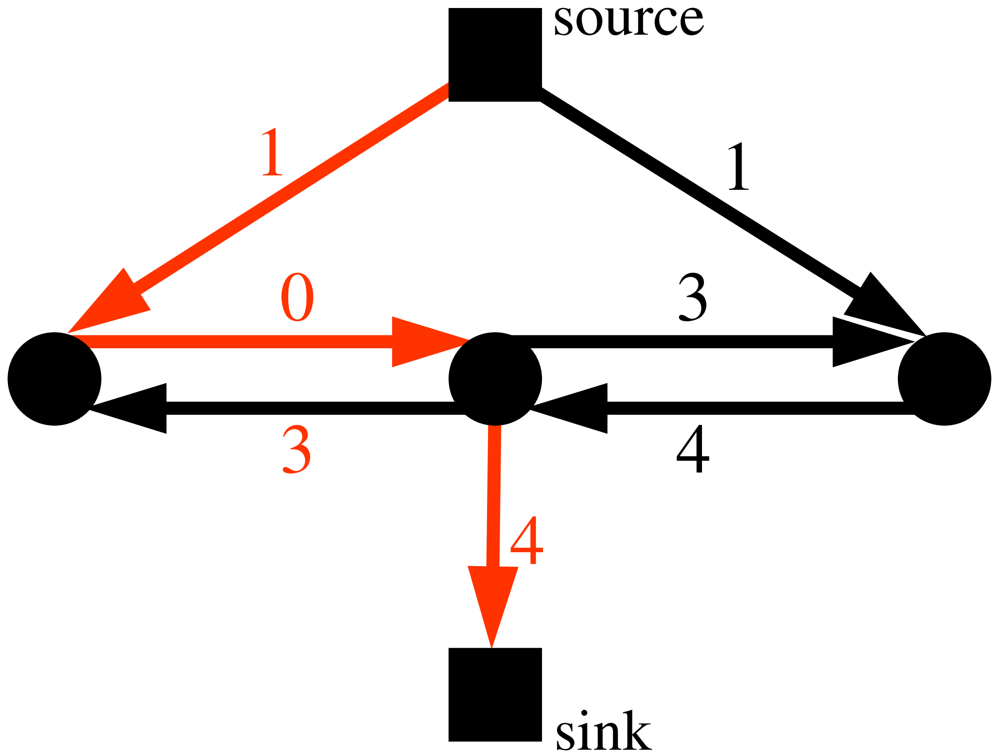
value(flow)=0

Maxflow algorithm



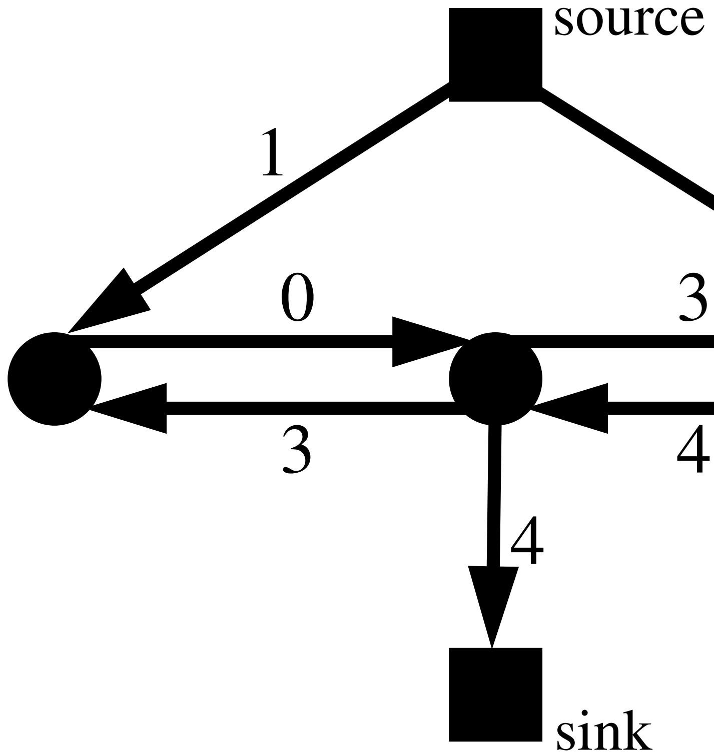
value(flow)=1

Maxflow algorithm



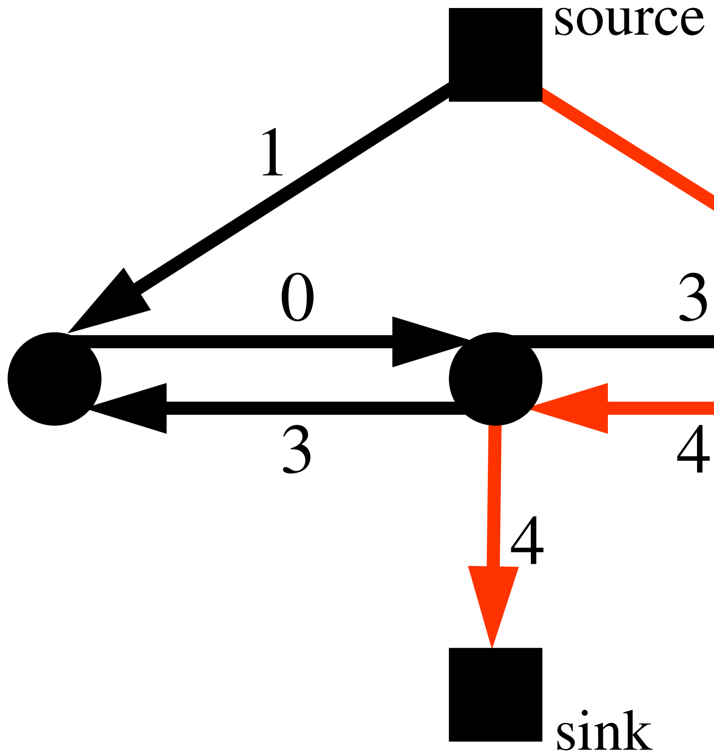
value(flow)=1

Maxflow algorithm



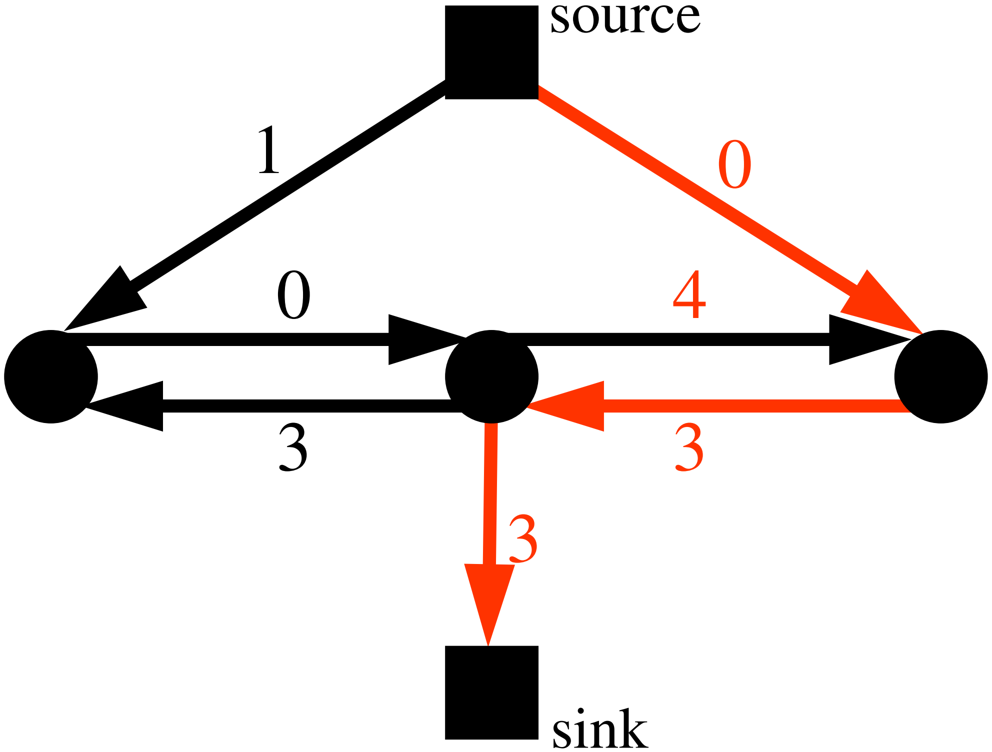
value(flow)=1

Maxflow algorithm



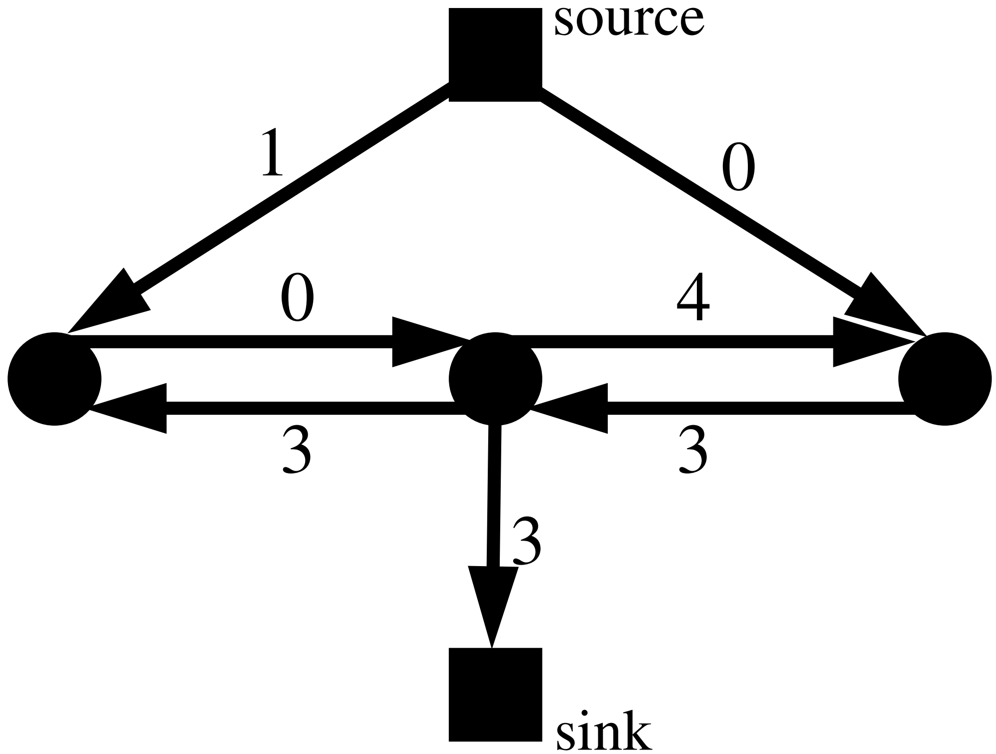
value(flow)=1

Maxflow algorithm



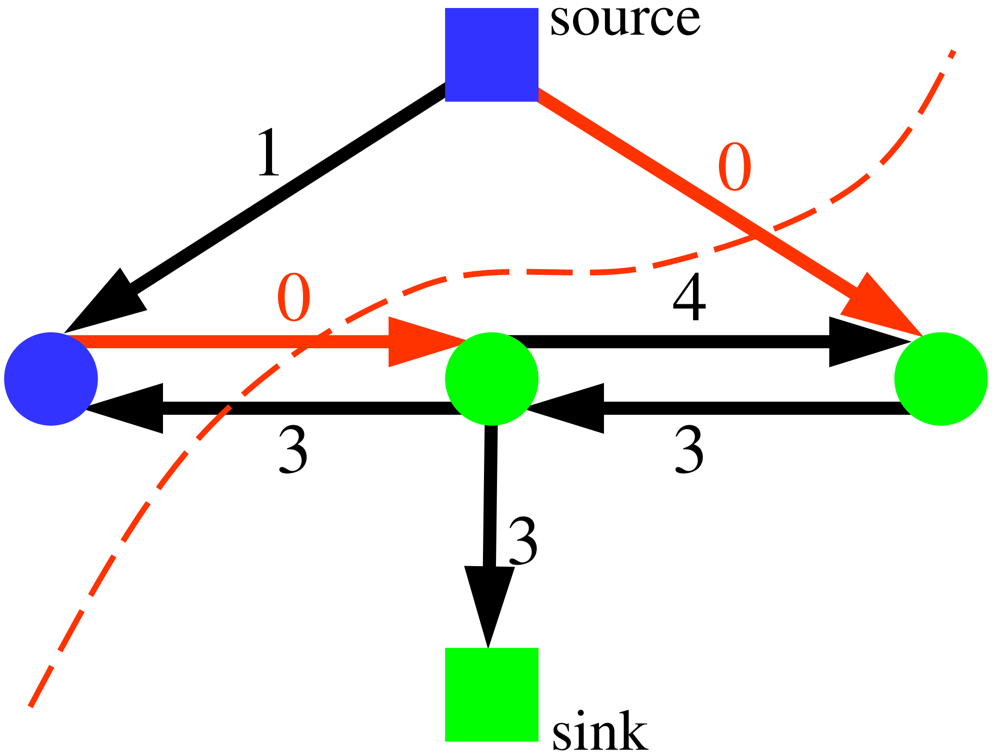
value(flow)=2

Maxflow algorithm



value(flow)=2

Maxflow algorithm

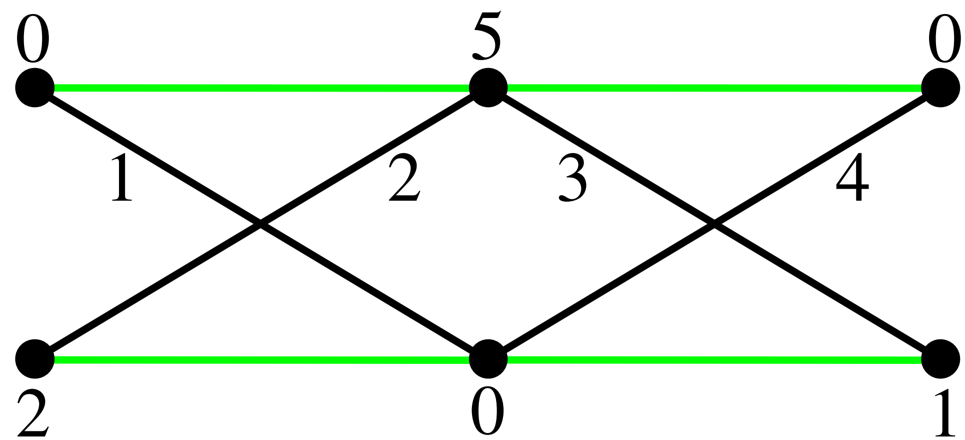


value(flow)=2

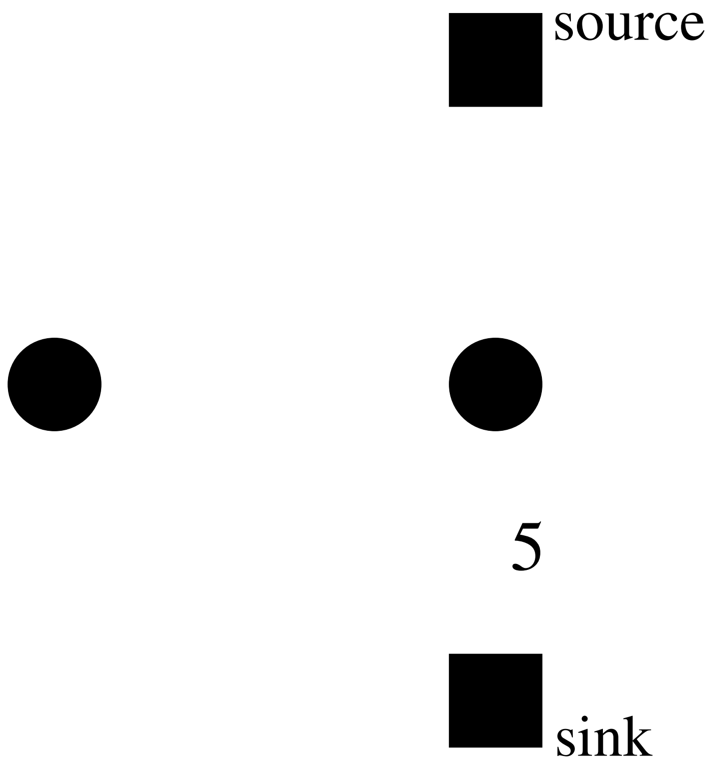
Binary variables,
non-submodular functions

