Graphical Models and Learning

Lecture 3:
Maximum Flow, Minimum Cut

Yuliya Tarabalka
yuliya.tarabalka@inria.fr

Slides courtesy of Pawan Kumar

Outline

- Preliminaries
  - Functions and Excess Functions
  - s-t Flow
  - s-t Cut
  - Flows vs. Cuts
- Maximum Flow
- Algorithms
- Energy minimization with max flow/min cut

Context

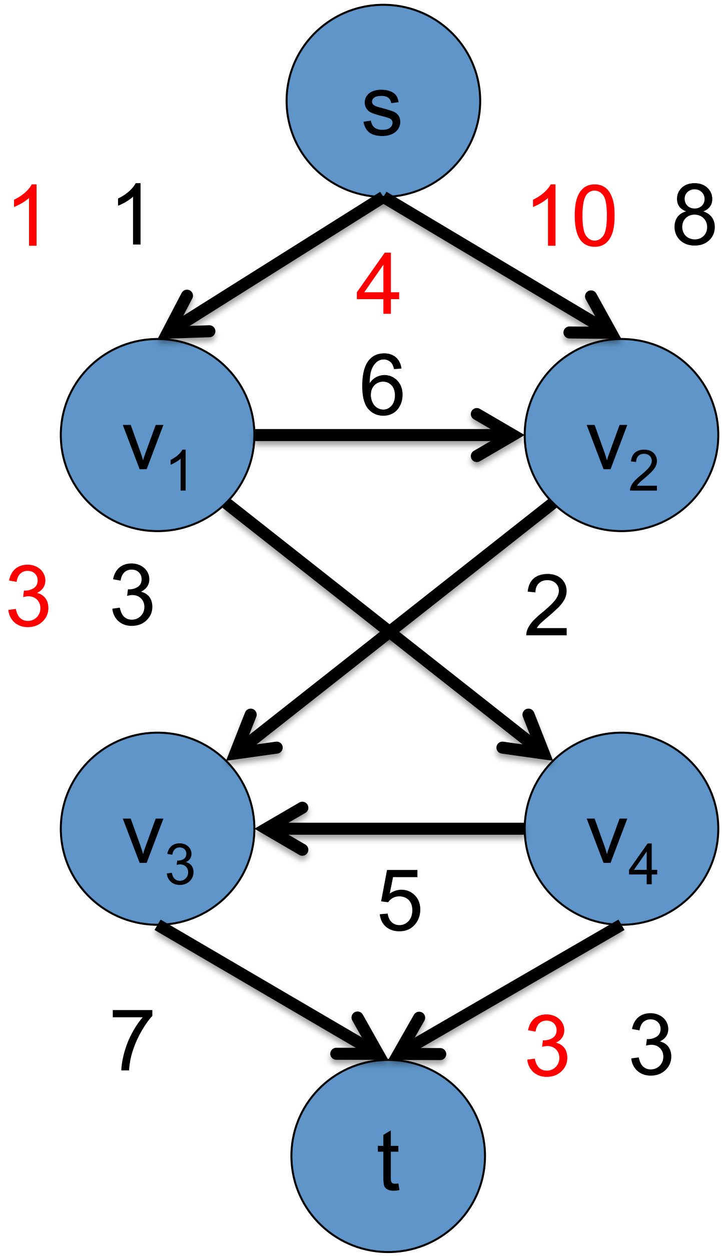
- Example: network optimization problems

Nodes	Arcs	Flow
Intersections	Roads	Vehicles
Airports	Air lanes	Aircraft
Switching points	Wires, channels	Messages
Pumping stations	Pipes	Fluids
Work centers	Materials-handling routes	Jobs

Maximum flow problem

- Applications
  - Maximize the flow through a company's distribution network from factories to customers
  - Maximize the flow of oil through a system of pipelines
  - Maximize the flow of vehicles through a transportation network

Functions on Arcs



D = (V, A)

Arc capacities c(a)

Function f: A ➔ Reals

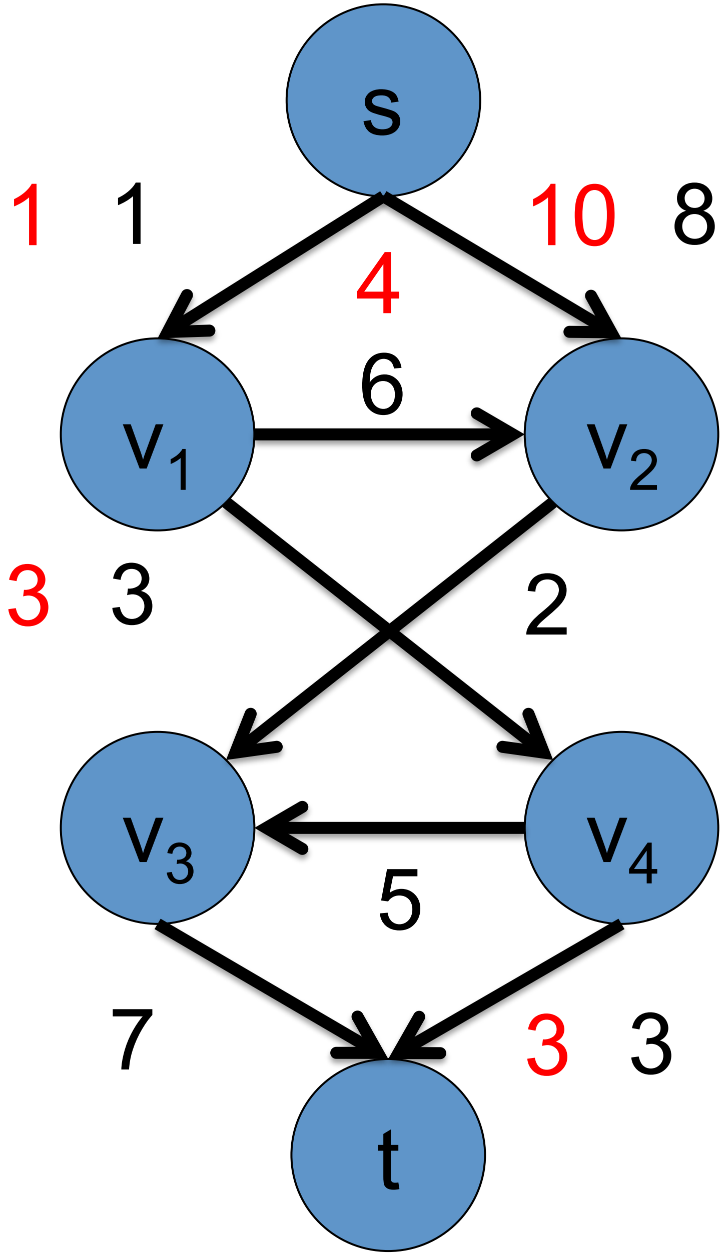
Excess function $E_f(v)$

Incoming value

-

Outgoing value

Functions on Arcs



D = (V, A)

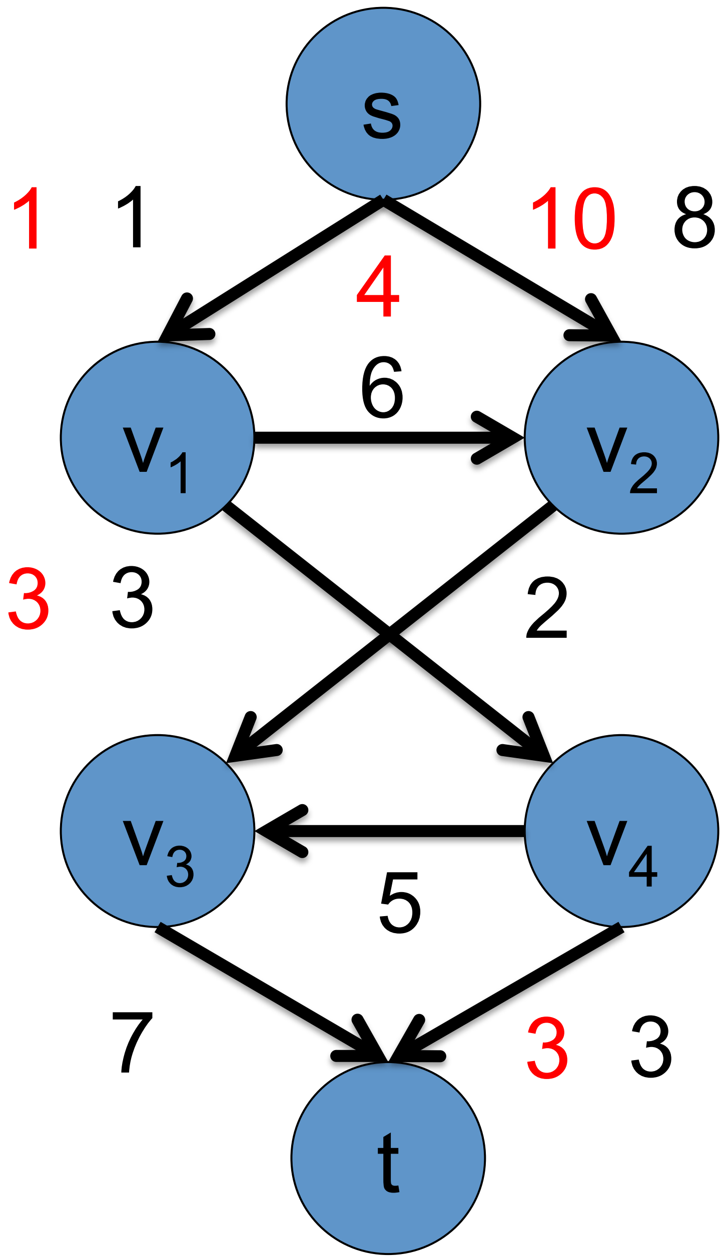
Arc capacities c(a)

Function f: A ➔ Reals

Excess function $E_f(v)$

$$\Sigma_{a \in \text{in-arcs}(v)} f(a)$$
$$-$$
$$\Sigma_{a \in \text{out-arcs}(v)} f(a)$$

Functions on Arcs



D = (V, A)

Arc capacities c(a)

Function f: A ➔ Reals

Excess function $E_f(v)$

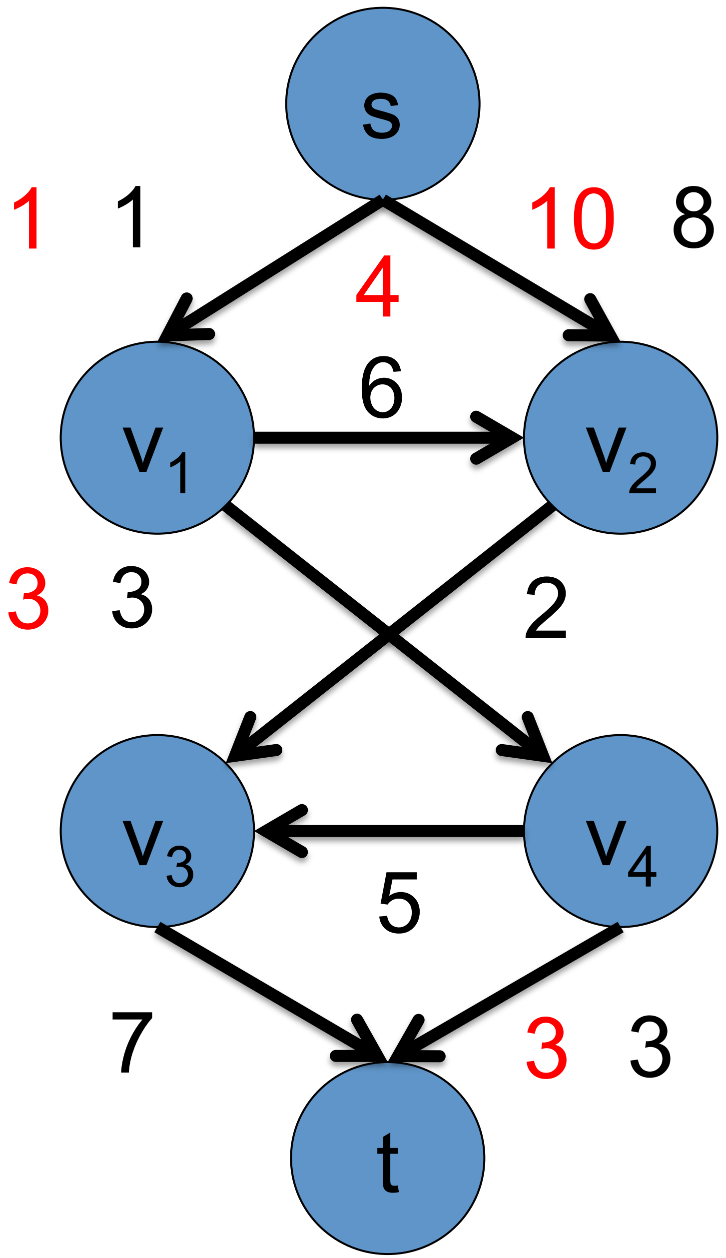
f(in-arcs(v))

\-

f(out-arcs(v))

$E_f(v_1)$ -6

Functions on Arcs



D = (V, A)

Arc capacities c(a)

Function f: A ➔ Reals

Excess function $E_f(v)$

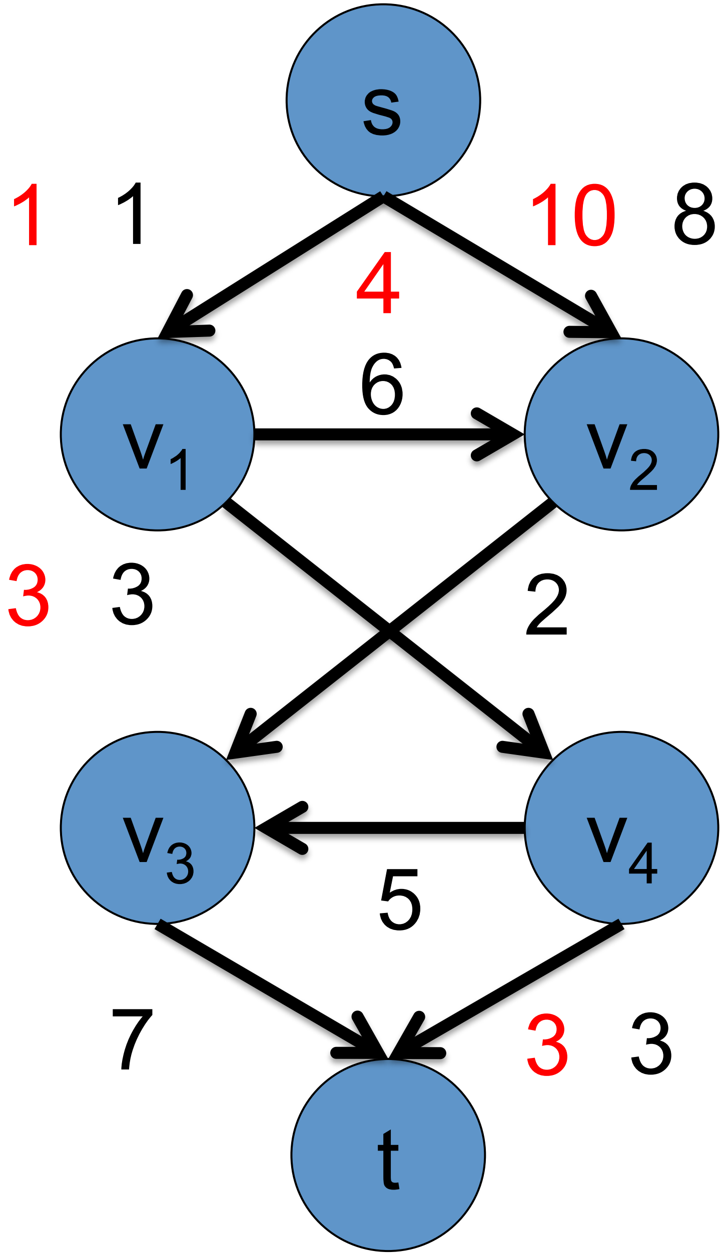
f(in-arcs(v))

-

f(out-arcs(v))

$E_f(v_2)$ 14

Excess Functions of Vertex Subsets



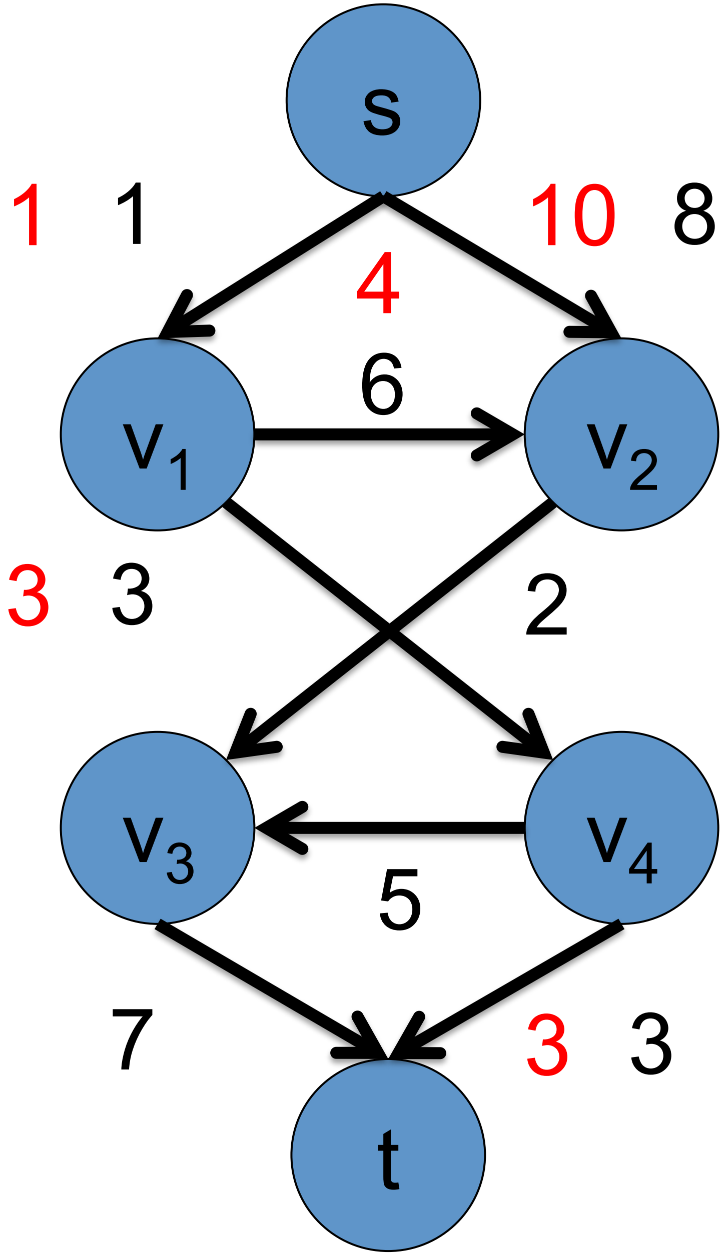
Excess function $E_f(U)$

Incoming Value

-

Outgoing Value

Excess Functions of Vertex Subsets



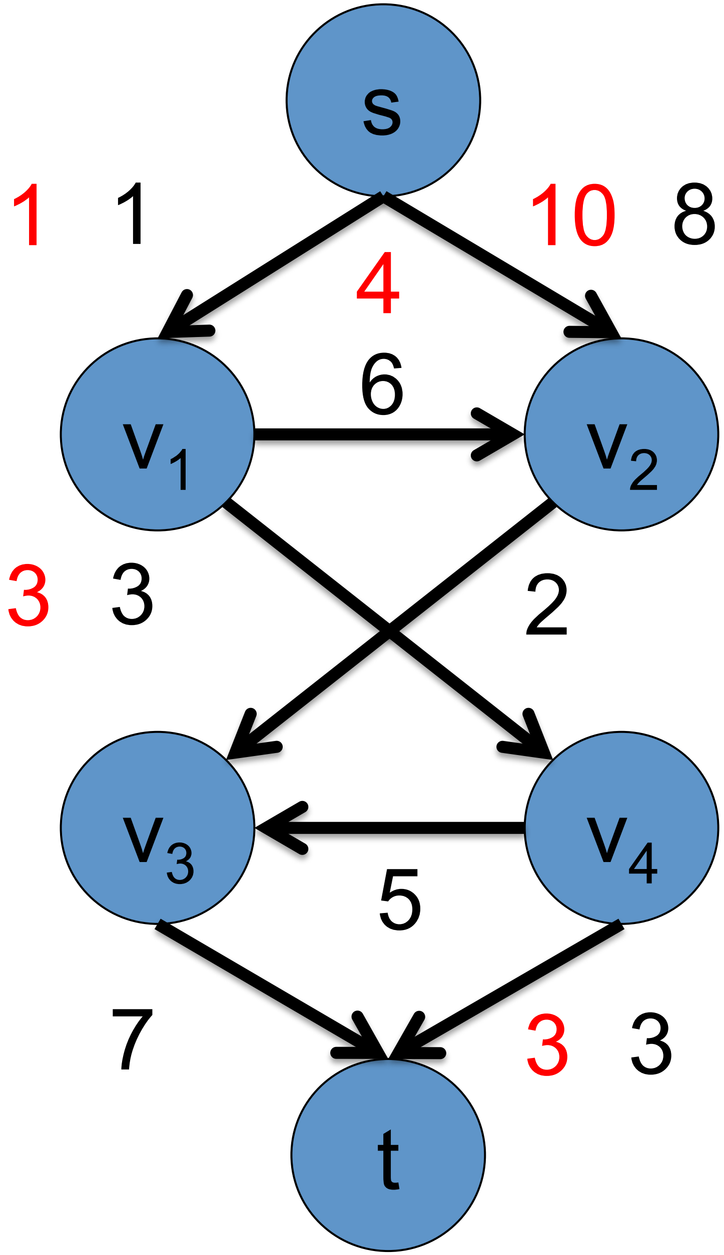
Excess function $E_f(U)$

$$\Sigma_{a \in \text{in-arcs}(U)} f(a)$$

$$-$$

$$\Sigma_{a \in \text{out-arcs}(U)} f(a)$$

Excess Functions of Vertex Subsets



Excess function $E_f(U)$

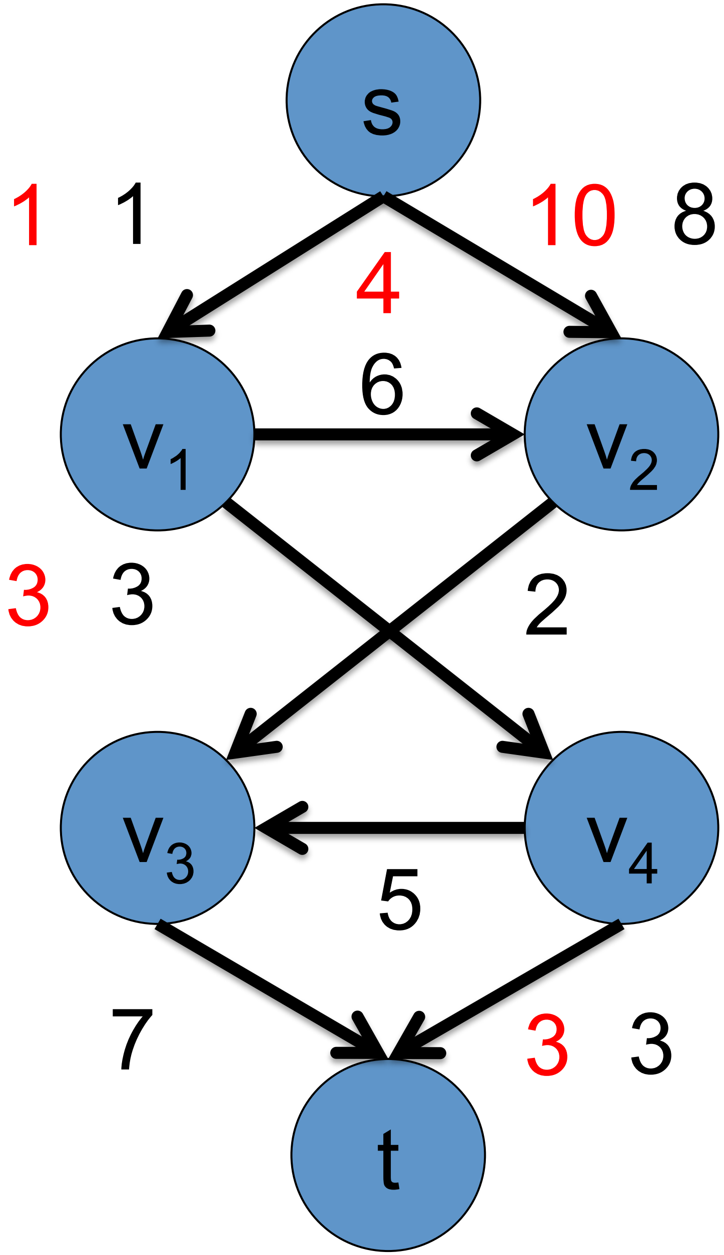
f(in-arcs(U))

-

f(out-arcs(U))

$E_f(\{v_1,v_2\})$ 8

Excess Functions of Vertex Subsets



Excess function $E_f(U)$

f(in-arcs(U))

-

f(out-arcs(U))

$E_f(\{v_1,v_2\})$ $-6 + 14$

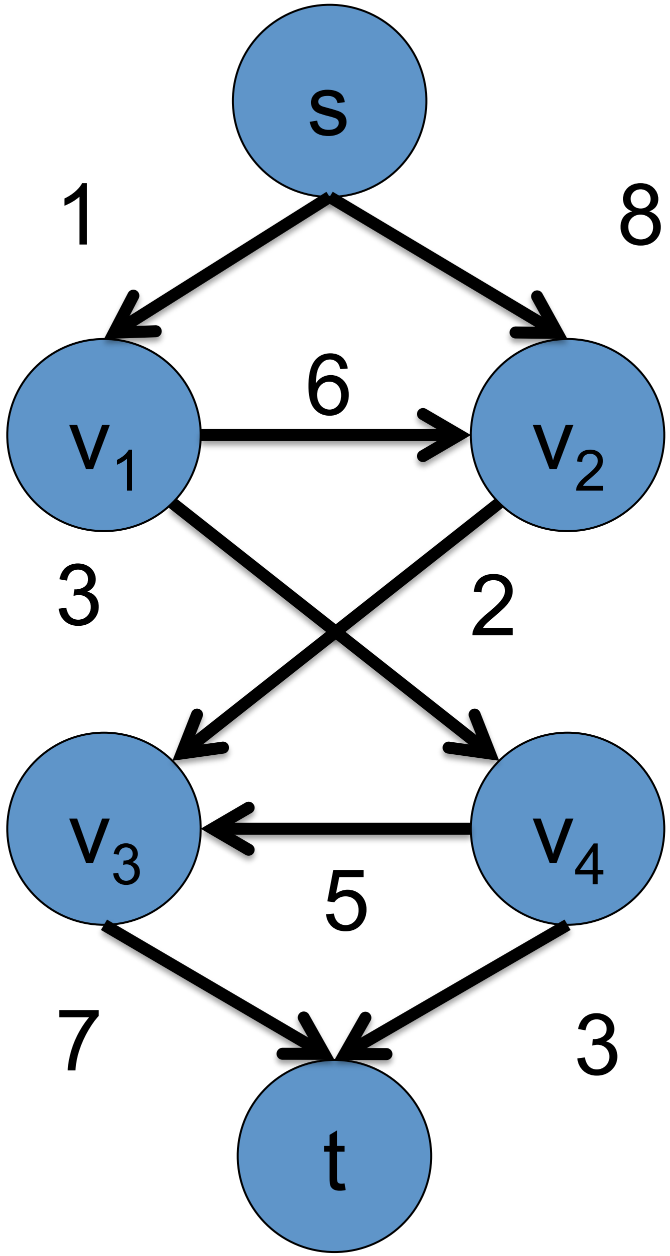
$E_f(U) = \Sigma_{v \in U} E_f(v)$

Outline

- Preliminaries
  - Functions and Excess Functions
  - **s-t Flow**
  - s-t Cut
  - Flows vs. Cuts

- Maximum Flow
- Algorithms
- Energy minimization with max flow/min cut

s-t Flow



Function flow: A ➔ R

Flow of arc ≤ arc capacity

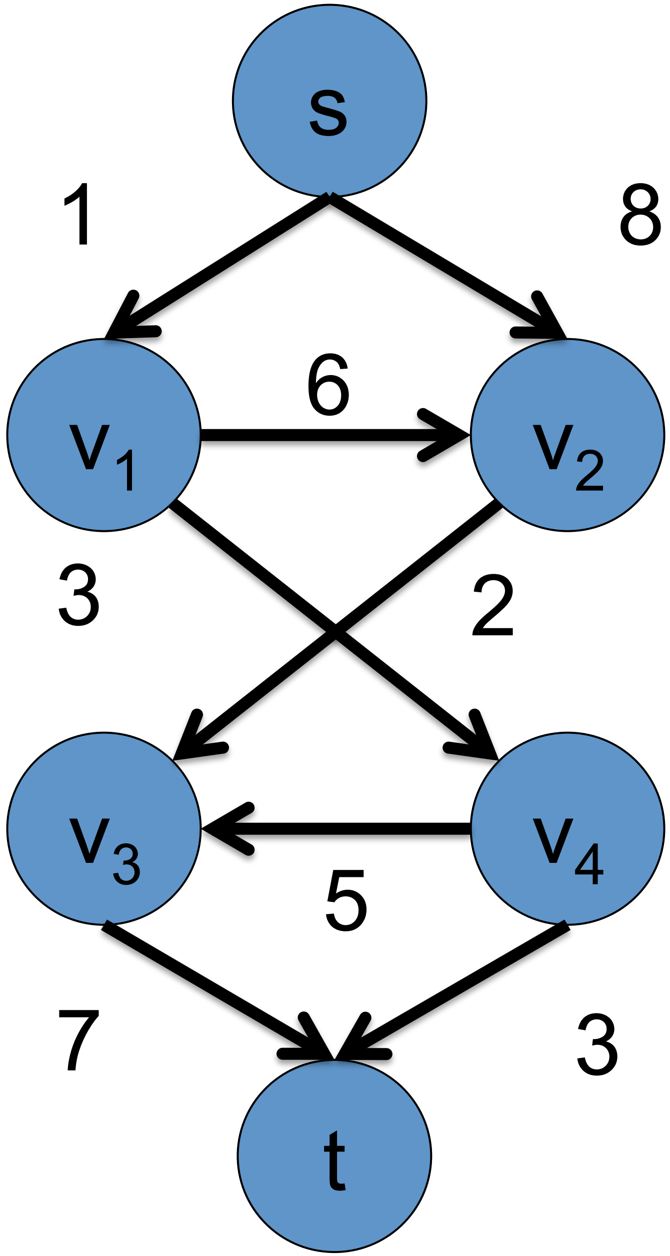
Flow is non-negative

For all vertex except s,t

Incoming flow

= Outgoing flow

s-t Flow



Function flow: A ➔ R

$\text{flow}(a) \leq c(a)$

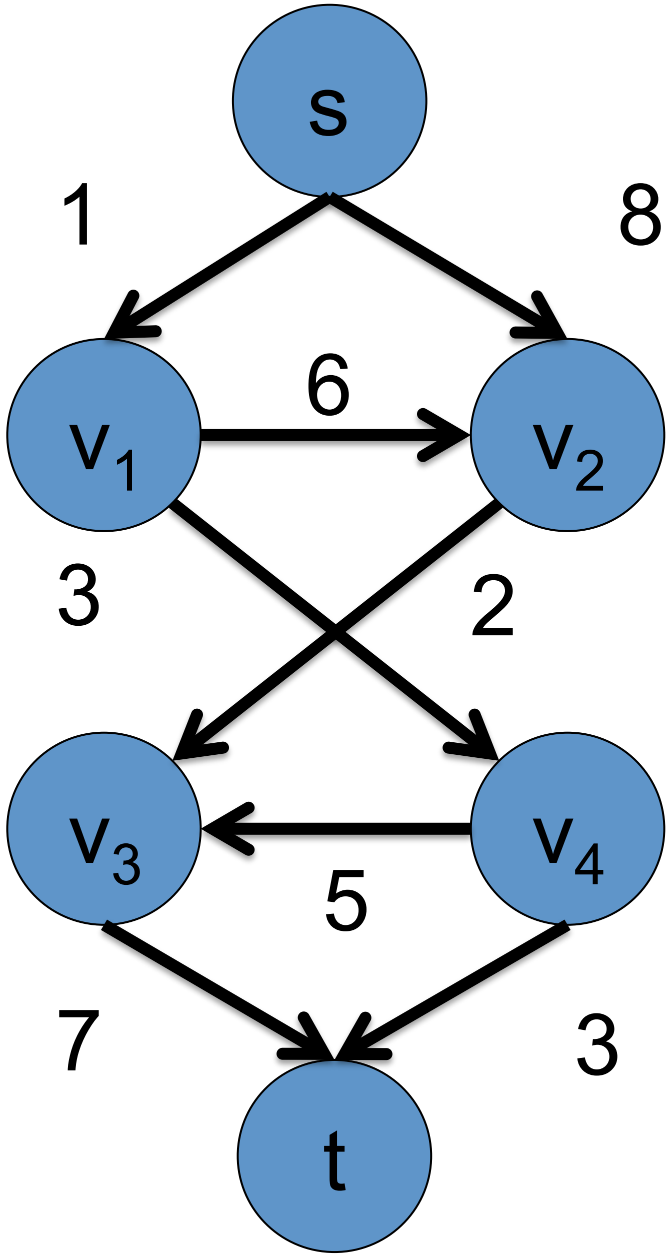
$\text{flow}(a) \geq 0$

For all $v \in V \setminus \{s,t\}$

$$\Sigma_{(u,v)\in A} \text{flow}((u,v))$$

$$= \Sigma_{(v,u)\in A} \text{flow}((v,u))$$

s-t Flow



Function flow: A ➔ R

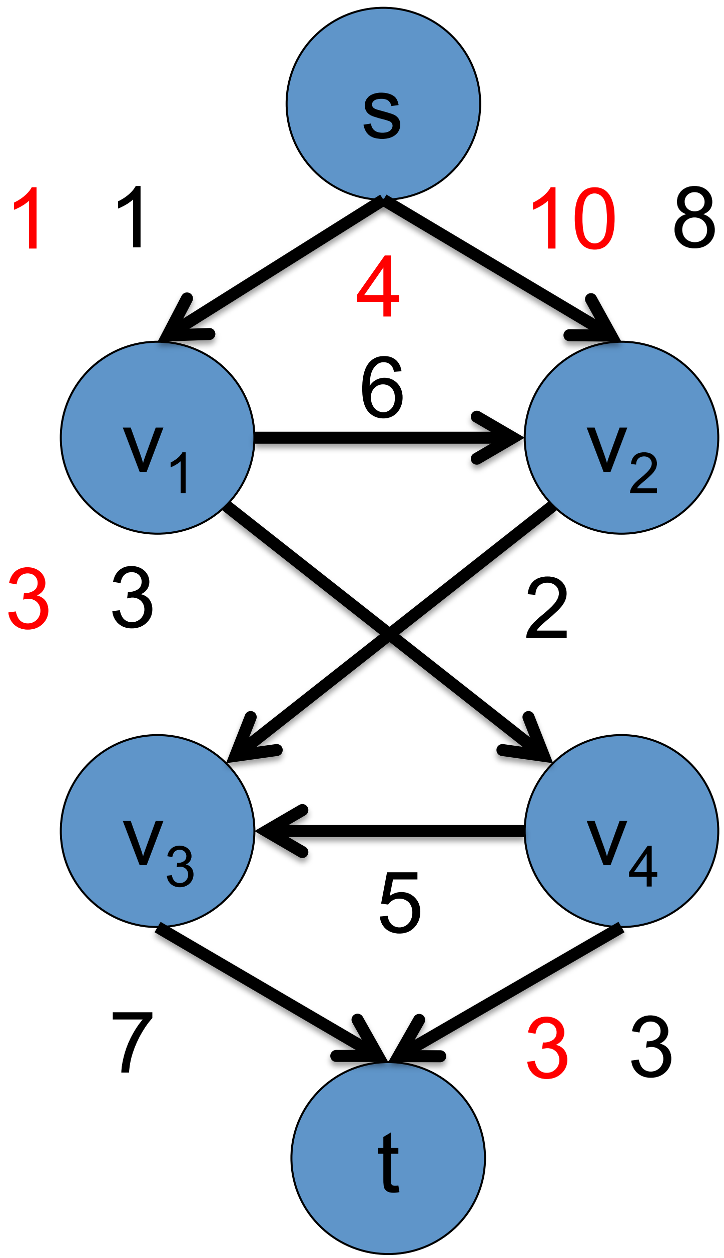
flow(a) ≤ c(a)

flow(a) ≥ 0

For all $v \in V \setminus \{s,t\}$

$E_{flow}(v) = 0$

s-t Flow



Function flow: A ➔ R

flow(a) ≤ c(a)

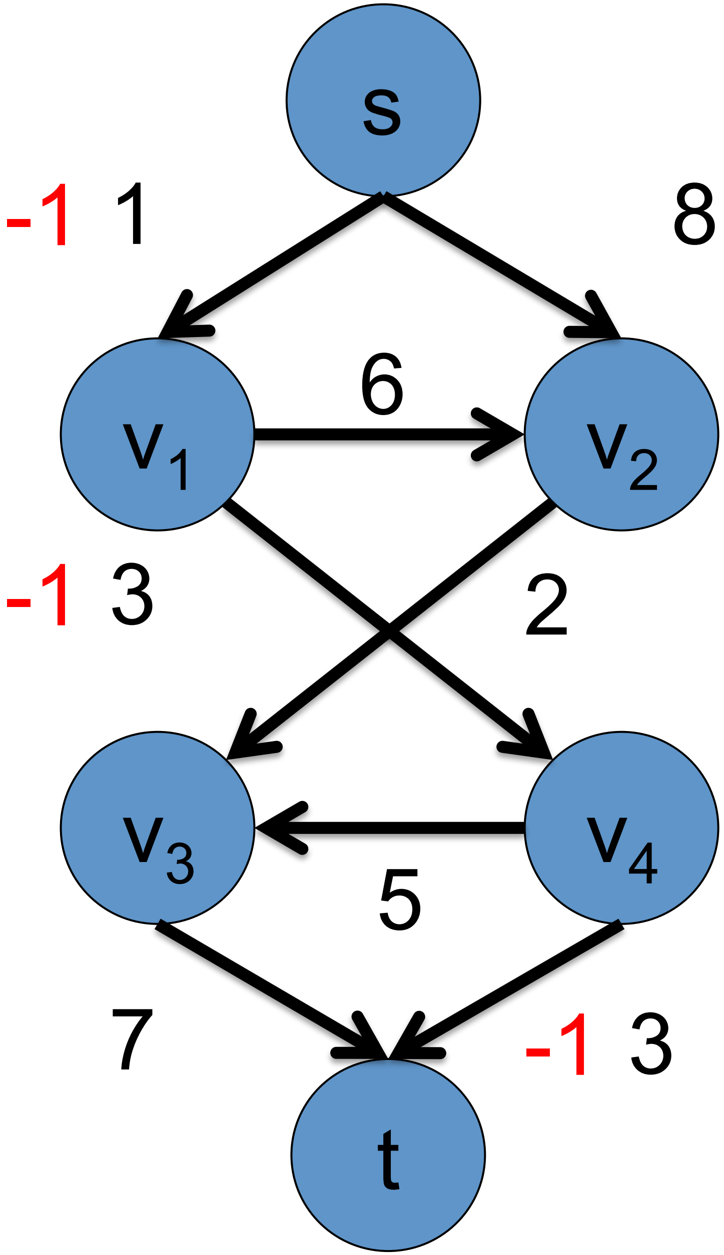
flow(a) ≥ 0

For all v ∈ V \ {s,t}

$E_{flow}(v) = 0$

✗

s-t Flow



Function flow: A ➔ R

flow(a) ≤ c(a)