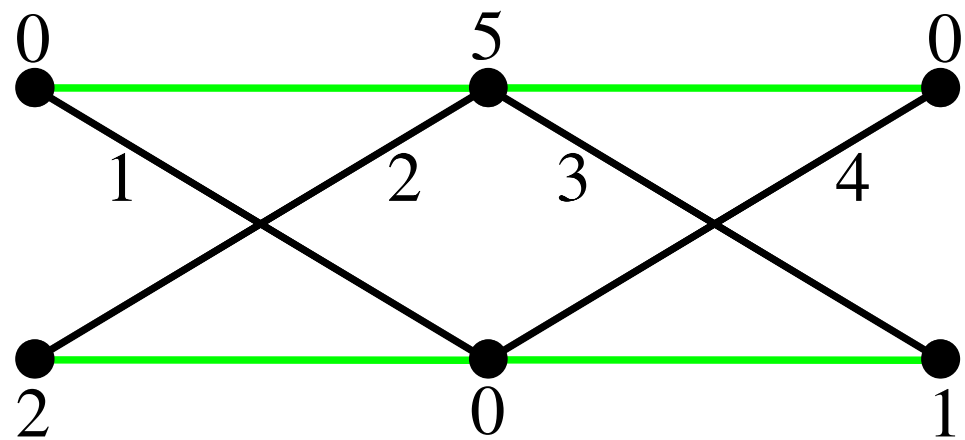
Reduction to maxflow

Maxflow algorithm and reparameterisation

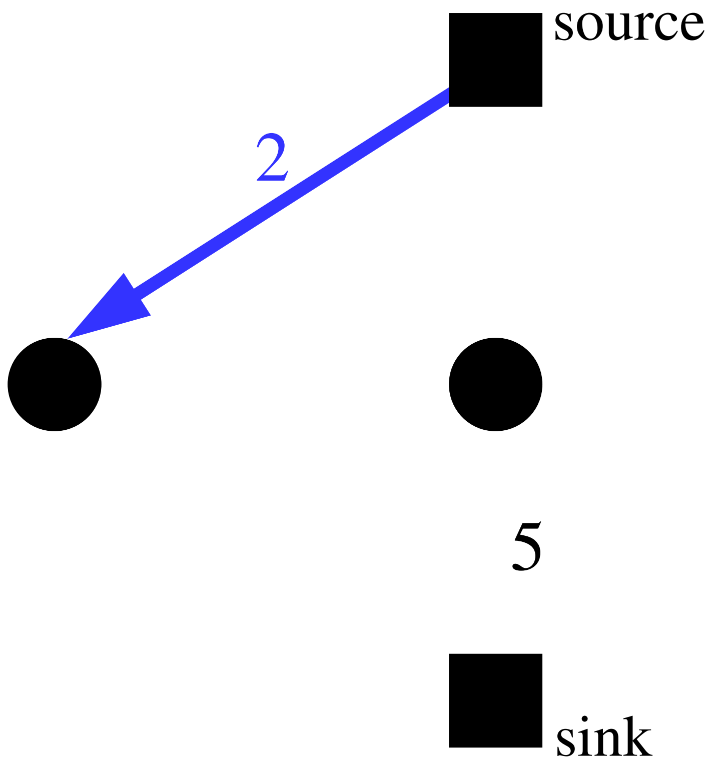


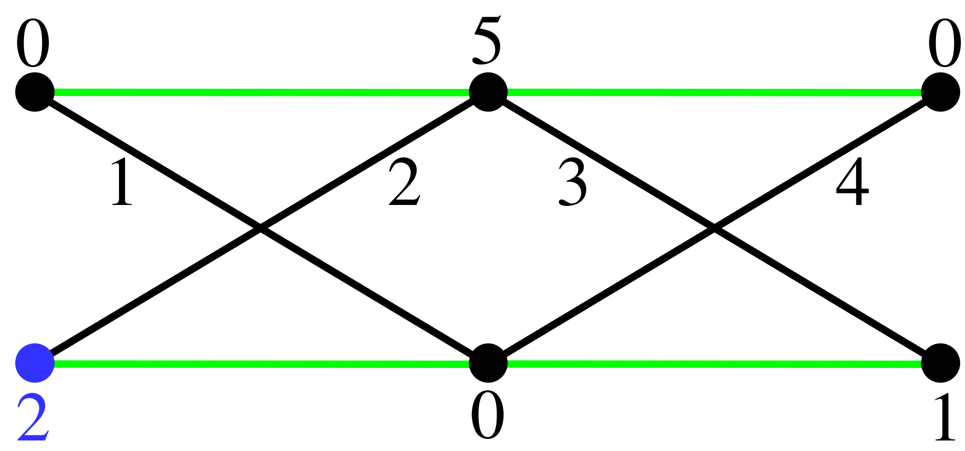
Maxflow algorithm and reparameterisation



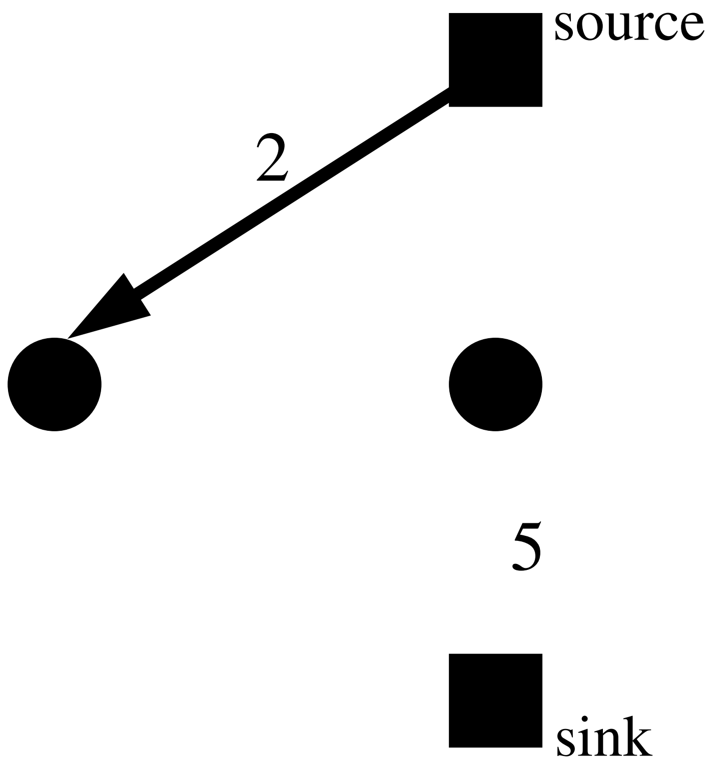


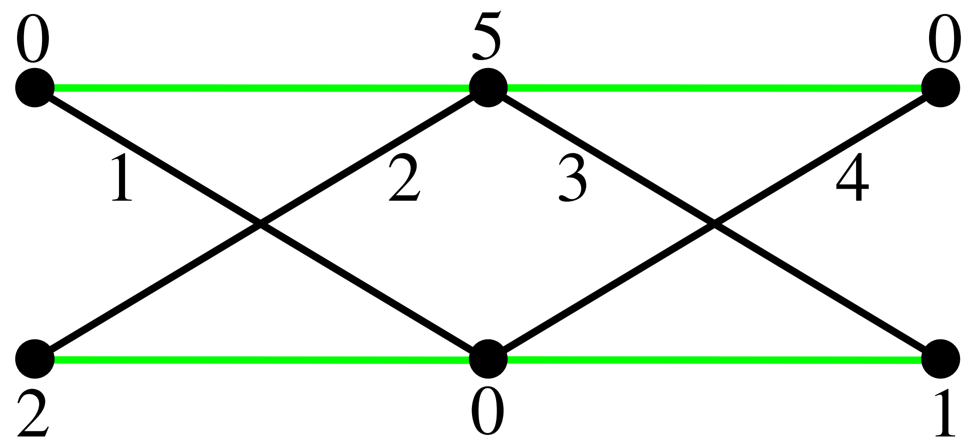
Maxflow algorithm
and reparameterisation



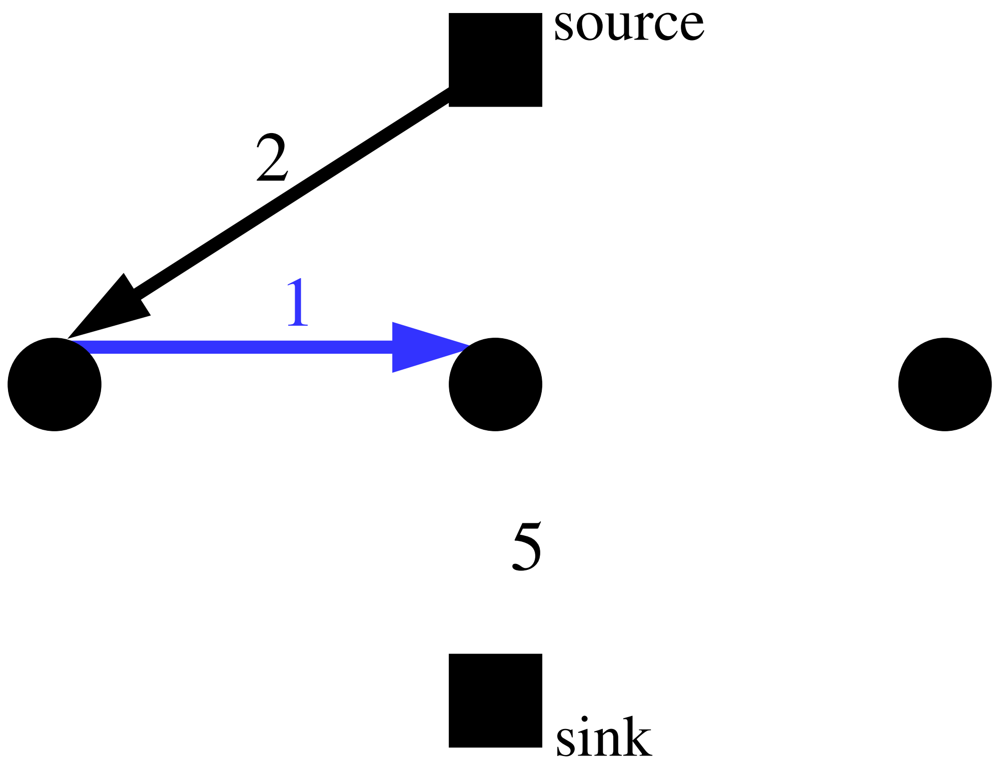


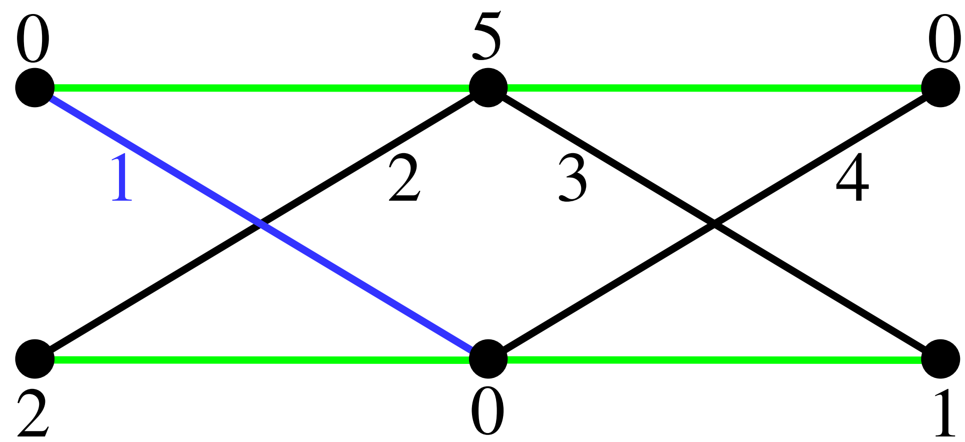
Maxflow algorithm and reparameterisation



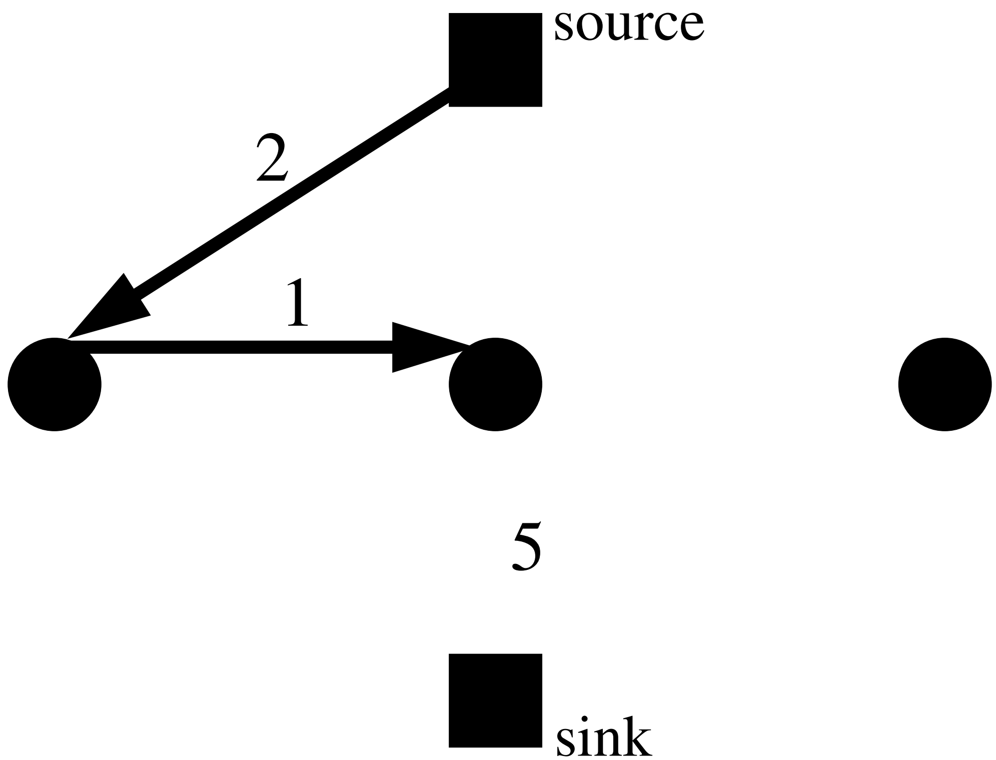


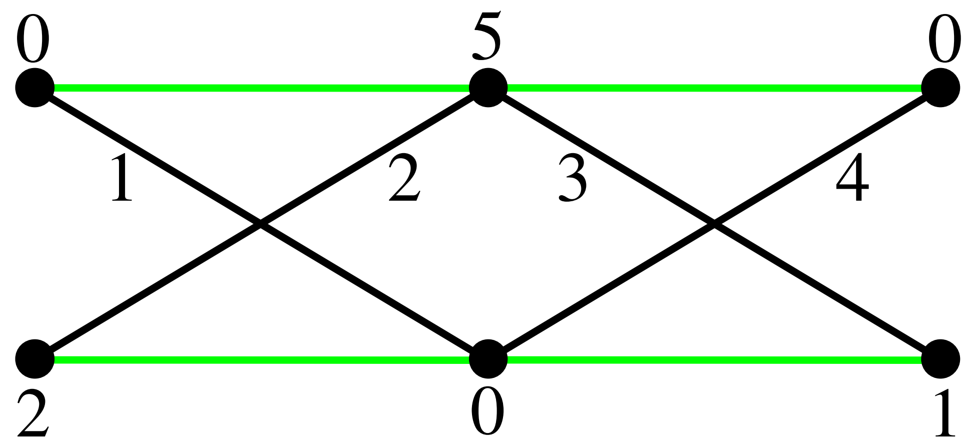
Maxflow algorithm and reparameterisation



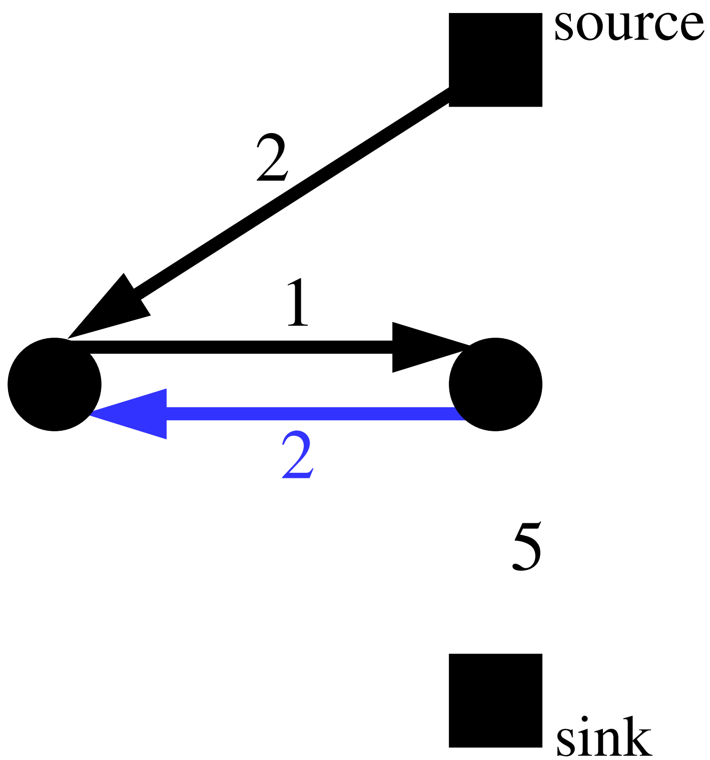


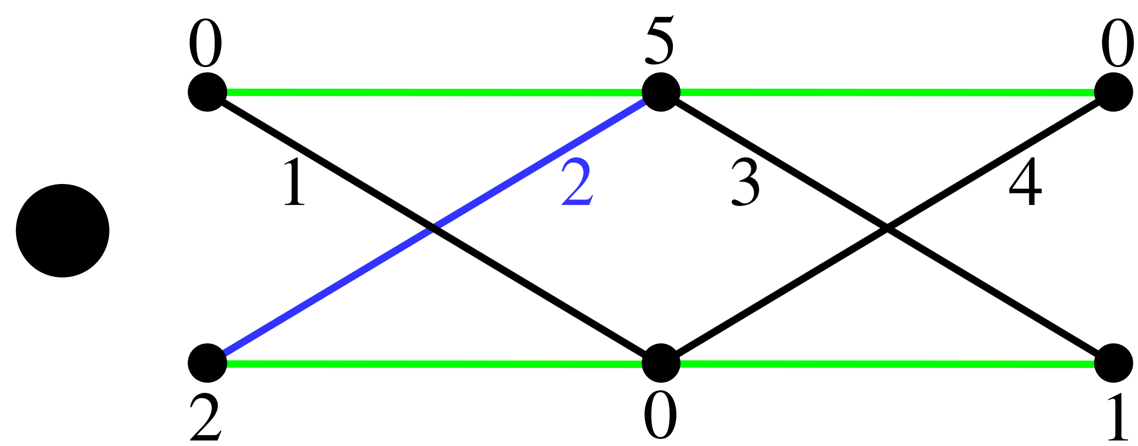
Maxflow algorithm and reparameterisation