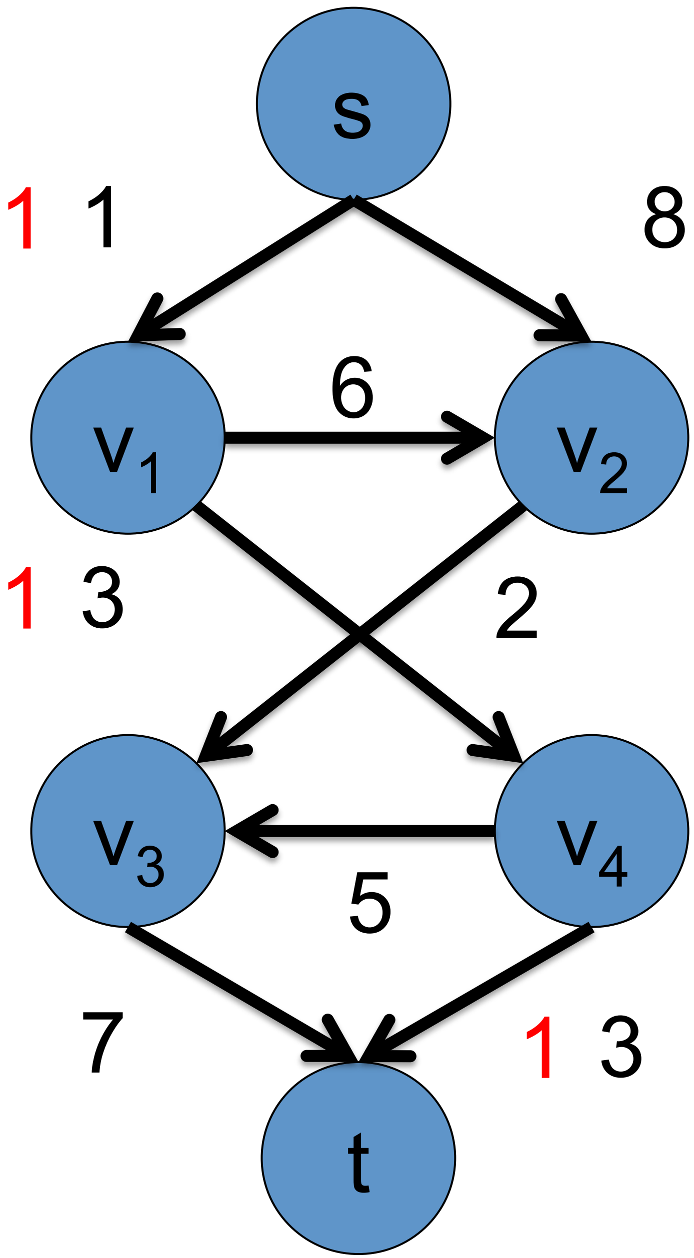
flow(a) ≥ 0

For all v ∈ V \ {s,t}

$E_{flow}(v) = 0$

✗

s-t Flow



Function flow: A ➔ R

$\text{flow}(a) \leq c(a)$

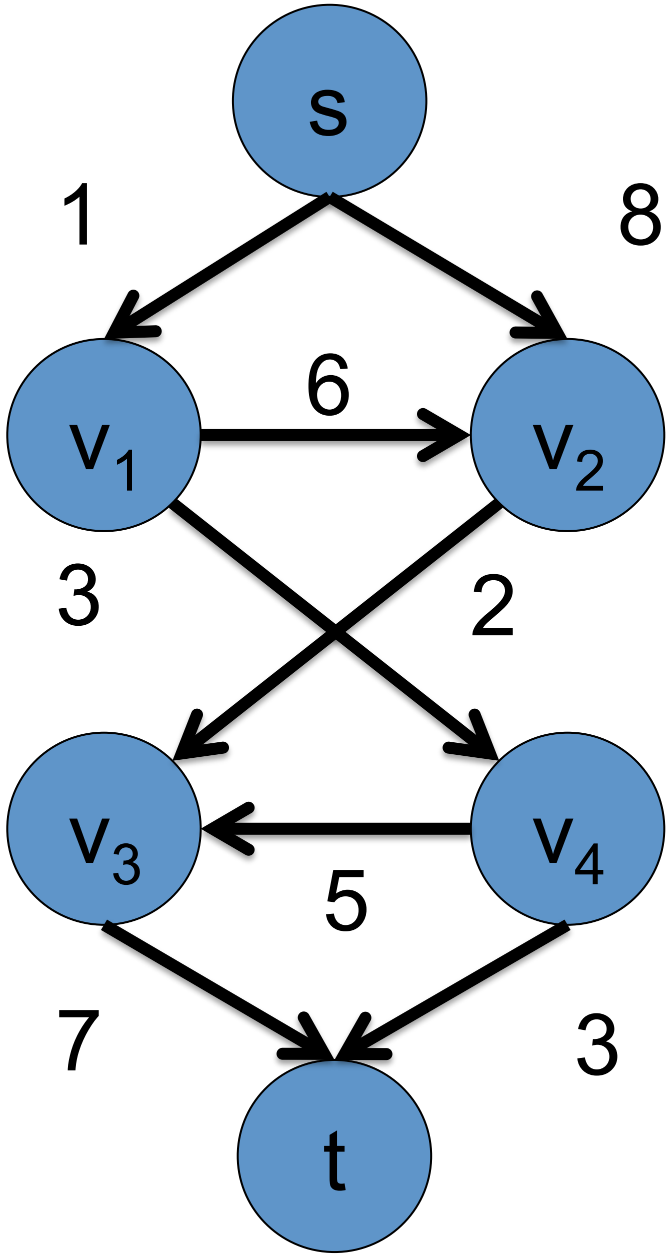
$\text{flow}(a) \geq 0$

For all $v \in V \setminus \{s,t\}$

$E_{\text{flow}}(v) = 0$

✓

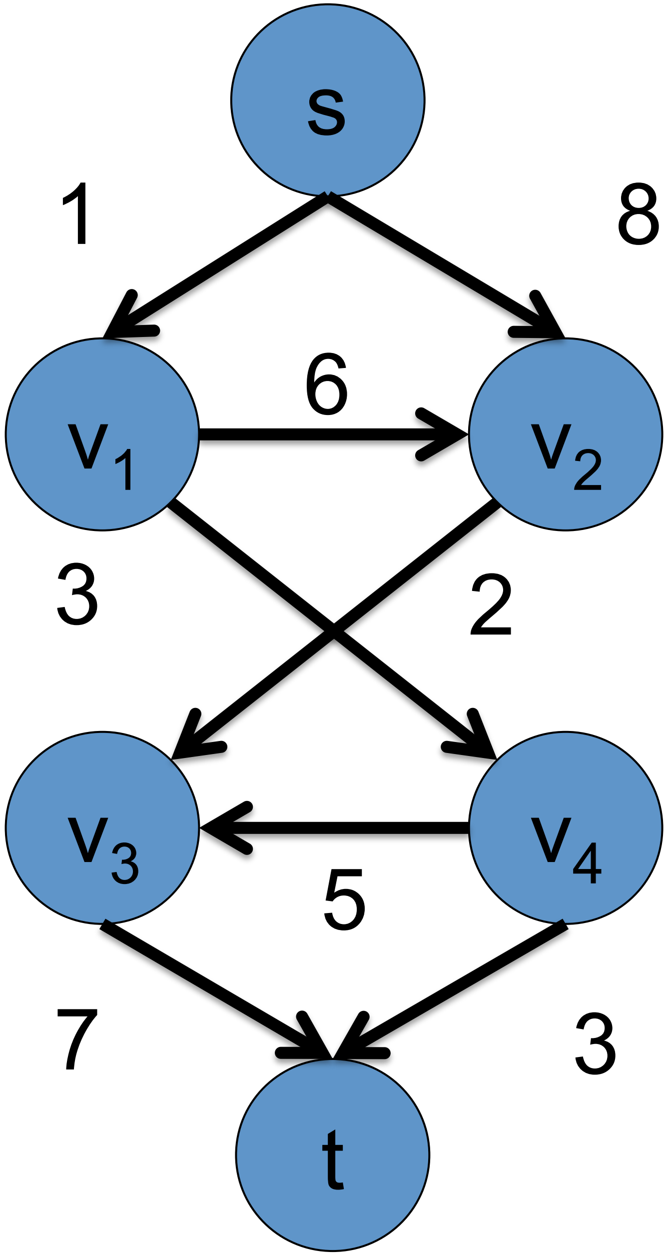
Value of s-t Flow



Outgoing flow of s

- Incoming flow of s

Value of s-t Flow

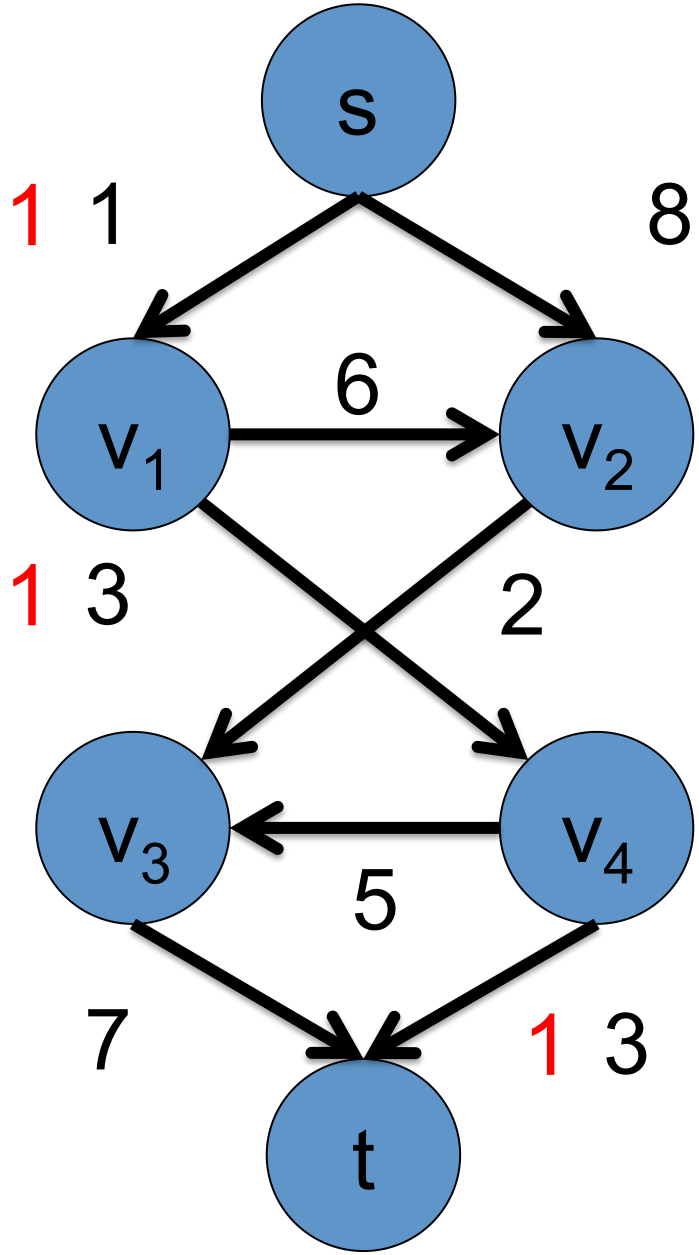


$-E_{flow}(s)$ $\quad$ $E_{flow}(t)$

$\Sigma_{(s,v)\in A}$ flow((s,v))

$- \Sigma_{(u,s)\in A}$ flow((u,s))

Value of s-t Flow



$-E_{flow}(s)$ $E_{flow}(t)$

$\Sigma_{(s,v)\in A}$ flow((s,v))

$- \Sigma_{(u,s)\in A}$ flow((u,s))

Value = 1

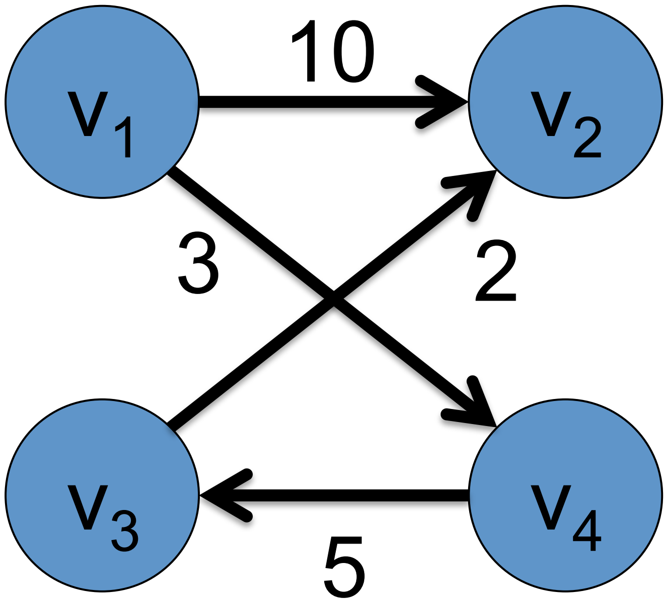
Outline

- Preliminaries
  - Functions and Excess Functions
  - s-t Flow
  - **s-t Cut**
  - Flows vs. Cuts

- Maximum Flow
- Algorithms
- Energy minimization with max flow/min cut

Cut

D = (V, A)



Let U be a subset of V

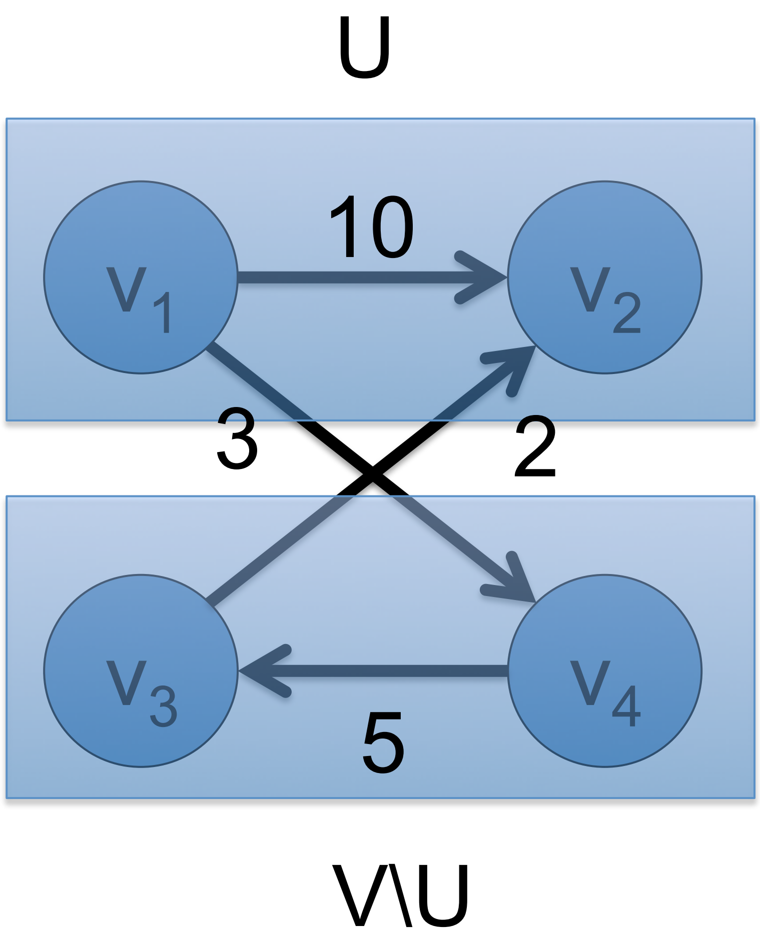
C is a set of arcs such that

- $(u,v) \in A$
- $u \in U$
- $v \in V \setminus U$

C is a cut in the digraph D

Cut

D = (V, A)



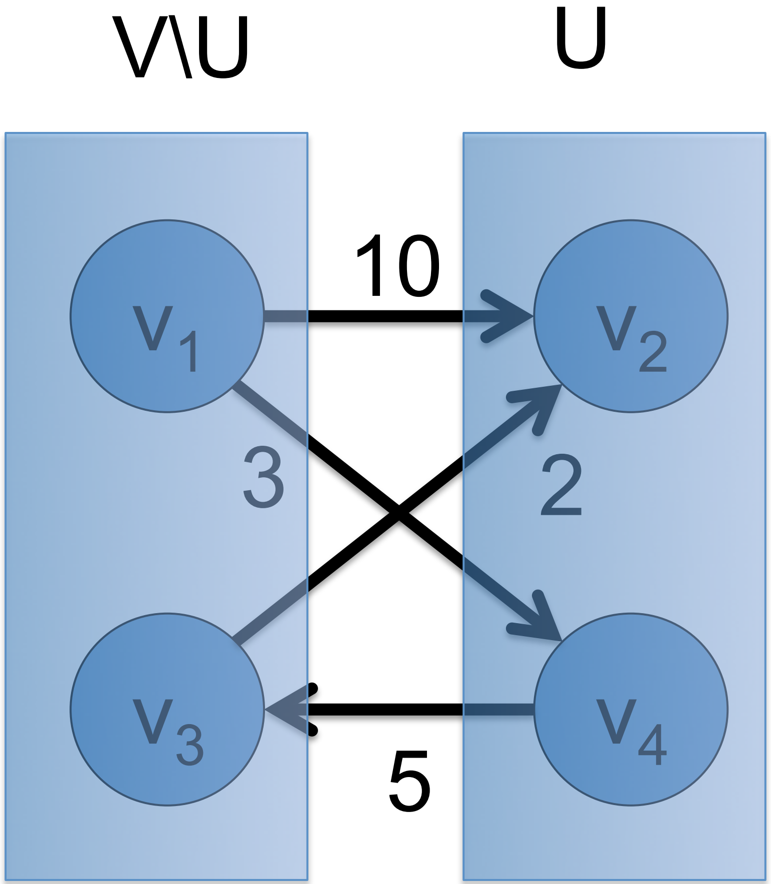
What is C?

$\{(v_1,v_2),(v_1,v_4)\}$?

$\{(v_1,v_4),(v_3,v_2)\}$?

✓ $\{(v_1,v_4)\}$?

Cut



D = (V, A)

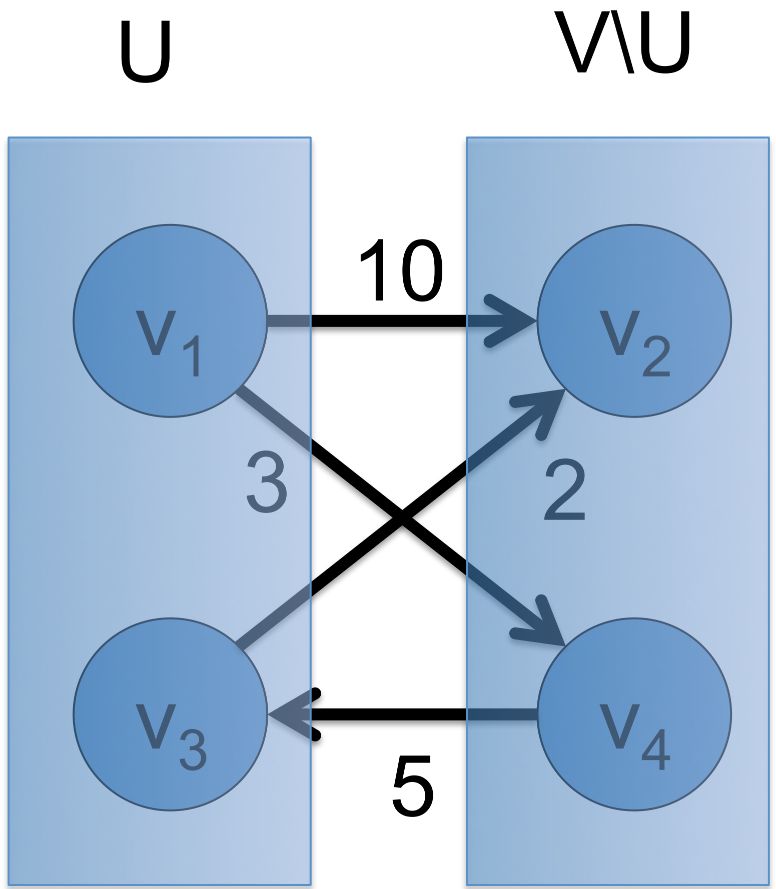
What is C?

$\{(v_1,v_2),(v_1,v_4),(v_3,v_2)\}$?

✓ $\{(v_4,v_3)\}$?

$\{(v_1,v_4),(v_3,v_2)\}$?

Cut



D = (V, A)

What is C?

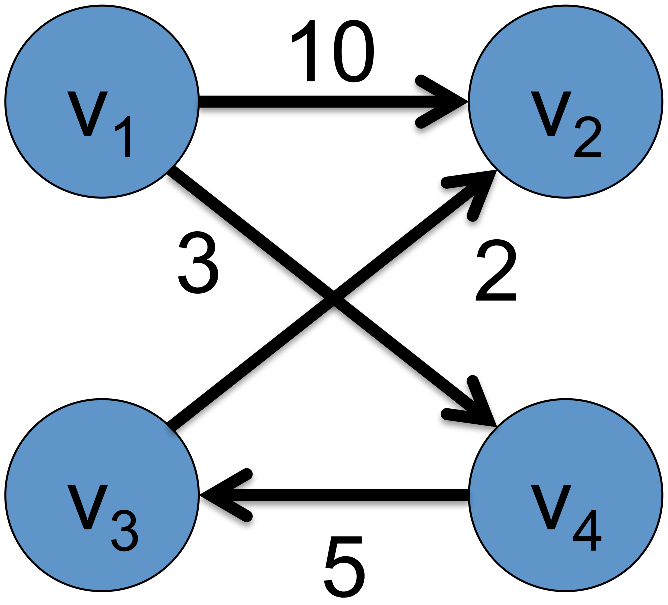
✓ $\{(v_1,v_2),(v_1,v_4),(v_3,v_2)\}$?

$\{(v_3,v_2)\}$?

$\{(v_1,v_4),(v_3,v_2)\}$?

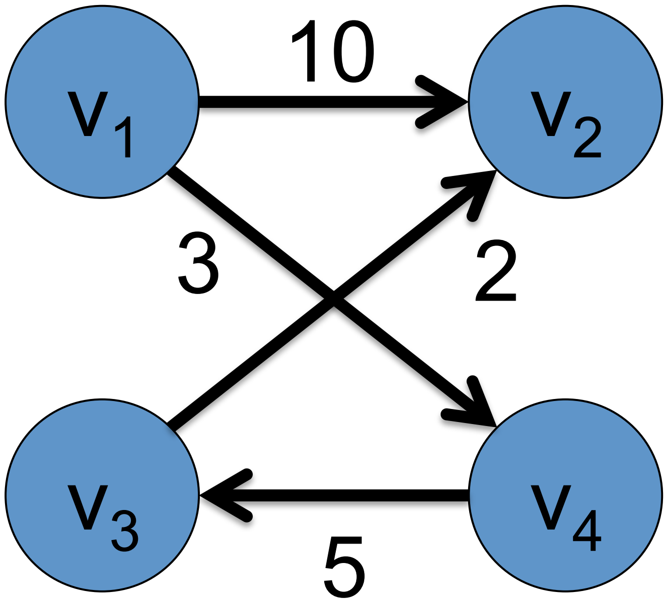
Cut

D = (V, A)



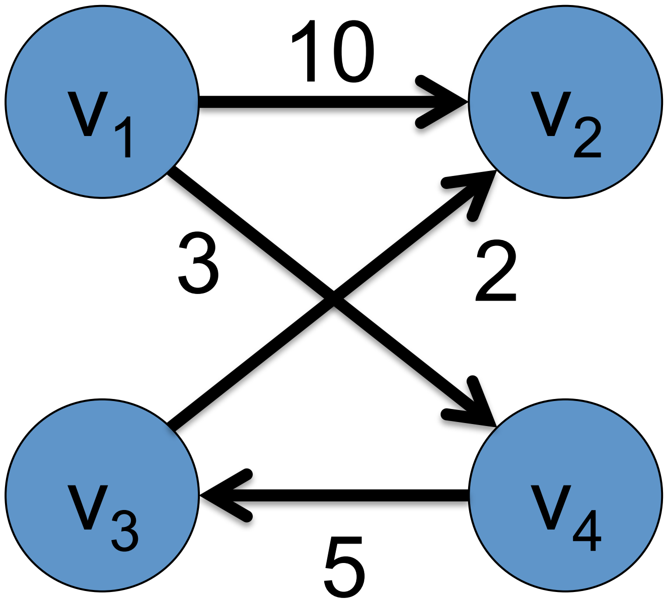
C = out-arcs(U)

Capacity of Cut



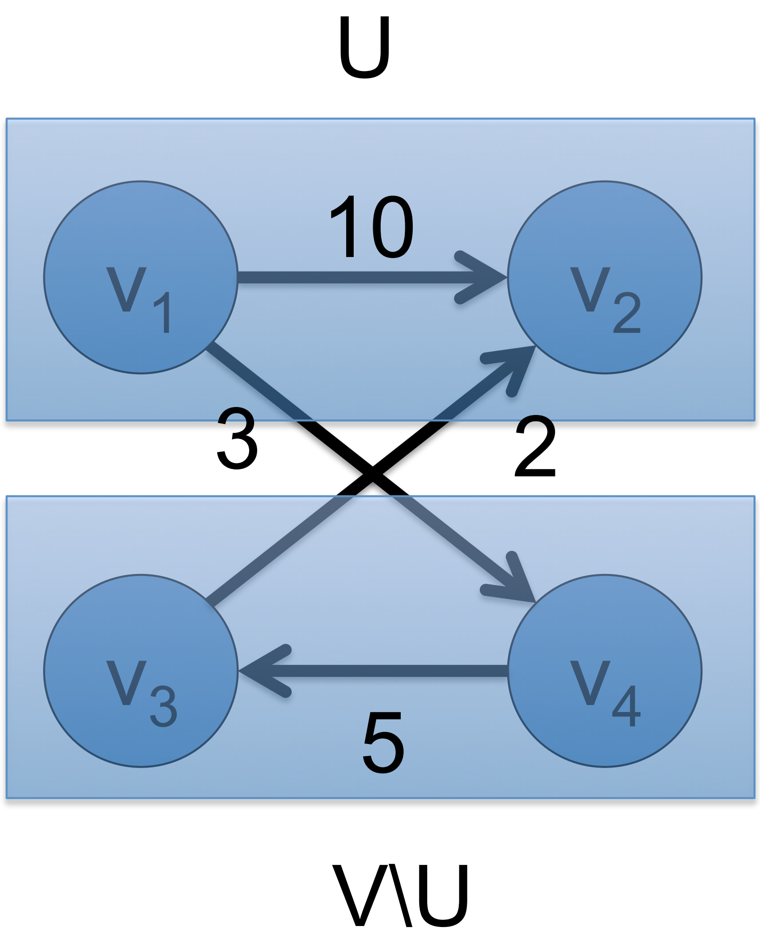
Sum of capacity of all arcs in C

Capacity of Cut



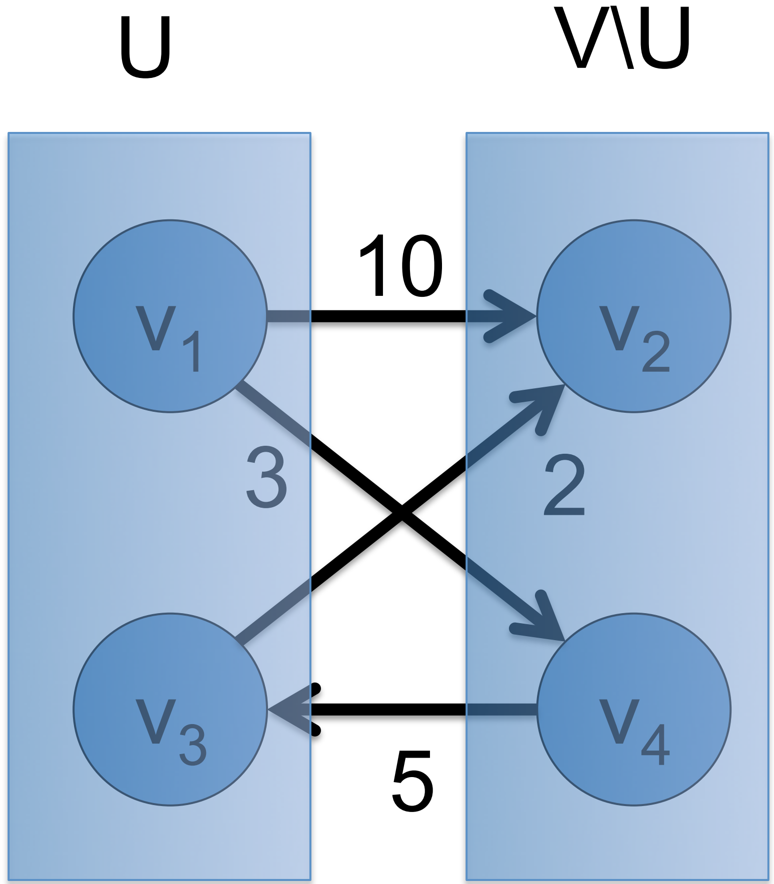
$\Sigma_{a \in C}\, c(a)$

Capacity of Cut



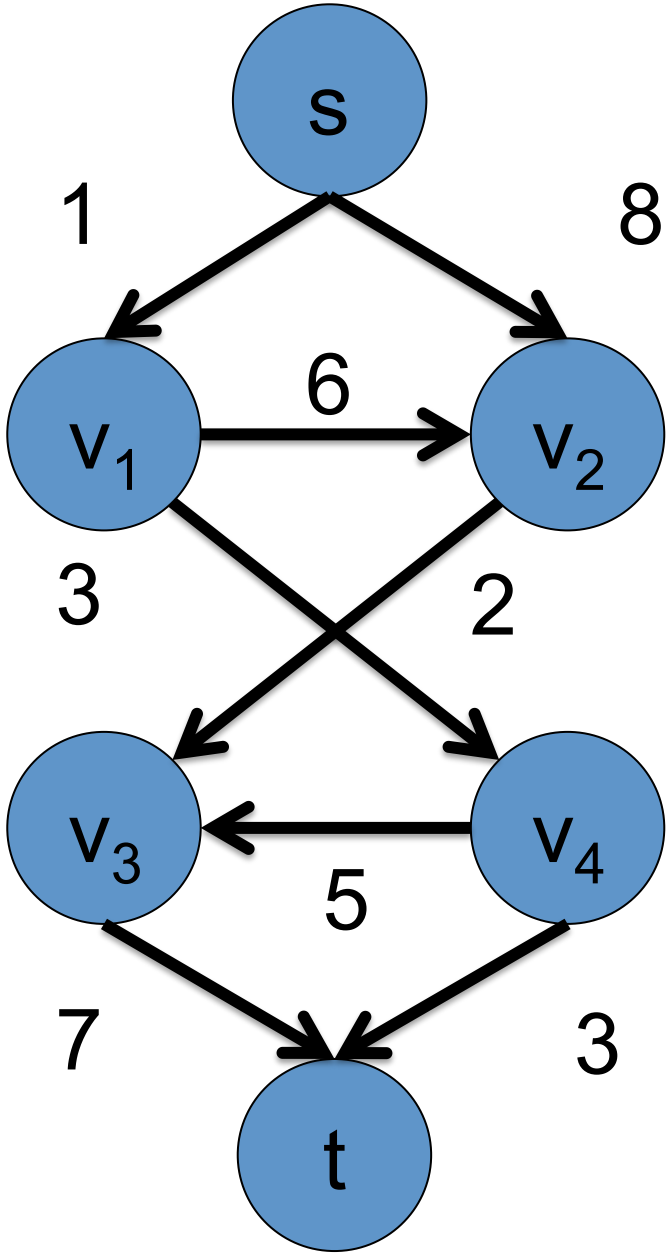
3

Capacity of Cut



15

s-t Cut



D = (V, A)

A source vertex "s"

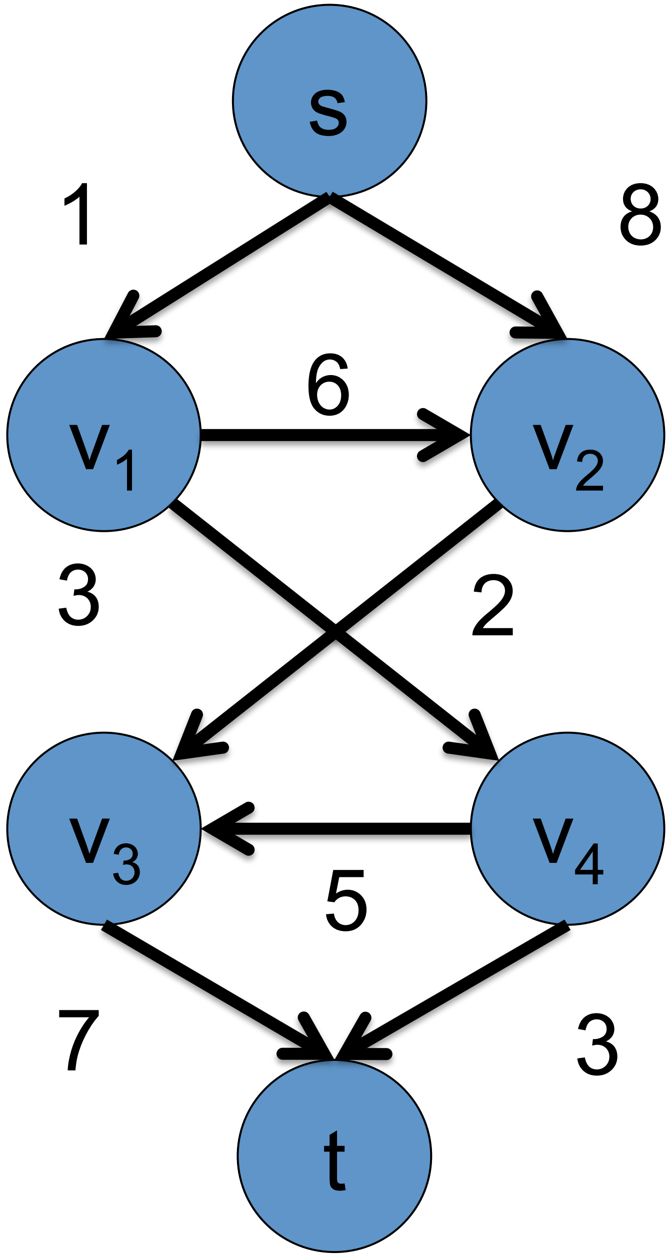
A sink vertex "t"

C is a cut such that

- $s \in U$
- $t \in V \backslash U$

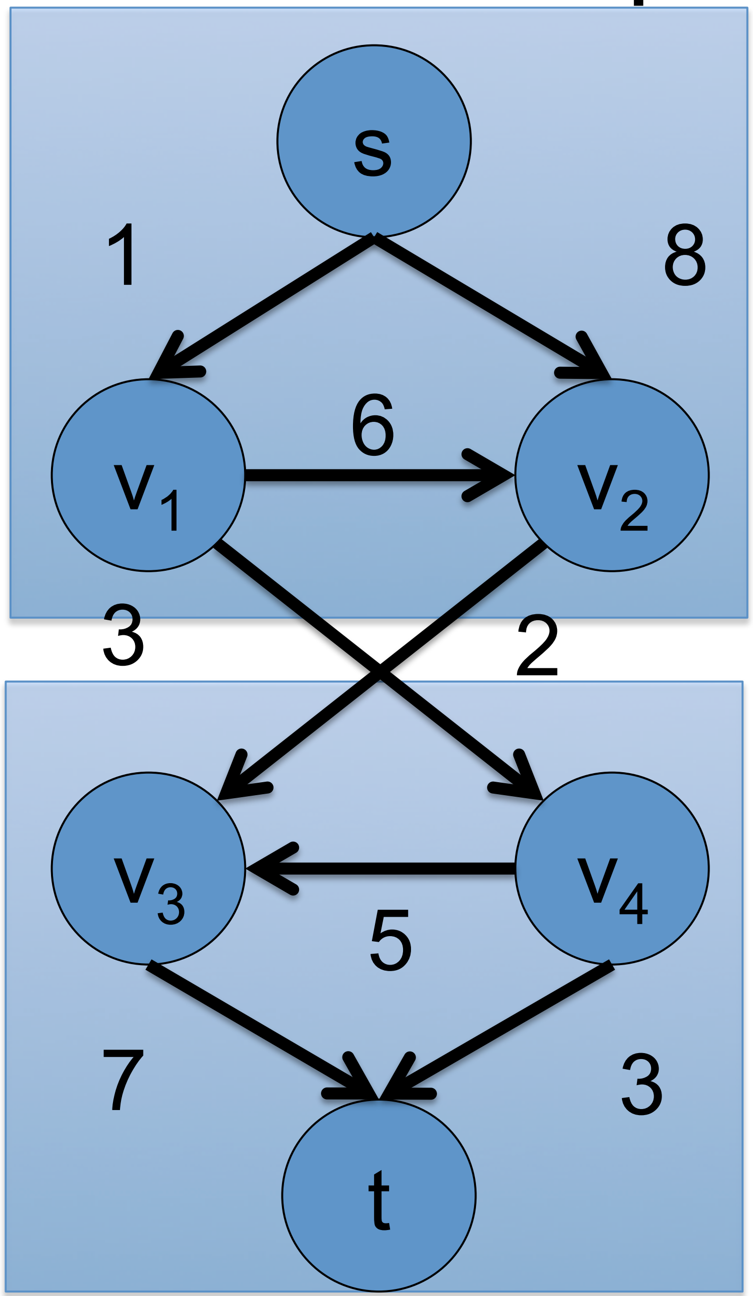
C is an s-t cut

Capacity of s-t Cut



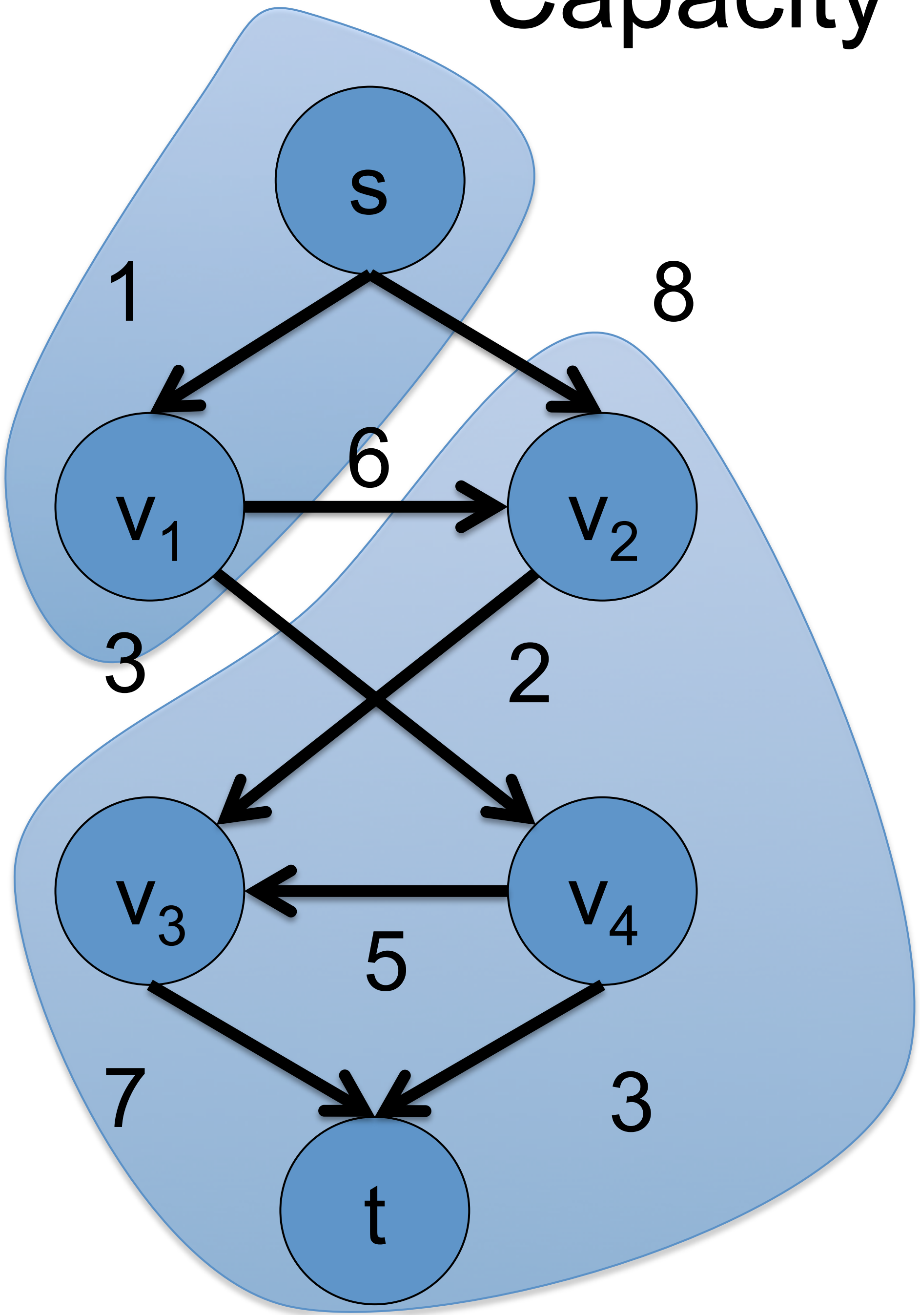
$\Sigma_{a \in C}\, c(a)$

Capacity of s-t Cut



5

Capacity of s-t Cut



17

Outline

- Preliminaries
  - Functions and Excess Functions
  - s-t Flow
  - s-t Cut
  - **Flows vs. Cuts**
- Maximum Flow
- Algorithms
- Energy minimization with max flow/min cut

Flows vs. Cuts

An s-t flow function : A ➔ Reals

An s-t cut C such that $s \in U$, $t \in V \setminus U$

Value of flow ≤ Capacity of C

Flows vs. Cuts

Value of flow $= -E_{flow}(s)$

$= -E_{flow}(s) - \Sigma_{v \in U \setminus \{s\}} E_{flow}(v)$

$= -E_{flow}(U)$

$=$ flow(out-arcs(U))
 - flow(in-arcs(U))

$\leq$ Capacity of C
 - flow(in-arcs(U))

Flows vs. Cuts

Value of flow $= -E_{flow}(s)$

$= -E_{flow}(s) - \Sigma_{v \in U \setminus \{s\}} E_{flow}(v)$

$= -E_{flow}(U)$

$=$ flow(out-arcs(U))
 - flow(in-arcs(U))

$\leq$ Capacity of C

When does equality hold?

Flows vs. Cuts

Value of flow $= -E_{flow}(s)$

$= -E_{flow}(s) - \Sigma_{v \in U \setminus \{s\}} E_{flow}(v)$

$= -E_{flow}(U)$

$=$ flow(out-arcs(U))

$-$ flow(in-arcs(U))

$\leq$ Capacity of C

flow(a) = c(a), $a \in$ out-arcs(U)   flow(a) = 0, $a \in$ in-arcs(U)

Flows vs. Cuts

Value of flow $= -E_{flow}(s)$

$= -E_{flow}(s) - \Sigma_{v \in U \setminus \{s\}} E_{flow}(v)$

$= -E_{flow}(U)$

$=$ flow(out-arcs(U)) - flow(in-arcs(U))

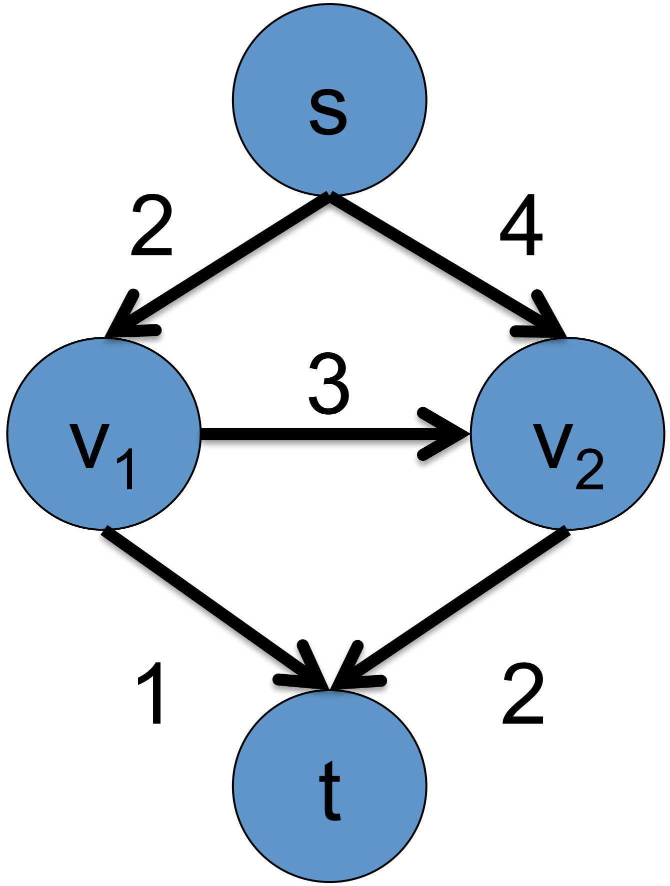
$=$ Capacity of C

flow(a) = c(a), $a \in$ out-arcs(U)   flow(a) = 0, $a \in$ in-arcs(U)

Outline

- Preliminaries
- **Maximum Flow**
  - Residual Graph
  - Max-Flow Min-Cut Theorem
- Algorithms
- Energy minimization with max flow/min cut

Maximum Flow Problem



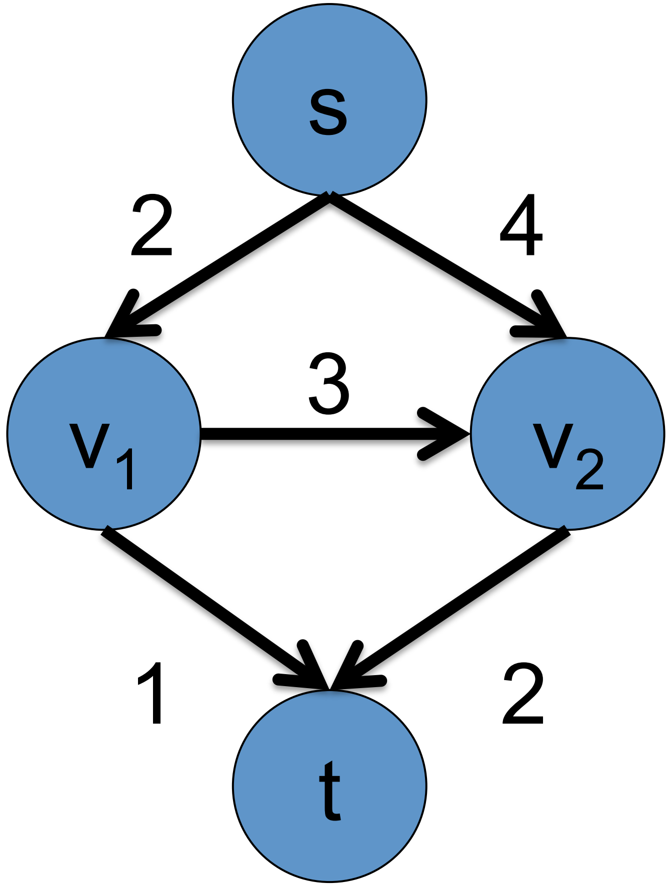
Find the flow with the maximum value !!

$\Sigma_{(s,v)\in A}$ flow((s,v))

- $\Sigma_{(u,s)\in A}$ flow((u,s))

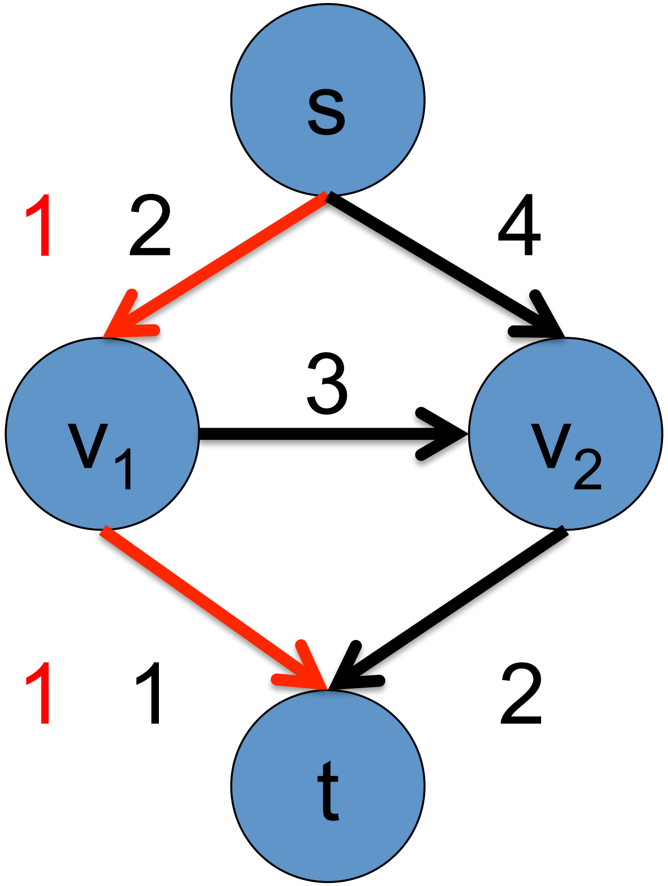
First suggestion to solve this problem !!

Passing Flow through s-t Paths



Find an s-t path where flow(a) < c(a) for all arcs

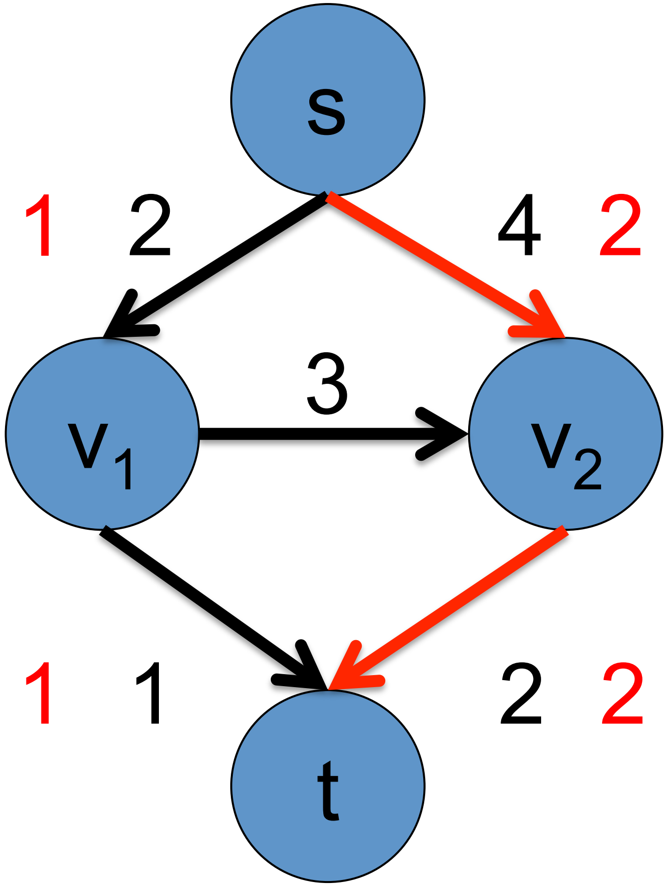
Passing Flow through s-t Paths



Find an s-t path where flow(a) < c(a) for all arcs

Pass maximum allowable flow through the arcs

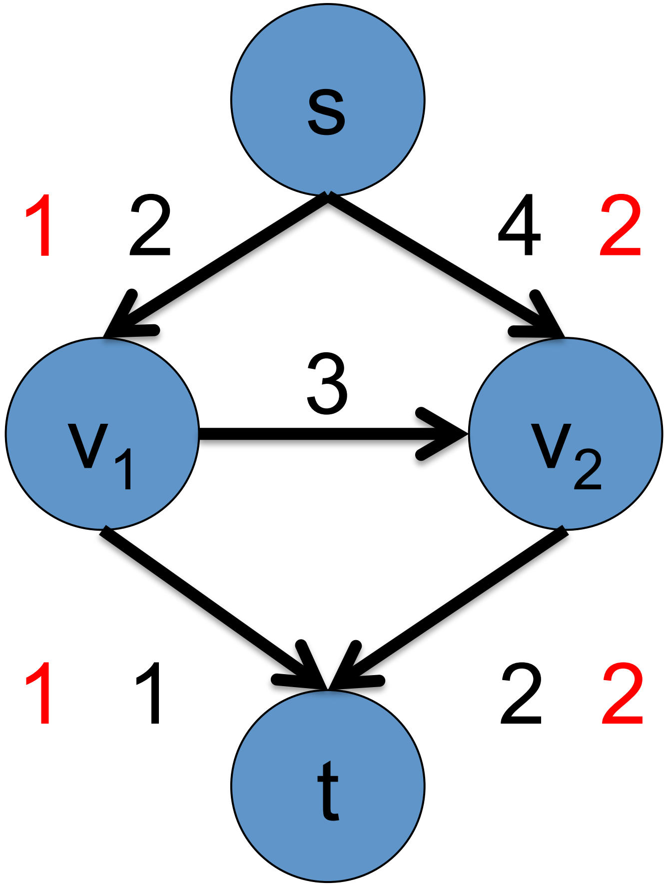
Passing Flow through s-t Paths



Find an s-t path where flow(a) < c(a) for all arcs

Pass maximum allowable flow through the arcs

Passing Flow through s-t Paths

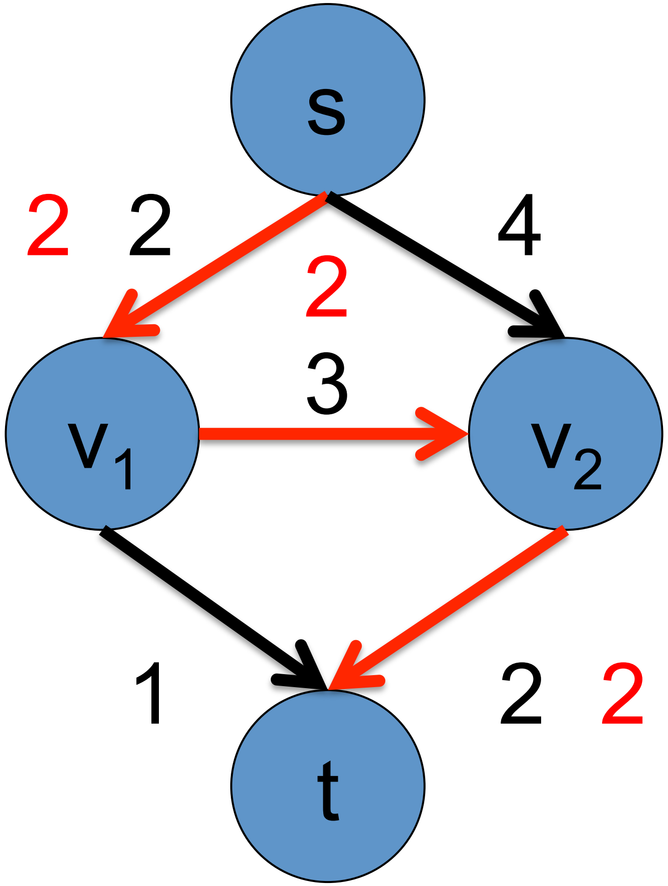


Find an s-t path where flow(a) < c(a) for all arcs

No more paths. Stop.

Will this give us maximum flow? **NO !!!**

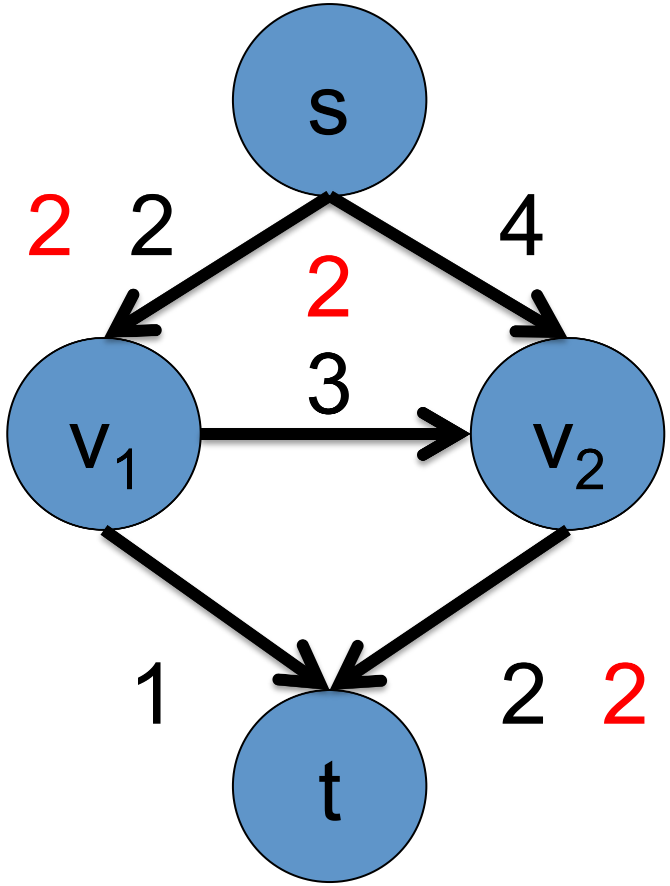
Passing Flow through s-t Paths



Find an s-t path where flow(a) < c(a) for all arcs

Pass maximum allowable flow through the arcs

Passing Flow through s-t Paths



Find an s-t path where flow(a) < c(a) for all arcs

No more paths. Stop.

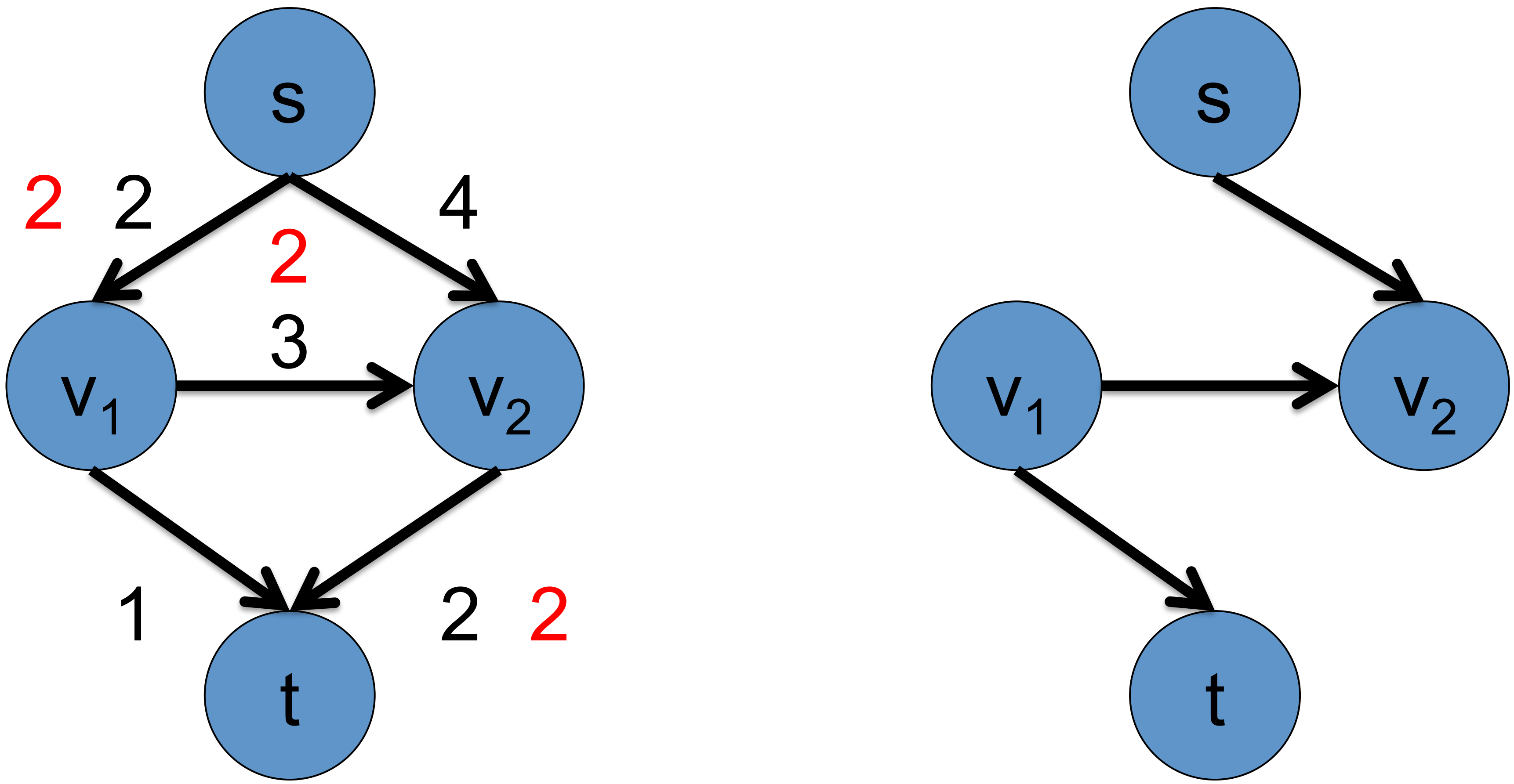
Another method?

Incorrect Answer !!

Outline

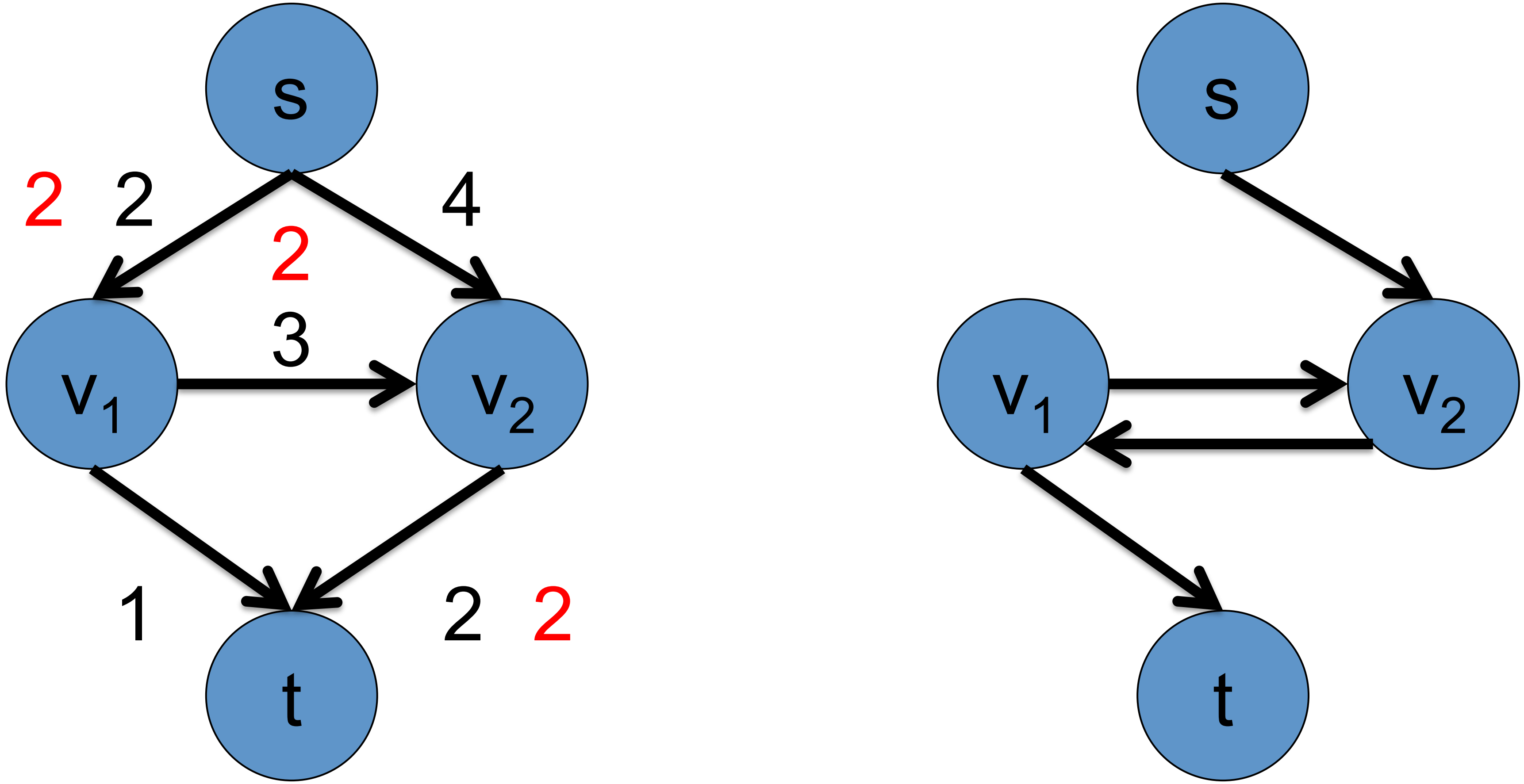
- Preliminaries
- Maximum Flow
  - **Residual Graph**
  - Max-Flow Min-Cut Theorem
- Algorithms
- Energy minimization with max flow/min cut

Residual Graph



Arcs where flow(a) < c(a)

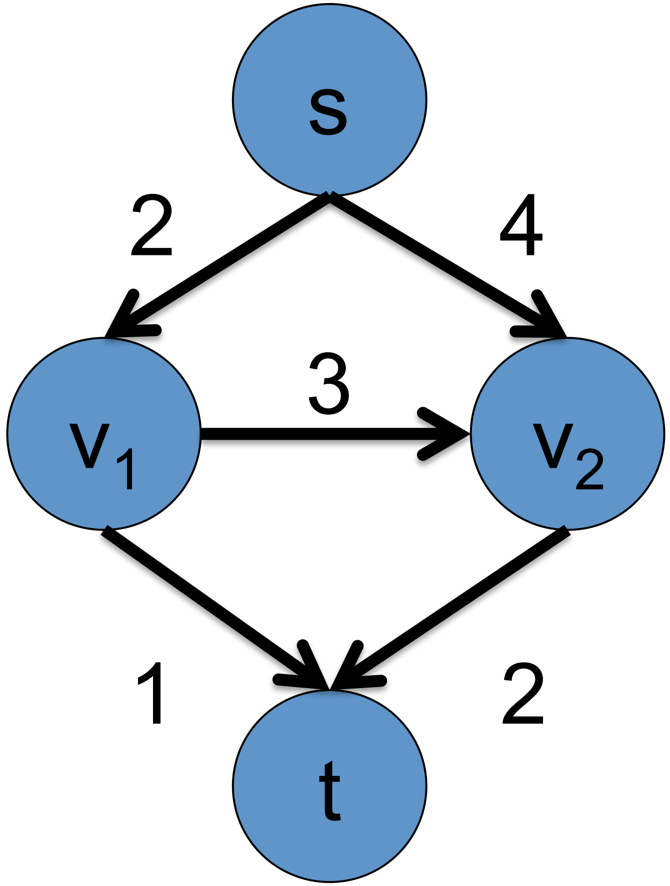
Residual Graph

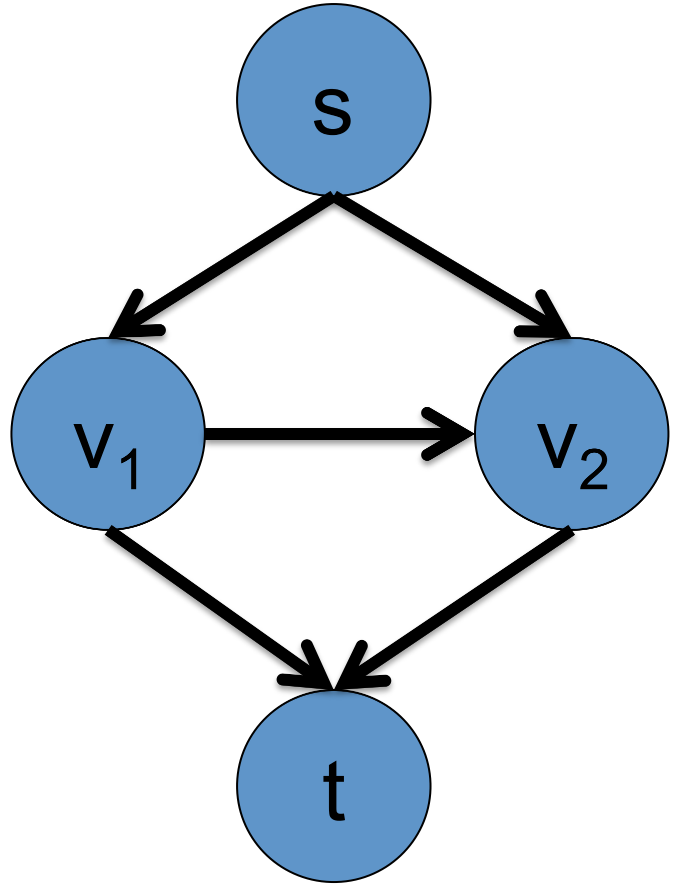


Including arcs to s and from t is not necessary

Inverse of arcs where flow(a) > 0

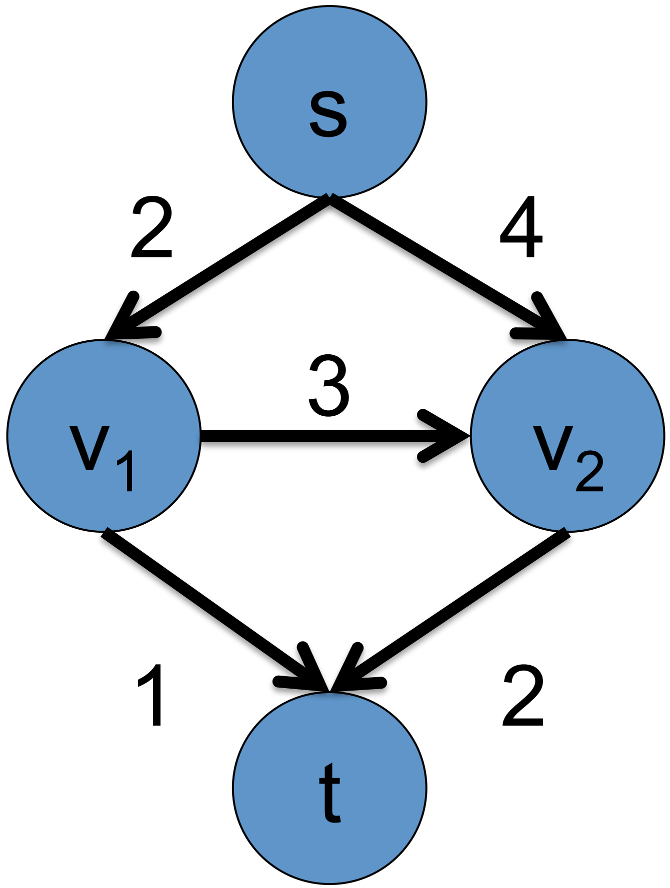
Maximum Flow using Residual Graphs

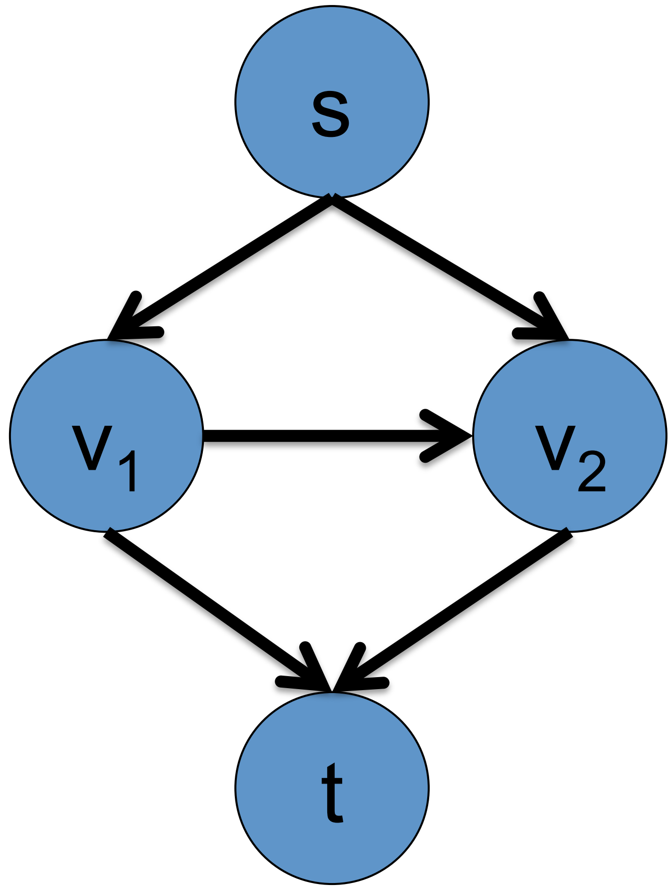




Start with zero flow.

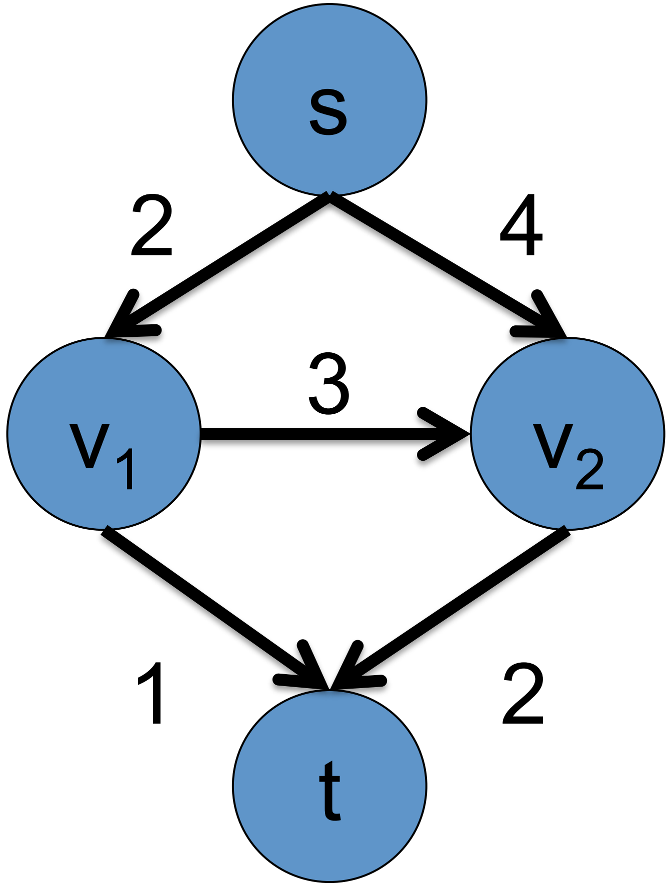
Maximum Flow using Residual Graphs

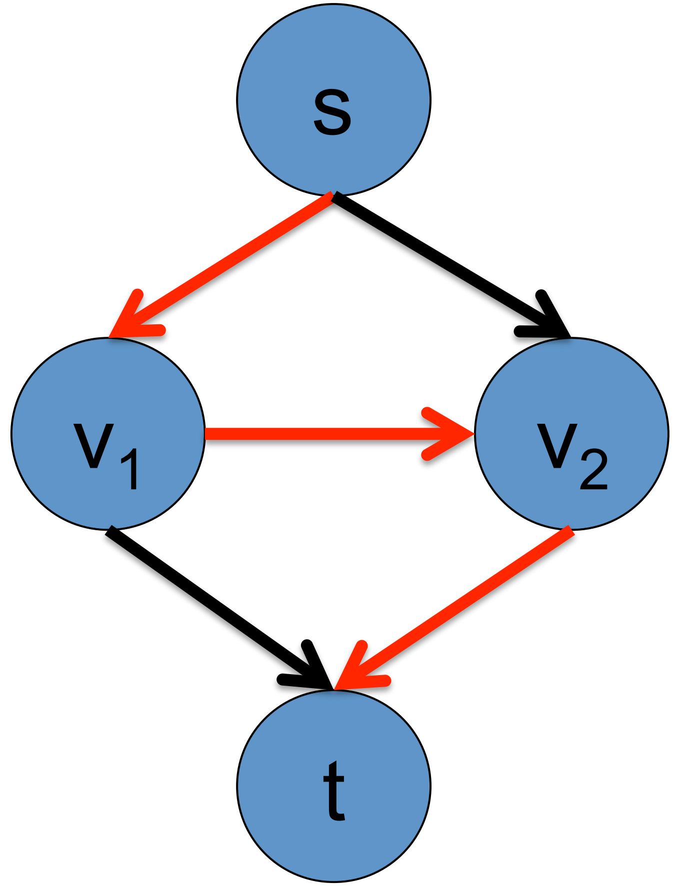




Find an s-t path in the residual graph.

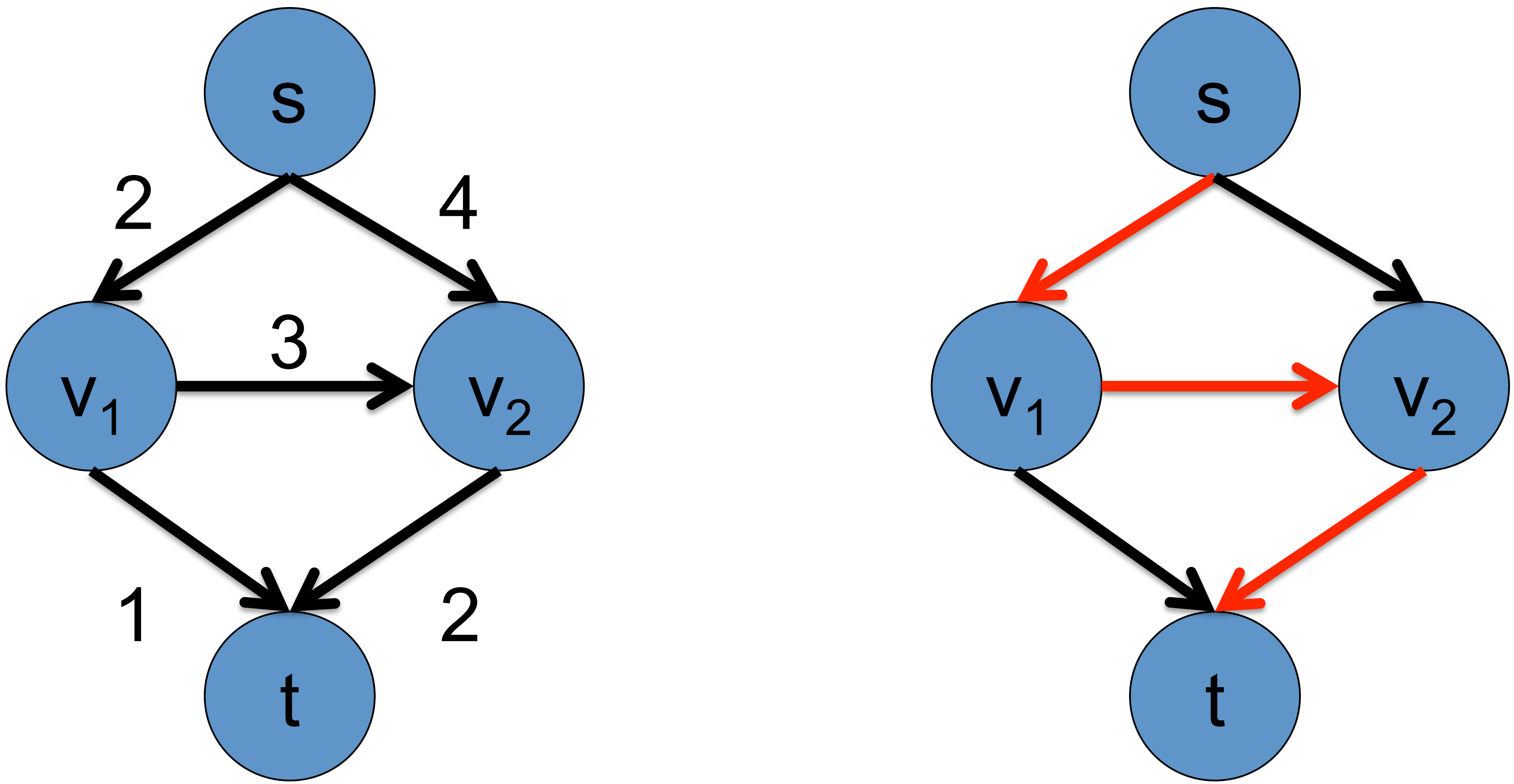
Maximum Flow using Residual Graphs





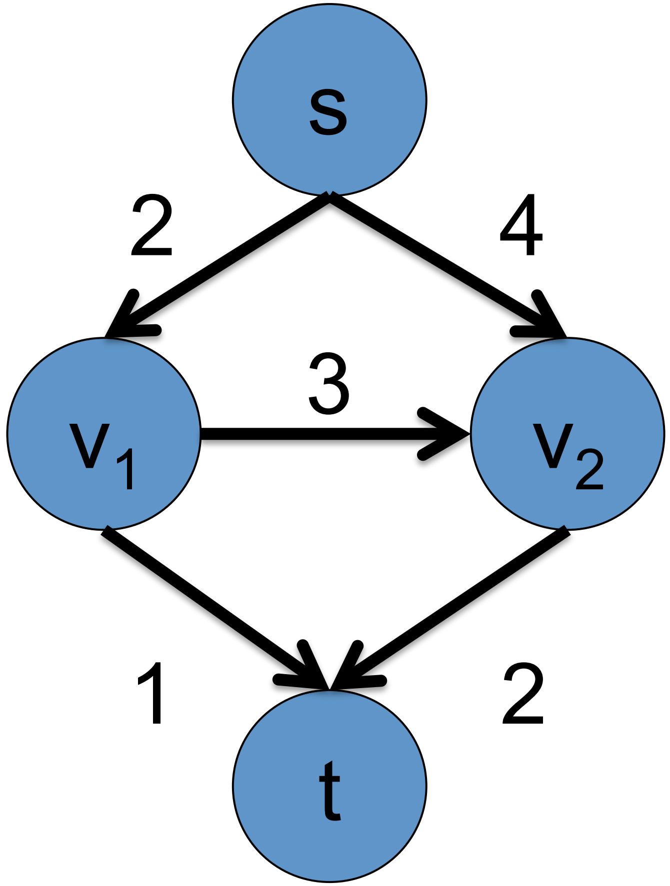
Find an s-t path in the residual graph.

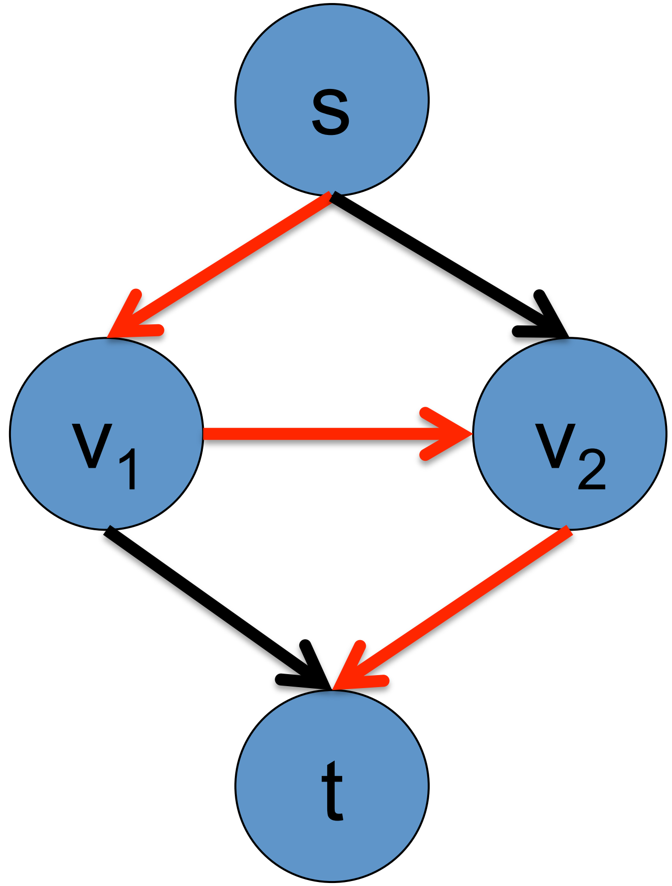
Maximum Flow using Residual Graphs



For inverse arcs in path, subtract flow K.

Maximum Flow using Residual Graphs

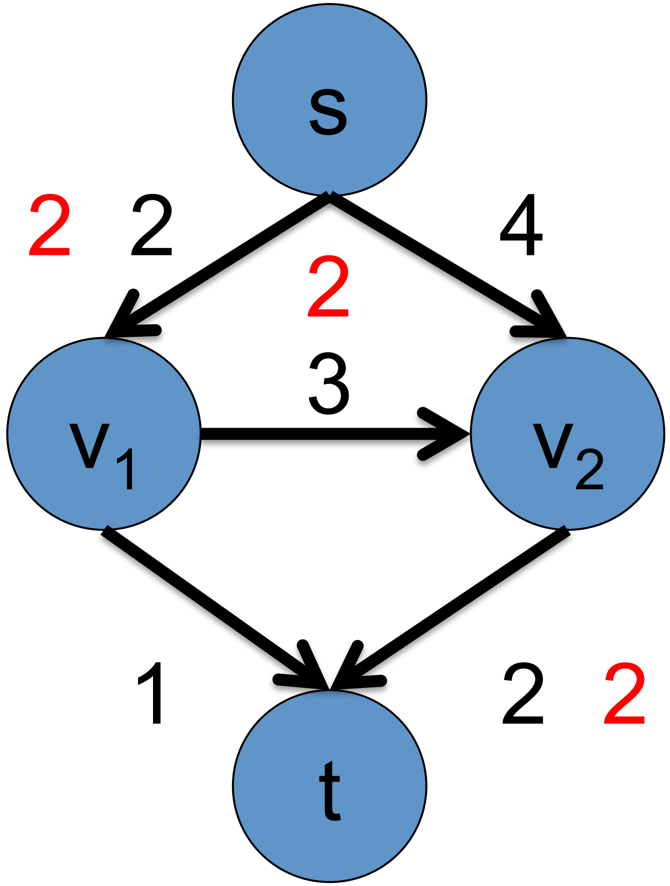


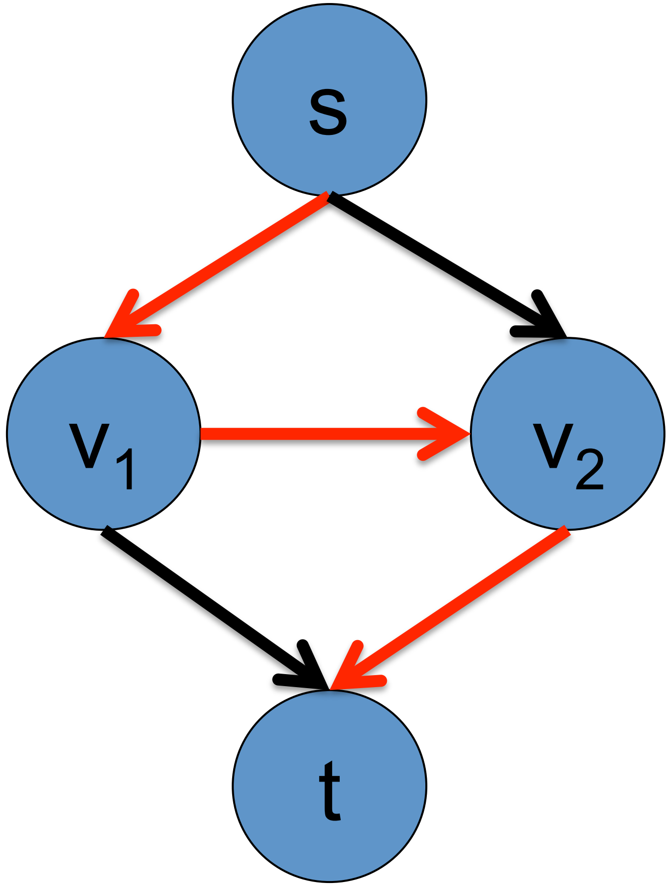


Choose maximum allowable value of K.