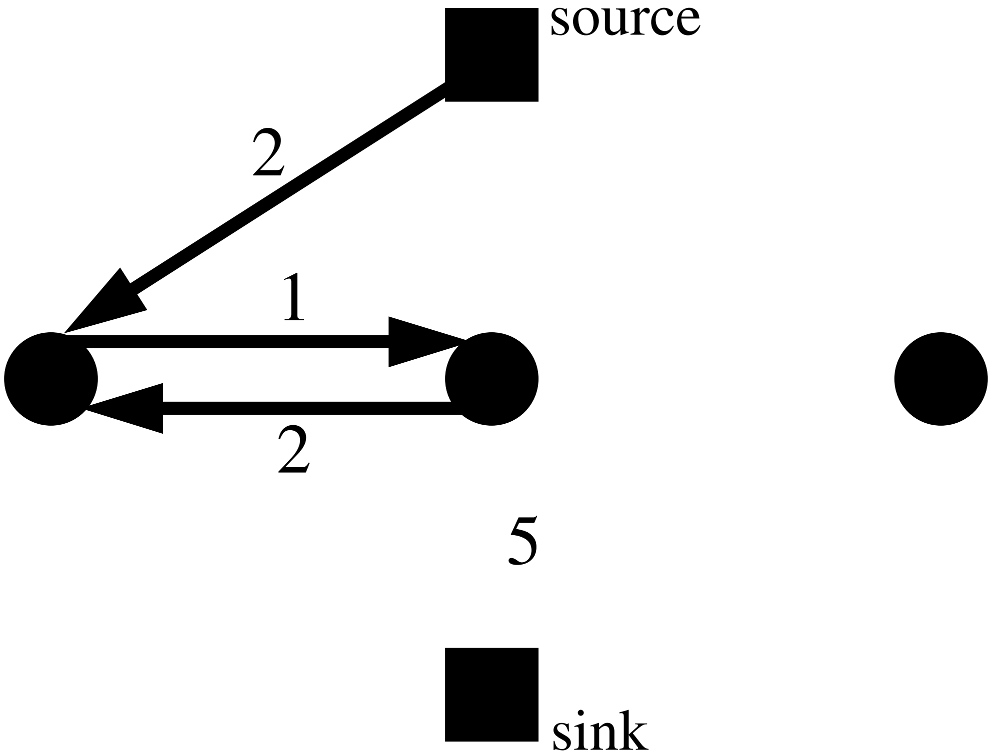


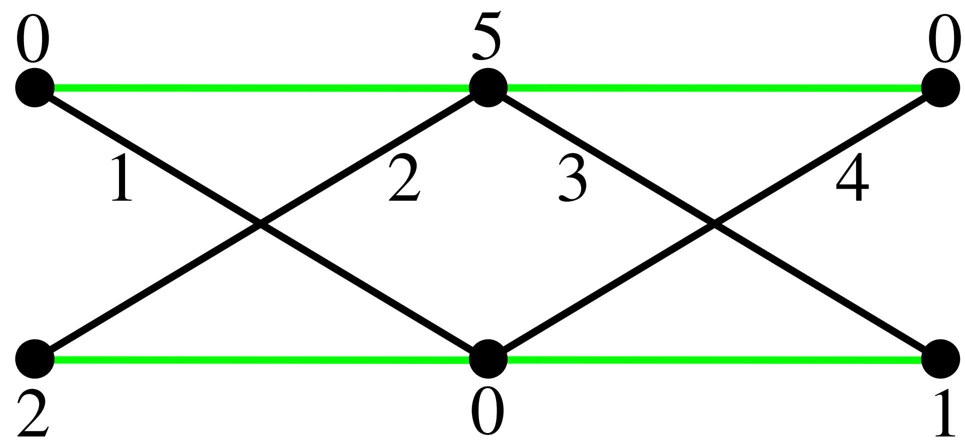
Maxflow algorithm and reparameterisation



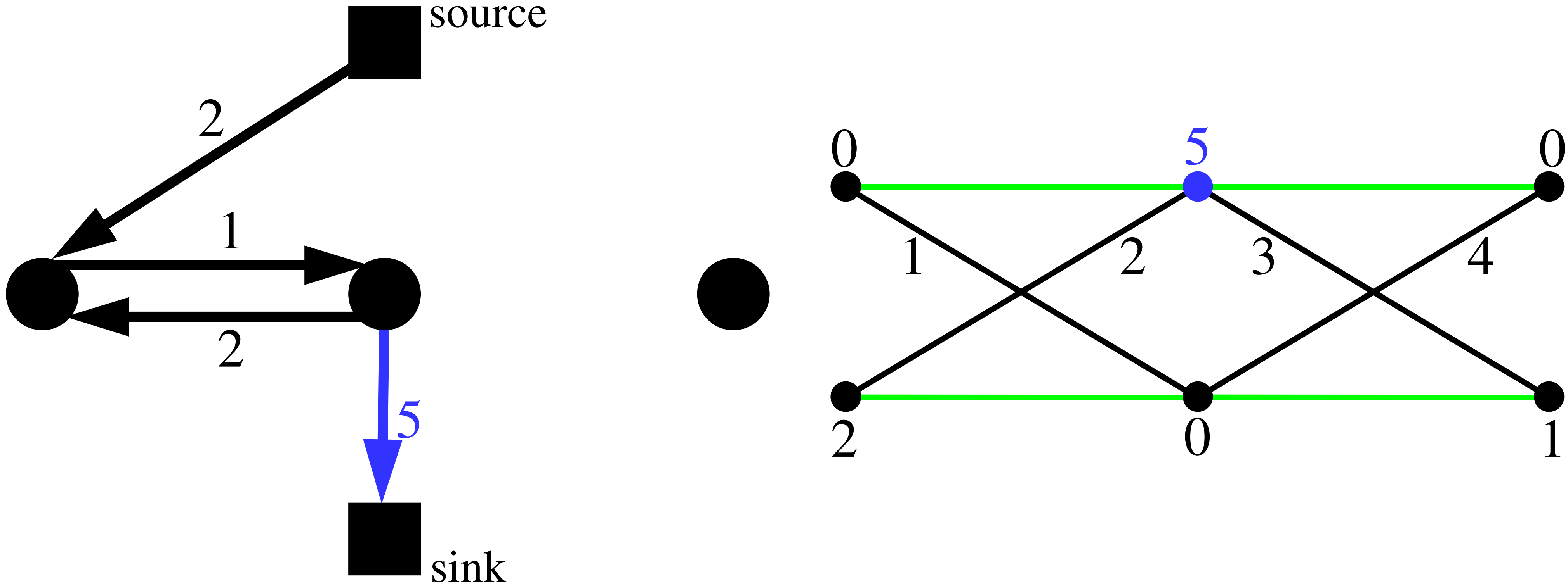


Maxflow algorithm and reparameterisation

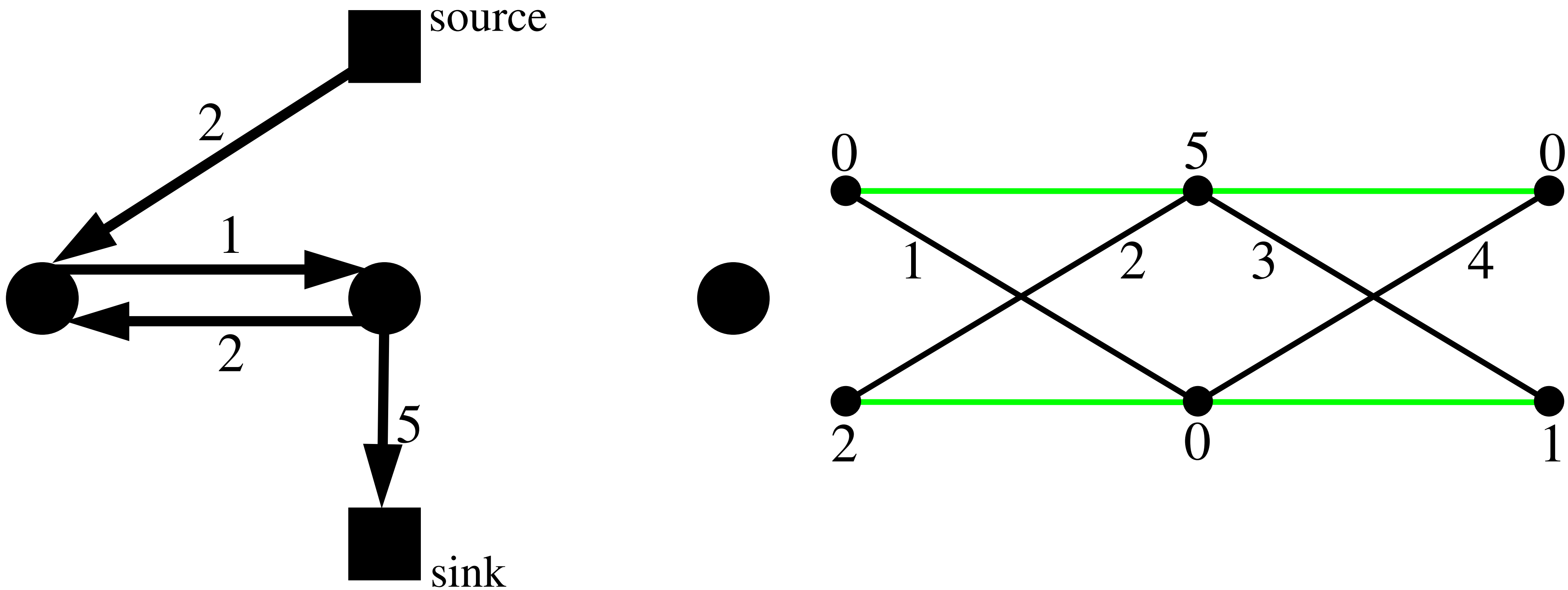




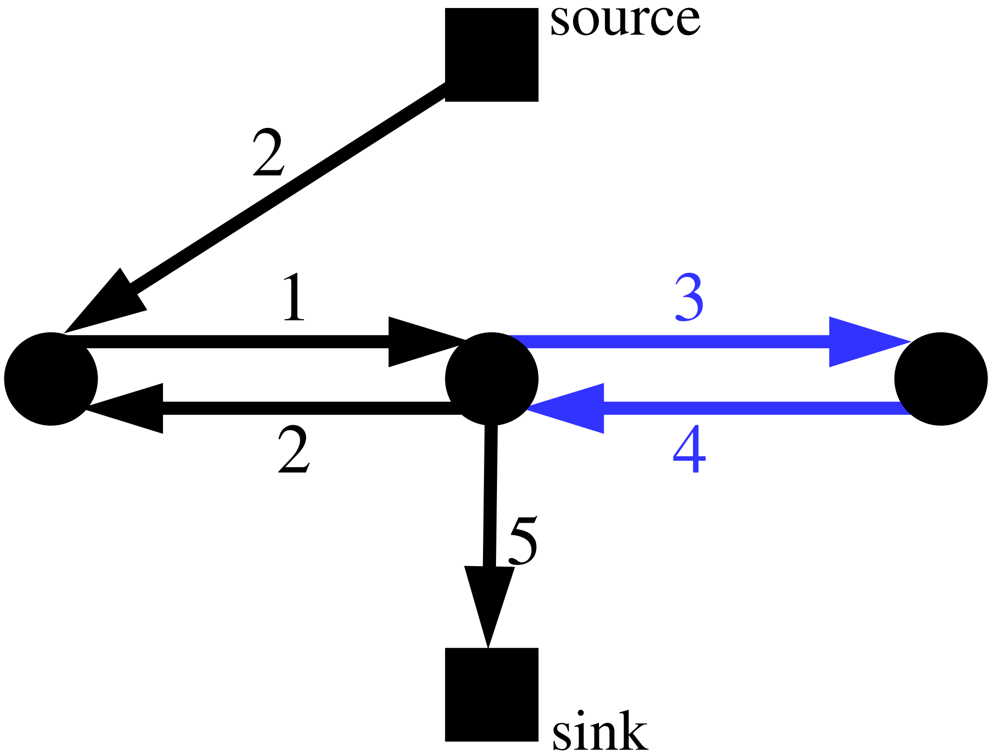
Maxflow algorithm and reparameterisation

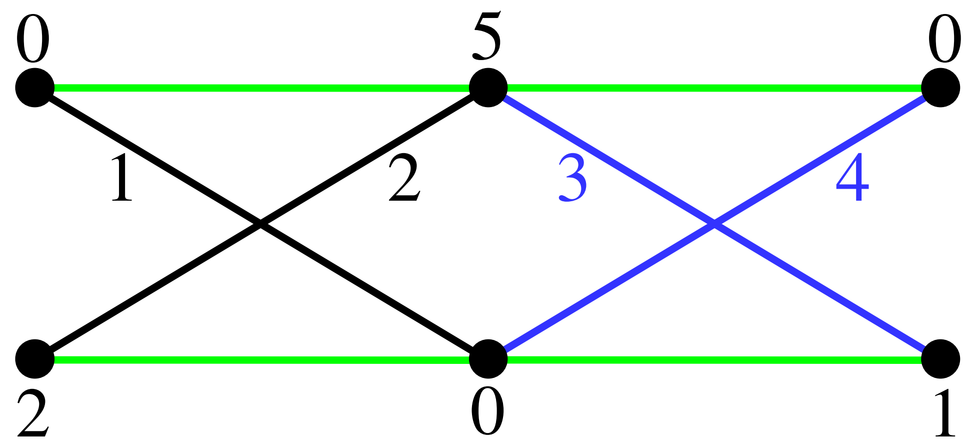


Maxflow algorithm and reparameterisation

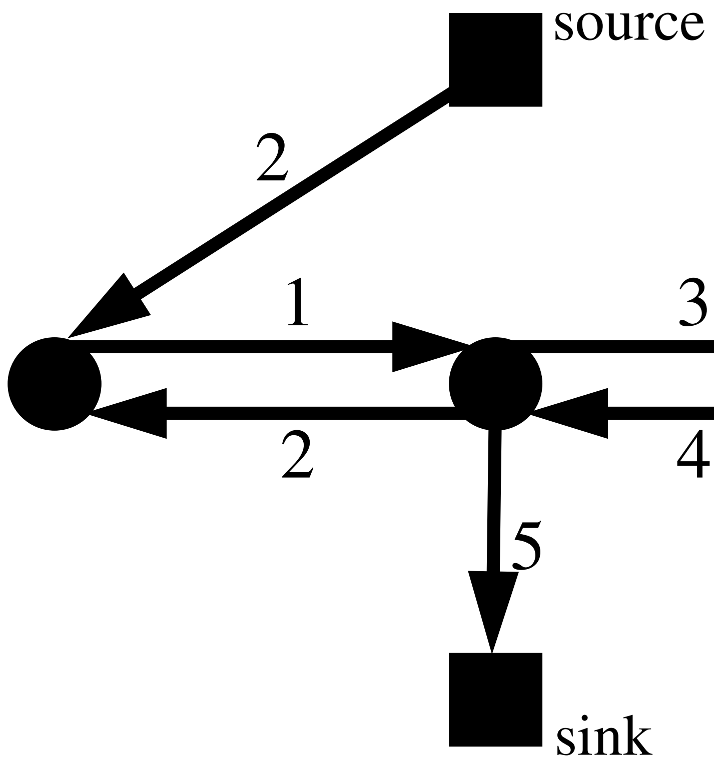


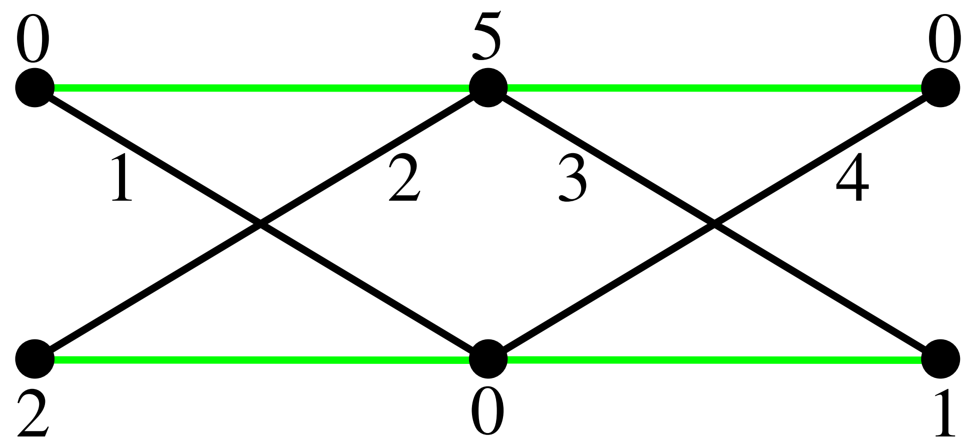
Maxflow algorithm and reparameterisation



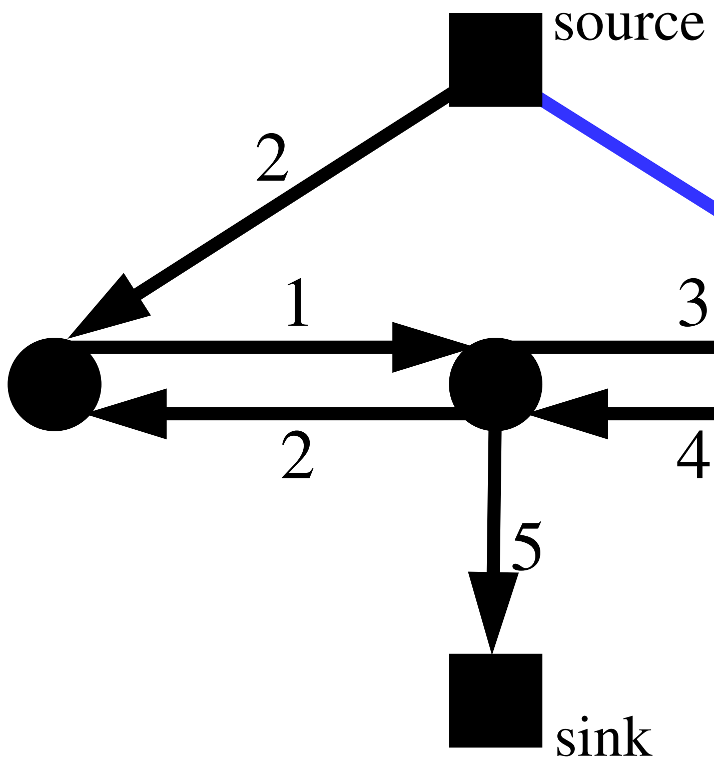


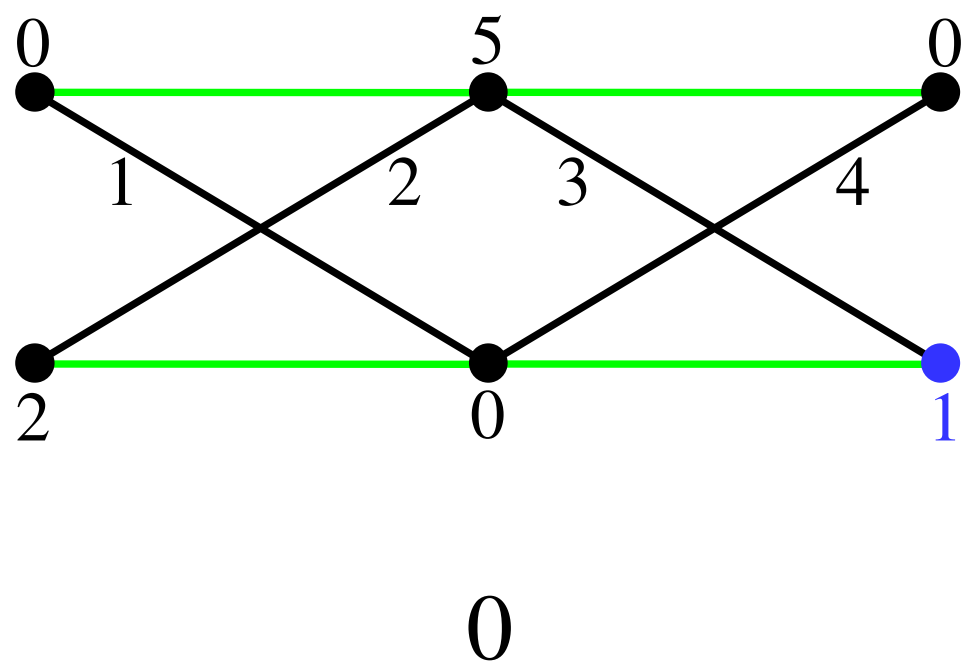
Maxflow algorithm and reparameterisation