For forward arcs in path, add flow K.

Maximum Flow using Residual Graphs

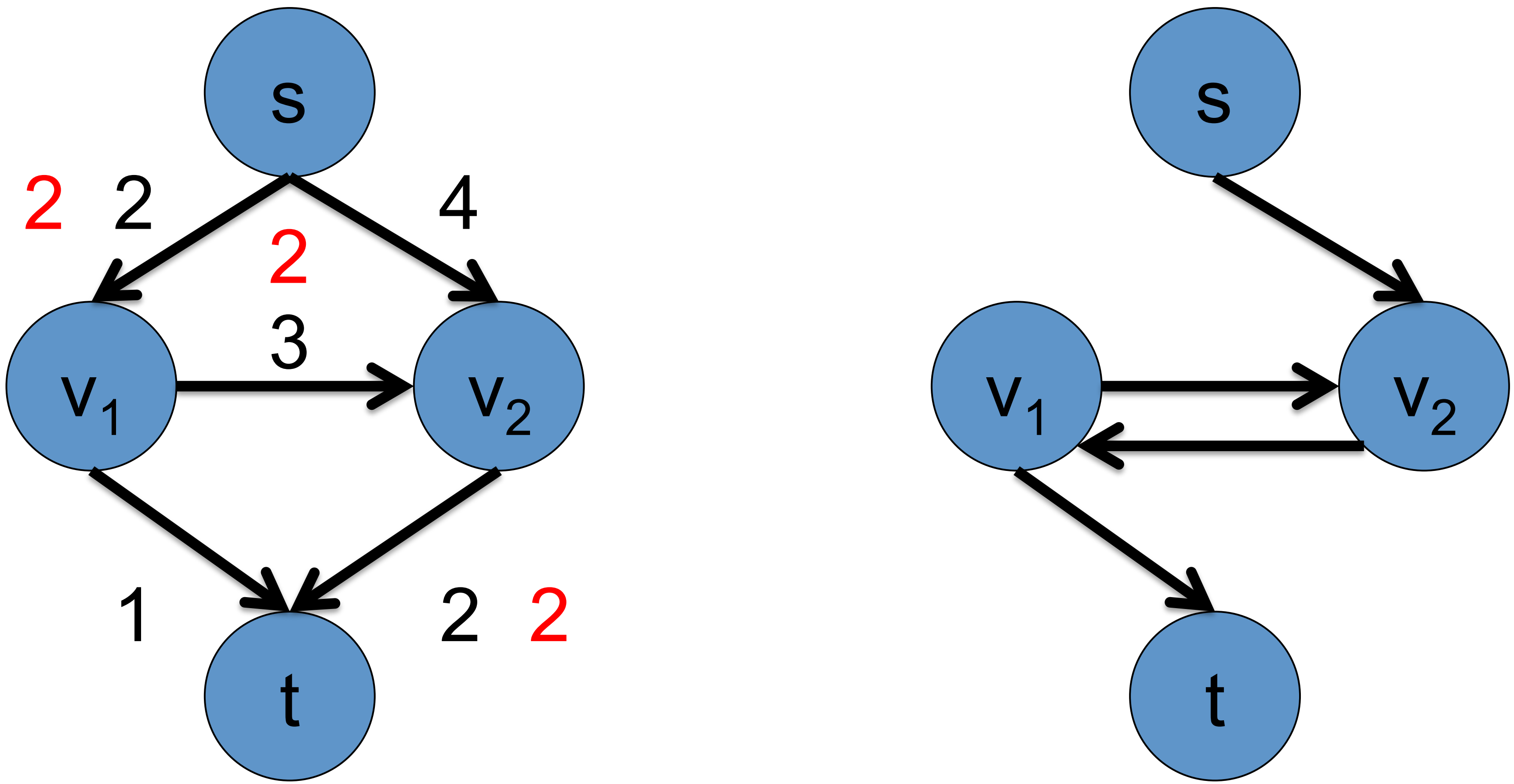




Choose maximum allowable value of K.

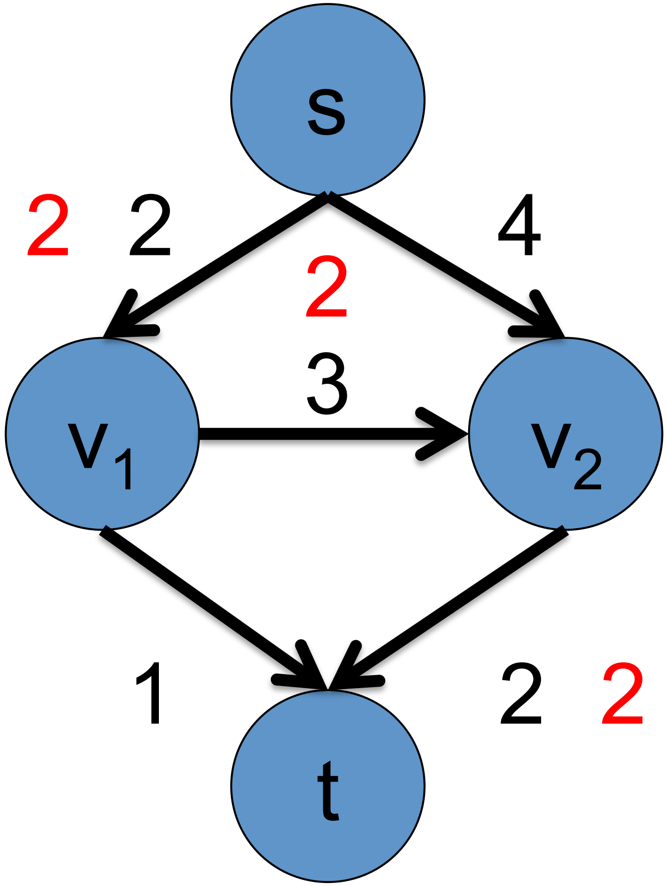
For forward arcs in path, add flow K.

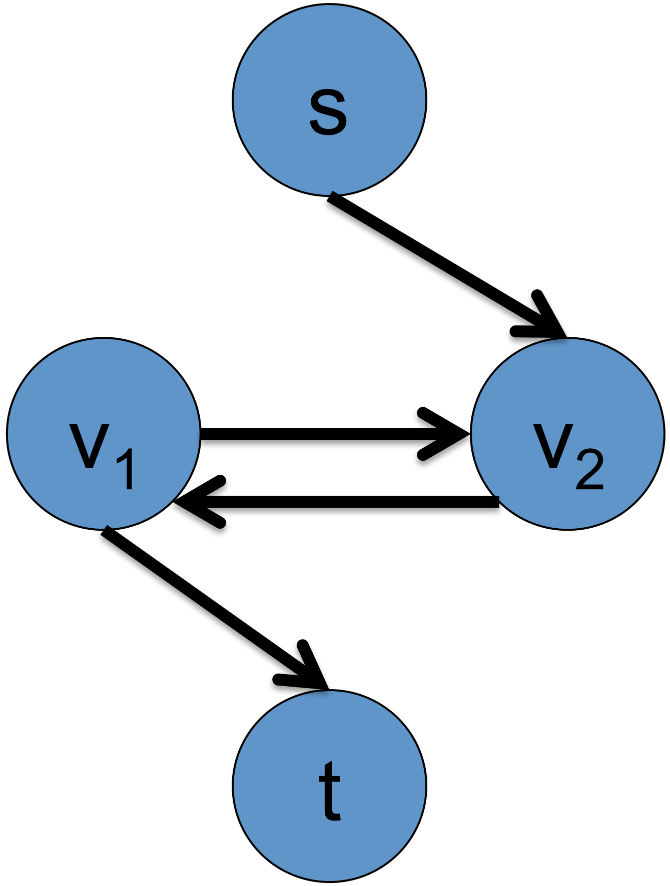
Maximum Flow using Residual Graphs



Update the residual graph.

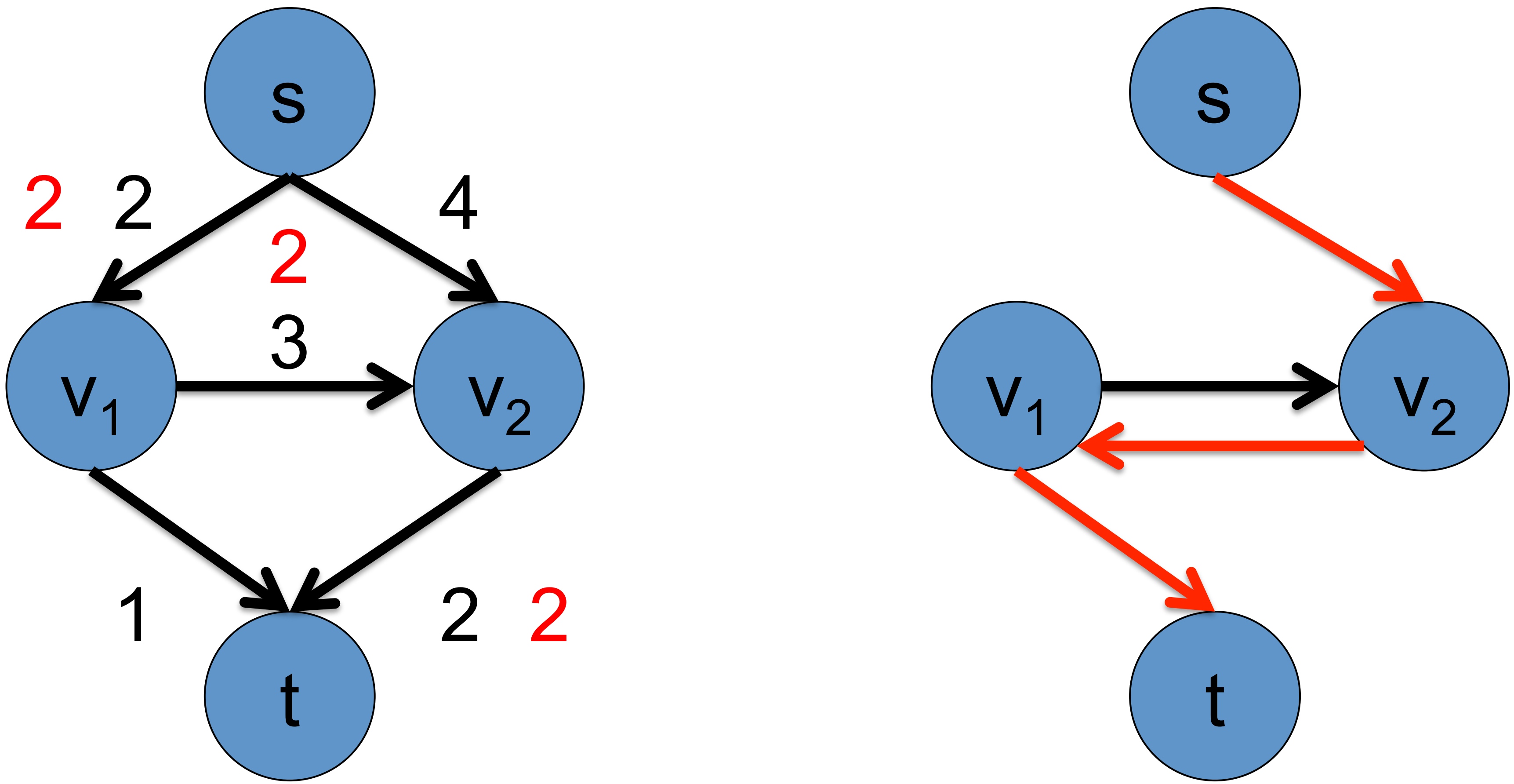
Maximum Flow using Residual Graphs





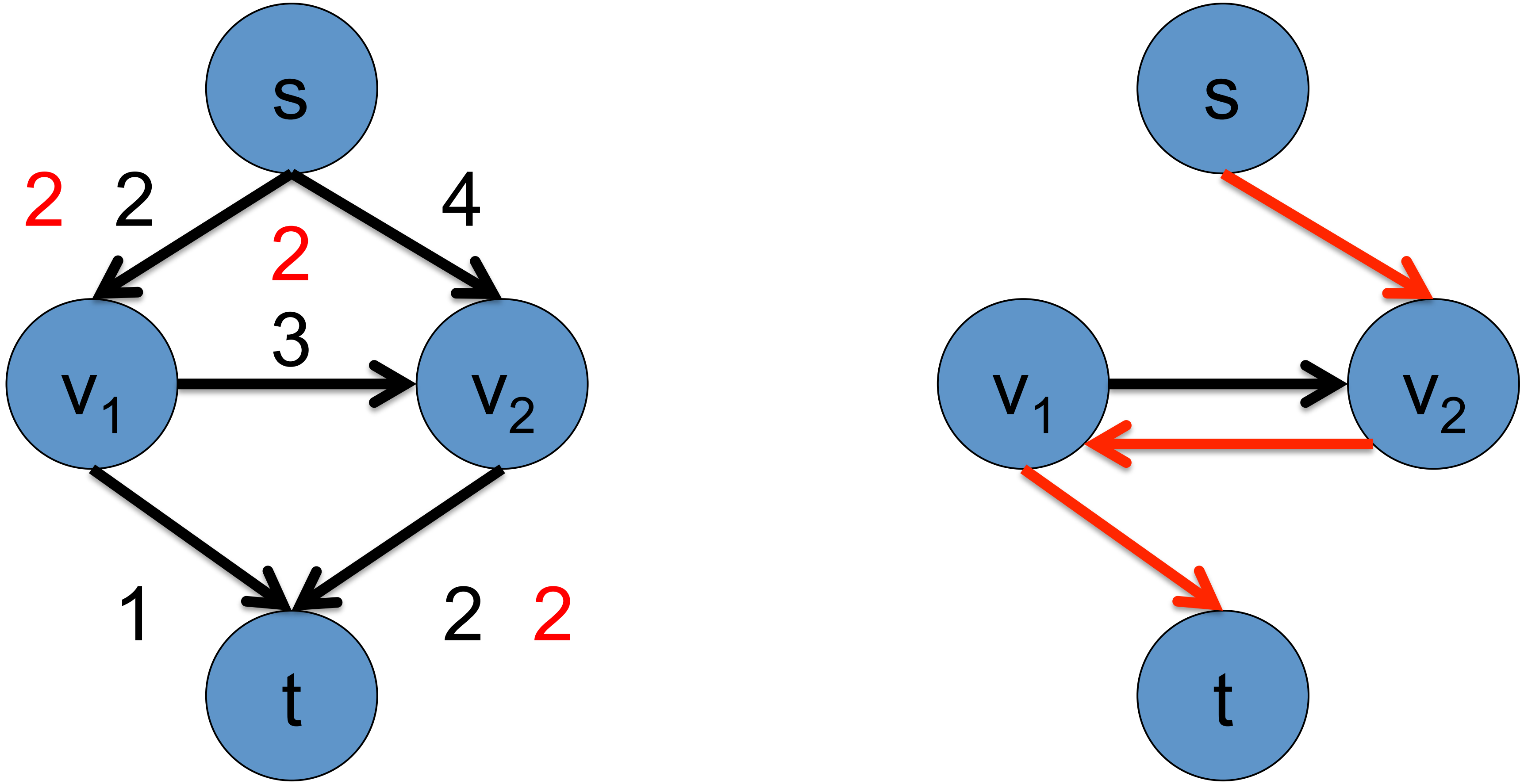
Find an s-t path in the residual graph.

Maximum Flow using Residual Graphs



Find an s-t path in the residual graph.

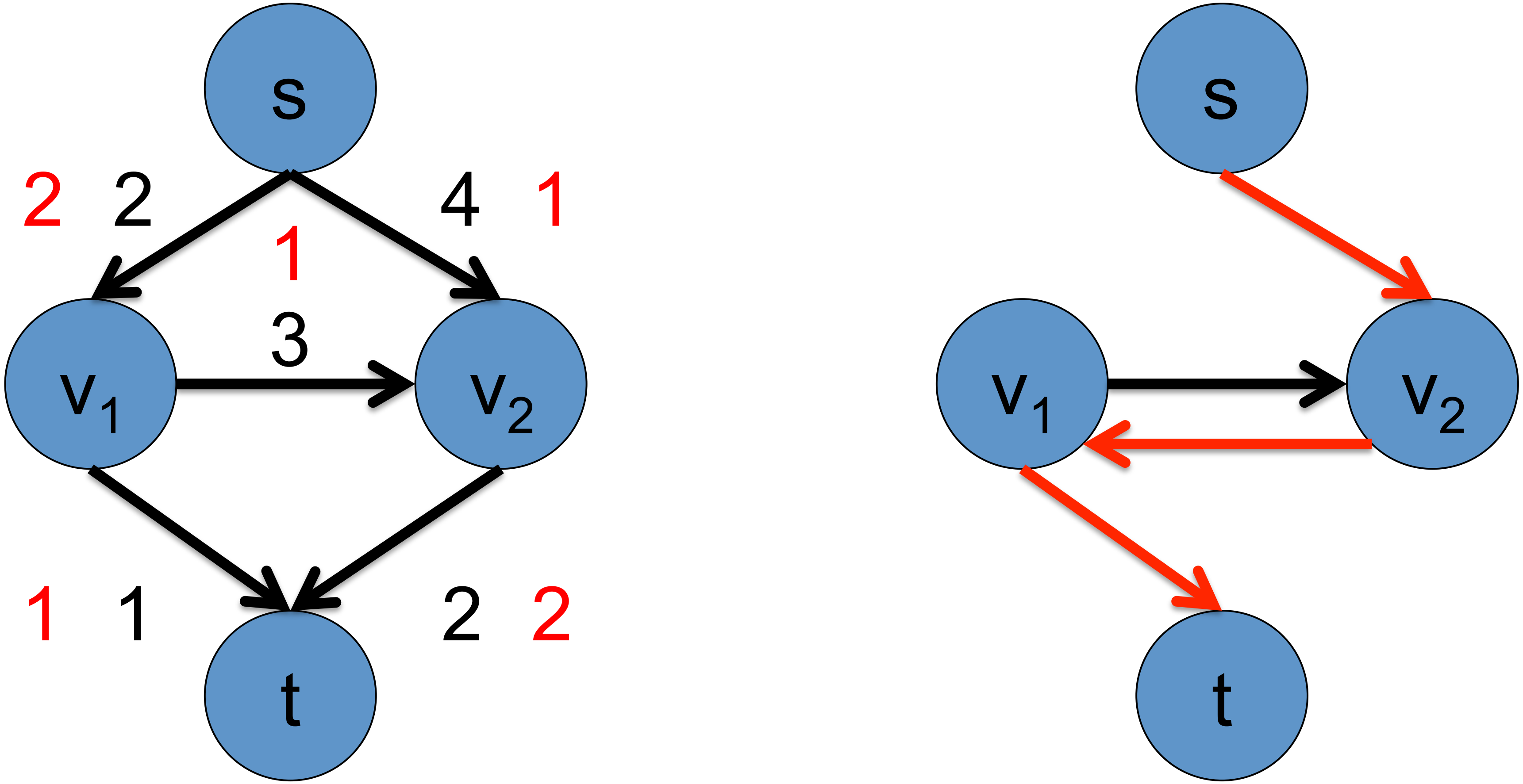
Maximum Flow using Residual Graphs



Choose maximum allowable value of K.

Add K to (s,v_2) and (v_1,t). Subtract K from (v_1,v_2).

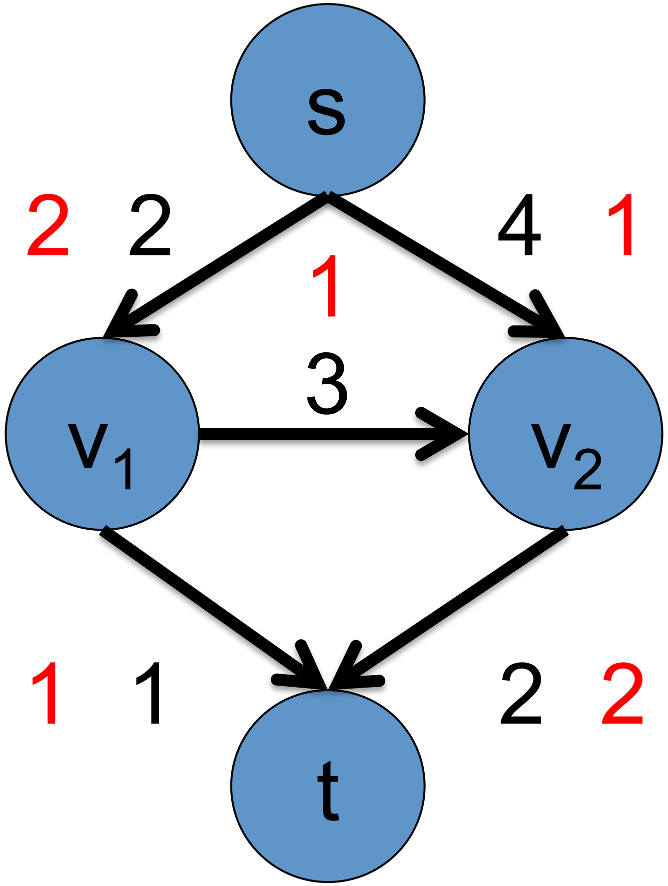
Maximum Flow using Residual Graphs

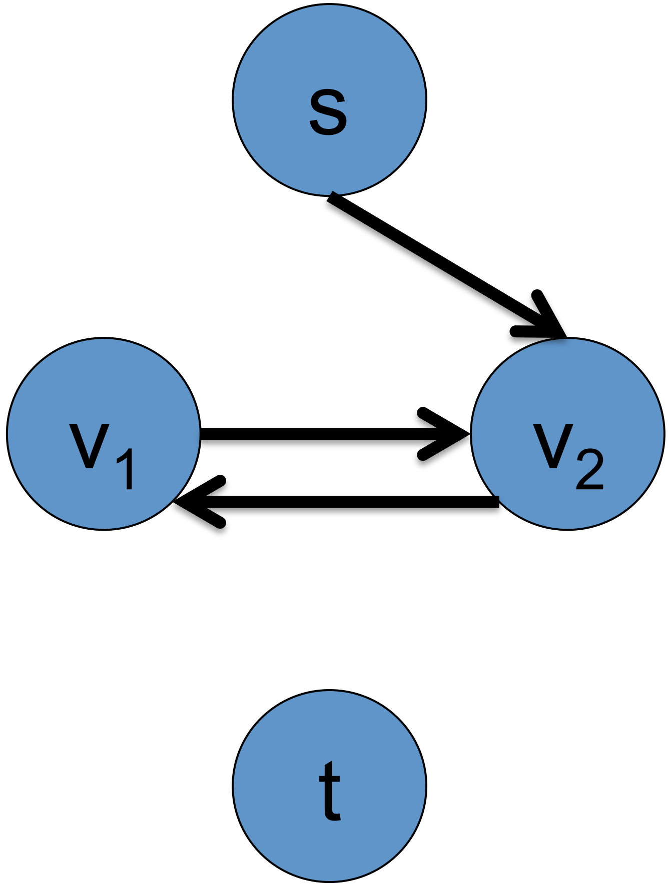


Choose maximum allowable value of K.

Add K to (s,v_2) and (v_1,t). Subtract K from (v_1,v_2).

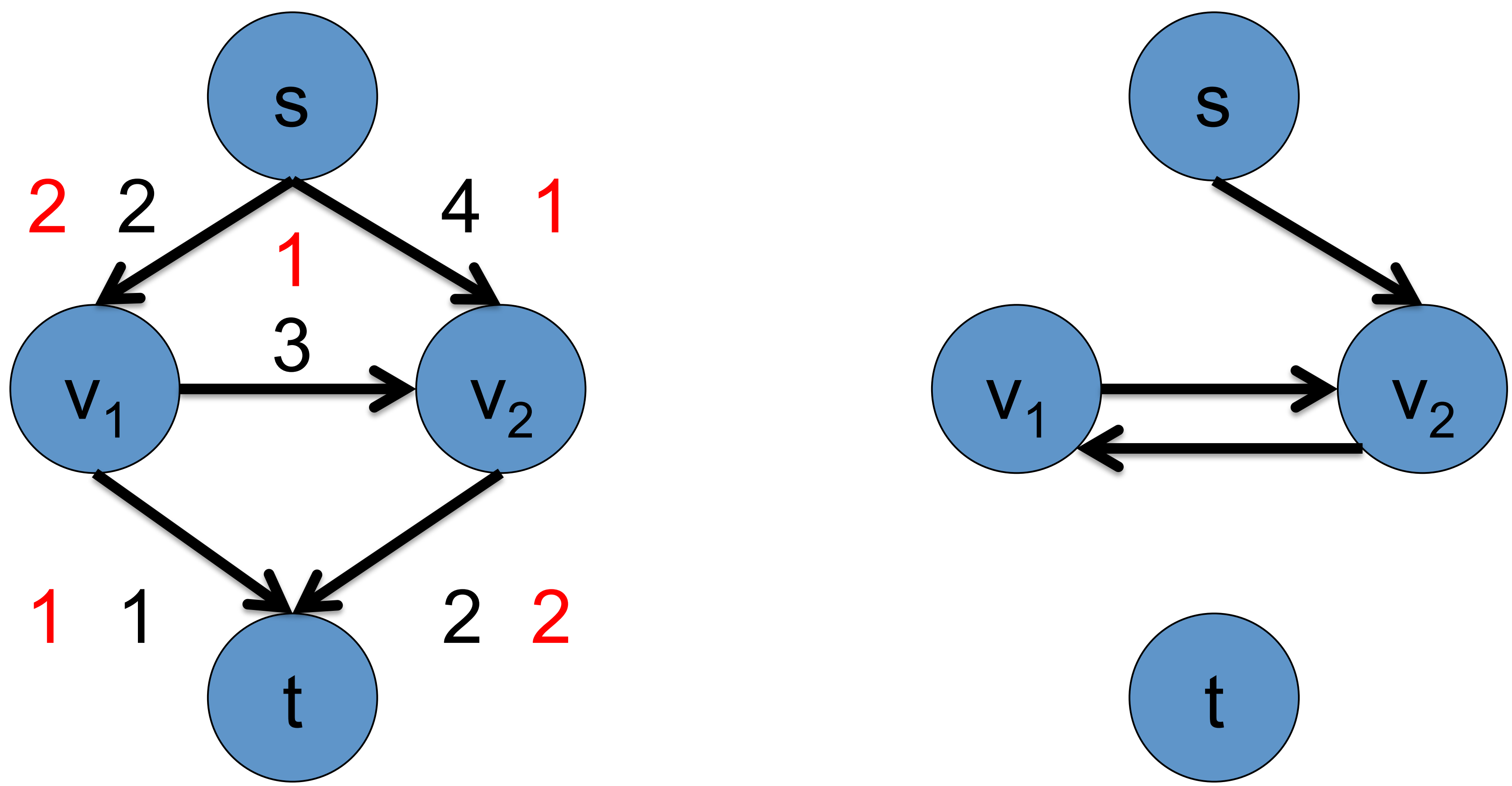
Maximum Flow using Residual Graphs





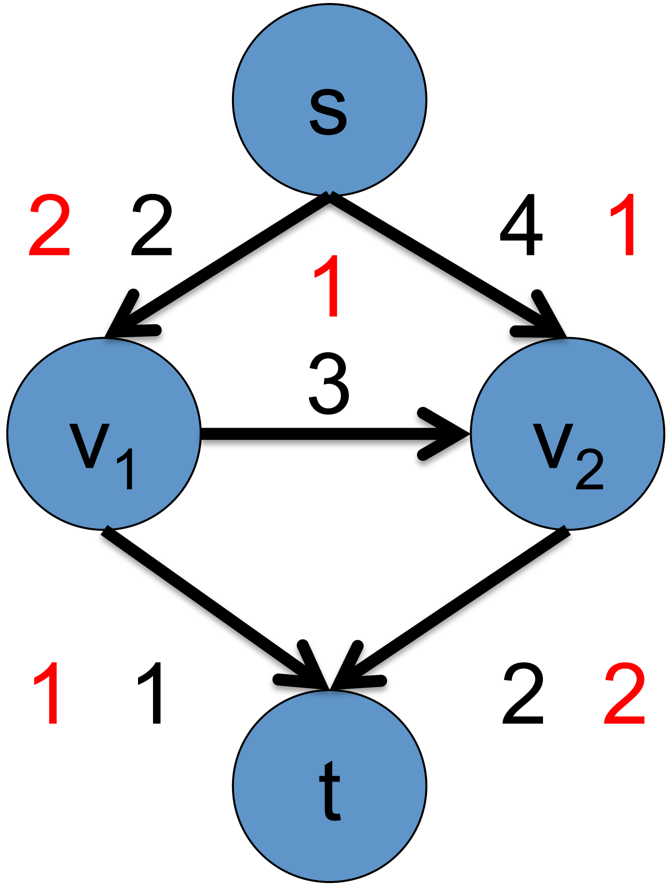
Update the residual graph.

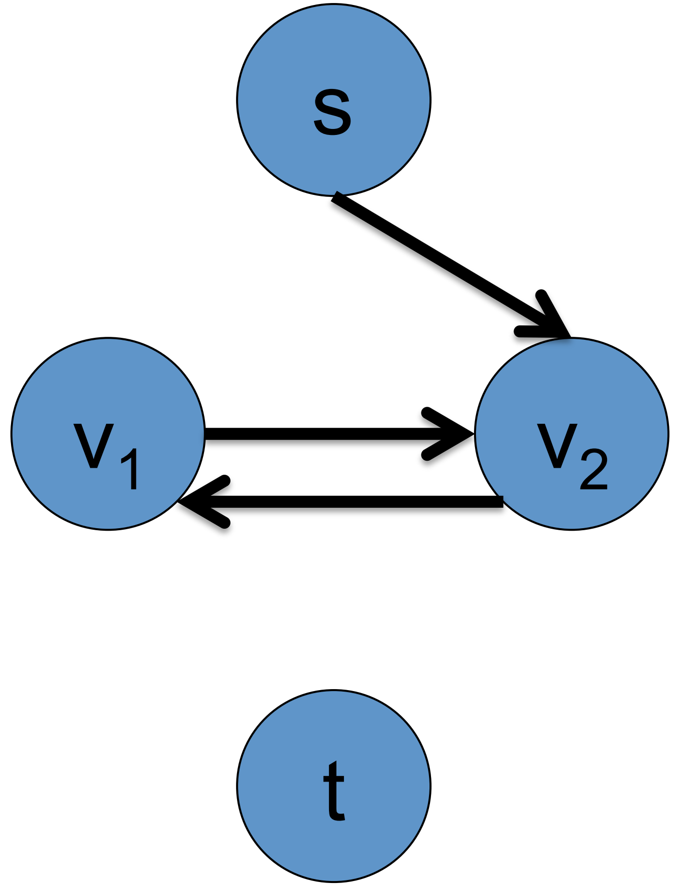
Maximum Flow using Residual Graphs



Find an s-t path in the residual graph.

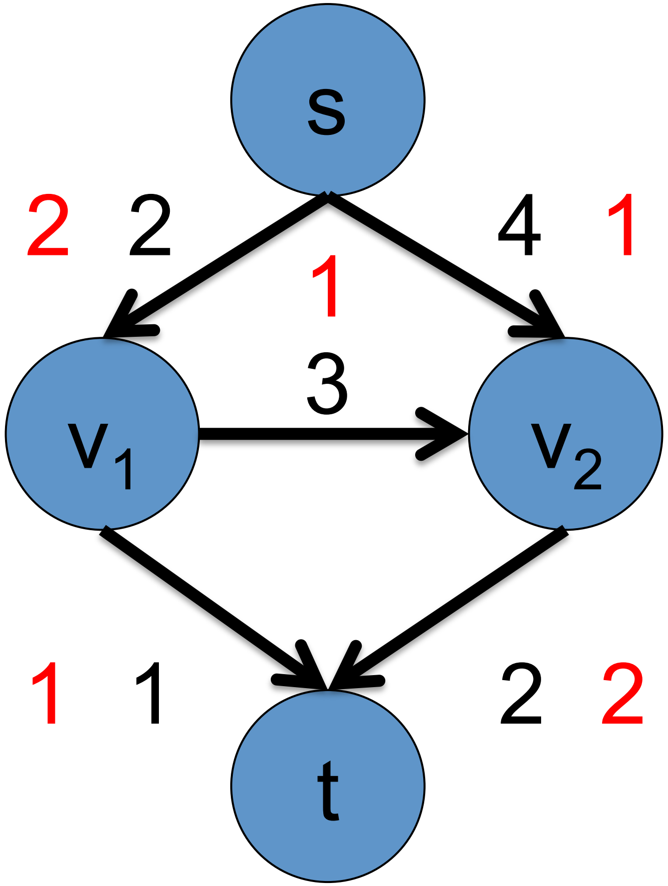
Maximum Flow using Residual Graphs

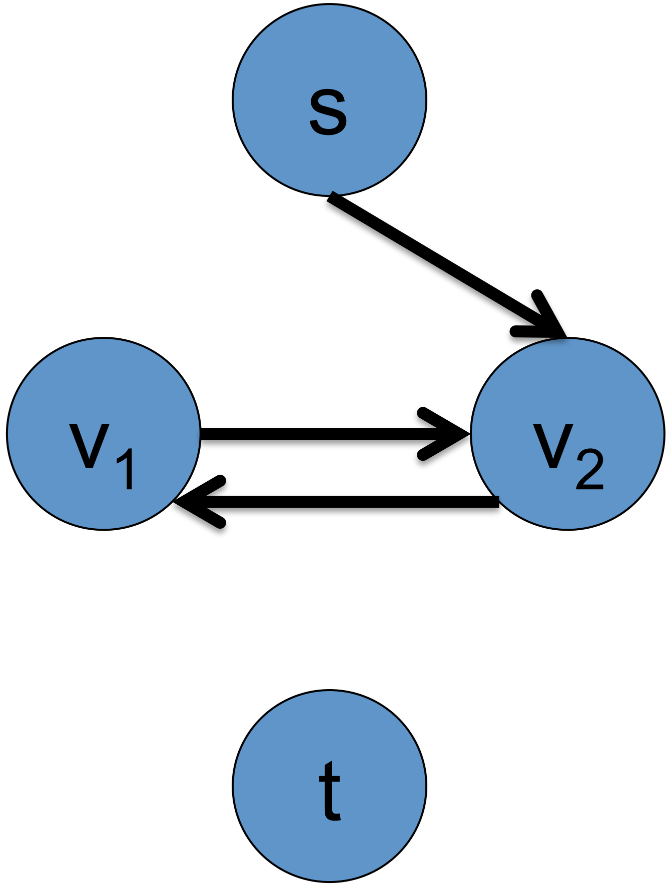




No more s-t paths. Stop.

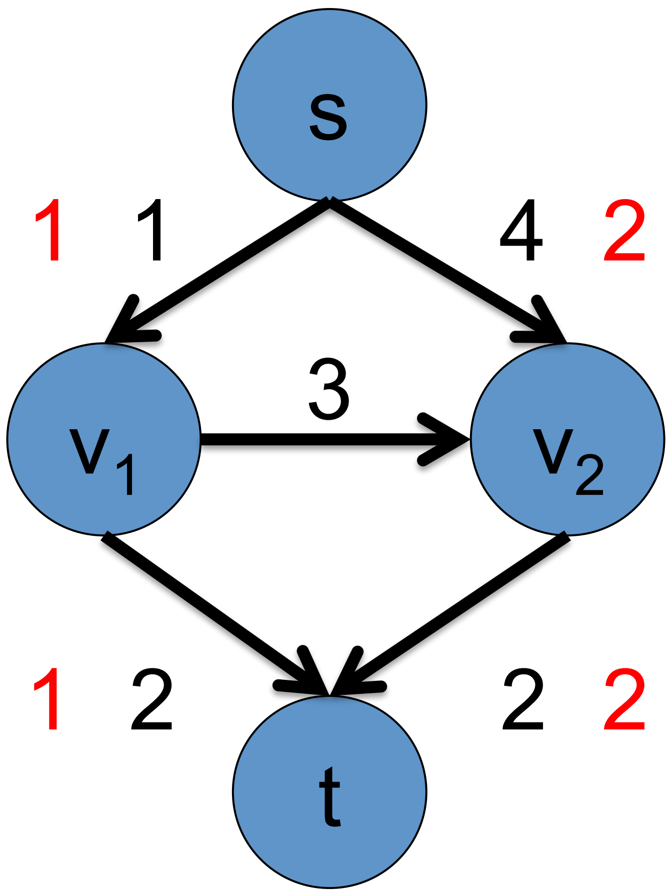
Maximum Flow using Residual Graphs

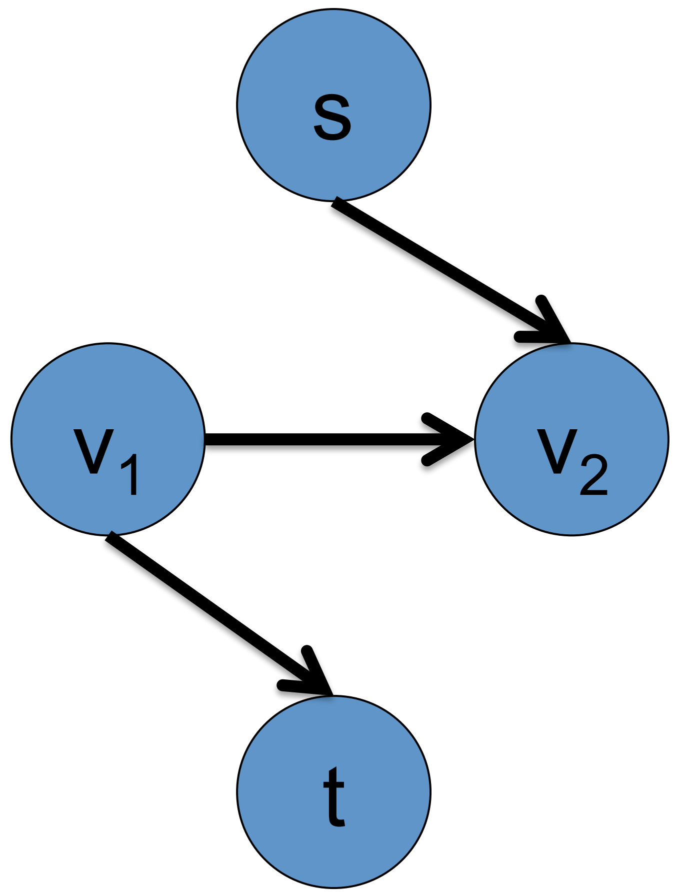




Correct Answer.

Maximum Flow using Residual Graphs



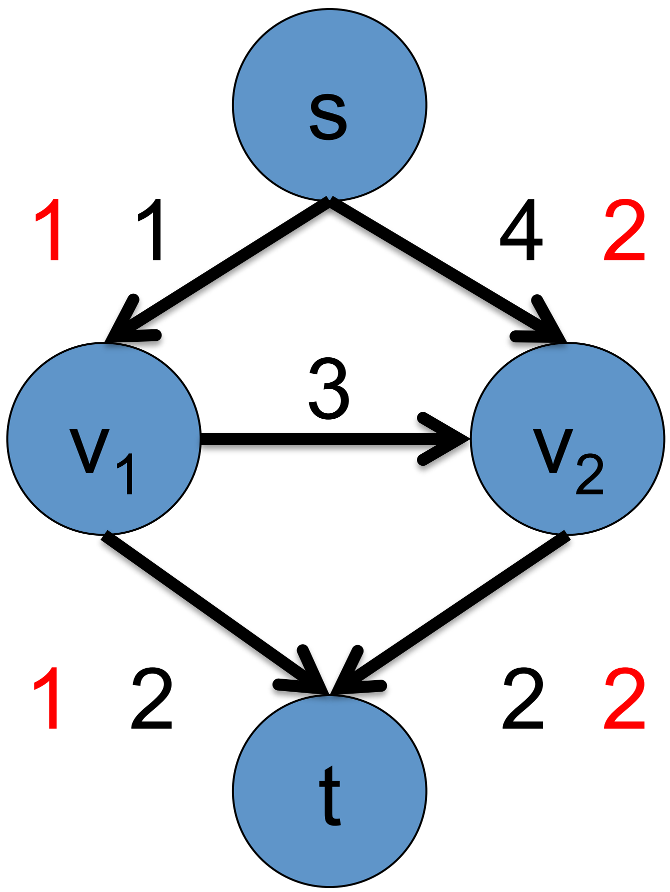


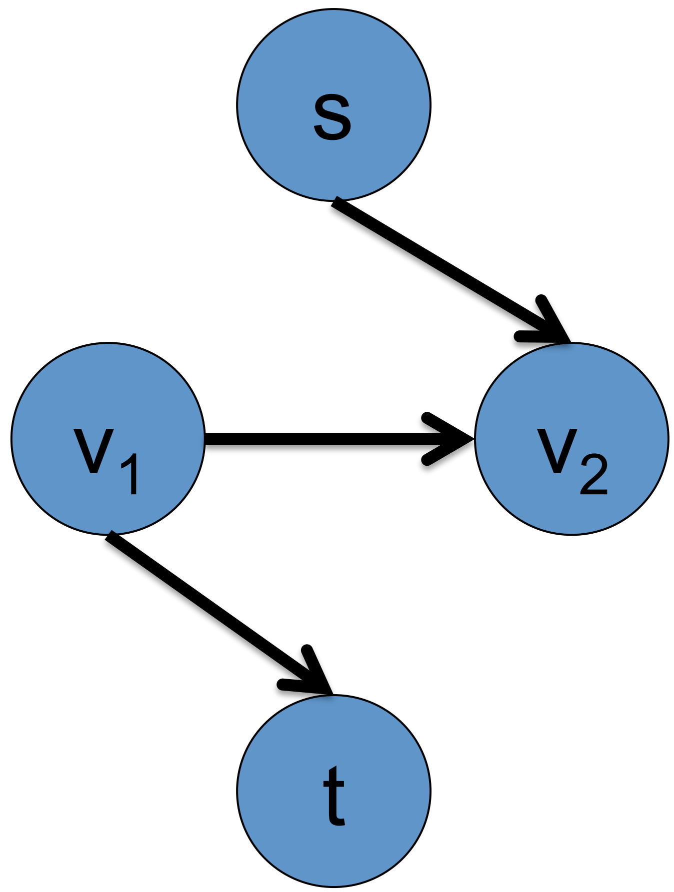
How can I be sure this will always work?

Outline

- Preliminaries
- Maximum Flow
  - Residual Graph
  - **Max-Flow Min-Cut Theorem**
- Algorithms
- Energy minimization with max flow/min cut

Max-Flow Min-Cut

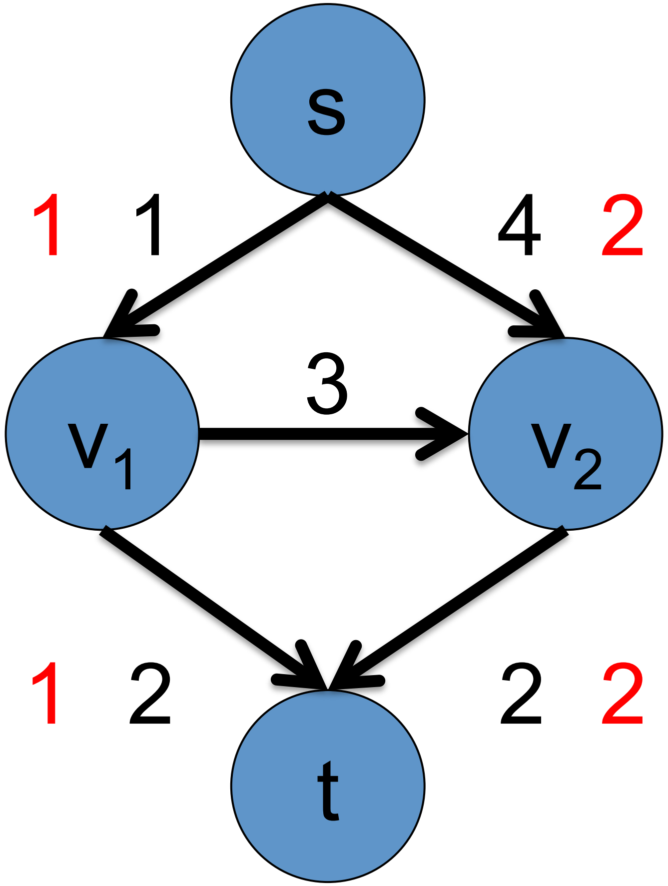


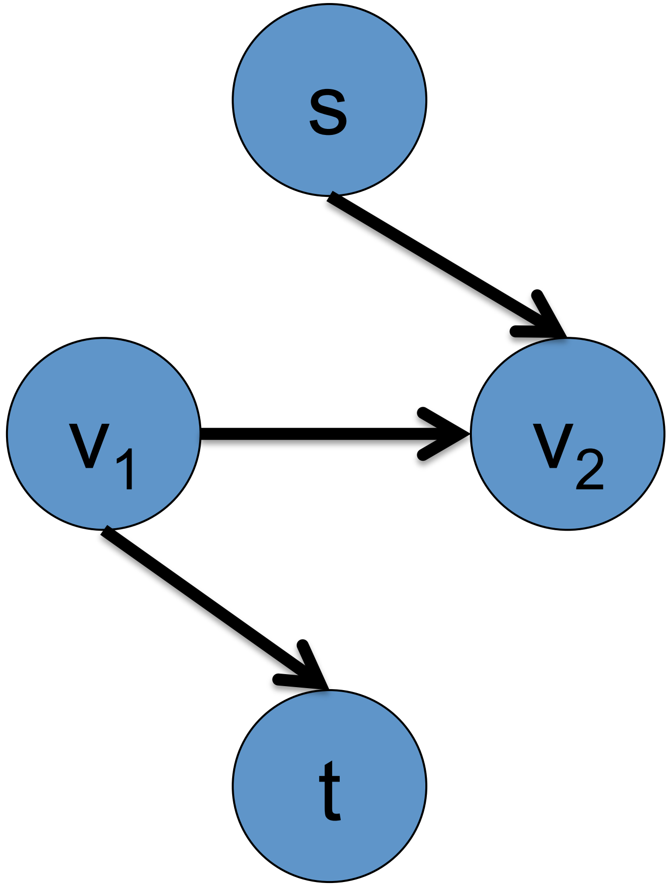


t is not in U.

Let the subset of vertices U be reachable from s.

Max-Flow Min-Cut

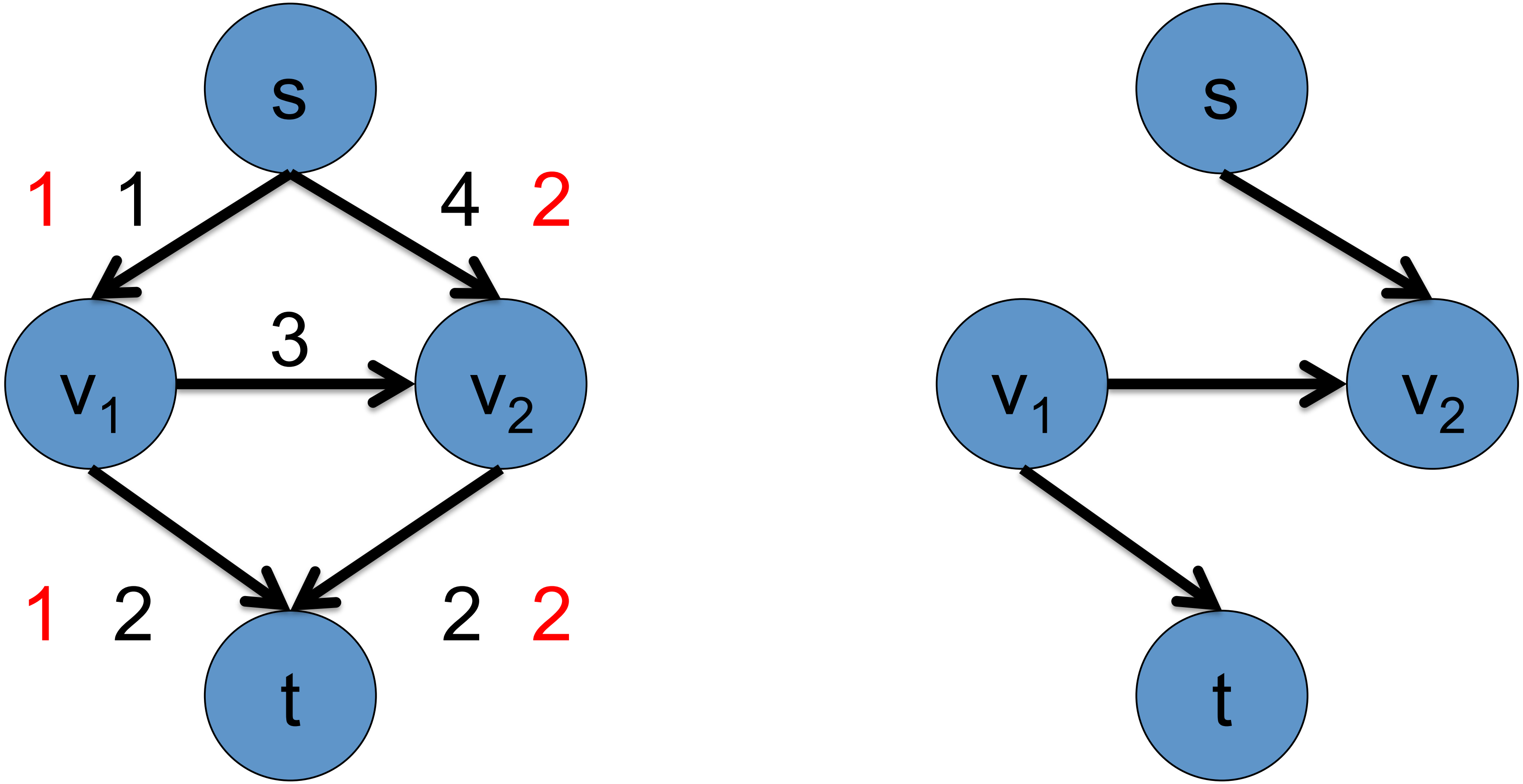




Or else a will be in the residual graph.

For all a ∈ out-arcs(U), flow(a) = c(a).

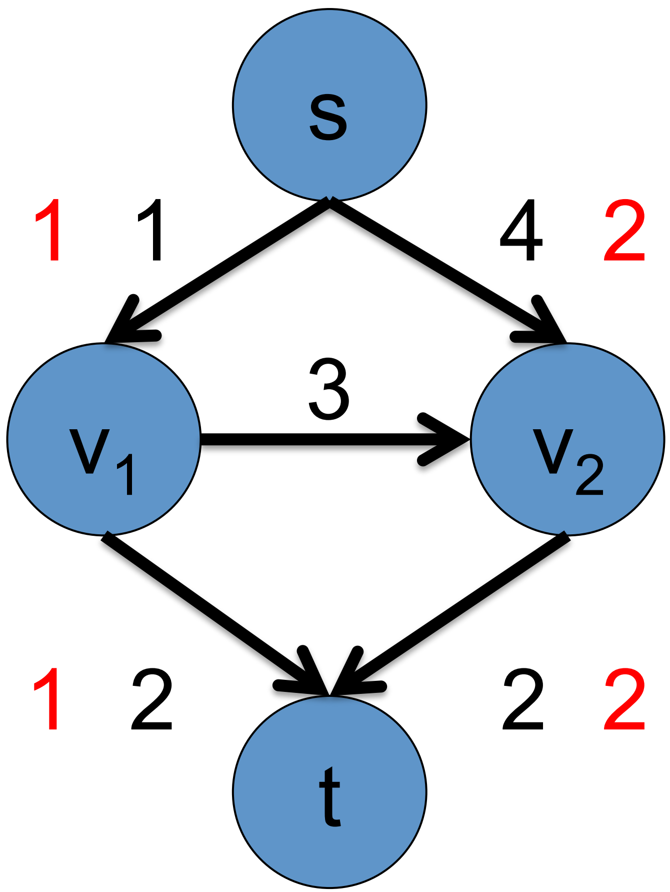
Max-Flow Min-Cut

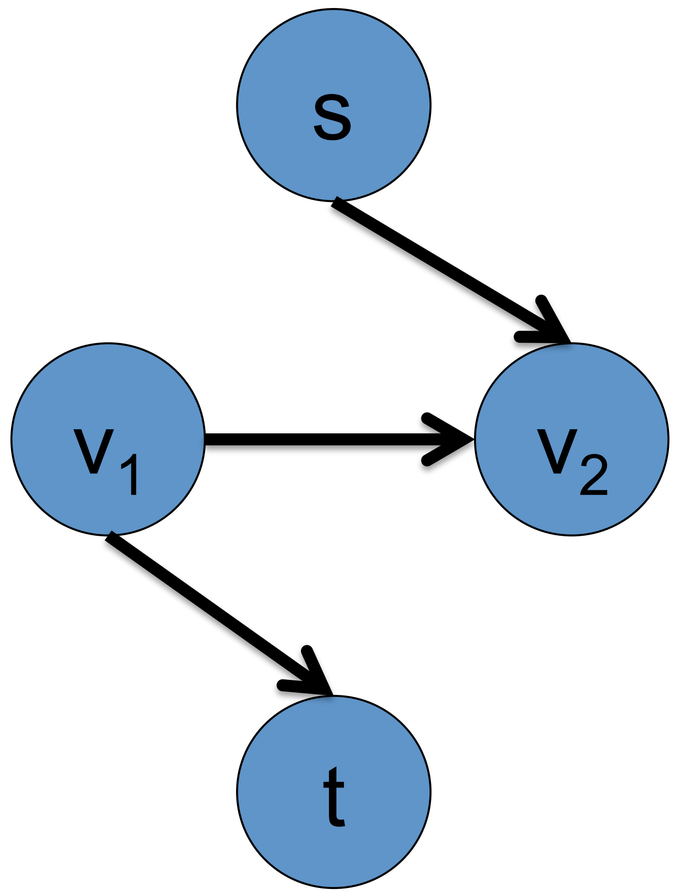


Or else inverse of a will be in the residual graph.

For all a ∈ in-arcs(U), flow(a) = 0.

Max-Flow Min-Cut





For all a ∈ out-arcs(U), flow(a) = c(a).

For all a ∈ in-arcs(U), flow(a) = 0.

Flows vs. Cuts

Value of flow $= -E_{flow}(s)$

$= -E_{flow}(s) - \Sigma_{v \in U \setminus \{s\}} E_{flow}(v)$

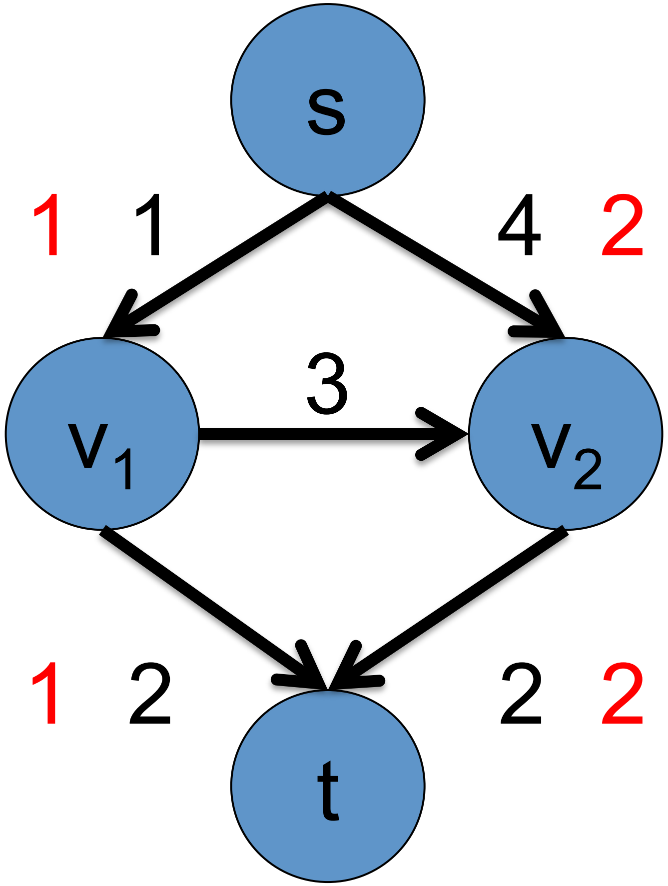
$= -E_{flow}(U)$

$=$ flow(out-arcs(U)) - flow(in-arcs(U))

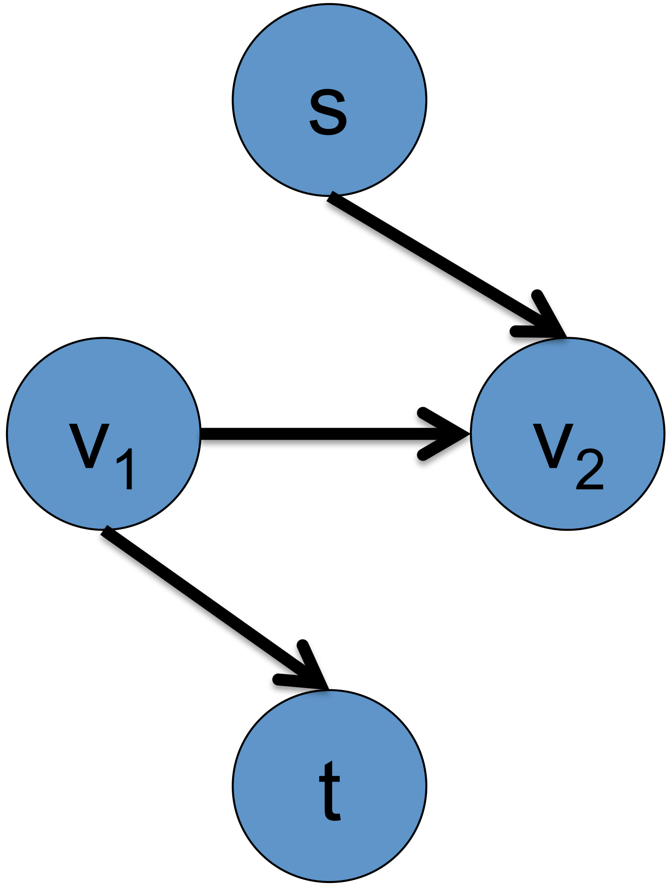
$=$ Capacity of C

flow(a) = c(a), a $\in$ out-arcs(U)   flow(a) = 0, a $\in$ in-arcs(U)

Max-Flow Min-Cut



Minimum Cut



Maximum Flow

Capacity(C) = Value(flow)

Outline

- Preliminaries
- Maximum Flow
- **Algorithms**
  - **Ford-Fulkerson Algorithm**
  - Dinits Algorithm
- Energy minimization with max flow/min cut

Ford-Fulkerson Algorithm

Start with flow = 0 for all arcs.

Find an s-t path in the residual graph.

Pass maximum allowable flow.

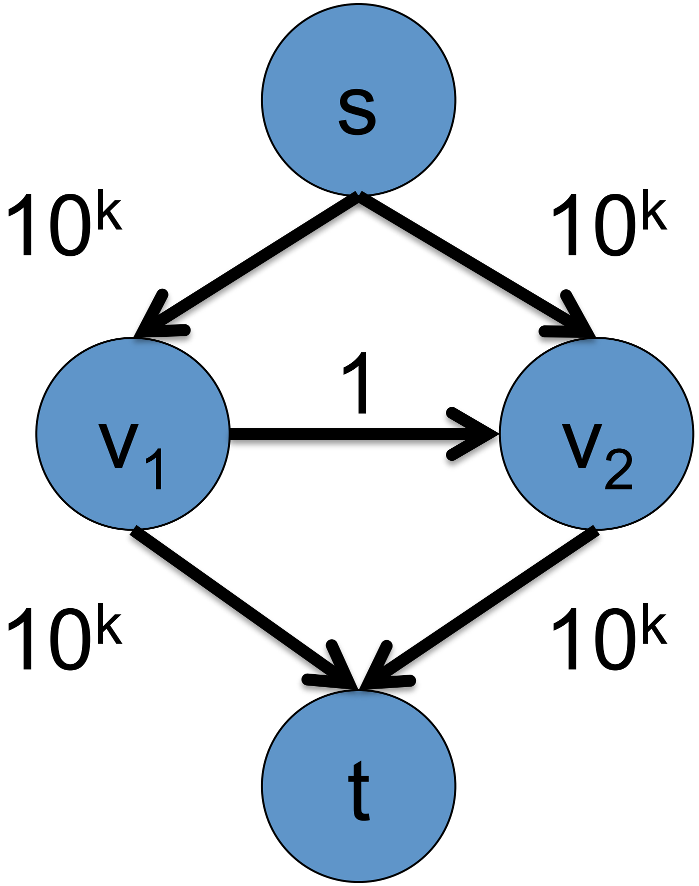
Subtract from inverse arcs.

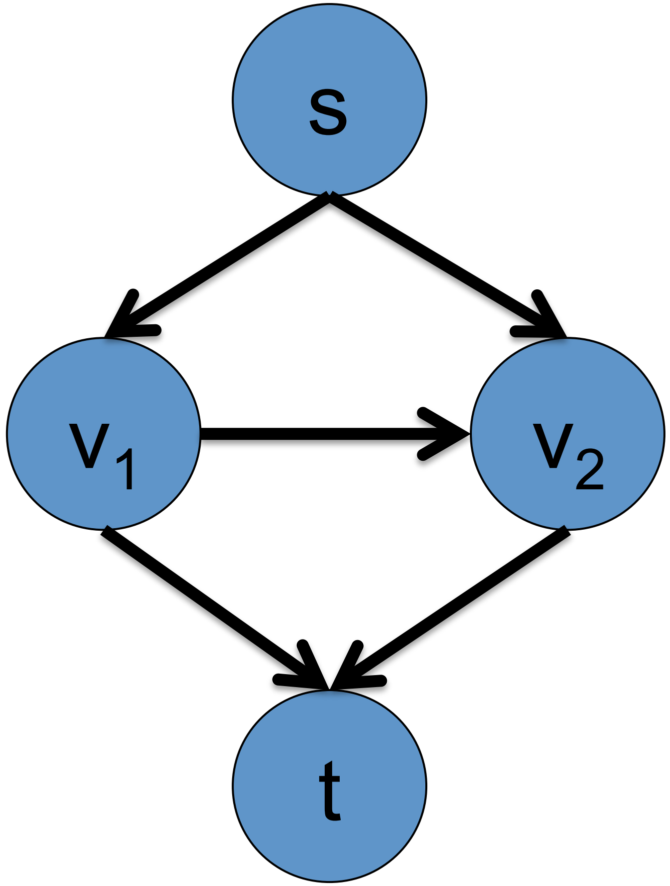
Add to forward arcs.

REPEAT

Until s and t are disjoint in the residual graph.

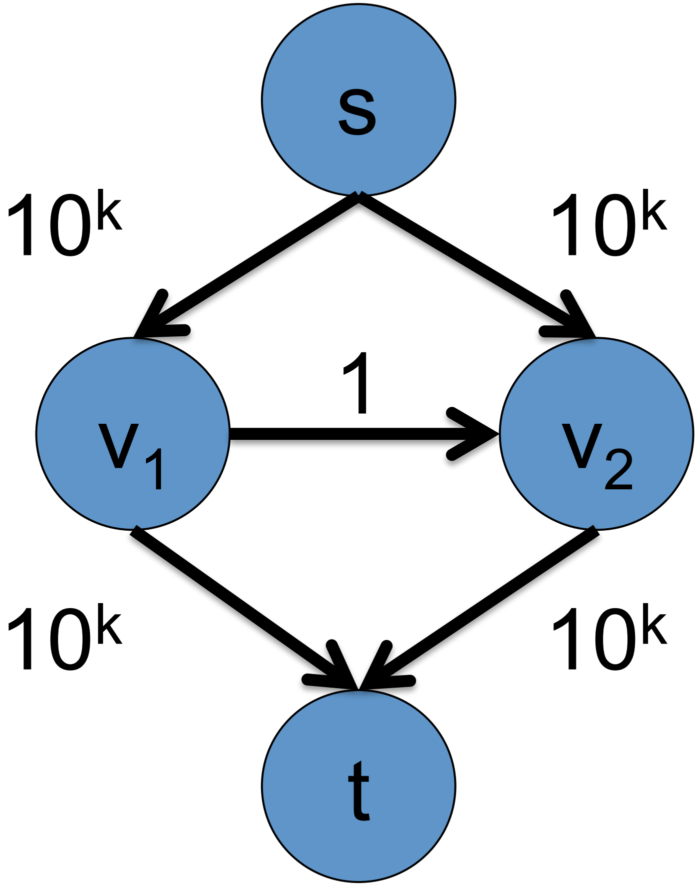
Ford-Fulkerson Algorithm

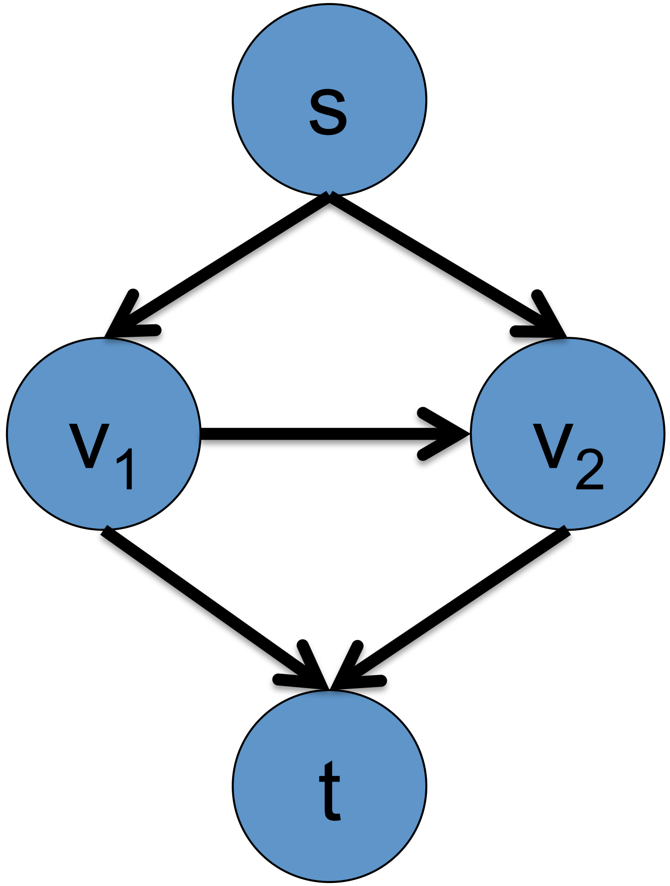




Start with zero flow

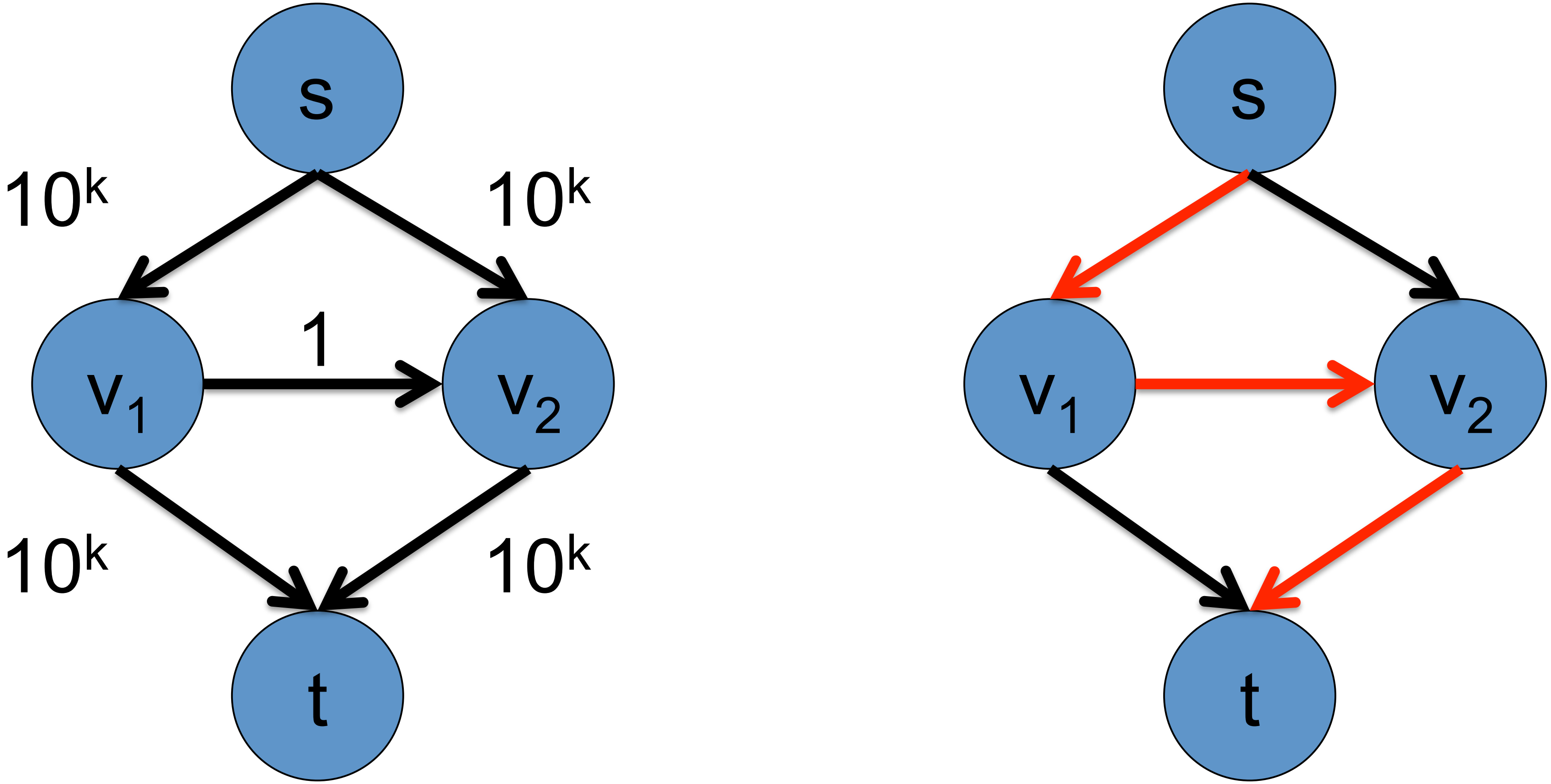
Ford-Fulkerson Algorithm





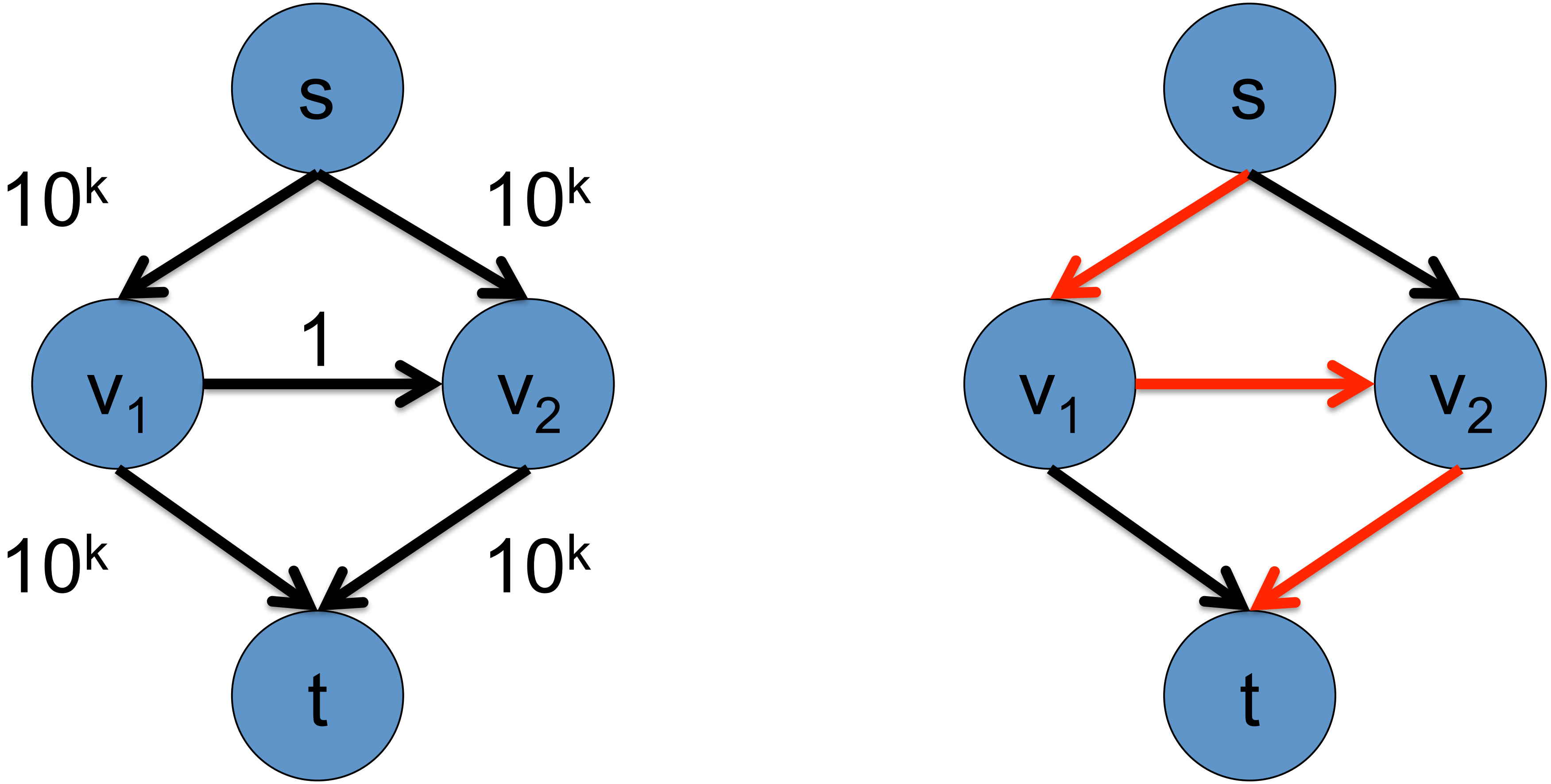
Find an s-t path in the residual graph.

Ford-Fulkerson Algorithm



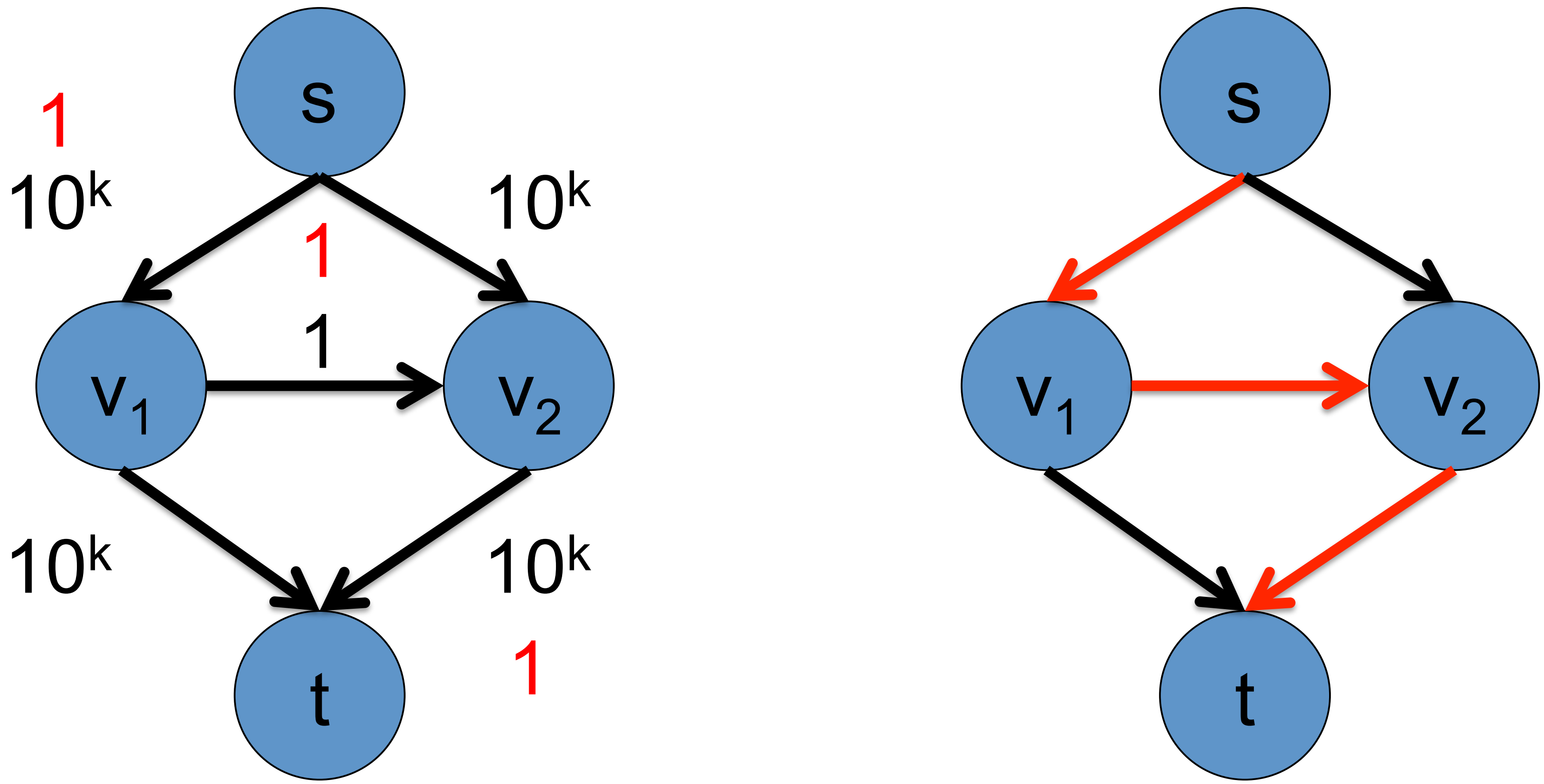
Find an s-t path in the residual graph.

Ford-Fulkerson Algorithm



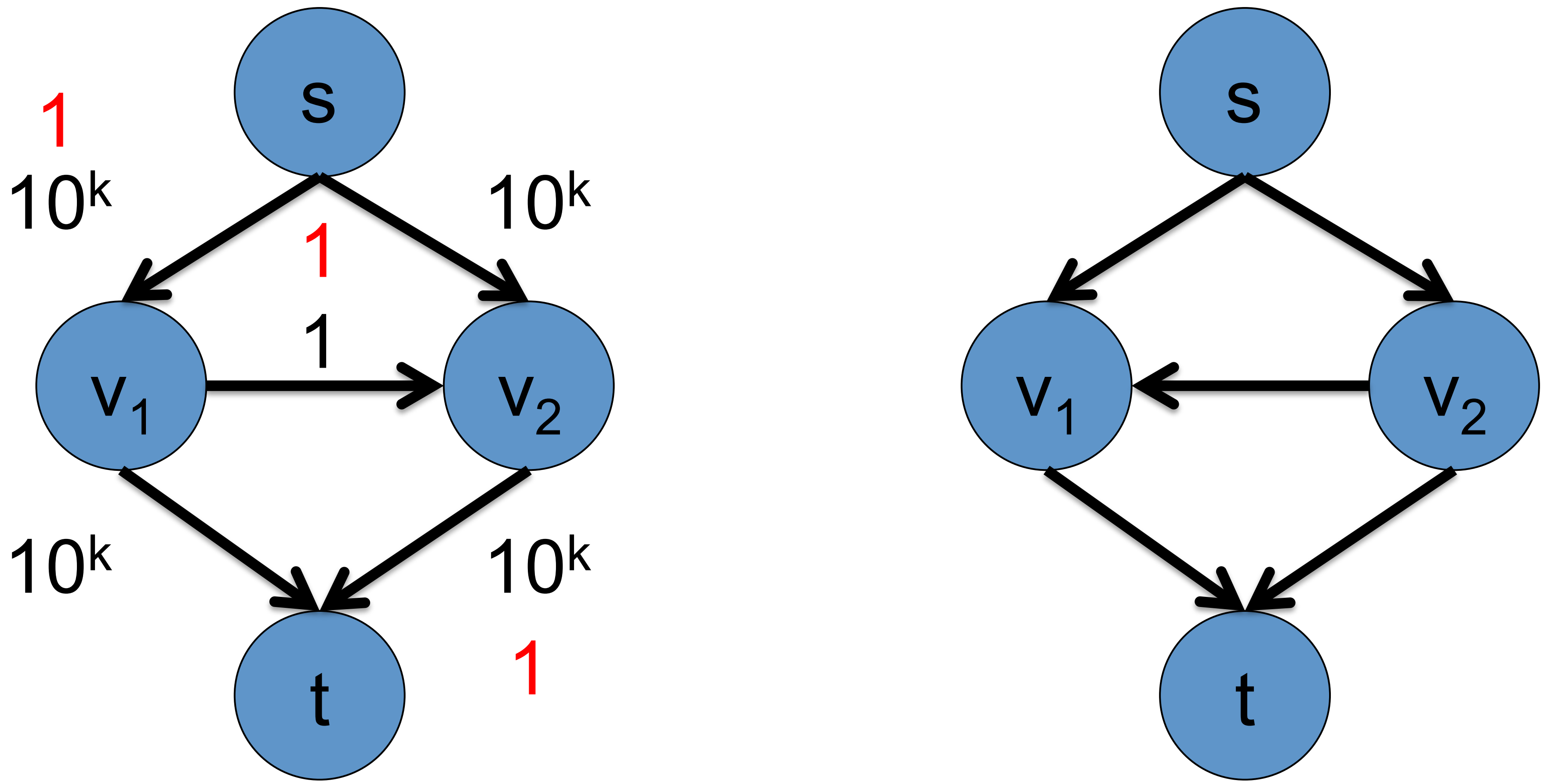
Pass the maximum allowable flow.