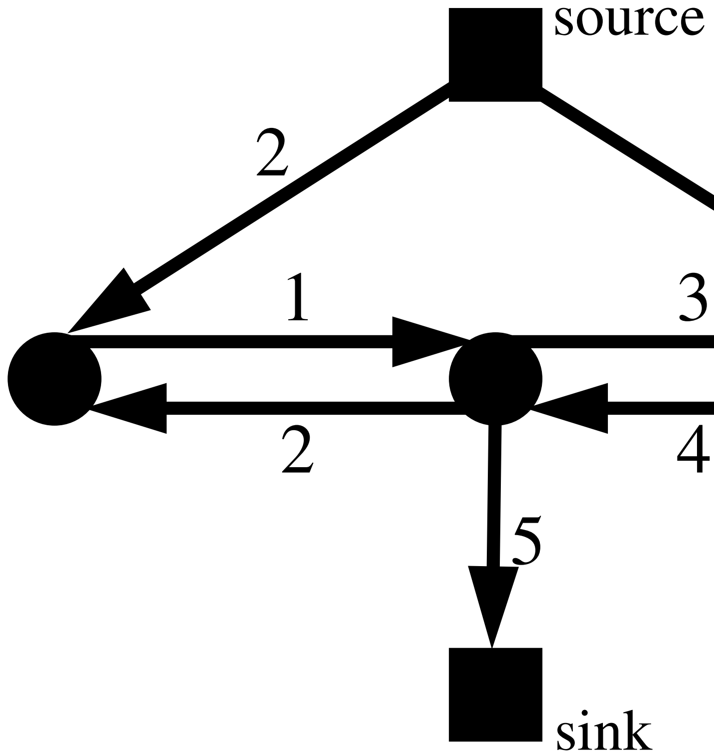


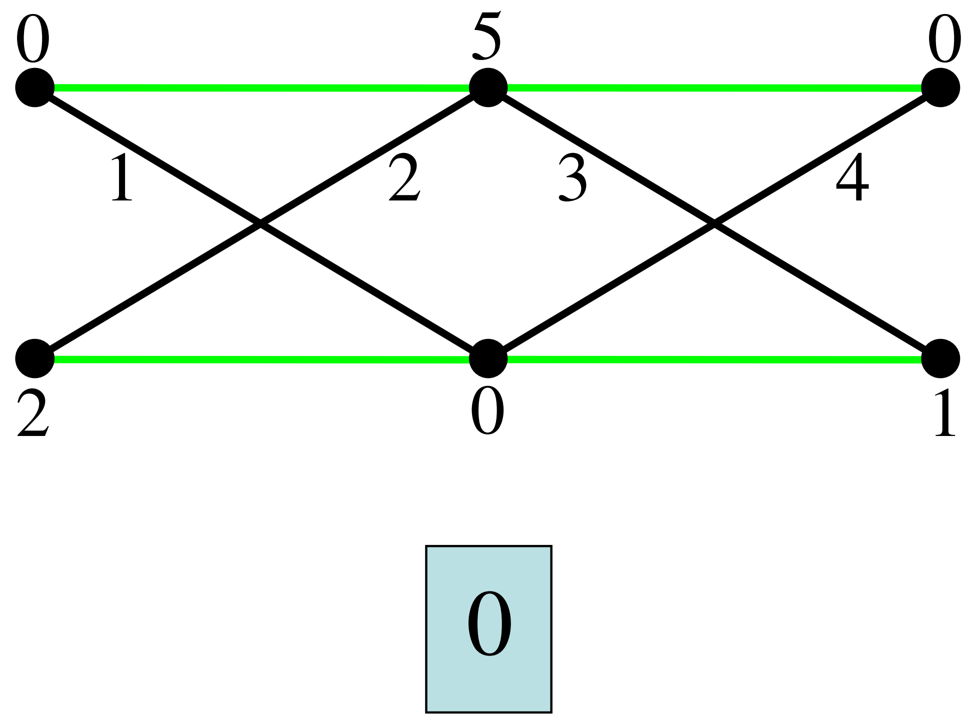
Maxflow algorithm and reparameterisation





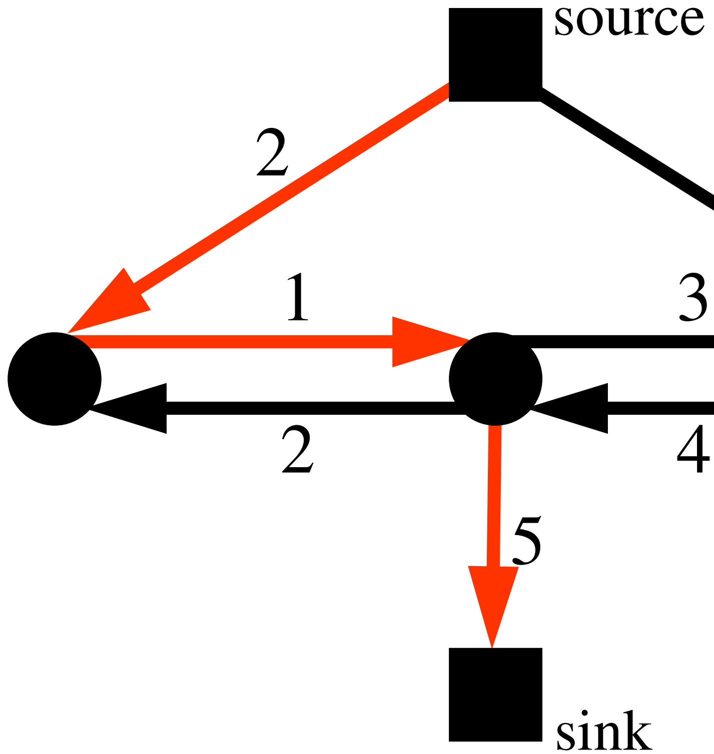
Maxflow algorithm and reparameterisation

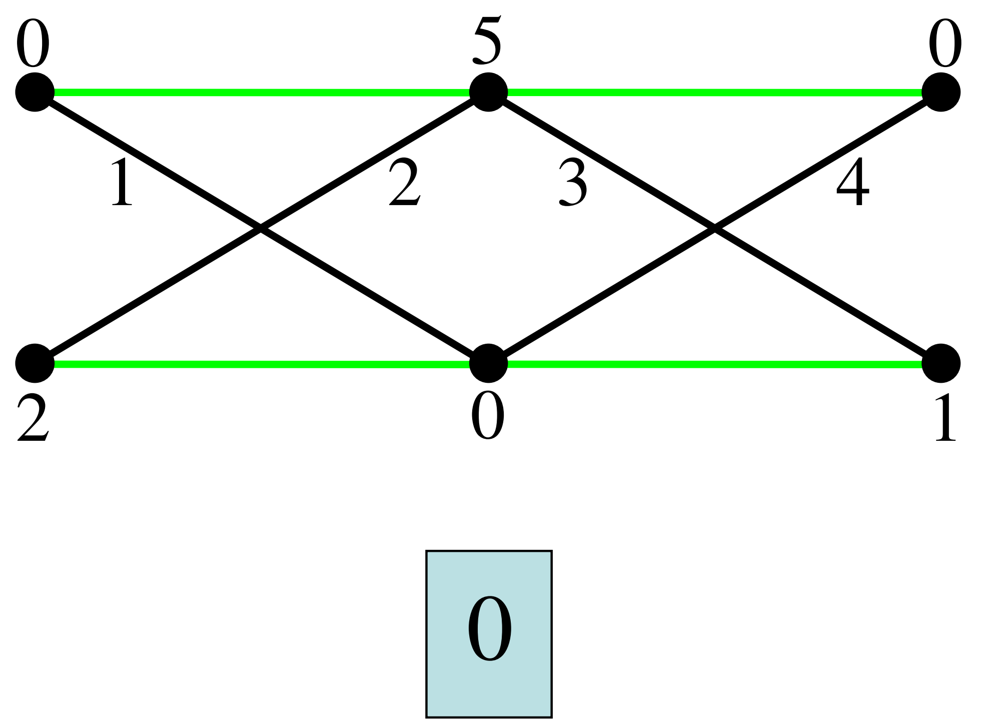




value(flow)=0

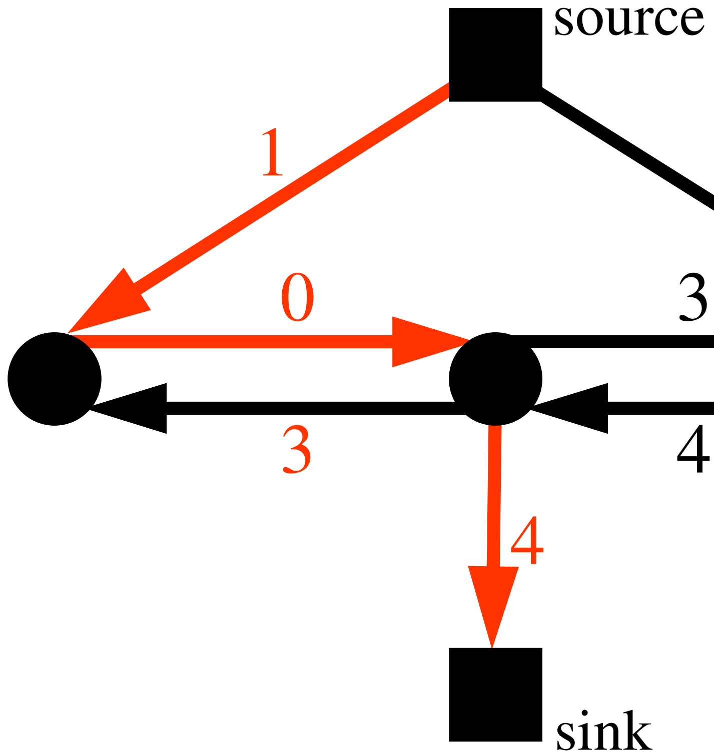
Maxflow algorithm
and reparameterisation

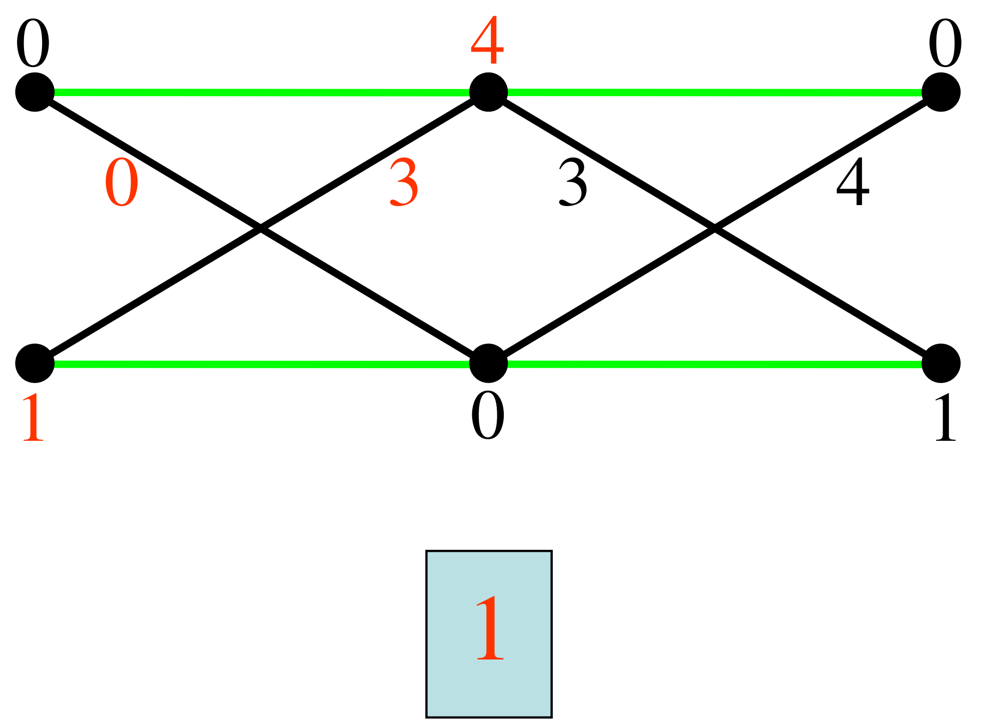




value(flow)=0

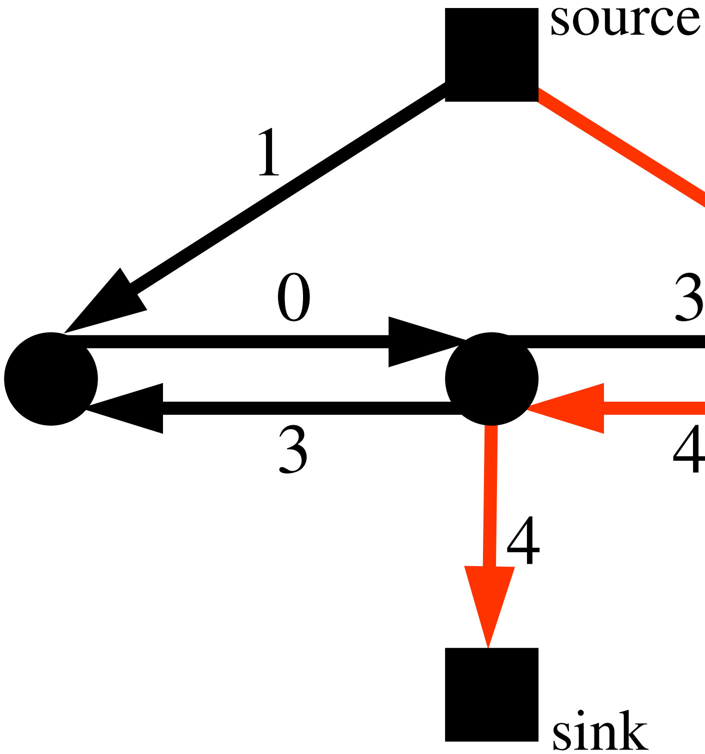
Maxflow algorithm and reparameterisation