Ford-Fulkerson Algorithm



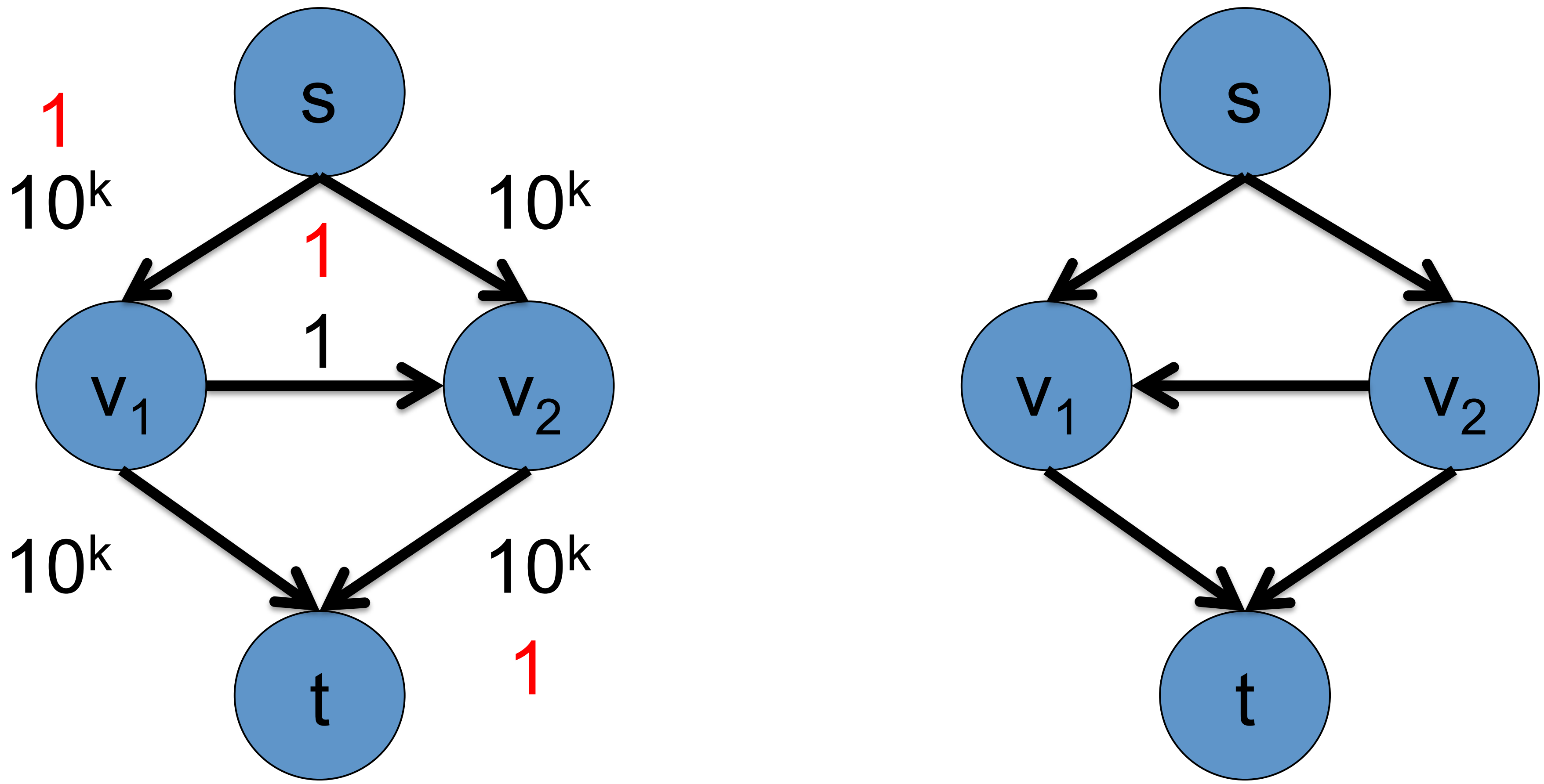
Pass the maximum allowable flow.

Ford-Fulkerson Algorithm



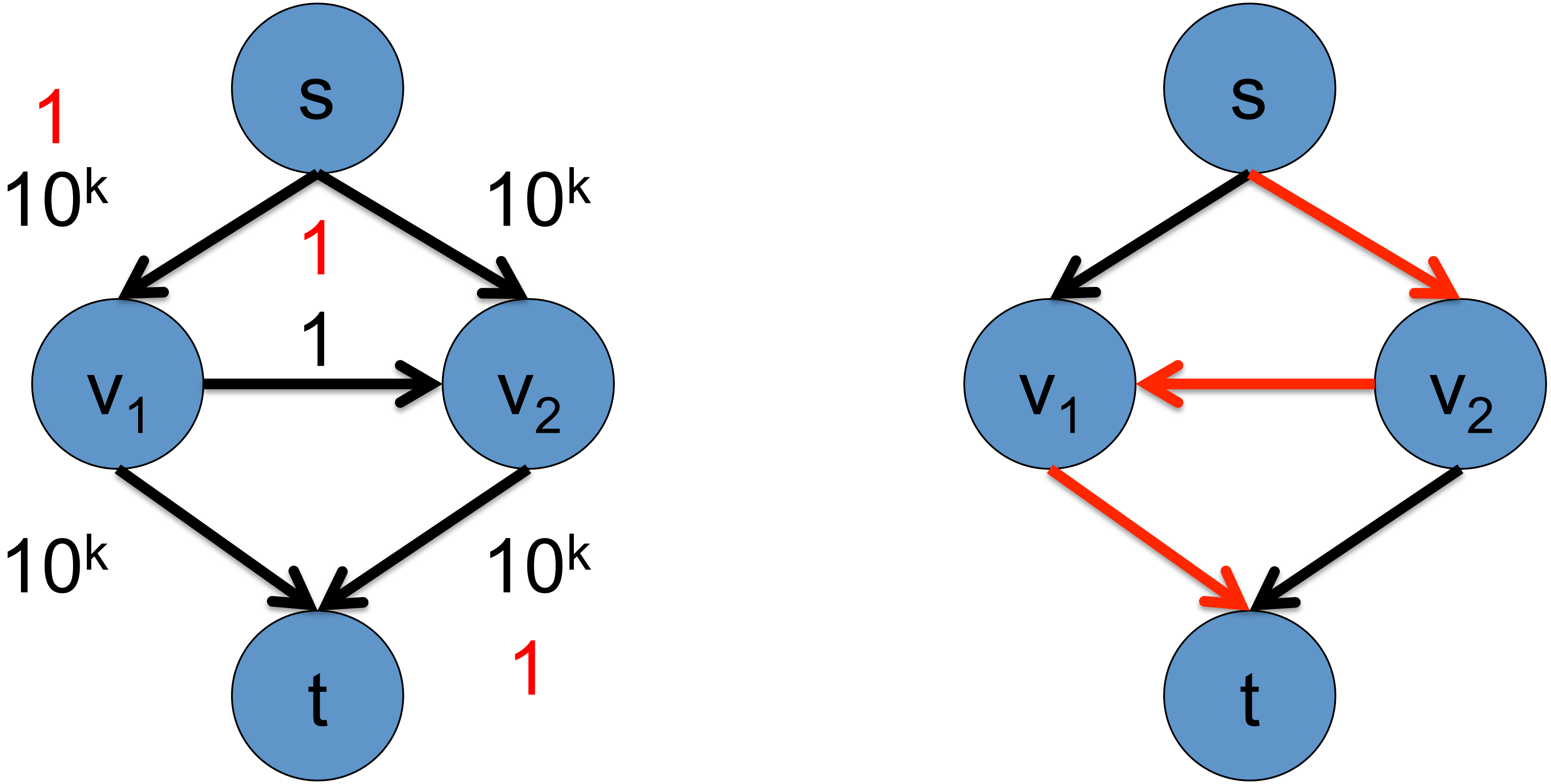
Update the residual graph.

Ford-Fulkerson Algorithm



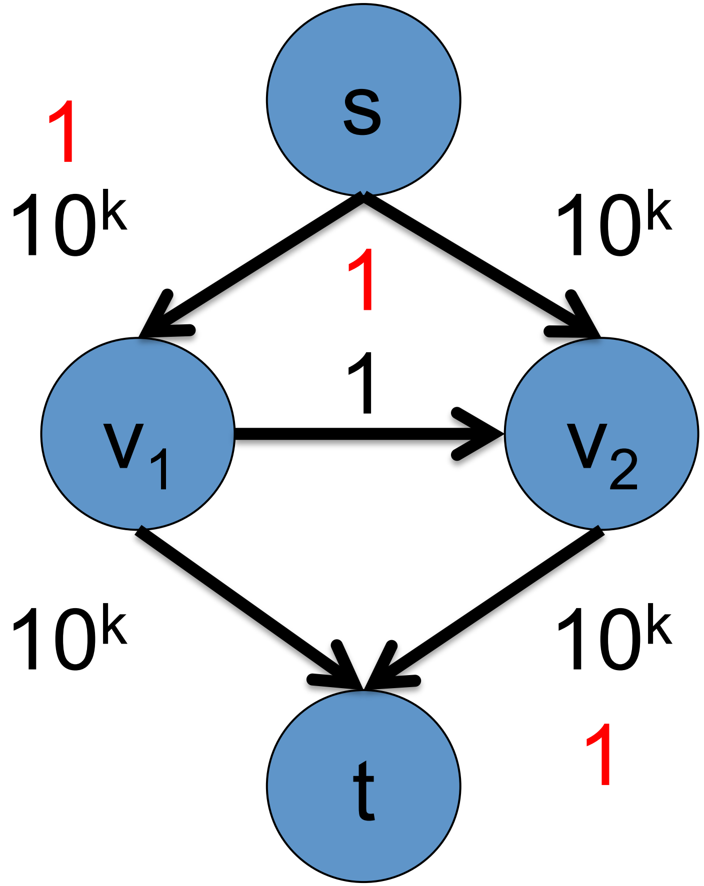
Find an s-t path in the residual graph.

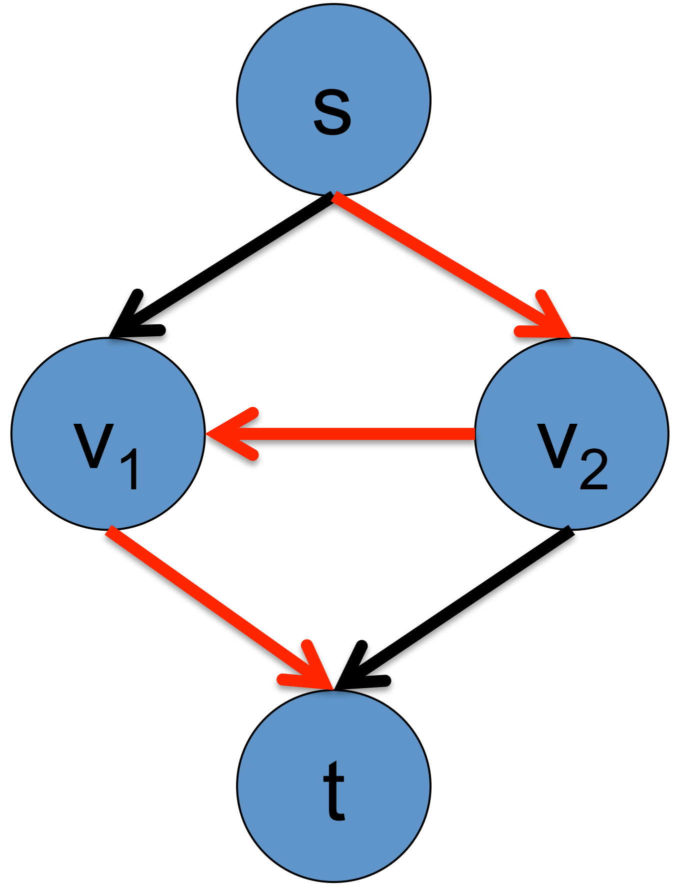
Ford-Fulkerson Algorithm



Find an s-t path in the residual graph.

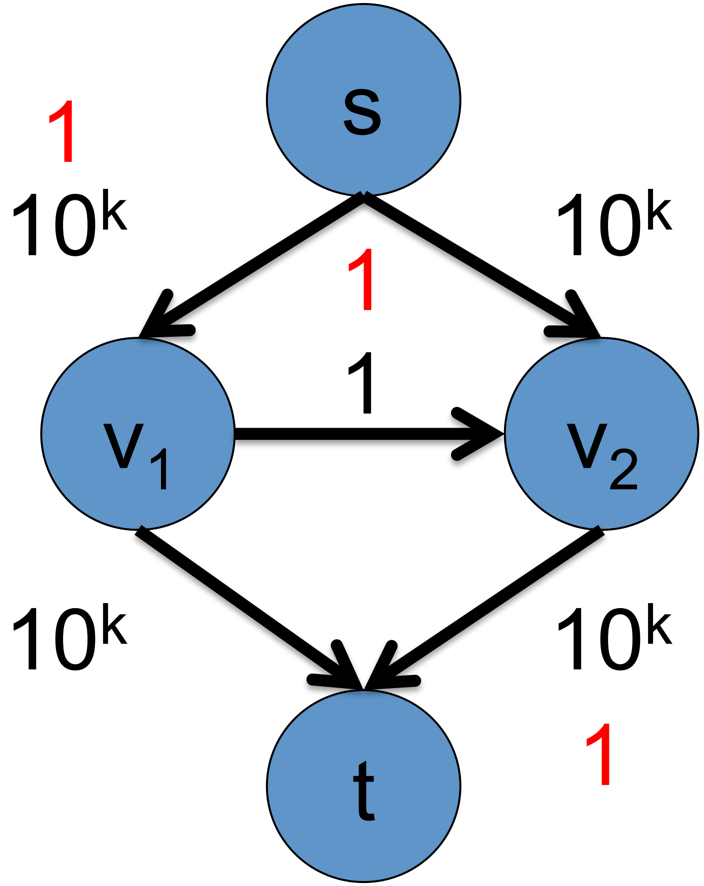
Ford-Fulkerson Algorithm

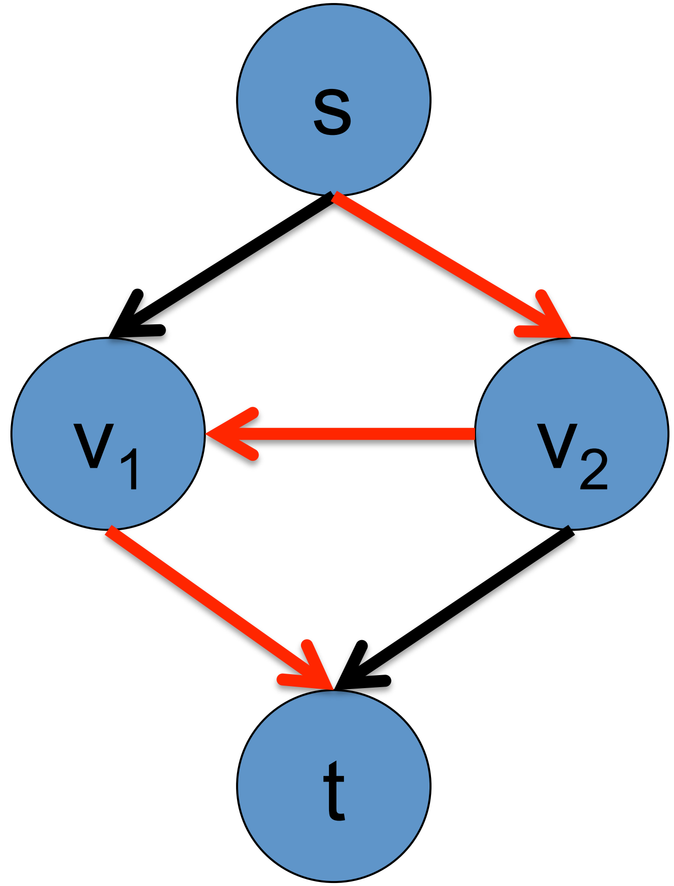




Complexity is exponential in k.

Ford-Fulkerson Algorithm

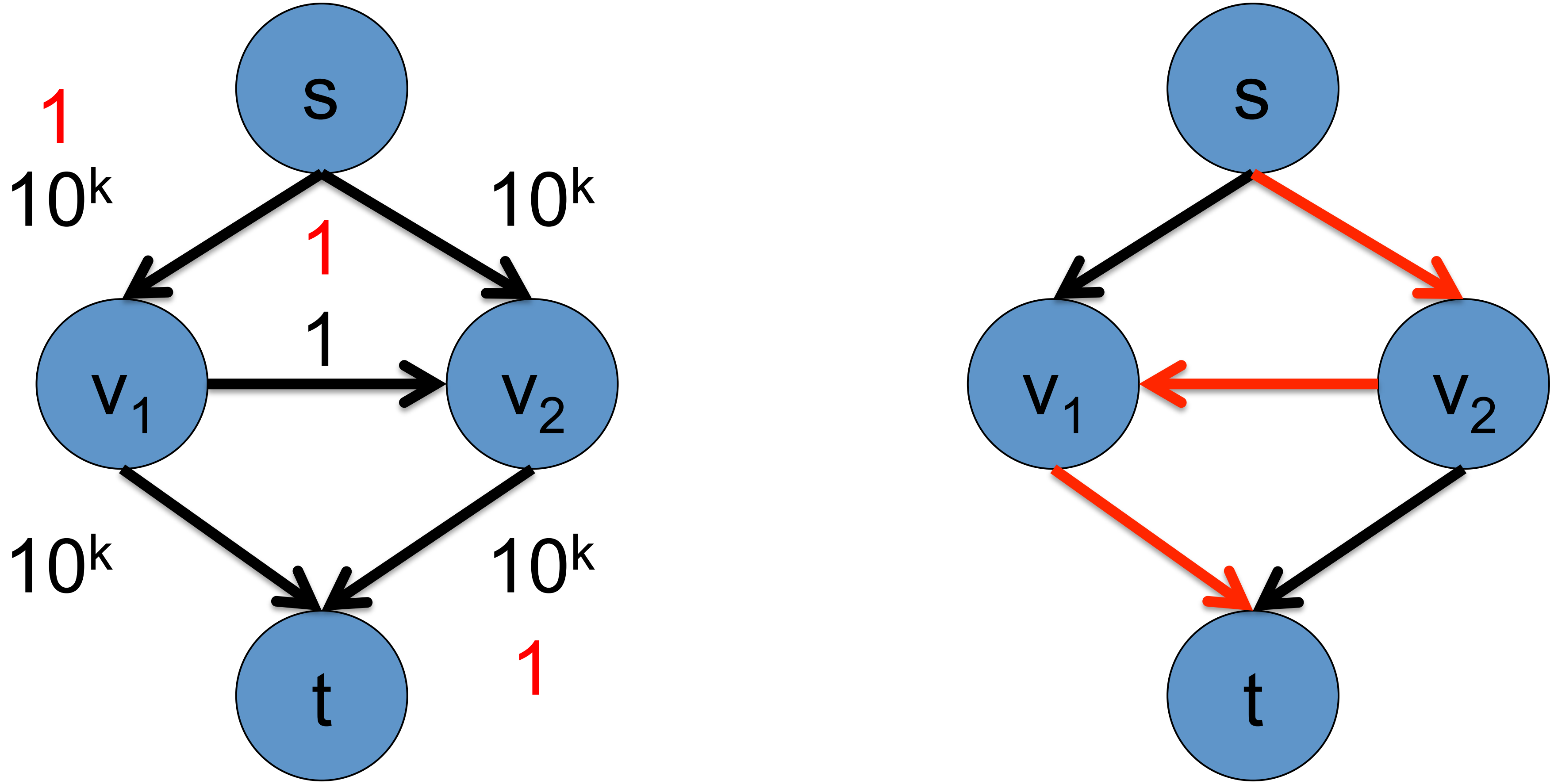




For examples, see Uri Zwick, 1993

Irrational arc lengths can lead to infinite iterations.

Ford-Fulkerson Algorithm



Choose wisely.

There are good paths and bad paths.

Outline

- Preliminaries
- Maximum Flow
- **Algorithms**
  - Ford-Fulkerson Algorithm
  - **Dinits Algorithm**
- Energy minimization with max flow/min cut

Dinits Algorithm

Start with flow = 0 for all arcs.

Find the **minimum s-t path**
in the residual graph.

Pass maximum allowable flow.

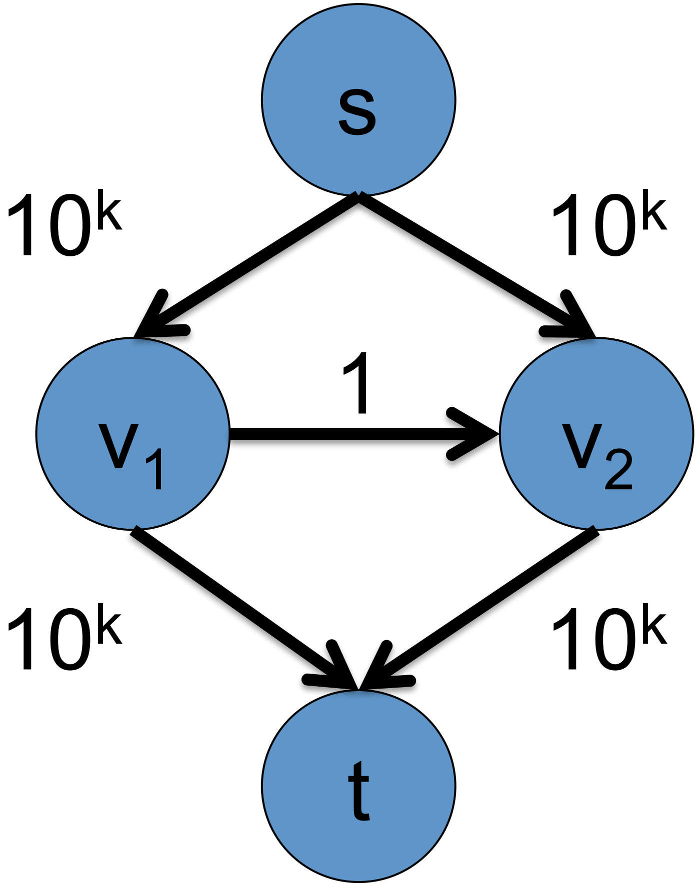
Subtract from inverse arcs.

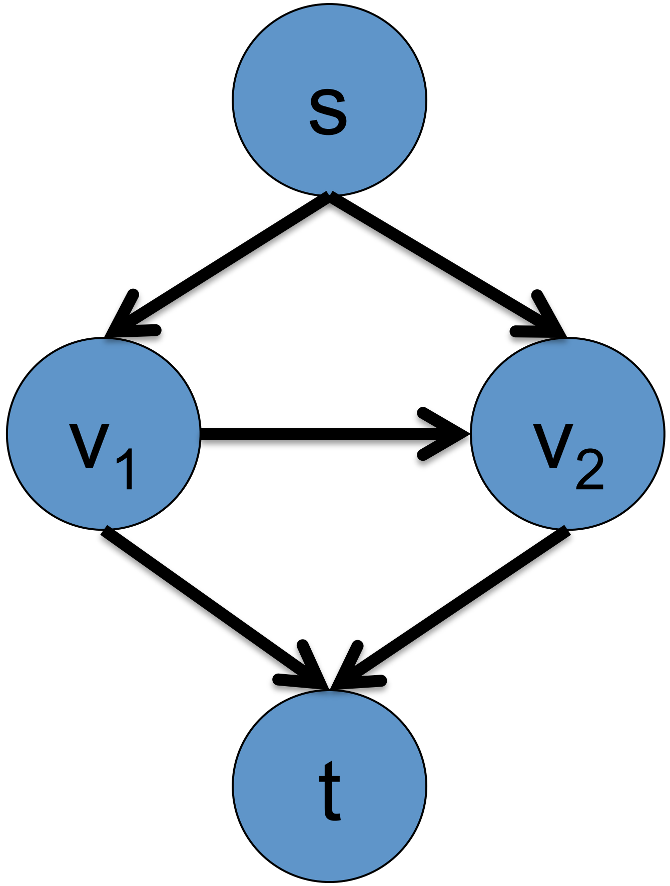
Add to forward arcs.

REPEAT

Until s and t are disjoint in the residual graph.

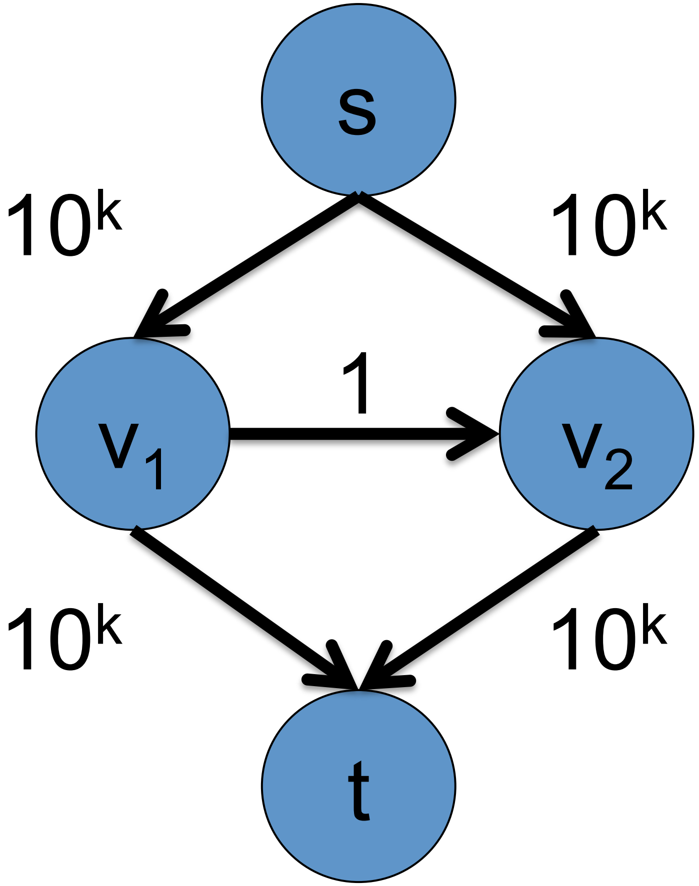
Dinits Algorithm

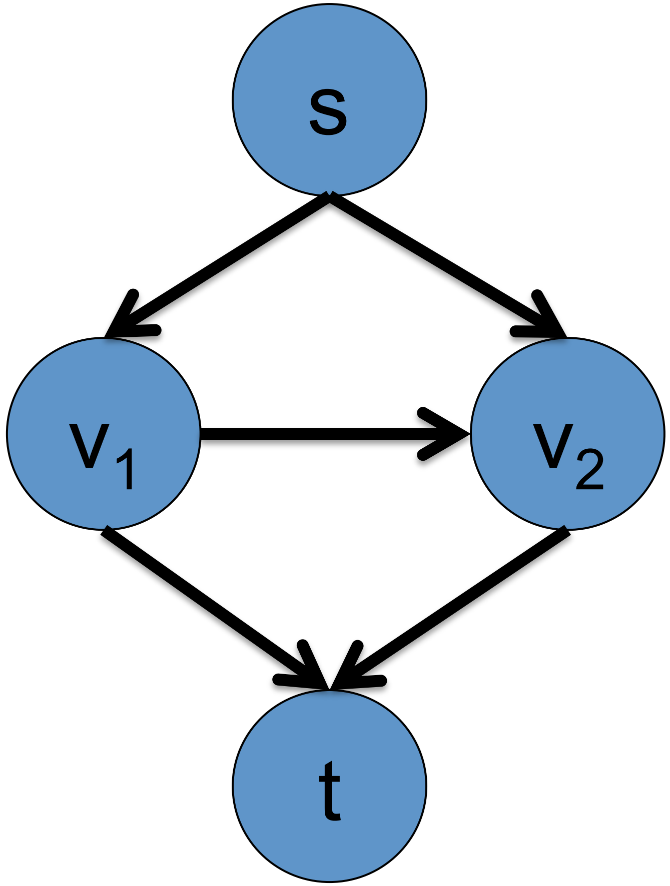




Start with zero flow

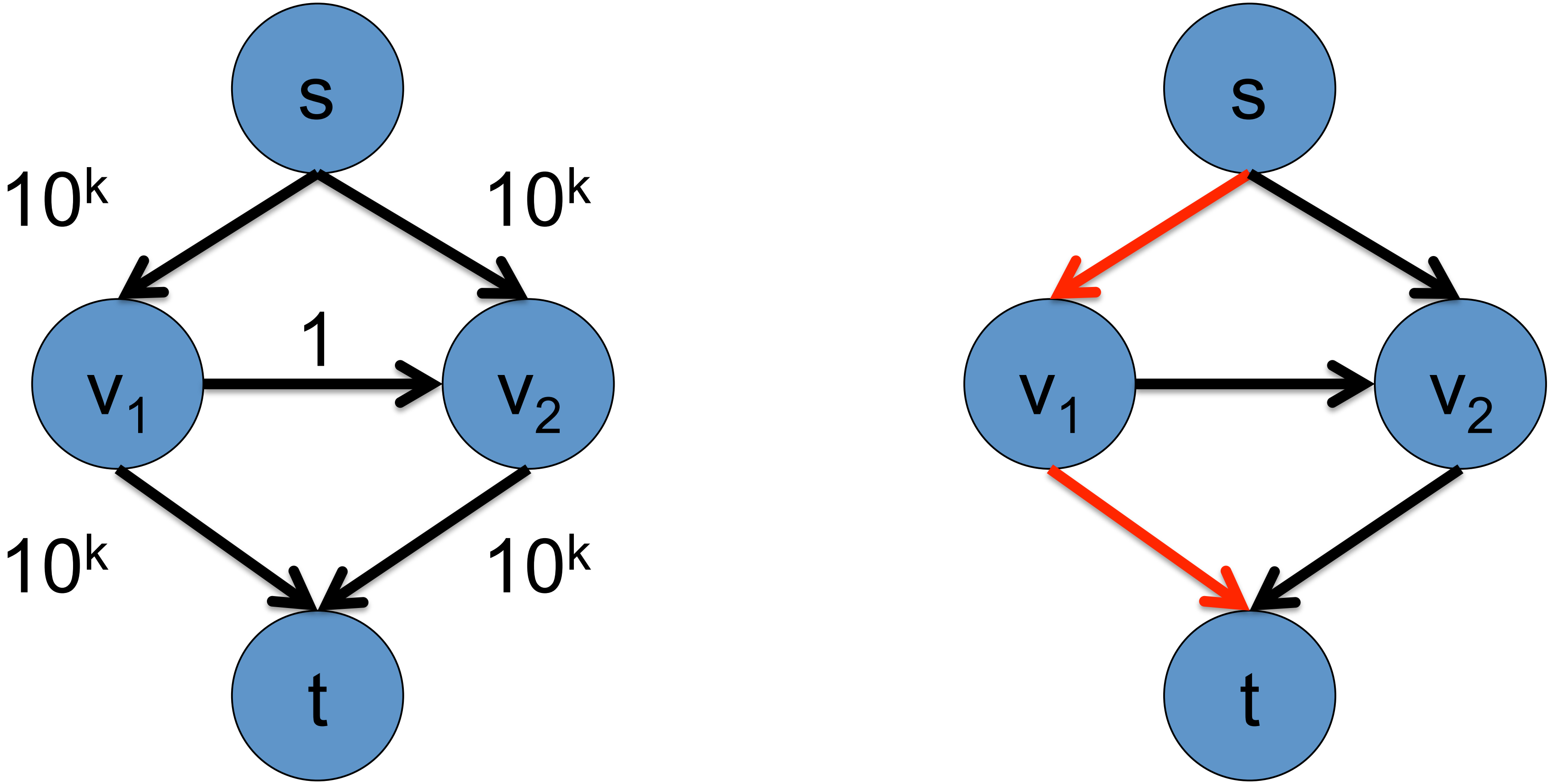
Dinits Algorithm





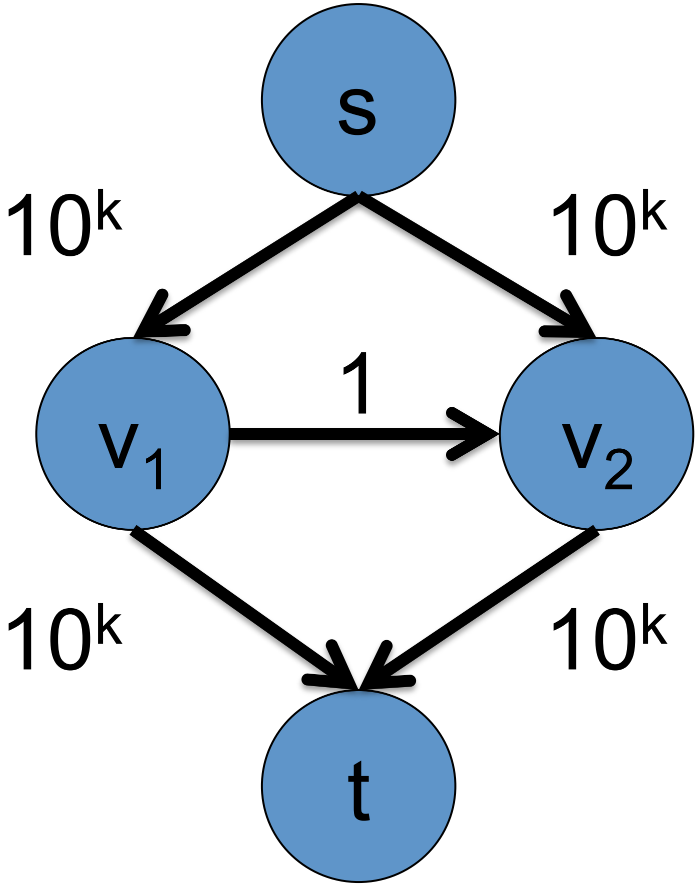
Find the minimum s-t path in the residual graph.

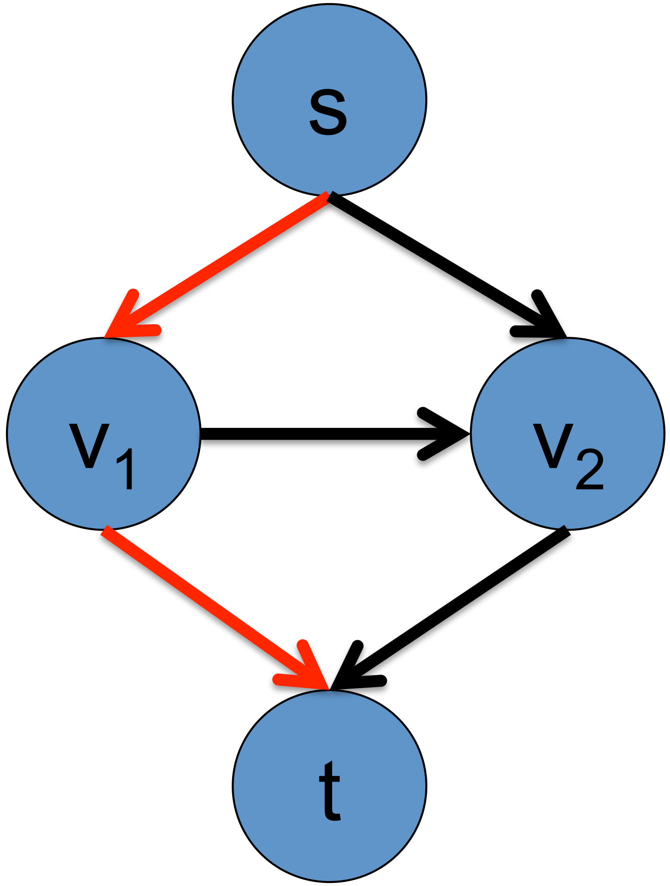
Dinits Algorithm



Find the minimum s-t path in the residual graph.

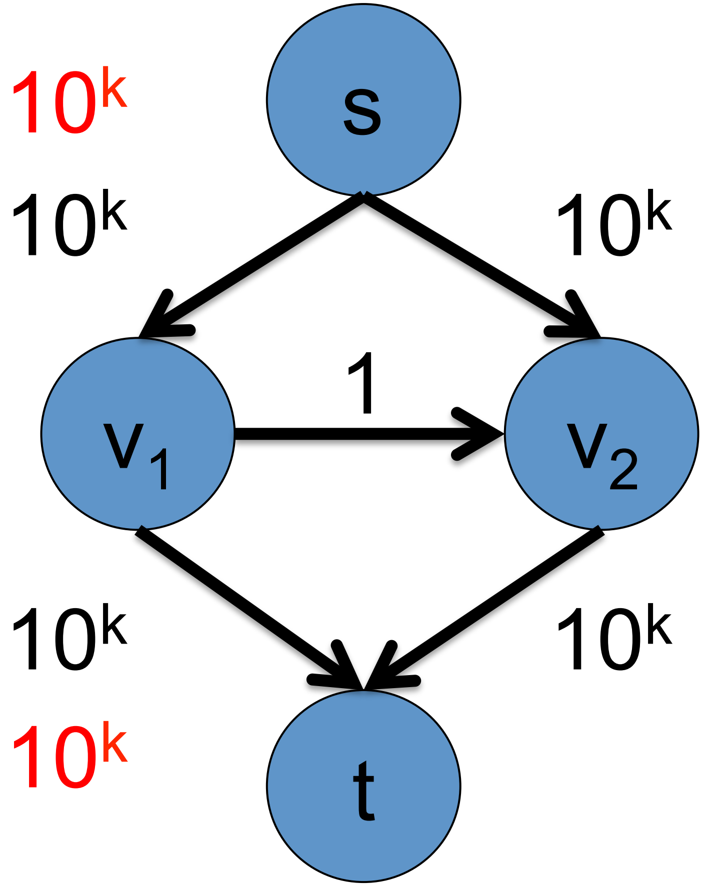
Dinits Algorithm

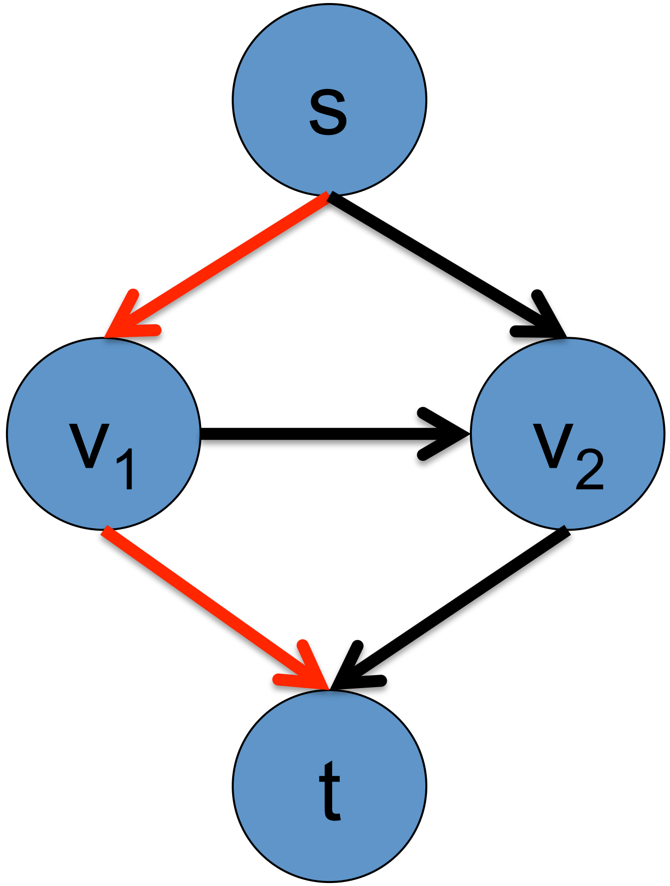




Pass the maximum allowable flow.

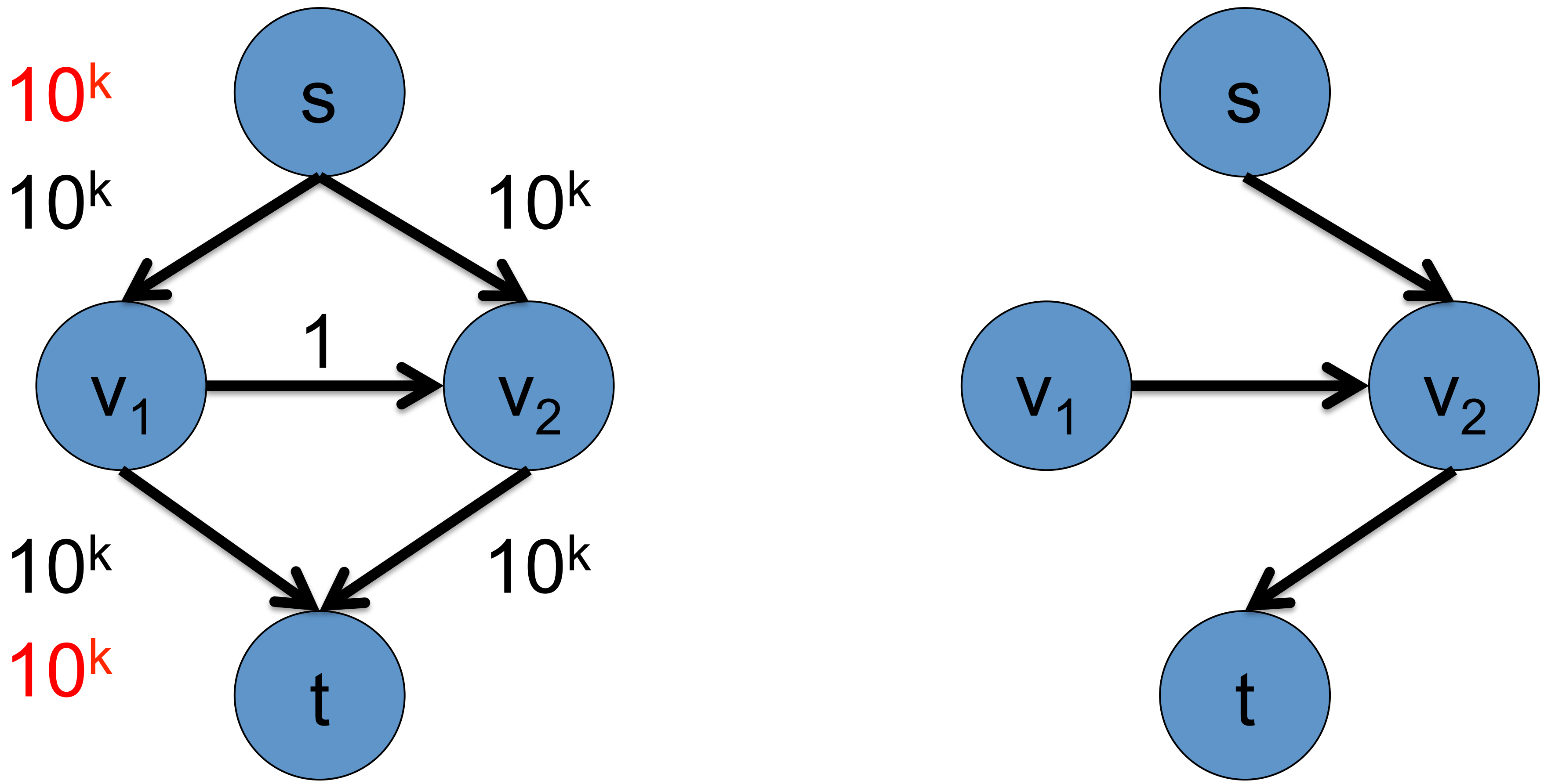
Dinits Algorithm





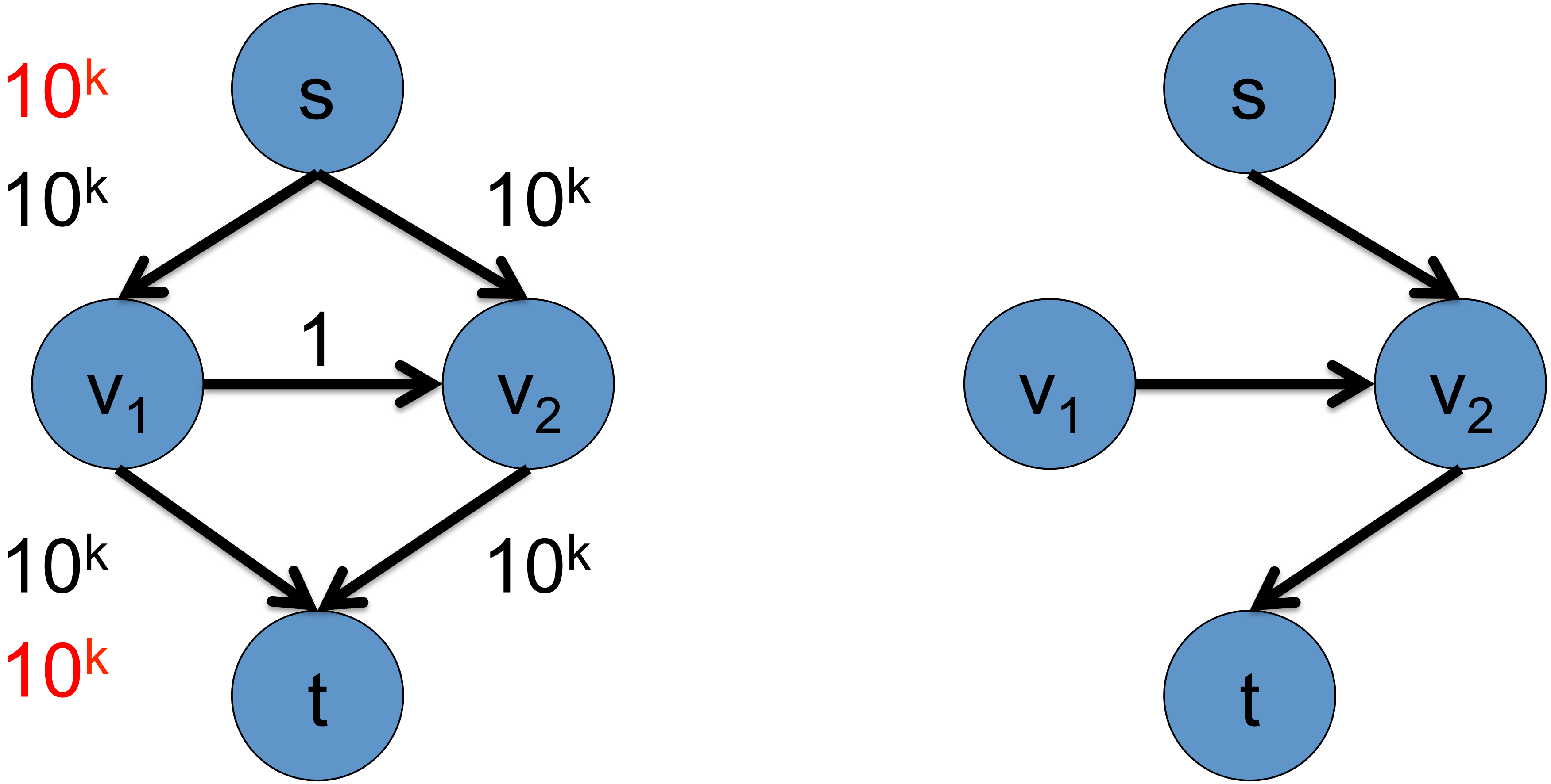
Pass the maximum allowable flow.

Dinits Algorithm



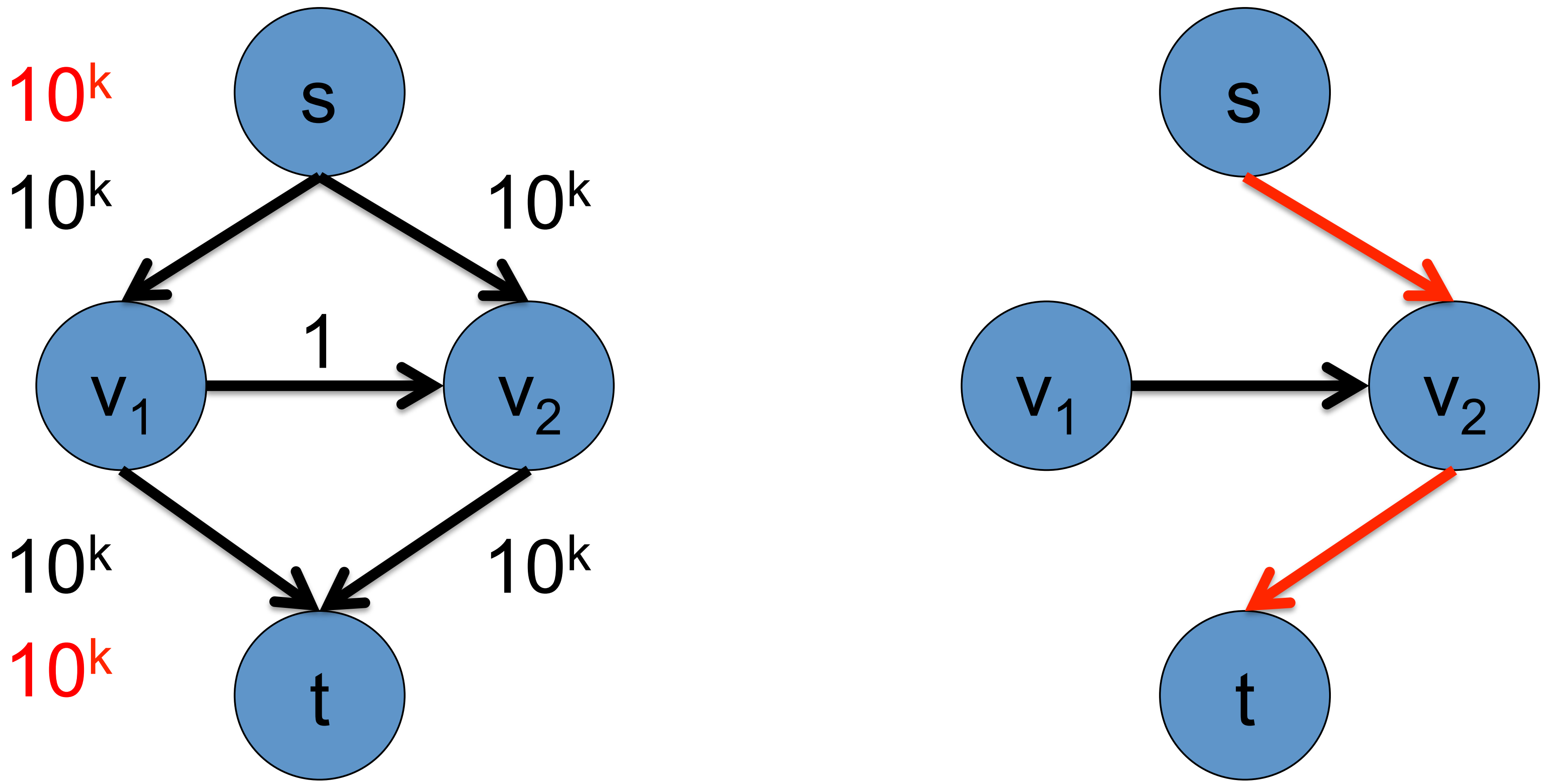
Update the residual graph.

Dinits Algorithm



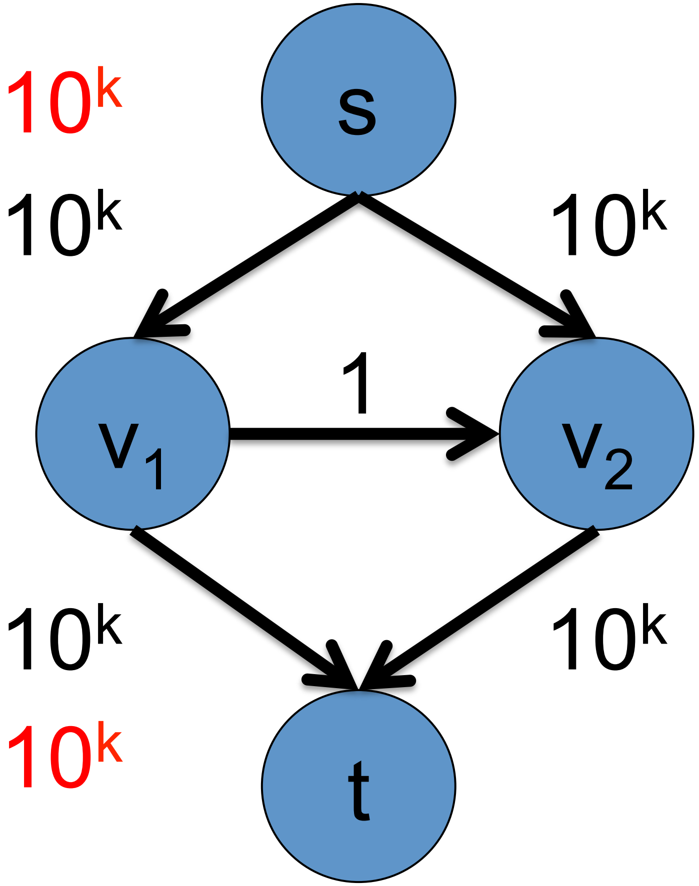
Find the minimum s-t path in the residual graph.

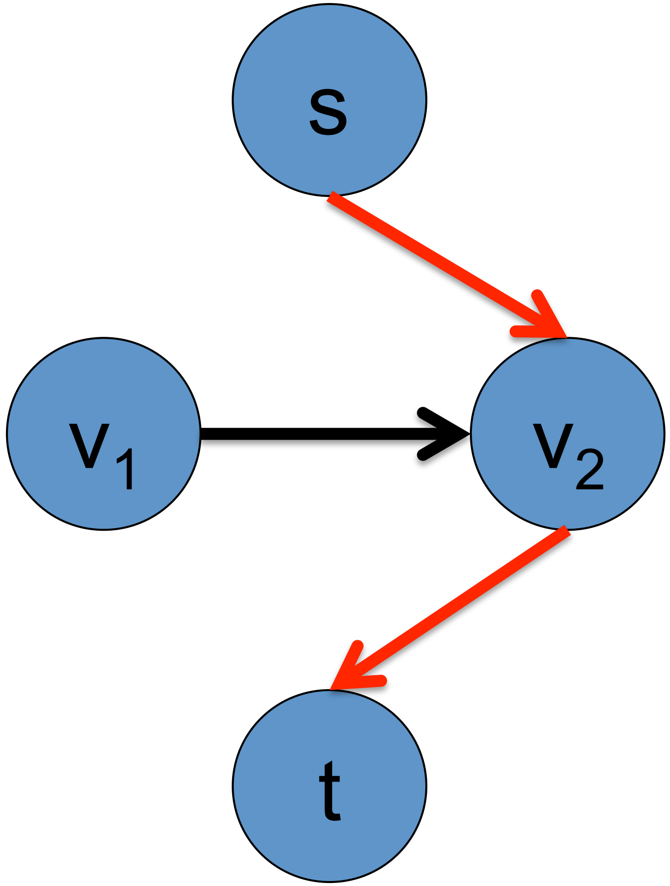
Dinits Algorithm



Find the minimum s-t path in the residual graph.

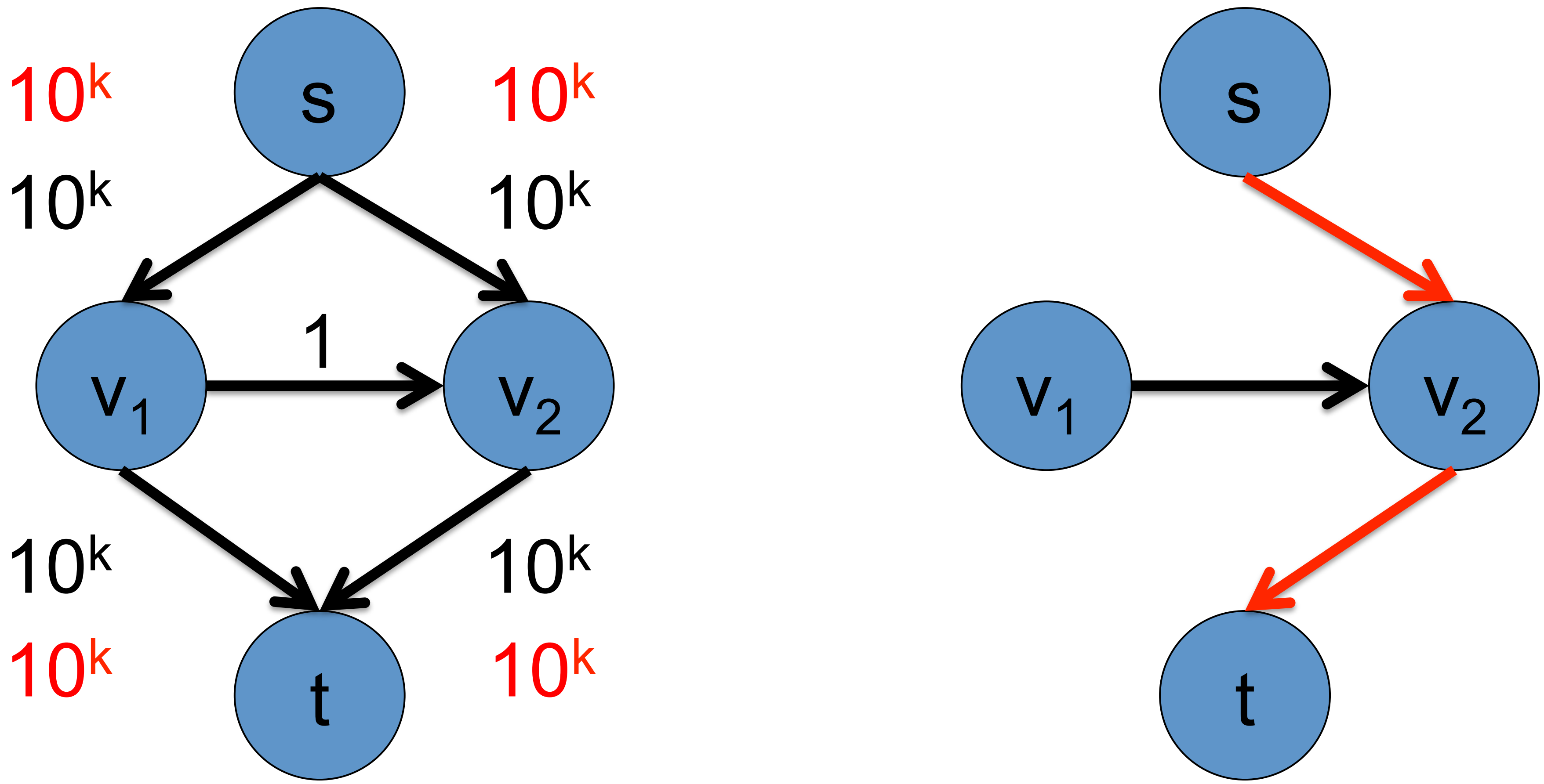
Dinits Algorithm





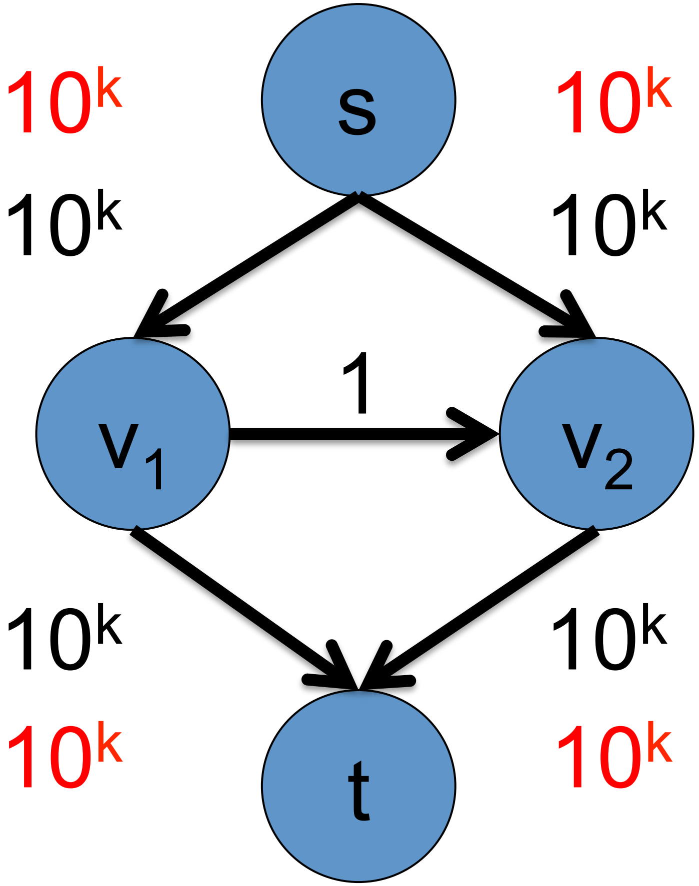
Pass the maximum allowable flow.

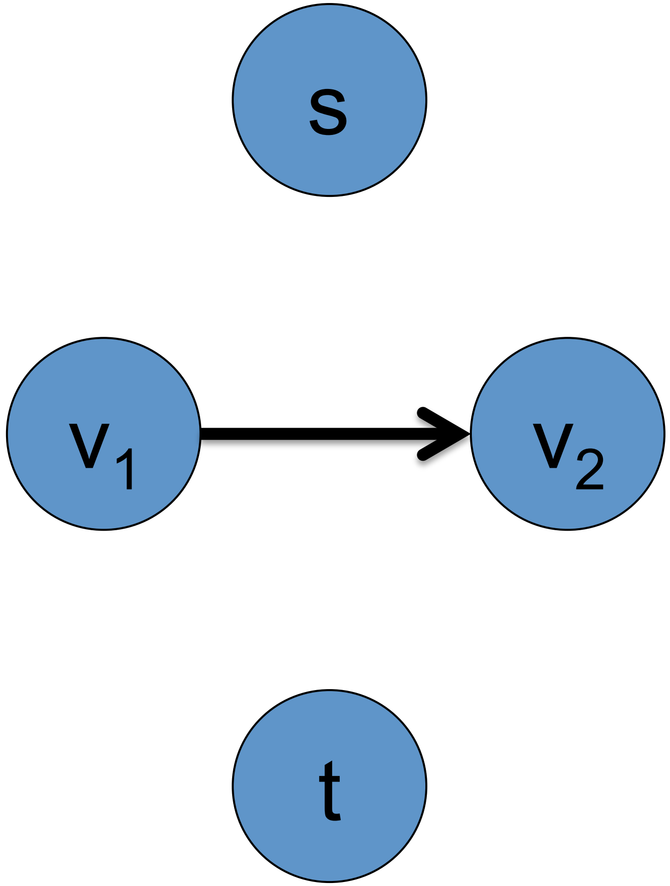
Dinits Algorithm



Pass the maximum allowable flow.

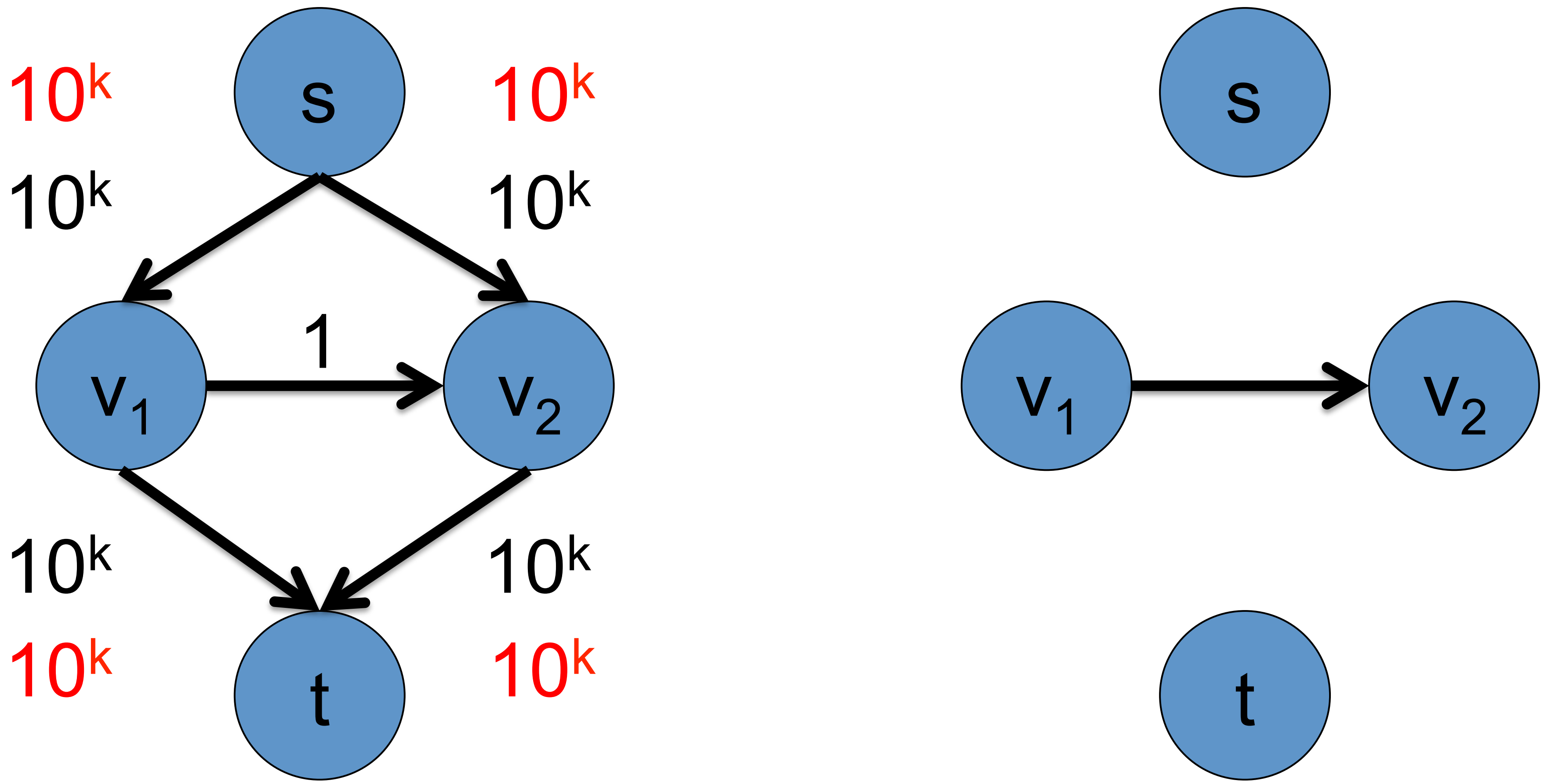
Dinits Algorithm





Update the residual graph.

Dinits Algorithm



No more s-t paths. Stop.

Solvers for the Minimum-Cut Problem

Augmenting Path and Push-Relabel

year	discoverer(s)	bound
1951	Dantzig	$O(n^2 mU)$
1955	Ford & Fulkerson	$O(m^2 U)$
1970	Dinitz	$O(n^2 m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2 m^{1/2})$
1980	Galil & Naamad	$O(nm \log^2 n)$
1983	Sleator & Tarjan	$O(nm \log n)$
1986	Goldberg & Tarjan	$O(nm \log(n^2/m))$
1987	Ahuja & Orlin	$O(nm + n^2 \log U)$
1987	Ahuja et al.	$O(nm \log(n\sqrt{\log U}/m))$
1989	Cheriyan & Hagerup	$E(nm + n^2 \log^2 n)$
1990	Cheriyan et al.	$O(n^3/\log n)$
1990	Alon	$O(nm + n^{8/3} \log n)$
1992	King et al.	$O(nm + n^{2+\epsilon})$
1993	Phillips & Westbrook	$O(nm(\log_{m/n} n + \log^{2+\epsilon} n))$
1994	King et al.	$O(nm \log_{m/(n \log n)} n)$
1997	Goldberg & Rao	$O(m^{3/2} \log(n^2/m) \log U)$ $O(n^{2/3} m \log(n^2/m) \log U)$

n: #nodes

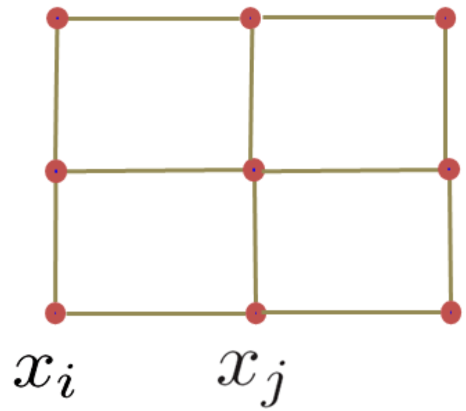
m: #arcs

U: maximum arc length

[Slide credit: Andrew Goldberg]

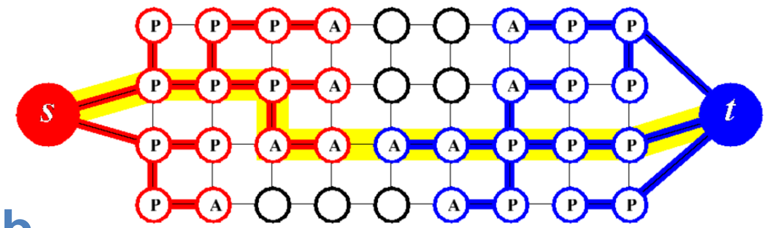
Max-Flow in Computer Vision

- **Specialized algorithms for vision problems**
  - Grid graphs
  - Low connectivity (m ~ O(n))



- **Dual search tree augmenting path algorithm**

  [Boykov and Kolmogorov PAMI 2004]
  - Finds approximate shortest augmenting paths efficiently
  - High worst-case time complexity
  - Empirically outperforms other algorithms on vision problems
  - Efficient code available on the web

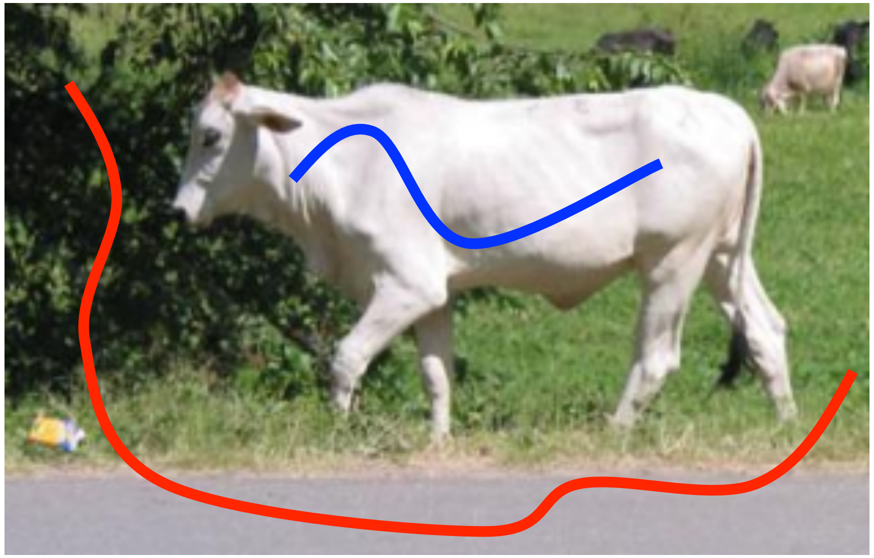


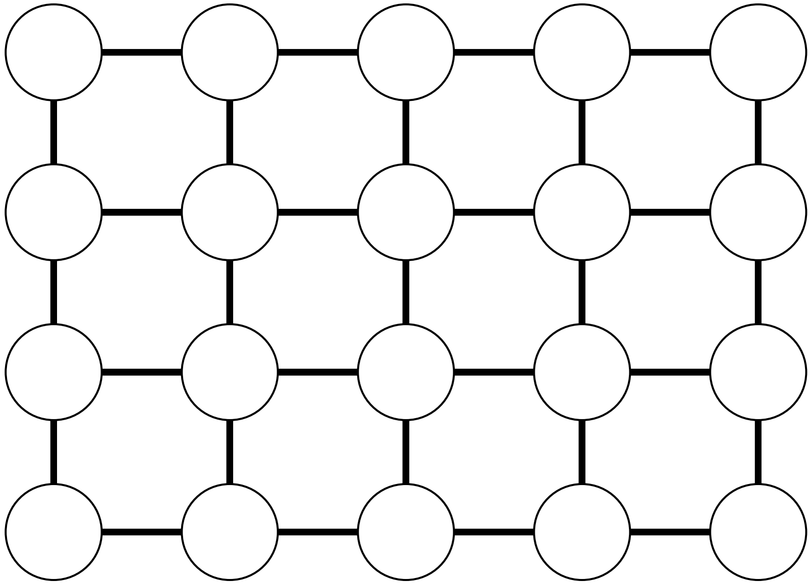
http://www.adastral.ucl.ac.uk/~vladkolm/software.html

Outline

- Preliminaries
- Maximum Flow
- Algorithms
- **Energy minimization with max flow/min cut**
  - Two-Label Energy Functions

Interactive Binary Segmentation



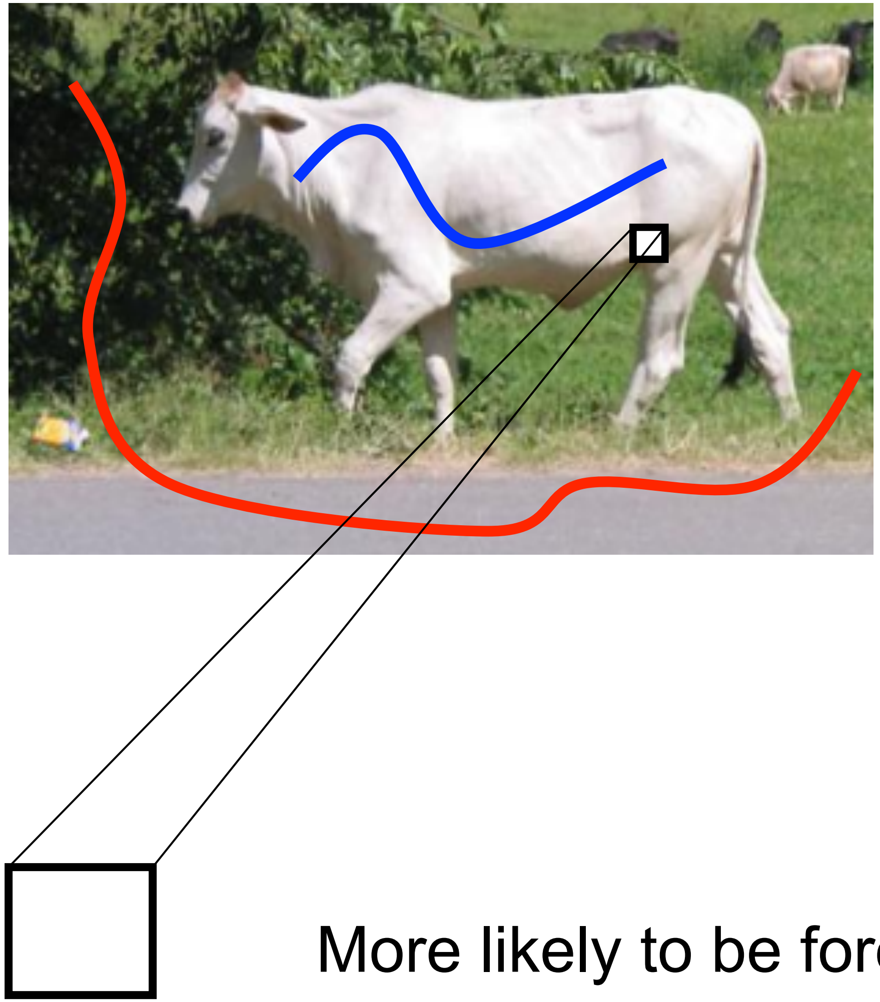


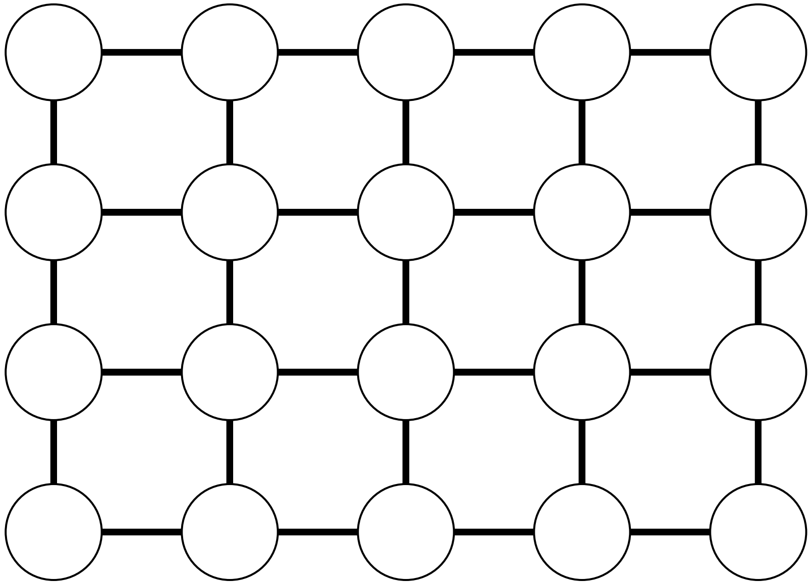
Foreground histogram of RGB values FG

Background histogram of RGB values BG

‘1’ indicates foreground and ‘0’ indicates background

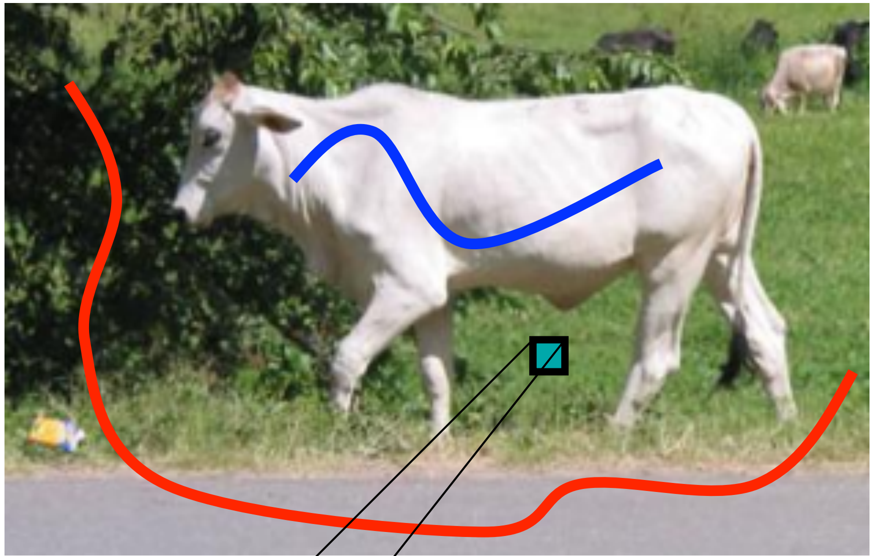
Interactive Binary Segmentation

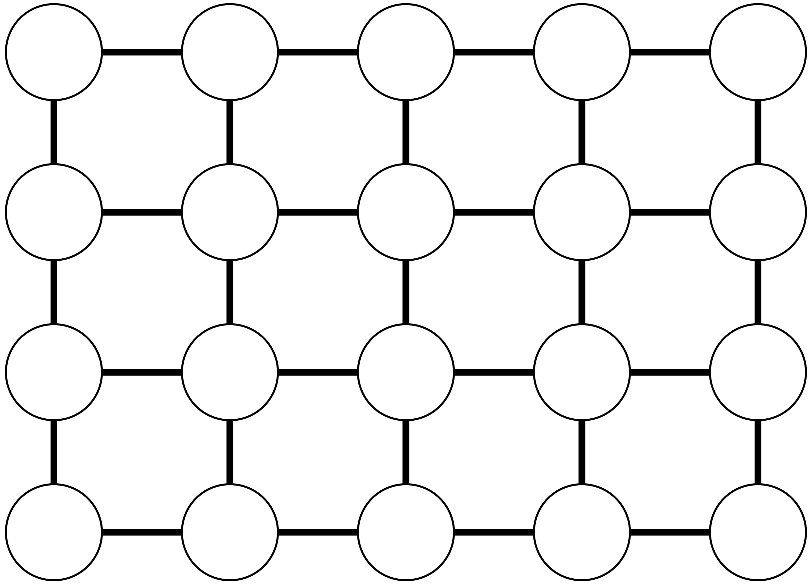




More likely to be foreground than background

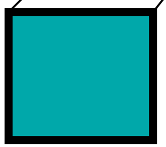
Interactive Binary Segmentation





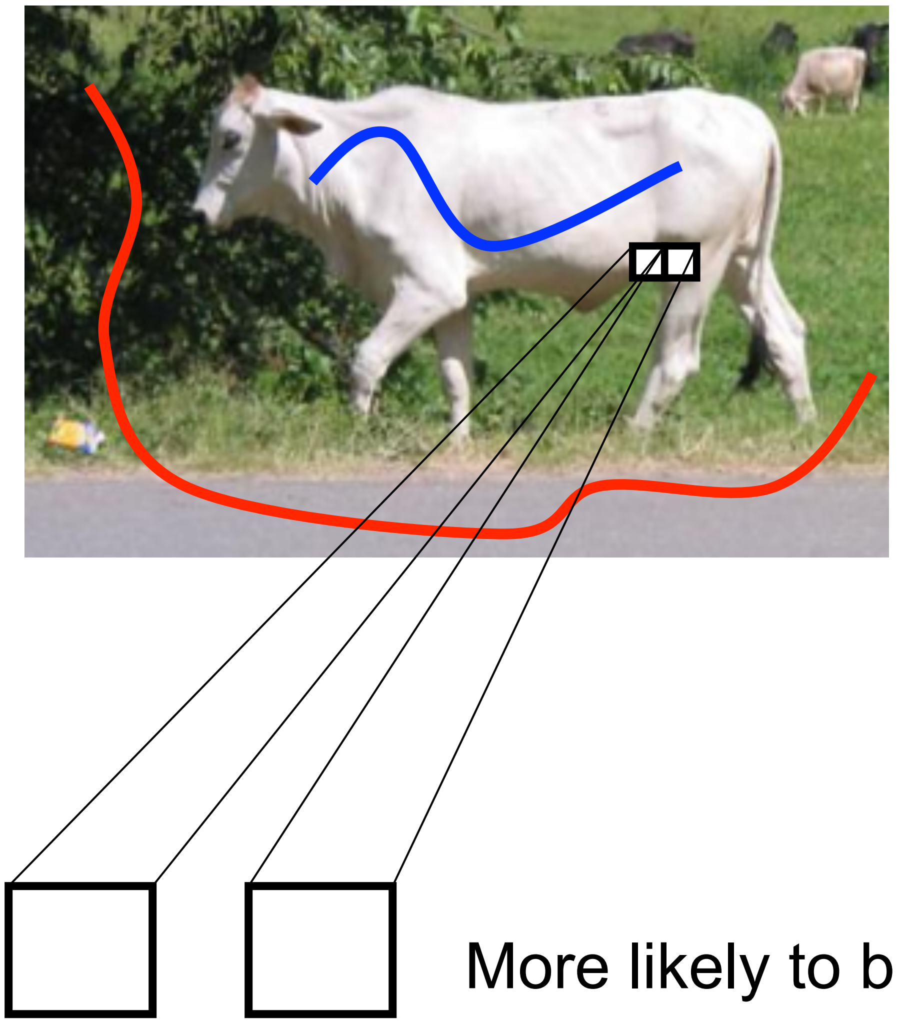
$\theta_p(0)$ proportional to $-\log(BG(d_a))$

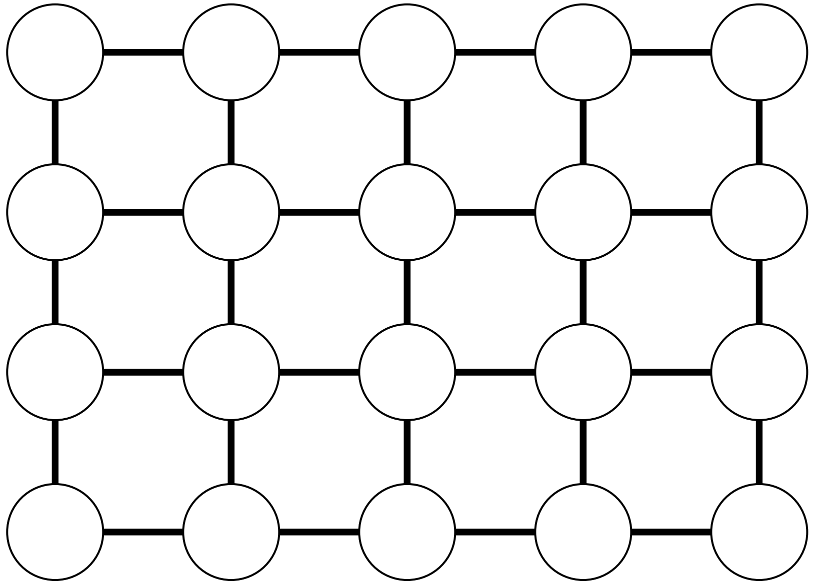
$\theta_p(1)$ proportional to $-\log(FG(d_a))$