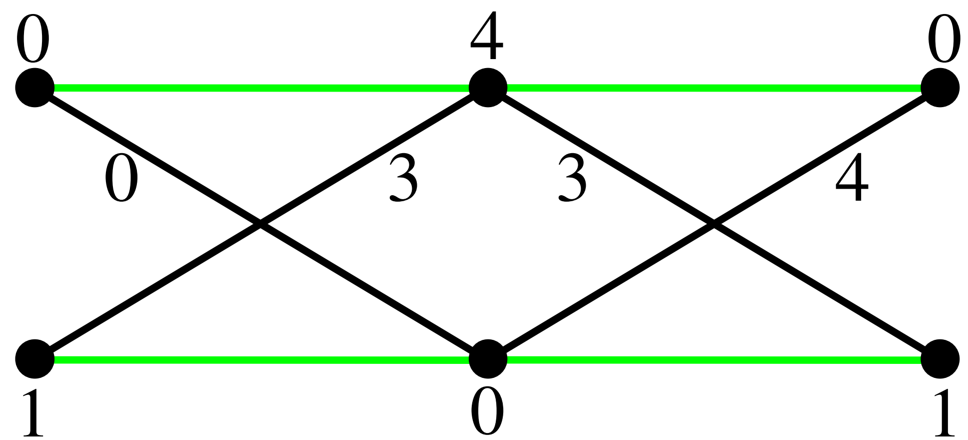




value(flow)=1

Maxflow algorithm and reparameterisation

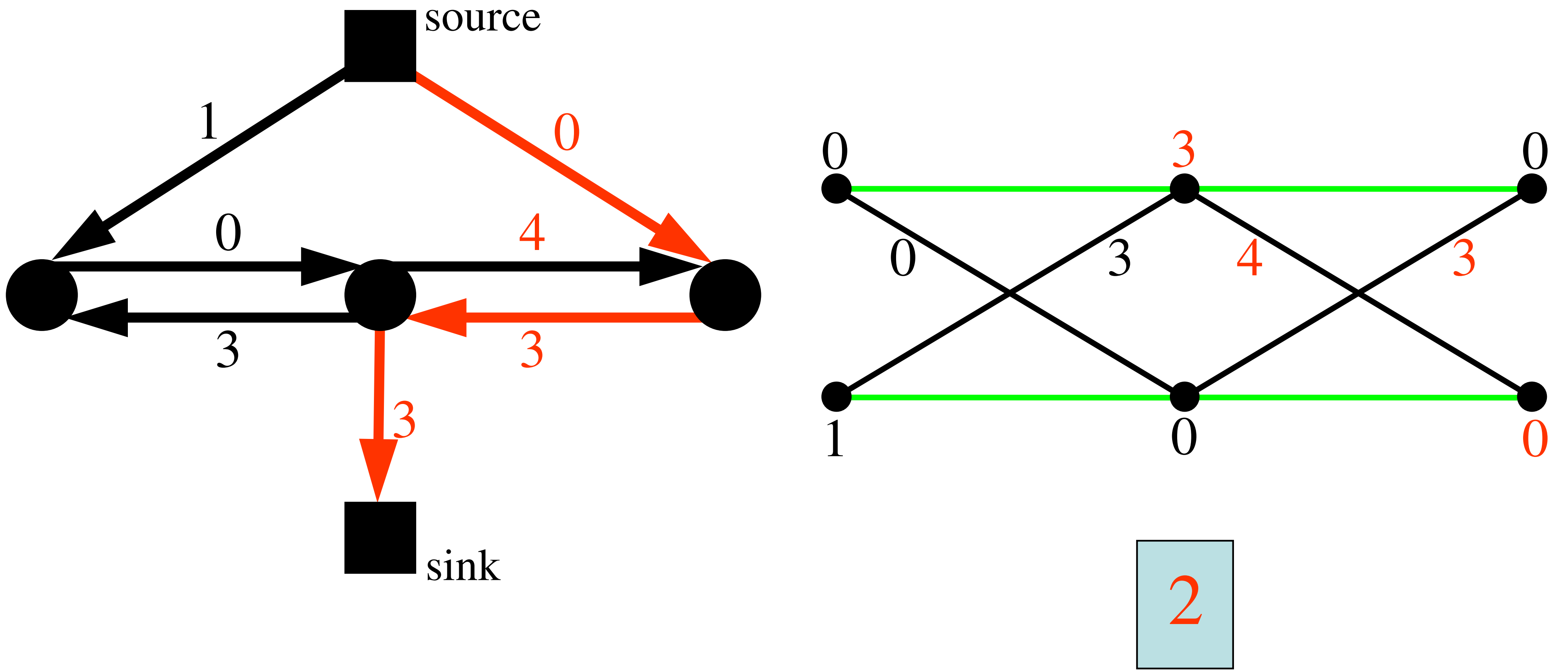




1

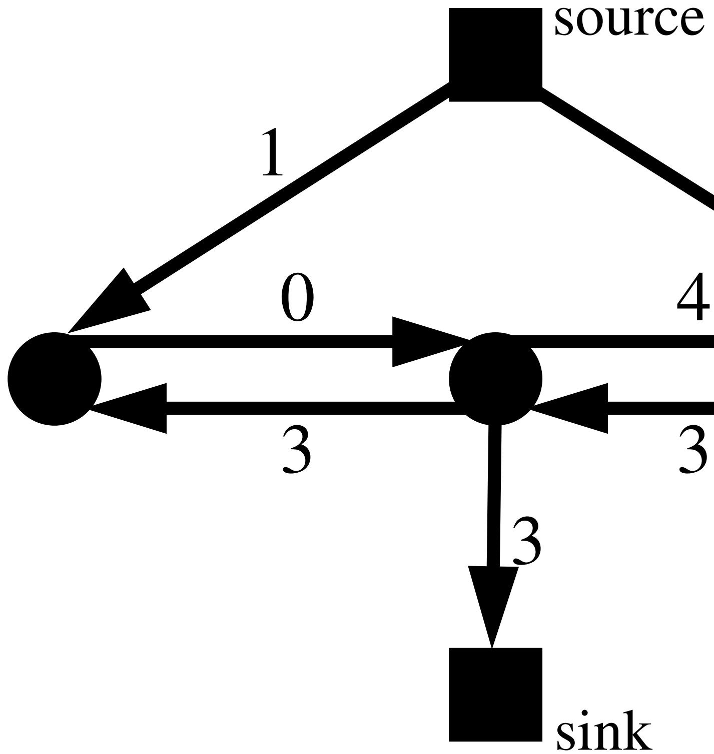
value(flow)=1

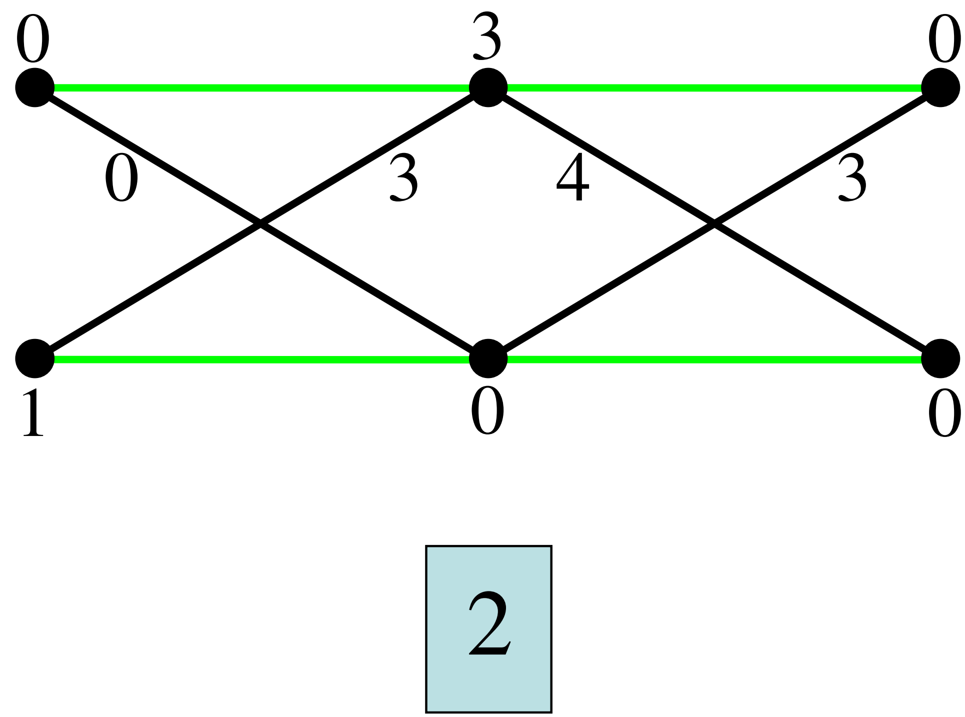
Maxflow algorithm and reparameterisation



value(flow)=2

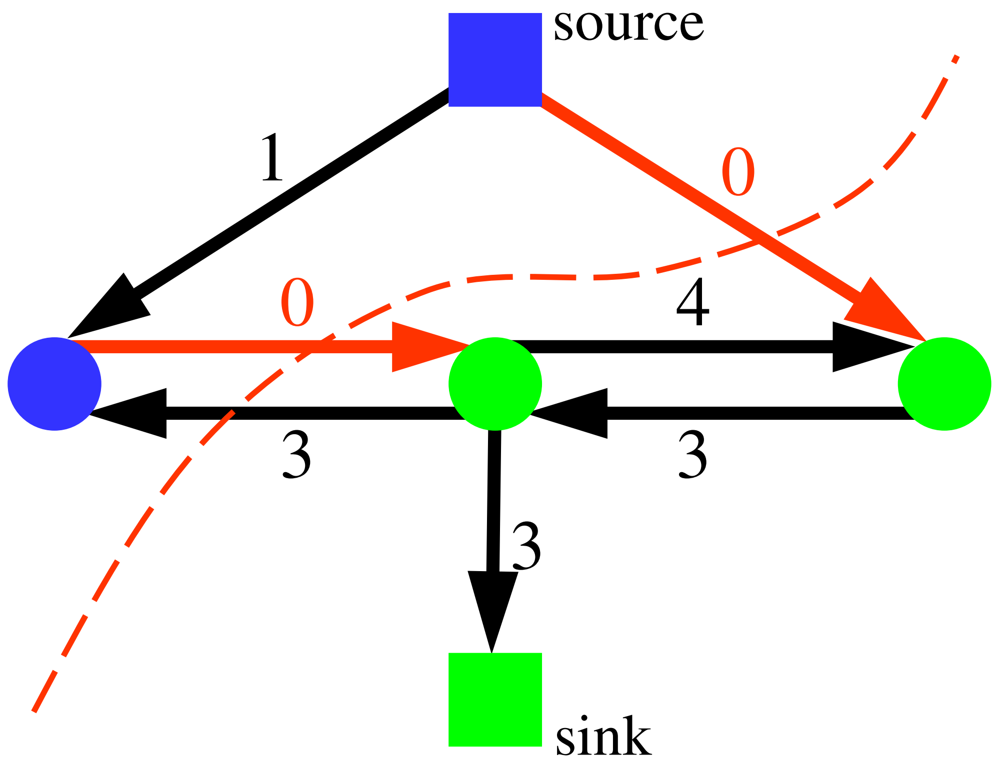
Maxflow algorithm and reparameterisation



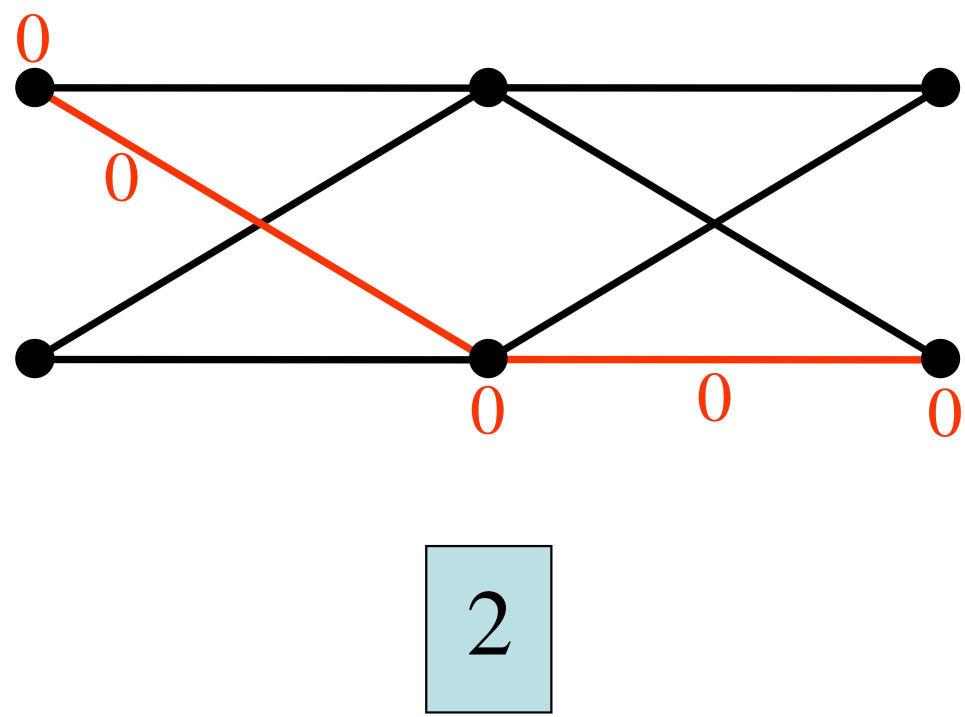


value(flow)=2

Maxflow algorithm and reparameterisation



value(flow)=2



minimum of the energy:

$$\mathbf{x} = (0,1,1)$$

Binary variables,
non-submodular functions

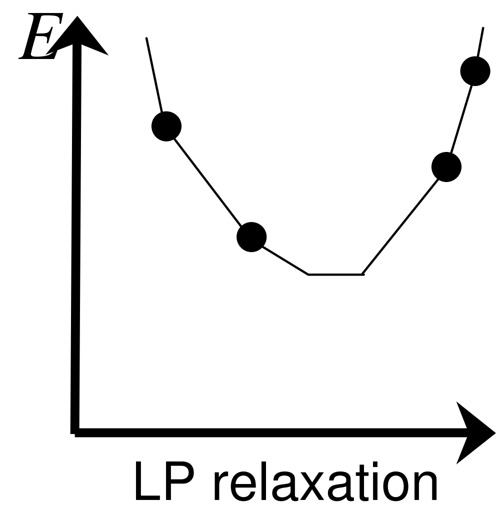
Arbitrary functions of binary variables

$$E(\mathbf{x} \mid \theta) = \theta_{const} + \sum_{p} \theta_p(x_p) + \sum_{p,q} \theta_{pq}(x_p, x_q)$$

maximize

non-negative

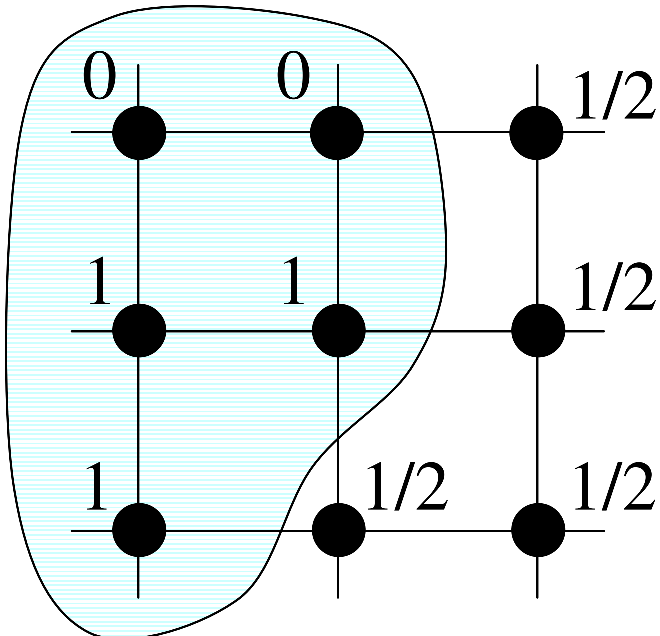
- Can be solved via maxflow [Hammer,Hansen,Simeone'84][Boros,Hammer,Sun'91]
  - Specially constructed graph



- Gives solution to LP relaxation: for each node

$$x_p \in \{0, 1/2, 1\}$$

Arbitrary functions of binary variables



Part of optimal solution
[Hammer, Hansen, Simeone'84]

Graph construction - Main idea

$$E(\{x_p\}) = \sum E_p(x_p)$$ — unary

$$+ \sum E_{pq}(x_p, x_q)$$ — pairwise submodular

$$+ \sum \tilde{E}_{pq}(x_p, x_q)$$ — pairwise non-submodular

- Double # of variables: $x_p \rightarrow x_p, x_{\bar{p}}$
  - Ideally, $x_{\bar{p}} = 1 - x_p$
- Write *E* as a function of both old and new variables
  - New function is submodular!

Graph construction - Main idea

$$E(\{x_p\}) = \sum E_p(x_p) + \sum E_{pq}(x_p, x_q) + \sum \tilde{E}_{pq}(x_p, x_q)$$

$$E(\{x_p\},\{x_{\overline{p}}\}) = \sum \frac{E_p(x_p) + E_p(1 - x_{\overline{p}})}{2} + \sum \frac{E_{pq}(x_p, x_p) + E_p(1 - x_{\overline{p}}, 1 - x_{\overline{q}})}{2} + \sum \frac{\tilde{E}_{pq}(x_p, 1 - x_{\overline{q}}) + \tilde{E}_p(1 - x_{\overline{p}}, x_q)}{2}$$

- Double # of variables: $x_p \rightarrow x_p, x_{\overline{p}}$
  - Ideally, $x_{\overline{p}} = 1 - x_p$
- Write *E* as a function of both old and new variables
  - New function is submodular!

Graph construction - Main idea

$$E(\{x_p\}) = \sum E_p(x_p)$$

$$E(\{x_p\},\{x_{\bar{p}}\}) = \sum \frac{E_p(x_p) + E_p(1 - x_{\bar{p}})}{2}$$

- Double # of variables: $x_p \rightarrow x_p, x_{\bar{p}}$
  - Ideally, $x_{\bar{p}} = 1 - x_p$
- Write *E* as a function of both old and new variables
  - New function is submodular!

Graph construction - Main idea

$$+\sum E_{pq}(x_p, x_q)$$

$$+\sum \frac{E_{pq}(x_p, x_p) + E_p(1 - x_{\overline{p}}, 1 - x_{\overline{q}})}{2}$$

- Double # of variables: $x_p \rightarrow x_p, x_{\overline{p}}$
  - Ideally, $x_{\overline{p}} = 1 - x_p$
- Write *E* as a function of both old and new variables
  - New function is submodular!

Graph construction - Main idea

$$+\sum \tilde{E}_{pq}(x_p, x_q)$$

$$+\sum \frac{\tilde{E}_{pq}(x_p, 1-x_{\bar{q}}) + \tilde{E}_p(1-x_{\bar{p}}, x_q)}{2}$$

- Double # of variables: $x_p \rightarrow x_p, x_{\bar{p}}$
  - Ideally, $x_{\bar{p}} = 1 - x_p$
- Write E as a function of both old and new variables
  - New function is submodular!

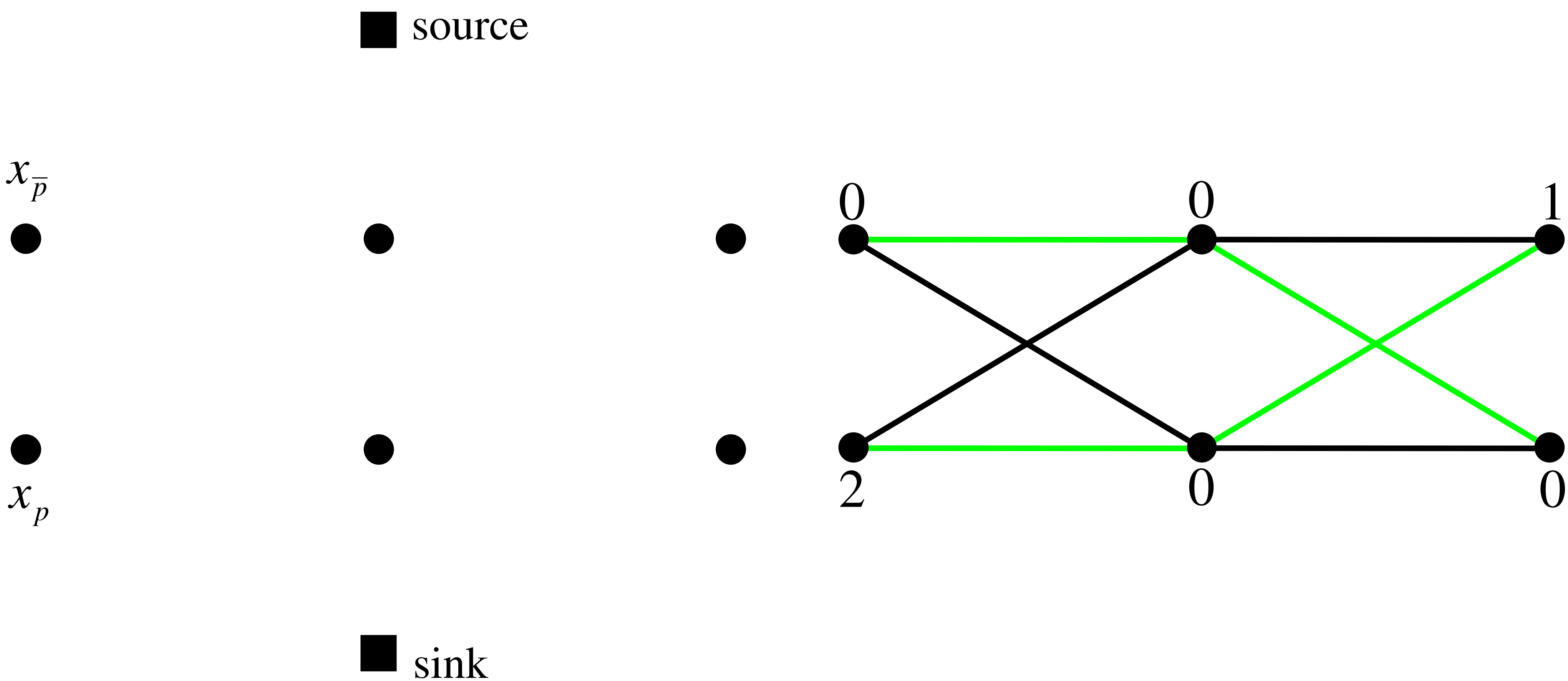
Graph construction - Main idea

$$E(\{x_p\}) = \sum E_p(x_p) + \sum E_{pq}(x_p, x_q) + \sum \tilde{E}_{pq}(x_p, x_q)$$

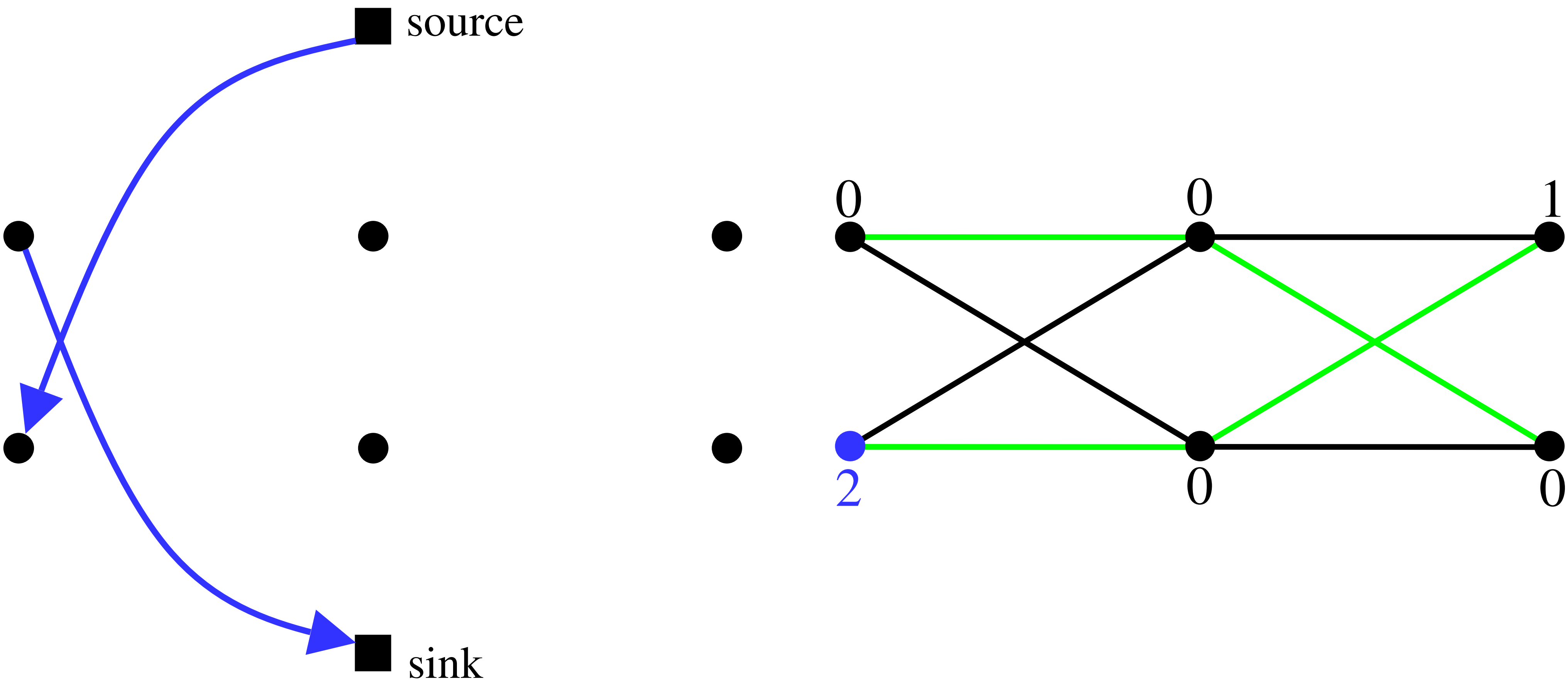
$$E(\{x_p\},\{x_{\bar{p}}\}) = \sum \frac{E_p(x_p) + E_p(1 - x_{\bar{p}})}{2} + \sum \frac{E_{pq}(x_p, x_p) + E_p(1 - x_{\bar{p}}, 1 - x_{\bar{q}})}{2} + \sum \frac{\tilde{E}_{pq}(x_p, 1 - x_{\bar{q}}) + \tilde{E}_p(1 - x_{\bar{p}}, x_q)}{2}$$

- Minimise new function $E(\{x_p\},\{x_{\bar{p}}\})$
  - Without constraint $x_{\bar{p}} = 1 - x_p$

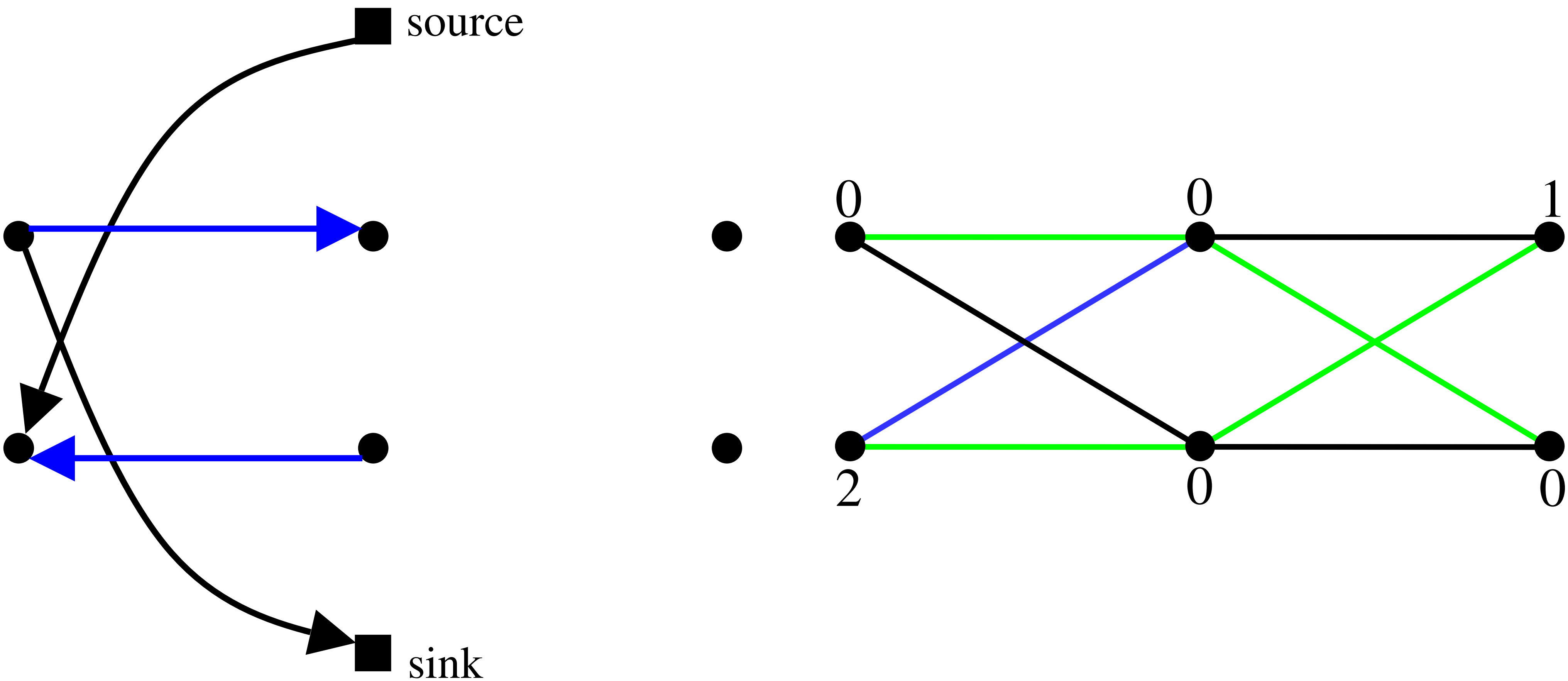
Graph construction



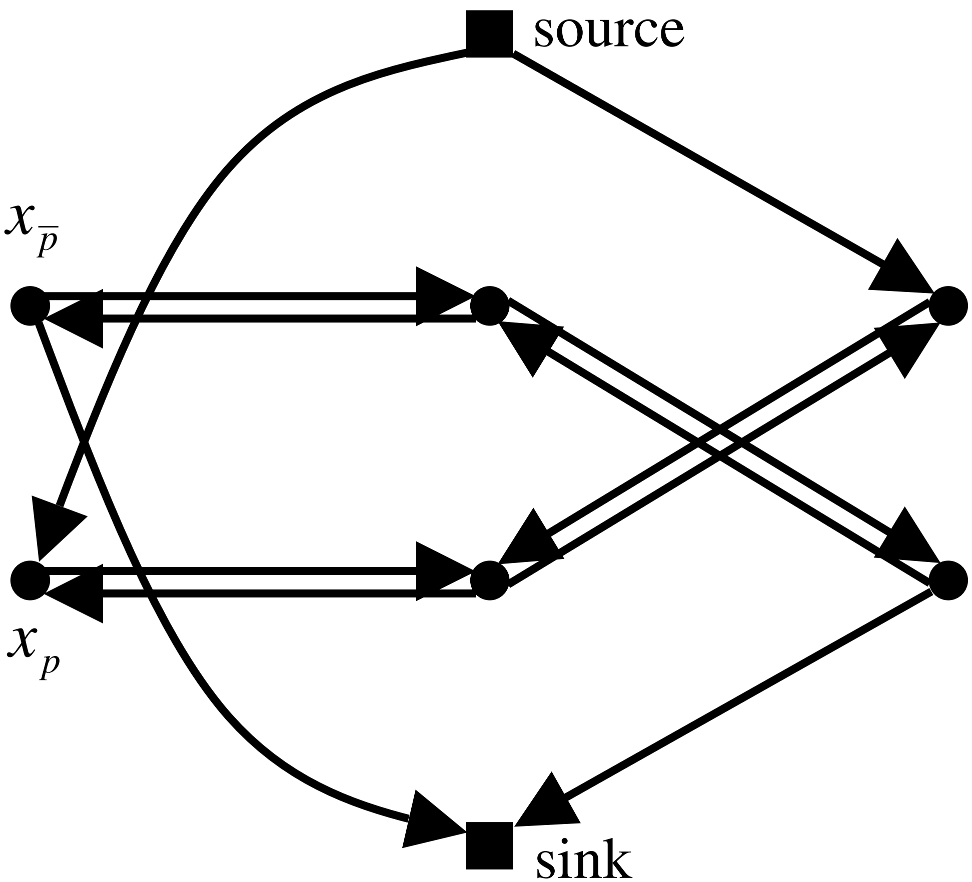
Graph construction

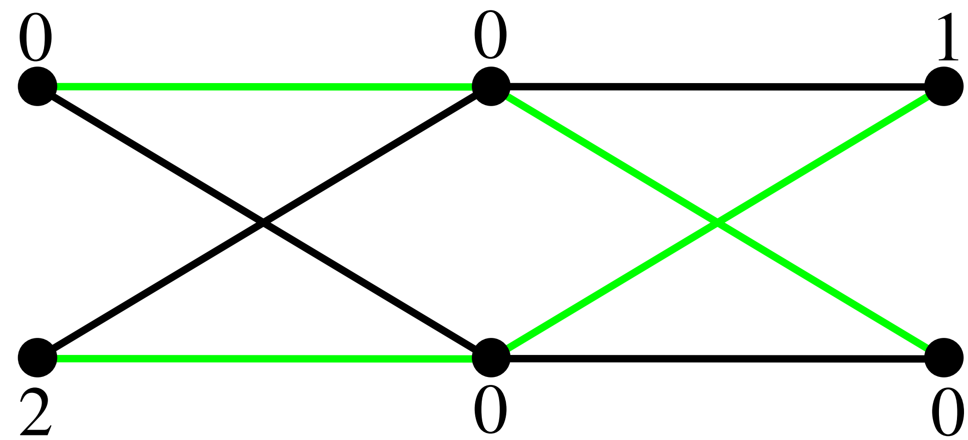


Graph construction



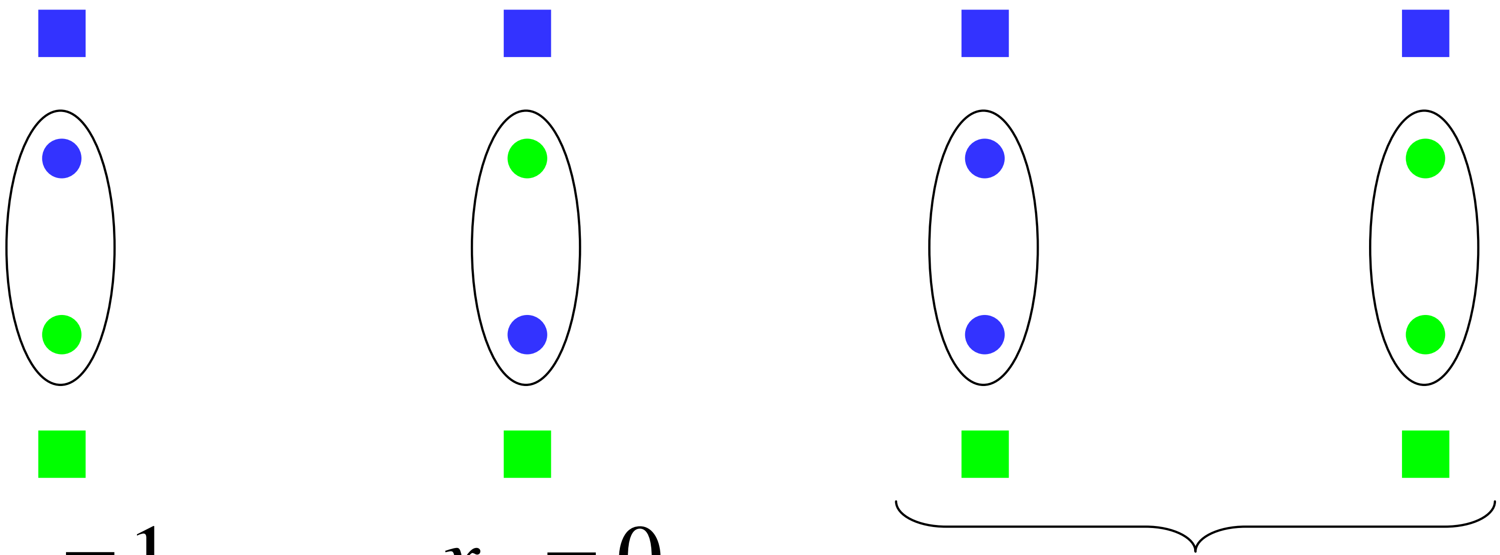
Graph construction





Assigning labels

- Assign labels based on minimum cut in auxiliary graph



$x_p = 1$

$x_{\bar{p}} = 0$

$x_p = 0$

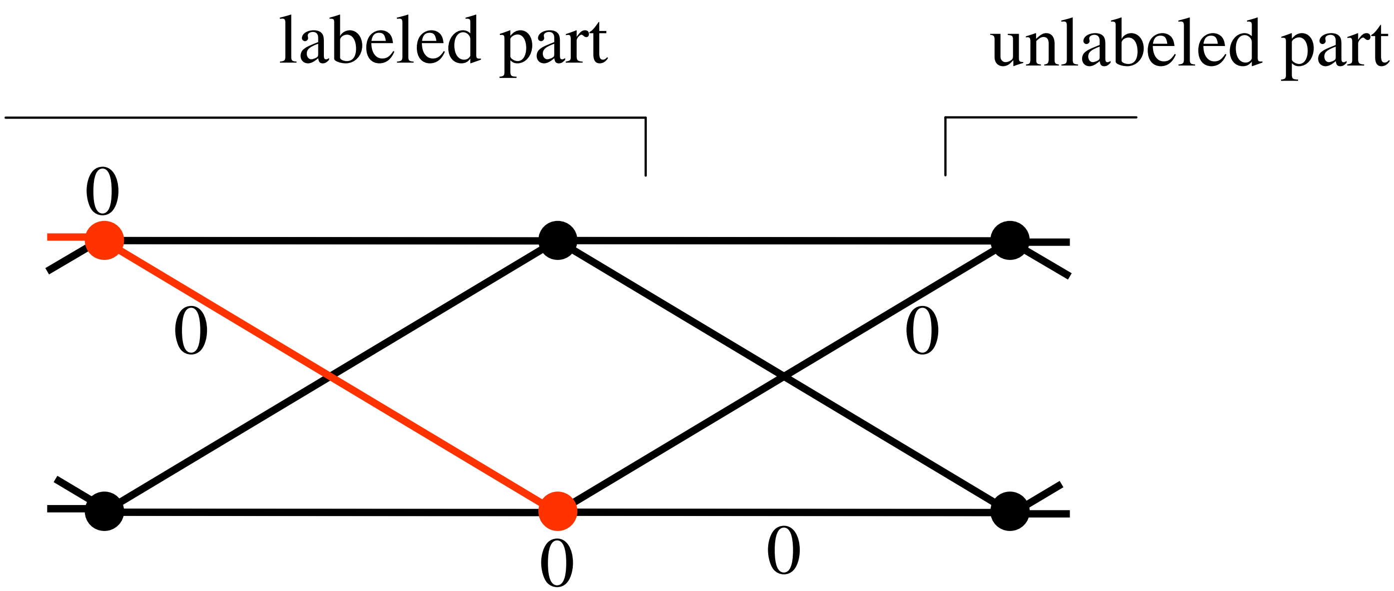
$x_{\bar{p}} = 1$

node p is unlabeled

- To maximize # of labeled nodes, choose a particular minimum cut

Optimality

Theorem [Hammer,Hansen,Simeone'84].
Labeling **x** is part of optimal labeling $\mathbf{x}^*$.



$x_p = 0$ $x_p = 1$ $x_p = ?$

Part B: Lower bound via convex combination of trees

($\Rightarrow$ tree-reweighted message passing)

Convex combination of trees [Wainwright, Jaakkola, Willsky '02]

- Goal: compute minimum of the energy for θ:

$$\Phi(\theta) = \min_{\mathbf{x}} E(\mathbf{x} \mid \theta)$$

- In general, intractable!

- Obtaining lower bound:
  - Split θ into several components: $\theta = \theta^1 + \theta^2 + \ldots$
  - Compute minimum for each component:

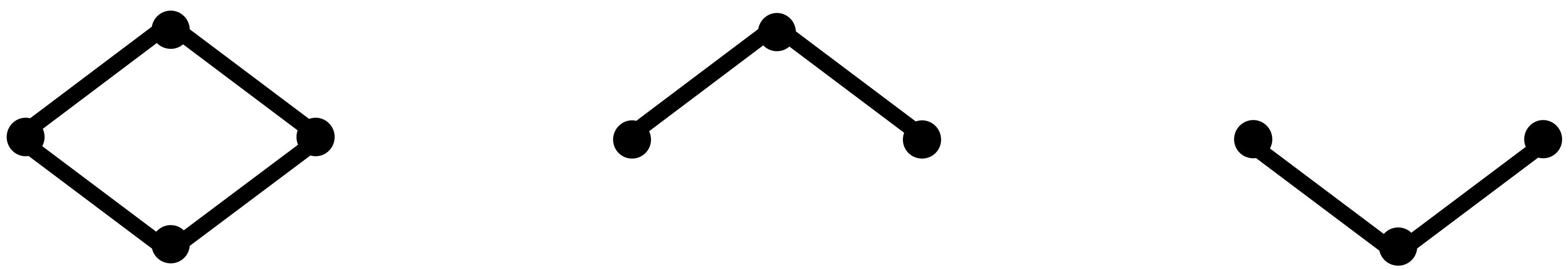
$$\Phi(\theta^i) = \min_{\mathbf{x}} E(\mathbf{x} \mid \theta^i)$$

  - Combine $\Phi(\theta^1)$, $\Phi(\theta^2)$, ... to get a bound on $\Phi(\theta)$

- Use trees!

Convex combination of trees (cont'd)

graph — tree T — tree T'



$$\theta \equiv \frac{1}{2}\theta^{T} + \frac{1}{2}\theta^{T'}$$

$$\Phi(\theta) \geq \underbrace{\frac{1}{2}\Phi(\theta^{T}) + \frac{1}{2}\Phi(\theta^{T'})}_{\text{lower bound on the energy}}$$

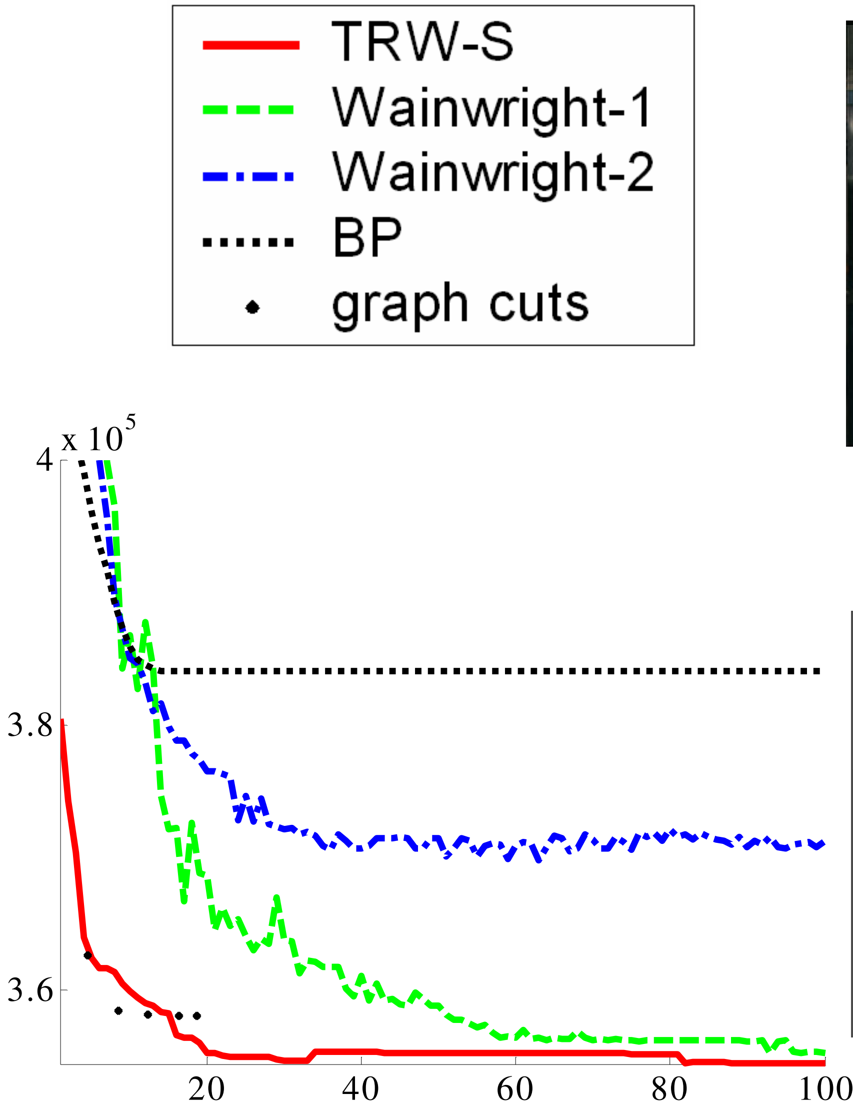
maximize

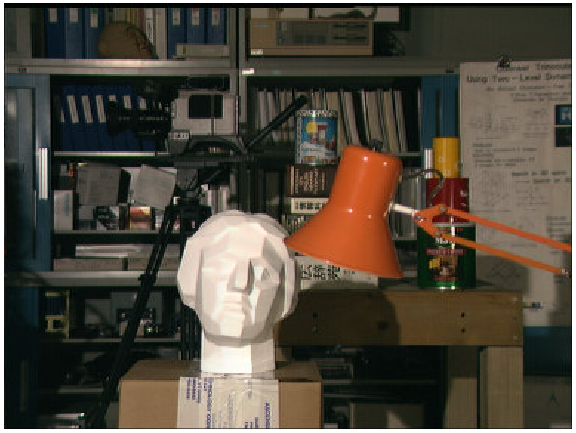
lower bound on the energy

TRW algorithms

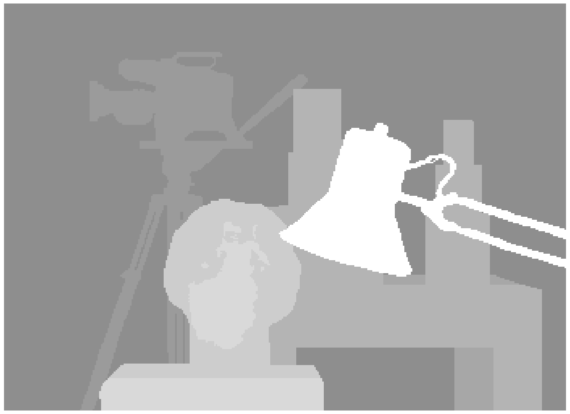
- Goal: find reparameterisation maximizing lower bound
- Apply sequence of different reparameterisation operations:
  - Node averaging
  - Ordinary BP on trees
- Order of operations?
  - Affects performance dramatically
- Algorithms:
  - [Wainwright *et al*. '02]: parallel schedule
    - May not converge
  - [Kolmogorov'05]: specific sequential schedule
    - Lower bound does not decrease, convergence guarantees
    - Needs half the memory

Experimental results: stereo

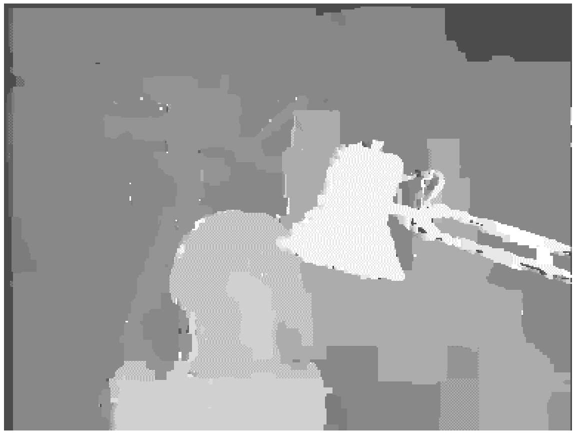




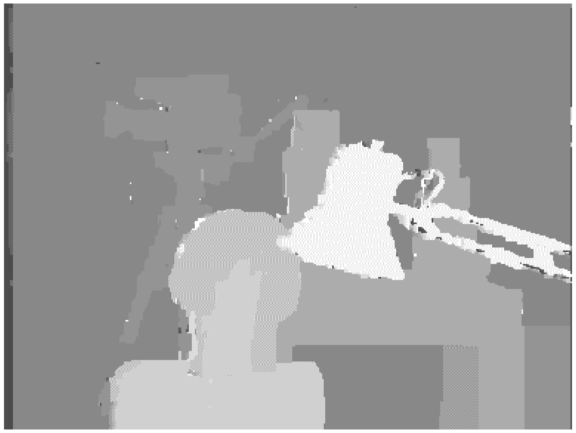
left image



ground truth



BP



TRW-S

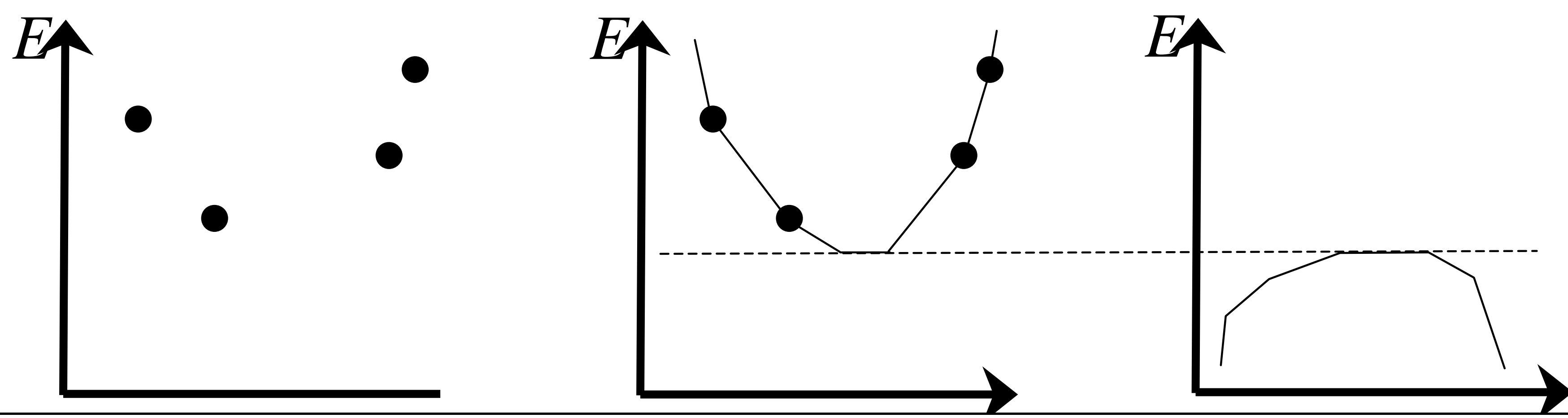
- Global minima for some instances with TRW [Meltzer,Yanover,Weiss'05]

Parts A and B: Summary

- MAP estimation algorithms are based on LP relaxation
  - Maximize lower bound
- Two ways to formulate lower bound
- Via posiforms: leads to maxflow algorithm (for binary variables)
  - Polynomial time solution
  - Submodular functions: global minimum
  - Non-submodular functions: part of optimal solution
- Via convex combination of trees: leads to TRW algorithm
  - Convergence in the limit (for TRW-S)
  - Applicable to arbitrary energy function

Non-binary variables: Other methods for solving LP

- No polynomial-time algorithm (except general purpose LP solvers)
- Iterative methods:
  - [Koval,Schlesinger'76]: *augmenting DAG algorithm*
  - [Kovalevsky,Koval'75, Flach'98] (unpublished): *max-sum diffusion*
    - See tech. report [Werner'05]
    - Not guaranteed to solve LP (only *arc constistent* solution) – same as TRW
- Special case: *submodular functions*
  - LP has integer optimal solution [Schlesinger,Flach'00]
  - Reduction to maxflow [Ishikawa'03, D.Schlesinger'05]



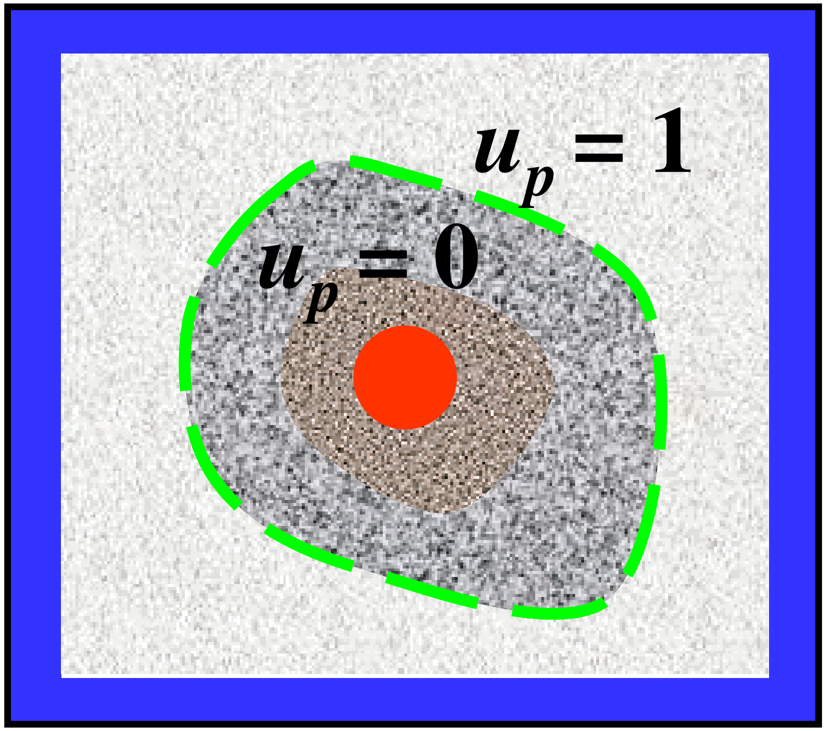
Continuous mincut/maxflow

Continuous mincut/maxflow

- Primal problem:

$$\int_C g(C(s))\, ds \rightarrow \min$$

subject to $\begin{cases} s \text{ inside } C \\ t \text{ outside } C \end{cases}$



Alternatively:

$$\int |\nabla u|_g \rightarrow \min$$

subject to $\begin{cases} u_p = 0,\ p \in s \\ u_p = 1,\ p \in t \end{cases}$

<u>total variation</u>

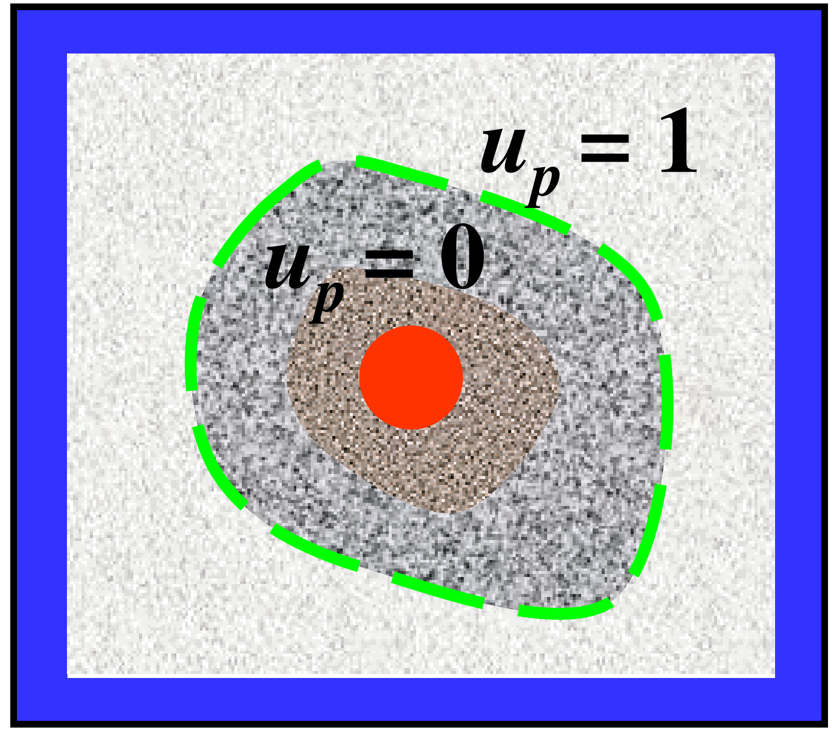
[Rudin,Osher,Fatemi'92] :
image restoration
[Amar,Belletini'94]:
definition for arbitrary metric

Continuous mincut/maxflow

- Primal problem:

$$\int_C g(C(s))\, ds \to \min$$

subject to $\begin{cases} s \text{ inside } C \\ t \text{ outside } C \end{cases}$



Alternatively:

$$\int |\nabla u|_g \to \min$$

subject to $\begin{cases} u_p = 0, p \in s \\ u_p = 1, p \in t \end{cases}$

- $u_p \in [0,1]$ – *fractional segmentations*
- Convex problem
- Integer optimal solution

Continuous mincut/maxflow

- Dual problem:

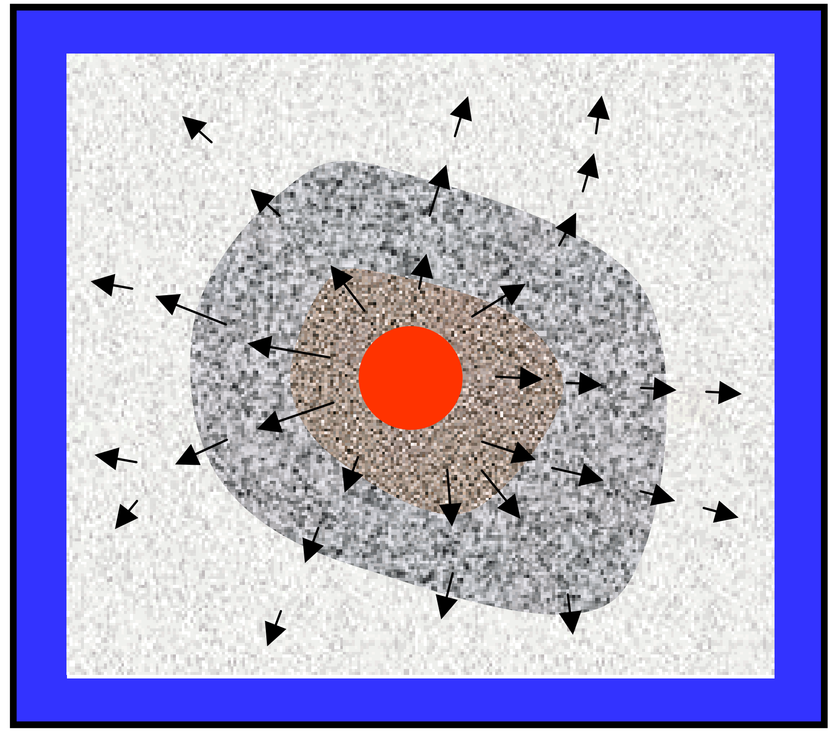
$$\int_{S} (\operatorname{div} \vec{f}_p)\, da \rightarrow \max$$

subject to

$$|\vec{f}_p| \leq g$$ *(capacity constraint)*

$$\operatorname{div} \vec{f}_p = 0$$ *(flow conservation)*

for $p \notin s,t$



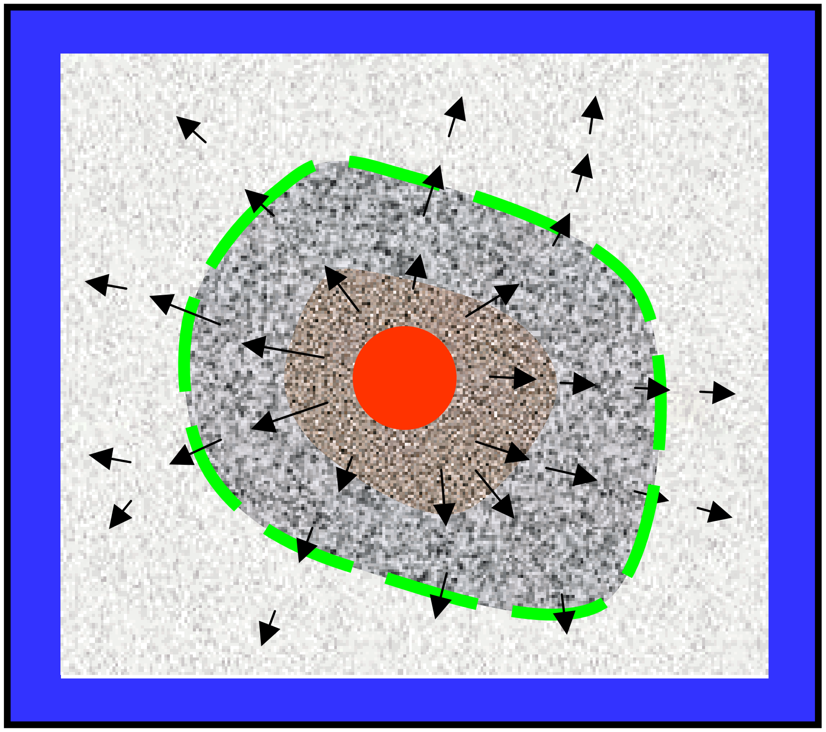
Reparameterisation

- Any flow with

$$\text{div}\,\vec{f}_p = 0 \quad \text{for} \quad p \notin s,t$$

defines reparameterisation

(*by the divergence theorem*):



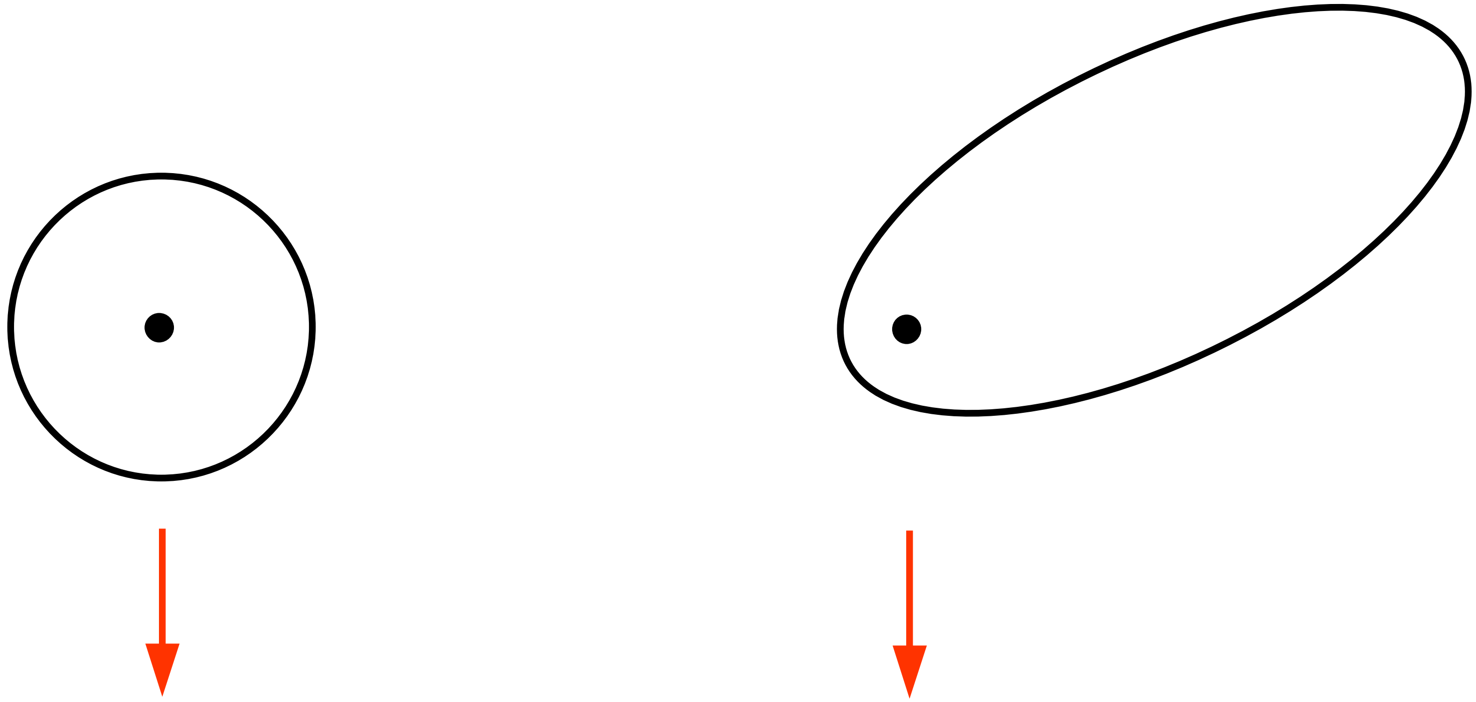
$$E(C) \equiv \int_C g\,ds = const + \int_C (g - \vec{f} \cdot \vec{N})\,ds$$

where

$$const = \int_s (\text{div}\,\vec{f})\,ds$$

Reparameterisation

distance maps



$$E(C) \equiv \int_C g \, ds = const + \int_C (g - \vec{f} \cdot \vec{N}) \, ds$$

lower bound on $E(C)$

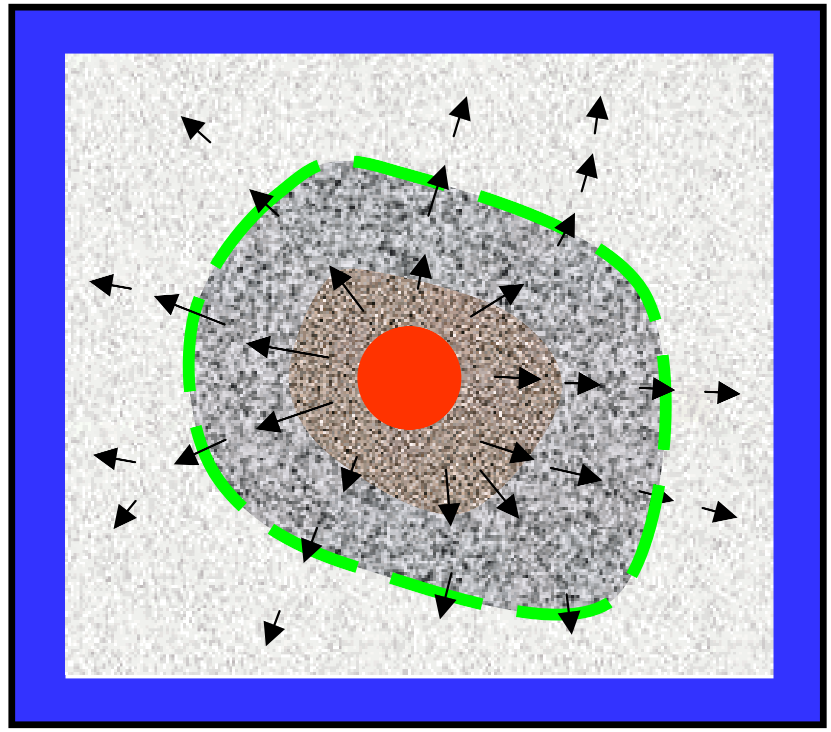
$| \vec{f} | \leq g \;\Rightarrow$ non-negative

Reparameterisation

Suppose flow saturates cut C^*

$(\vec{f}_p = g_p \vec{N} \quad \text{for } p \in C^*)$:

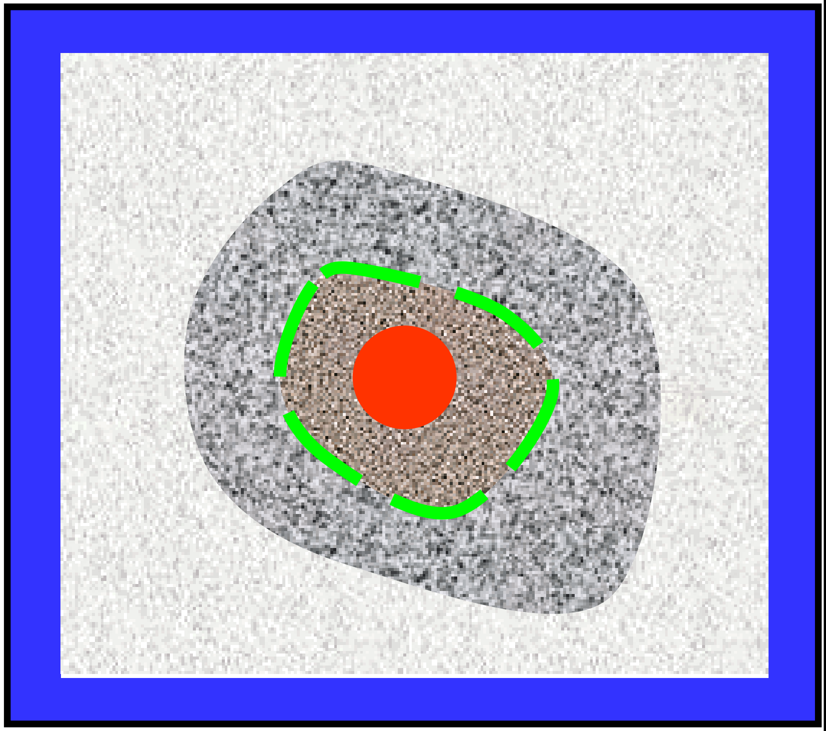
$\Rightarrow \; C^* = \text{minimum cut}$



$$E(C) \equiv \int_C g\, ds = const + \underbrace{\int_C (g - \vec{f} \cdot \vec{N})\, ds}_{\text{zero for } C^*}$$

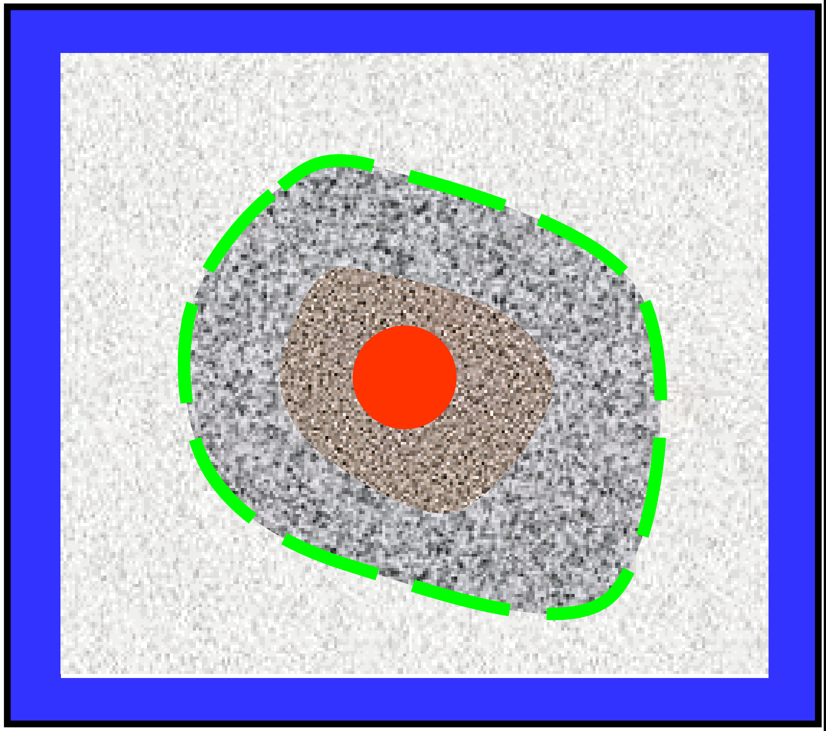
Global vs. local optimisation algorithms: Summary

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation
  - Graph cuts (e.g. geo-cuts)
    - Extended space (fractional segmentations)
    - Convex formulation
    - Integer solution (for submodular functions)



Global vs. local optimisation algorithms: Summary

- Geodesic active contours
  - Variational approach (e.g. level sets)
    - Gradient descent in the *space of contours*
    - Local minimum
    - Non-convex formulation
  - Graph cuts (e.g. geo-cuts)
    - Extended space (fractional segmentations)
    - Convex formulation
    - Integer solution (for submodular functions)



Other relaxations/extensions

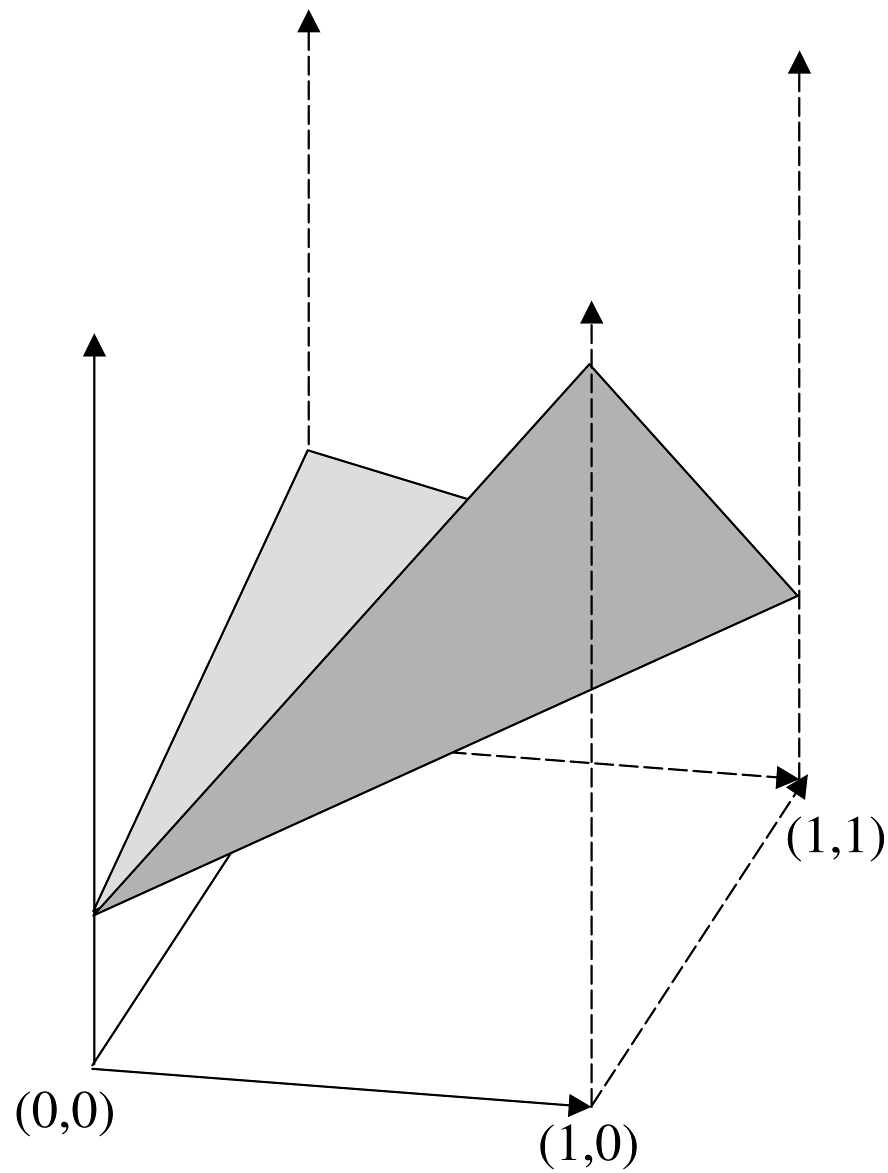
- Energy $E(\mathbf{x})$ defined for integer configurations $(x_p \in \{0,1\})$
- How to define for fractional configurations $(x_p \in [0,1])$?

Other relaxations/extensions

- LP relaxation [Schlesinger'76,Koster et al.'98,Chekuri et al.'00, Wainwright et al.'03]
  - Defined for multi-valued variables
  - Convex
  - E is submodular $\Rightarrow$ integer solution
- Lovász extension [Lovász'83]
  - Defined for binary variables
  - Always integer solution
  - E is submodular $\Leftrightarrow$ extension is convex
  - "Submodularity" – discrete analogue of convexity
- Sherali-Adams relaxation, semi-definite relaxation, SOCP relaxation, ...

LP relaxation and Lovász extension

Submodular function: $E(0,0) + E(1,1) \leq E(0,1) + E(1,0)$



LP relaxation and Lovász extension

Non-submodular function: $E(0,0) + E(1,1) \geq E(0,1) + E(1,0)$