More likely to be background than foreground

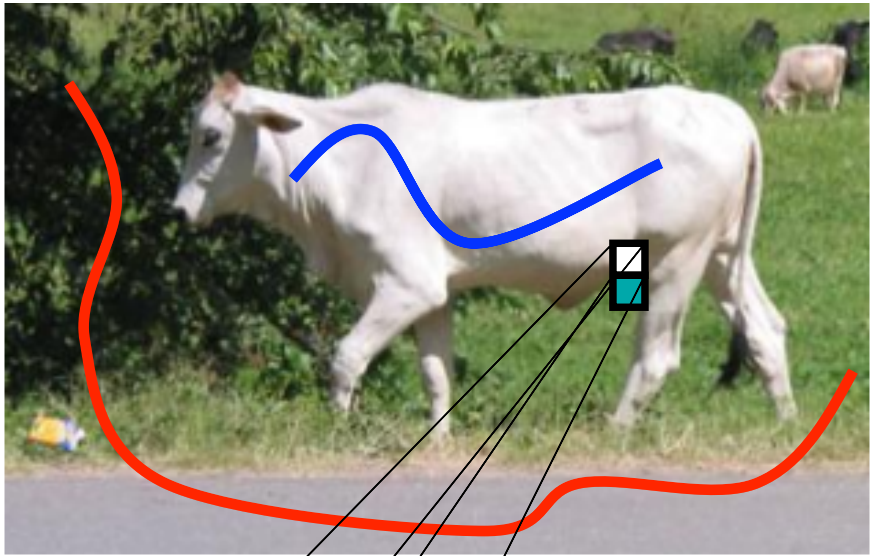
Interactive Binary Segmentation

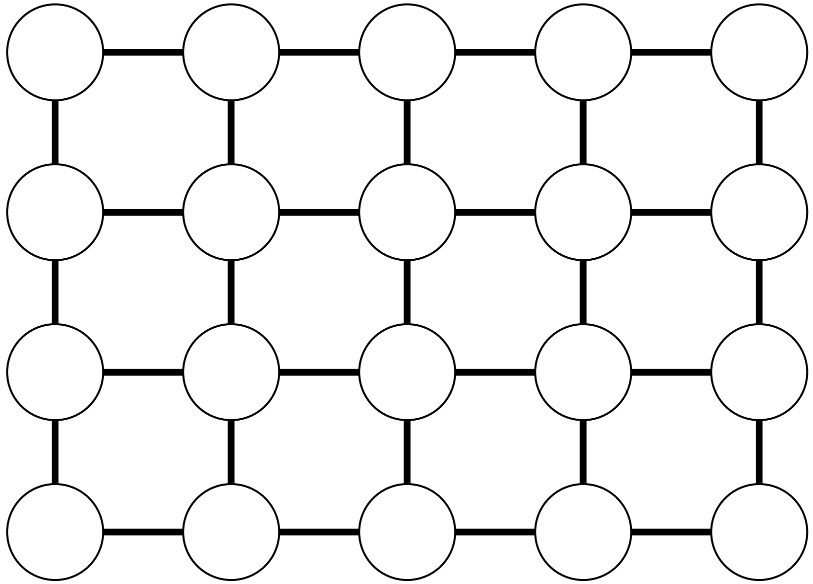




More likely to belong to same label

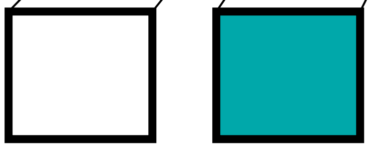
Interactive Binary Segmentation





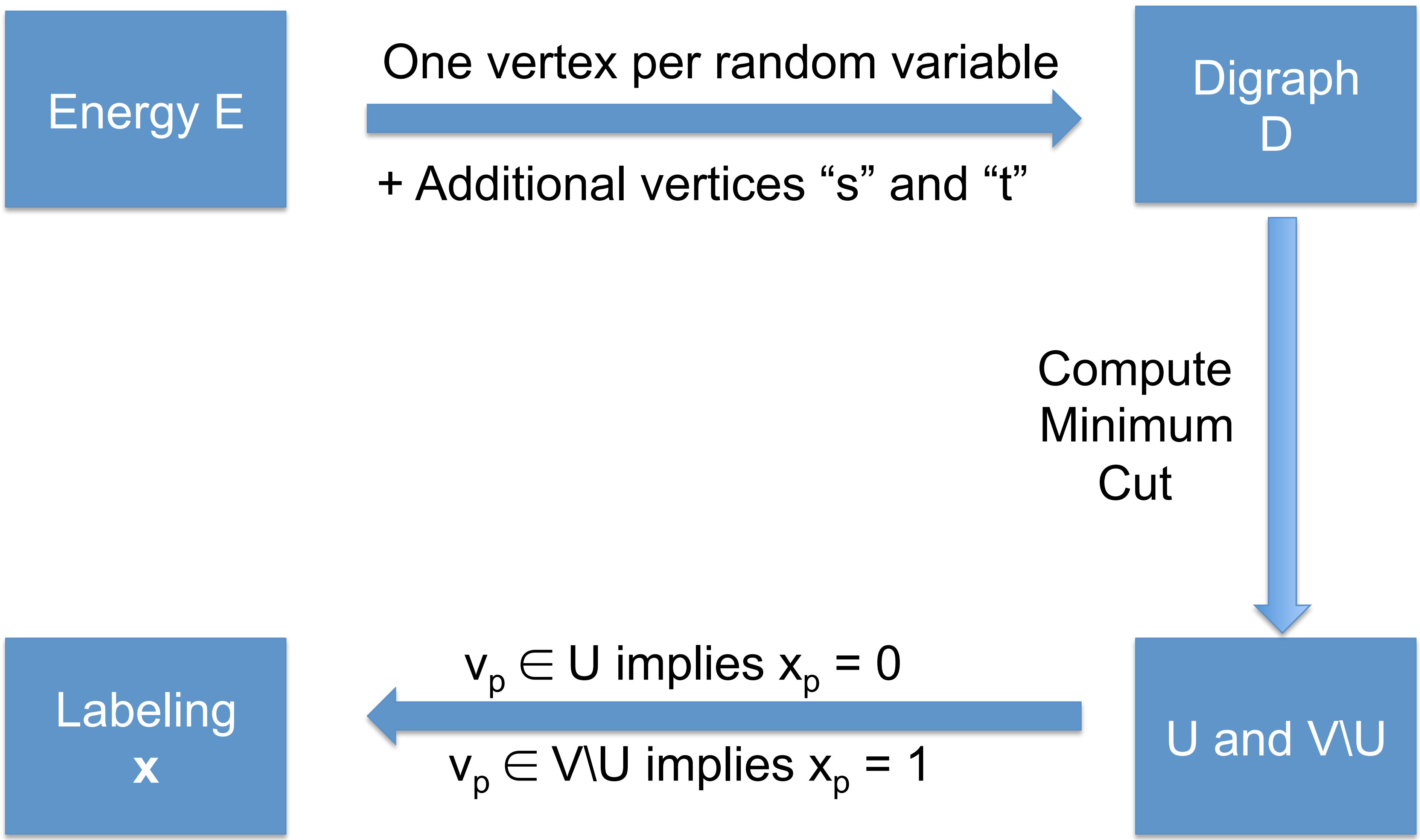
$\theta_{pq}(i,k)$ proportional to $\exp(-(d_a-d_b)^2)$ if $i \neq k$

$\theta_{pq}(i,k) = 0$ if $i = k$

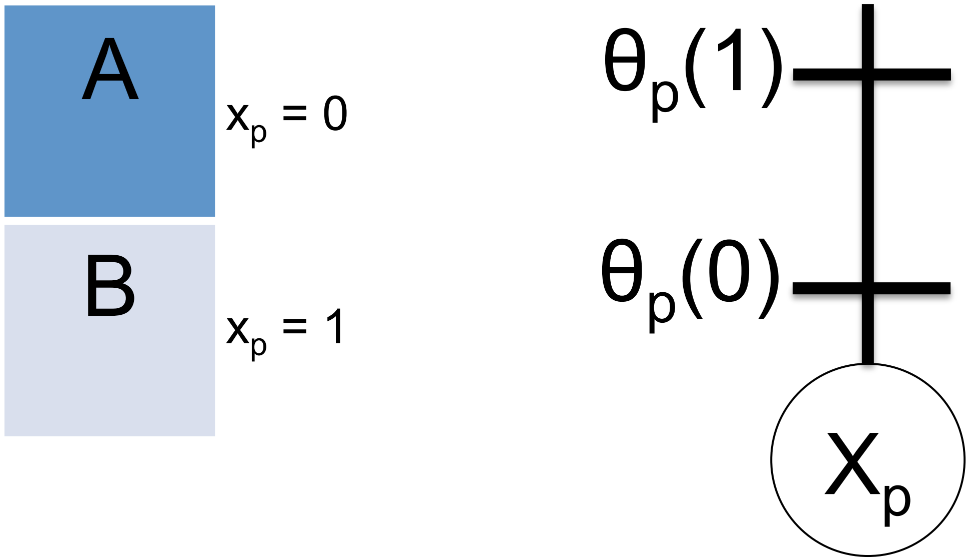


Less likely to belong to same label

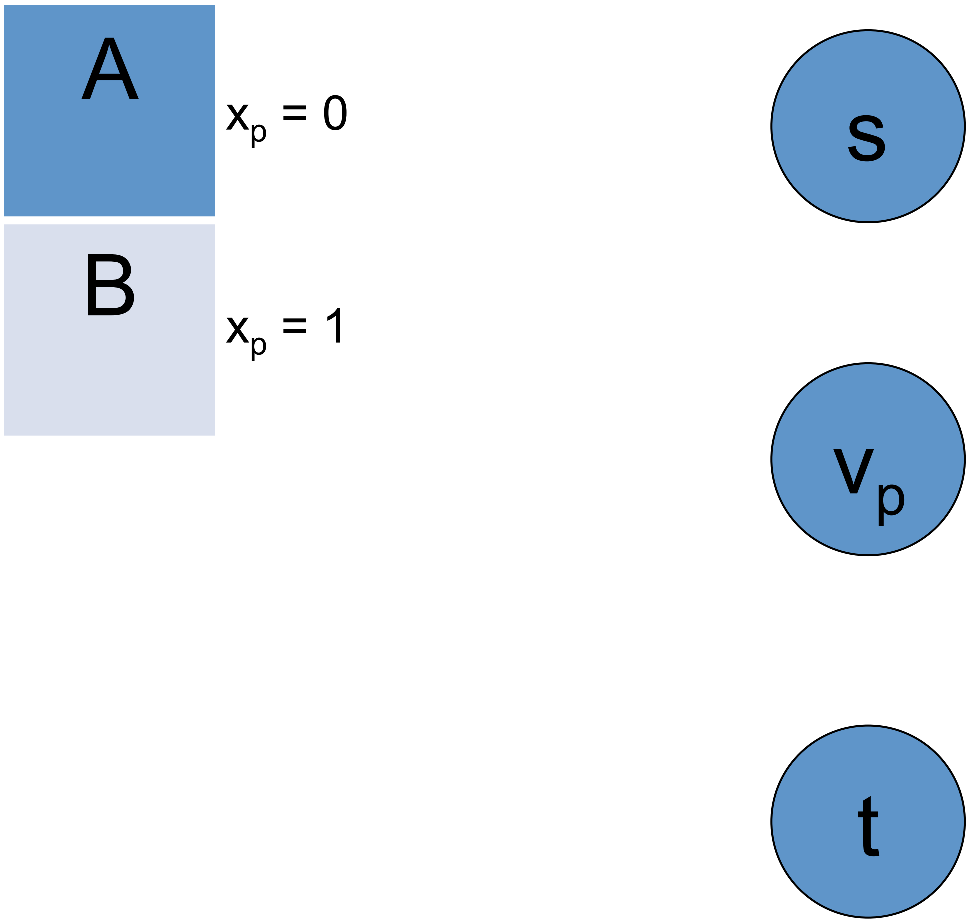
Overview


Energy E
One vertex per random variable
+ Additional vertices “s” and “t”
Digraph D
Compute Minimum Cut
U and V\U
$v_p \in U$ implies $x_p = 0$
$v_p \in V \setminus U$ implies $x_p = 1$
Labeling **x**

Digraph for Unary Potentials



Digraph for Unary Potentials

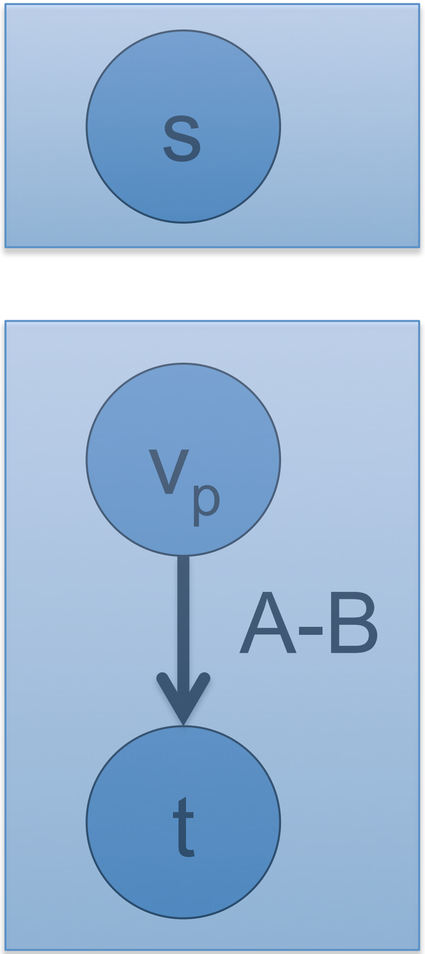


Digraph for Unary Potentials

A	$x_p = 0$
B	$x_p = 1$

Constant

$$\begin{bmatrix} A-B \\ 0 \end{bmatrix} + \begin{bmatrix} B \\ B \end{bmatrix}$$



Let A ≥ B

$x_p = 1$

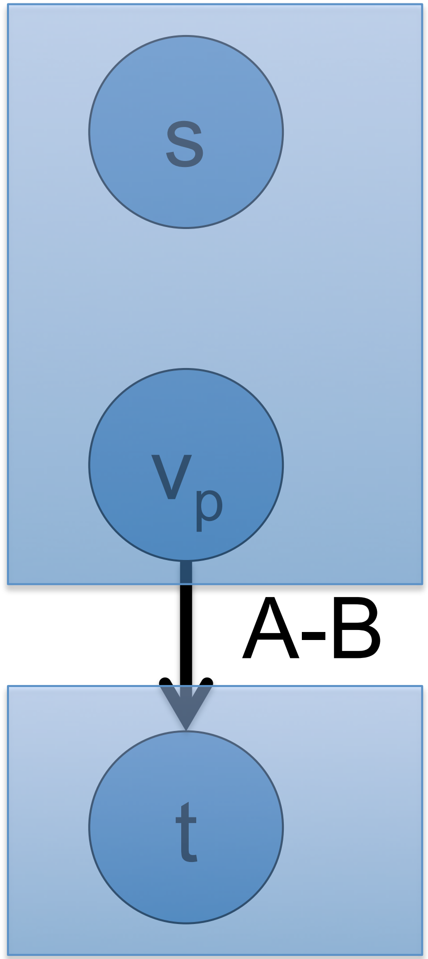
0

Digraph for Unary Potentials

A	$x_p = 0$
B	$x_p = 1$

Constant

A-B	+	B
0		B

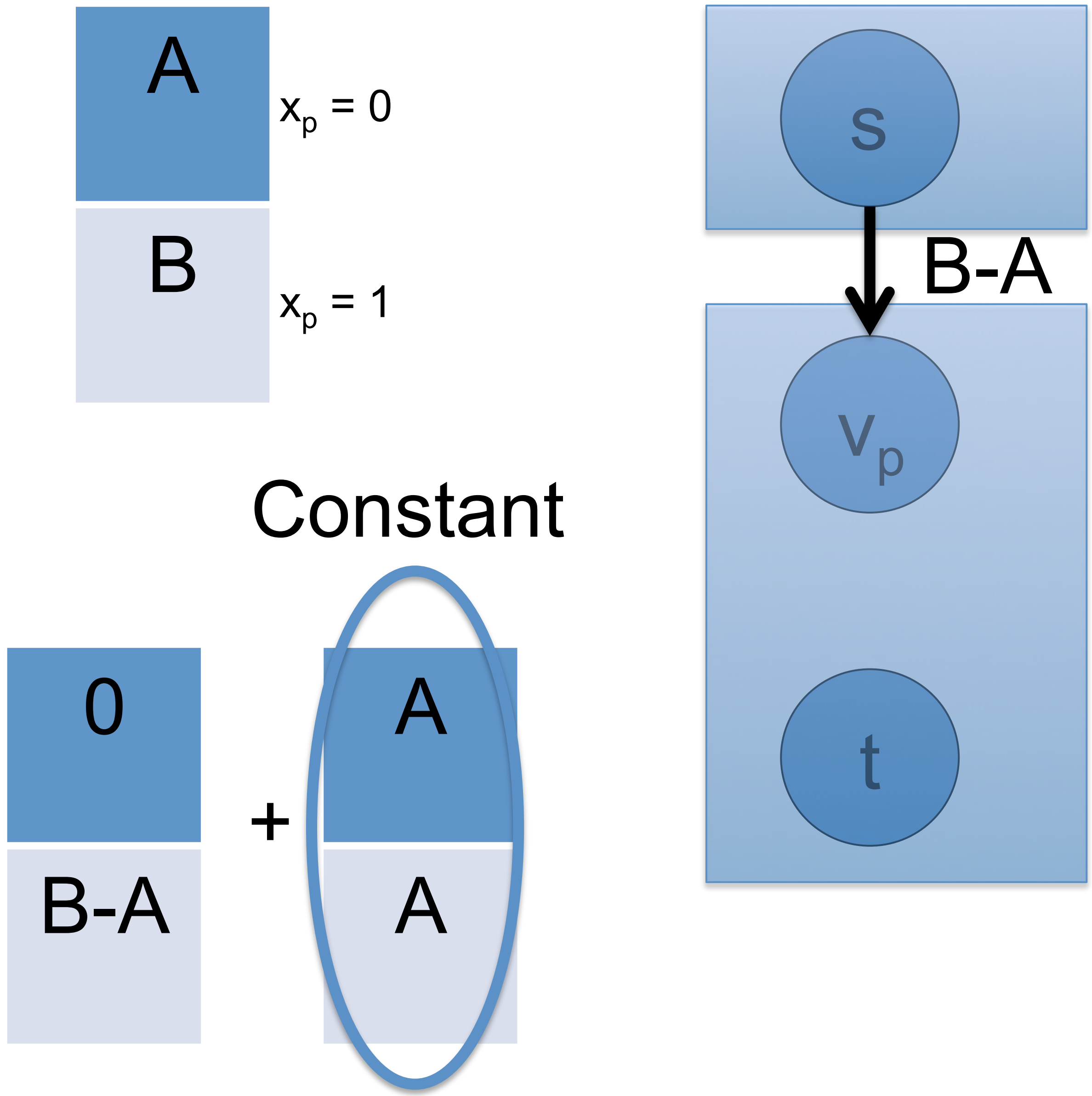


Let A ≥ B

$x_p = 0$

A-B

Digraph for Unary Potentials

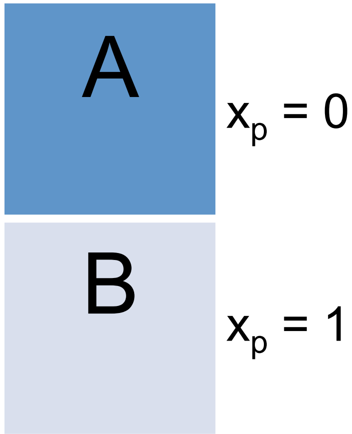


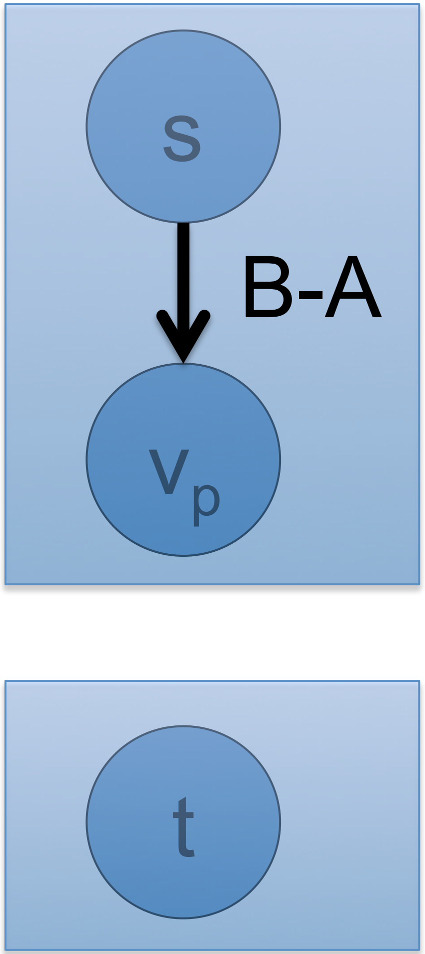
Let A < B

$x_p = 1$

B-A

Digraph for Unary Potentials



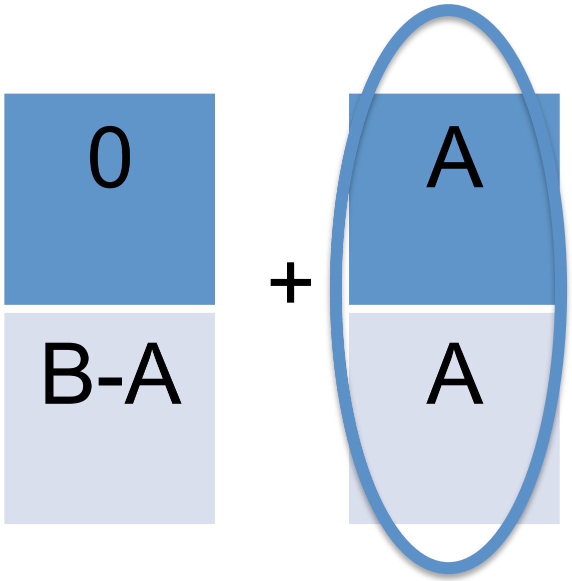


Let $A < B$

$x_p = 0$

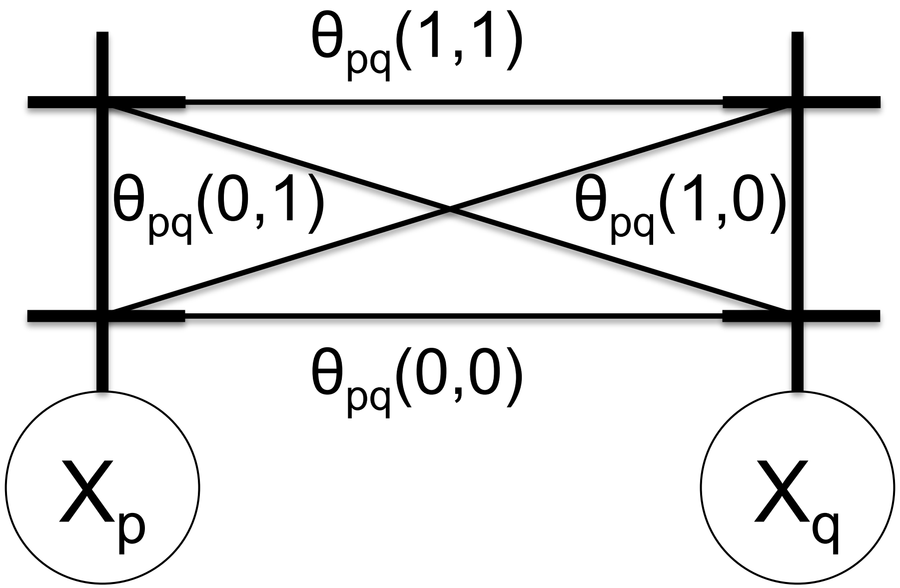
0

Constant



Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D



A	A
A	A

\+

0	0
B-A	B-A

\+

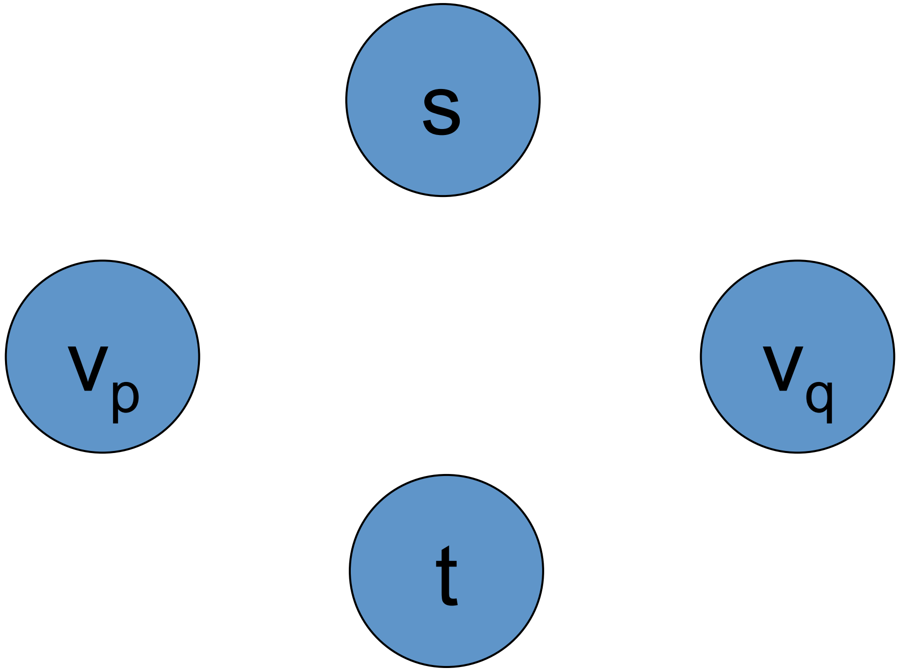
0	D-B
0	D-B

\+

0	C+B-D-A
0	0

Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D



Constant

A	A
A	A

+

0	0
B-A	B-A

+

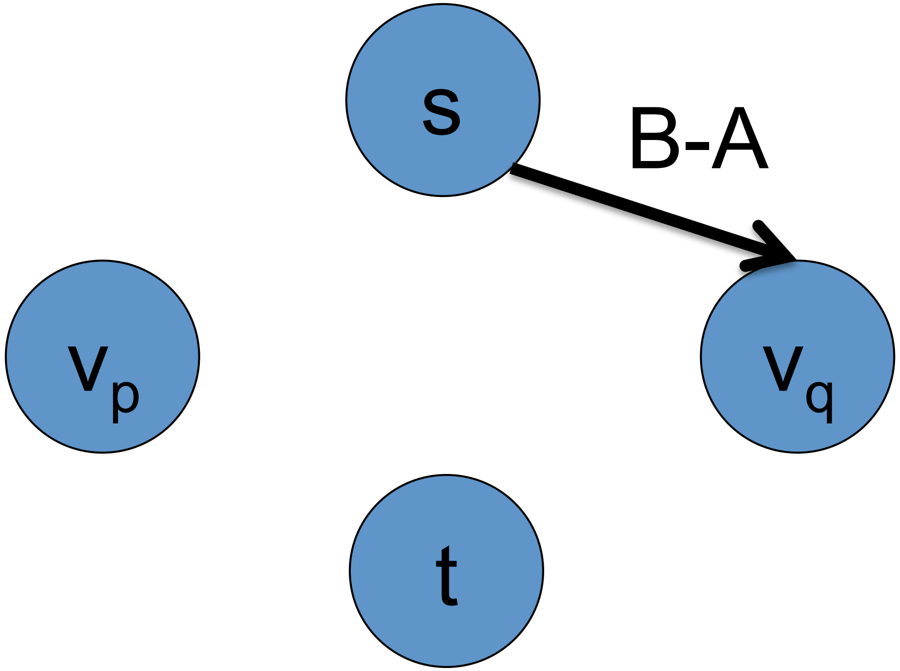
0	D-B
0	D-B

+

0	C+B-D-A
0	0

Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D



Unary Potential

$x_q = 1$

0	0
B-A	B-A

+

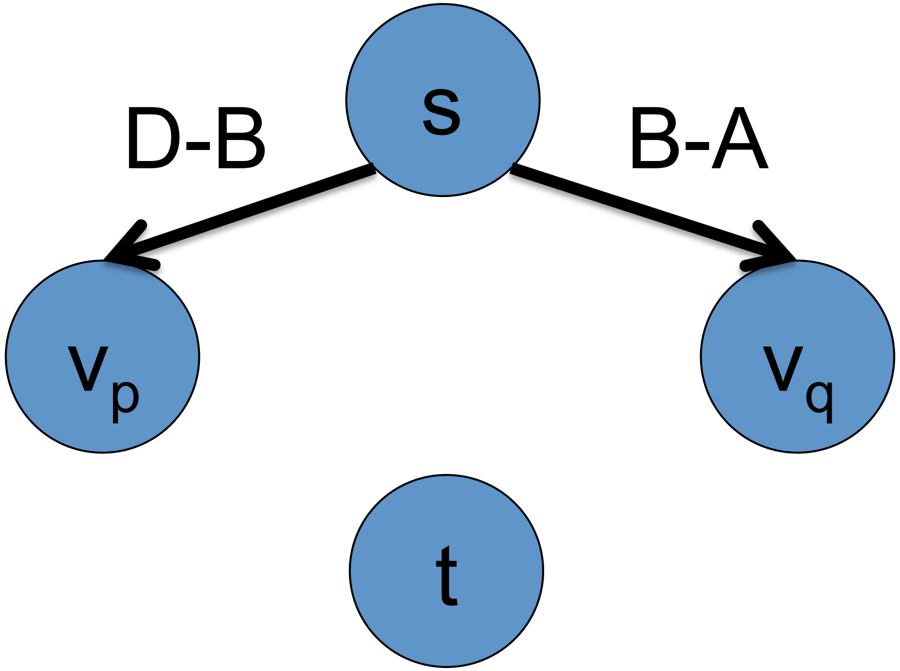
0	D-B
0	D-B

+

0	C+B-D-A
0	0

Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D



Unary Potential
$x_p = 1$

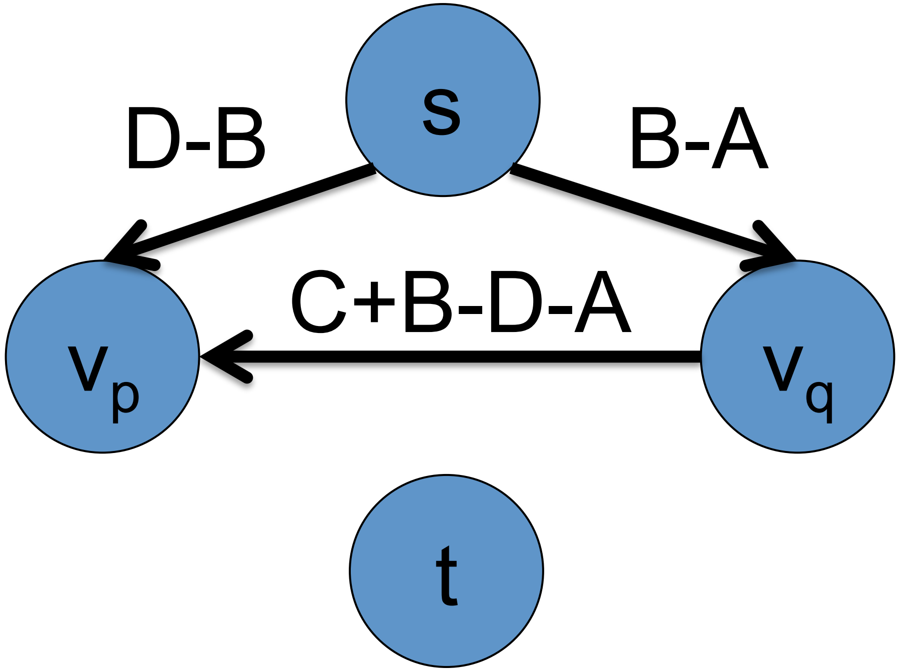
0	D-B
0	D-B

+

0	C+B-D-A
0	0

Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D



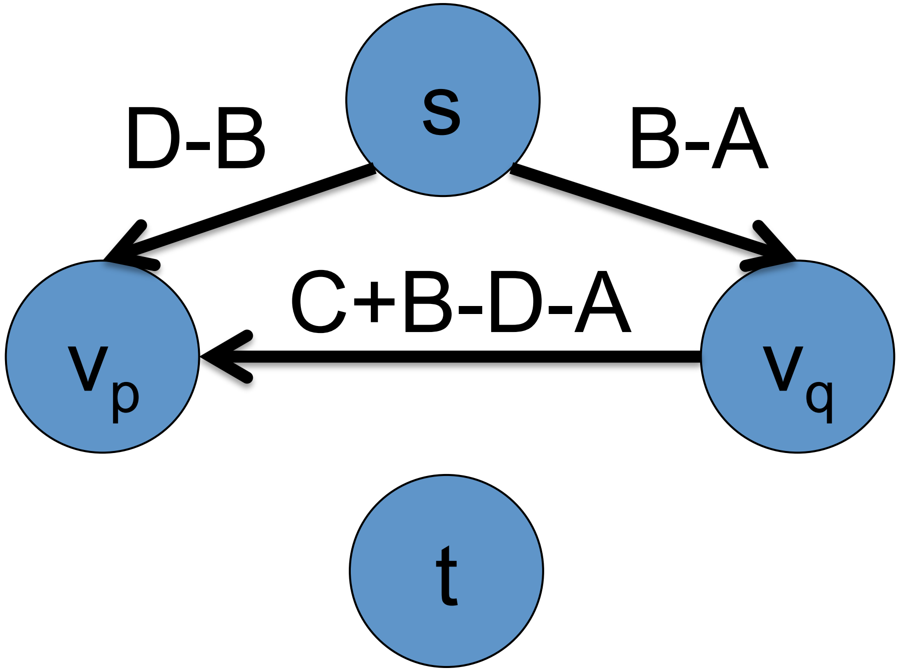
Pairwise Potential

$x_p = 1$, $x_q = 0$

0	C+B-D-A
0	0

Digraph for Pairwise Potentials

	$x_p = 0$	$x_p = 1$
$x_q = 0$	A	C
$x_q = 1$	B	D

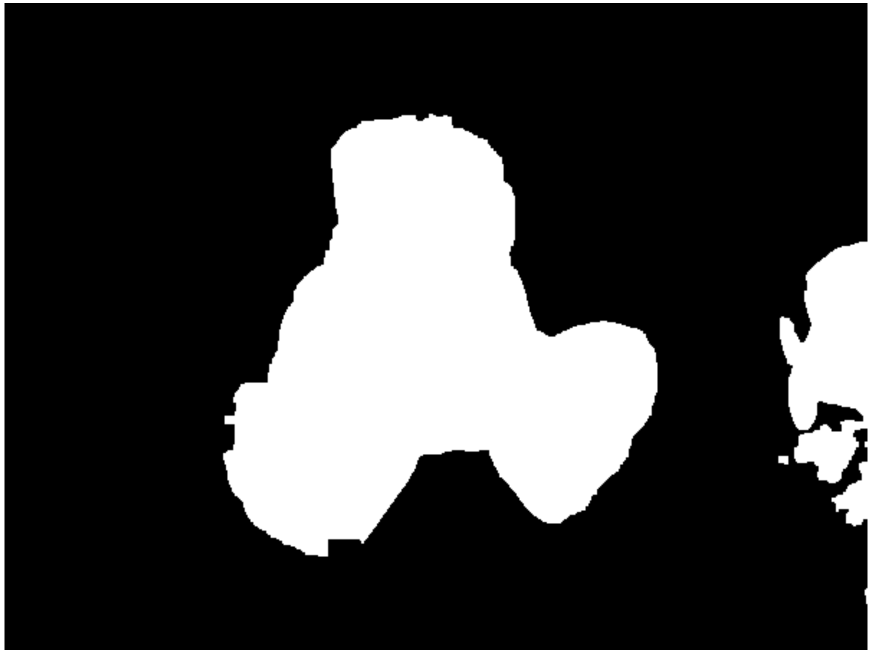


C+B-D-A ≥ 0

Submodular Energy

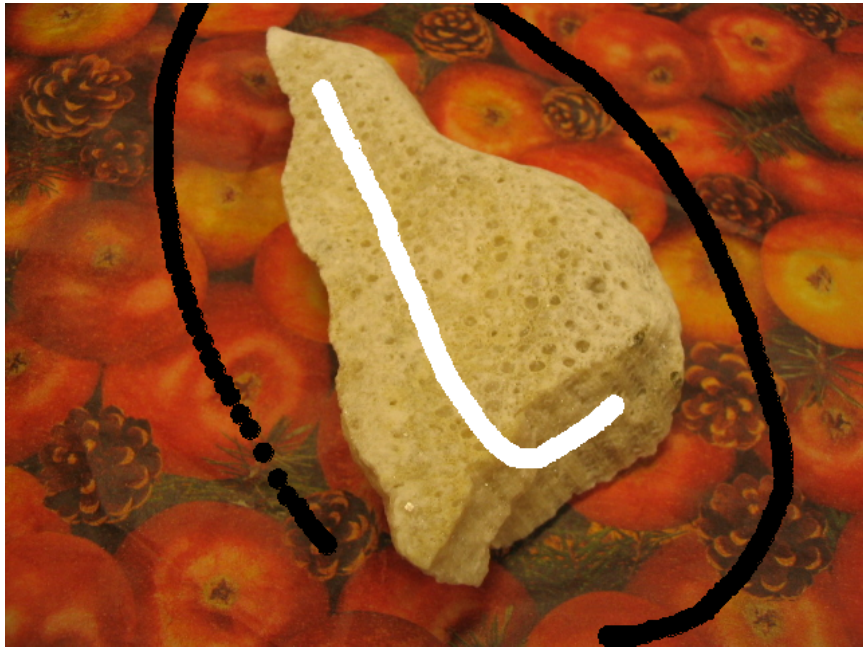
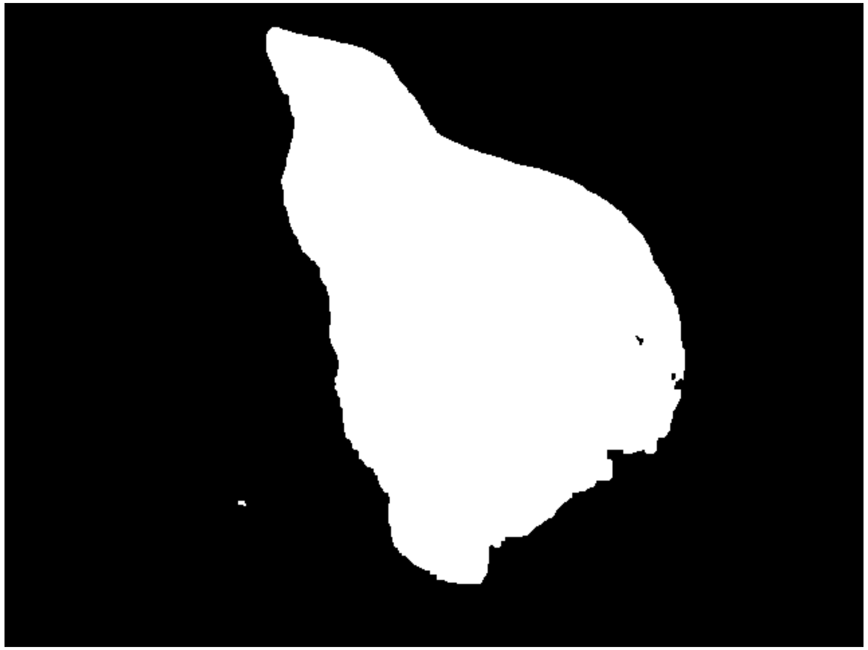
General 2-label MAP estimation is NP-hard

Results – Image Segmentation



Boykov and Jolly, ICCV 2001

Results – Image Segmentation

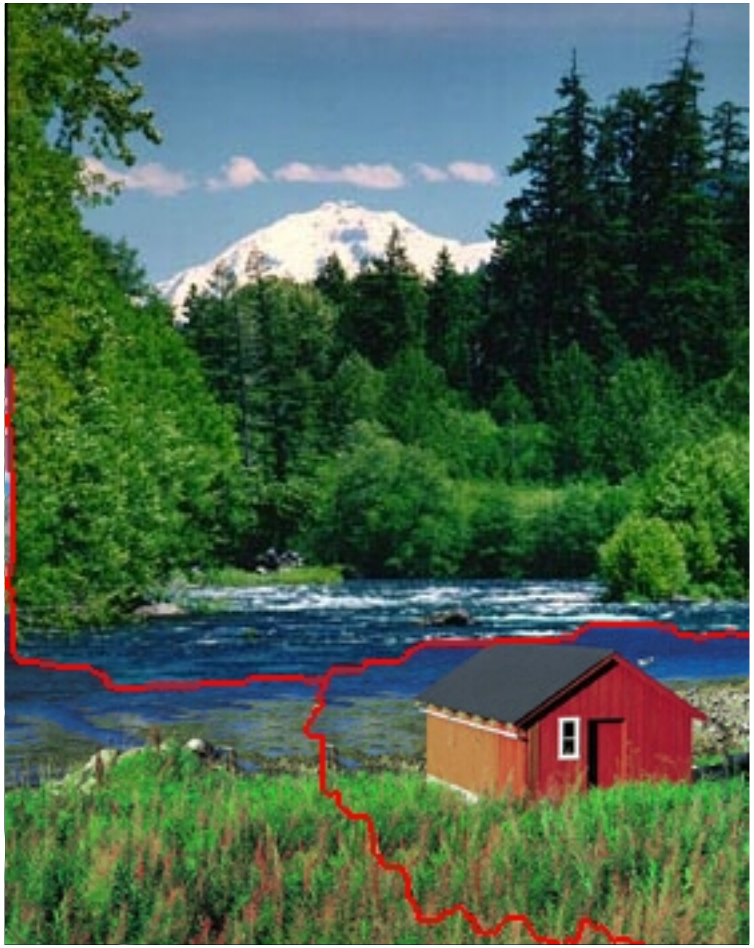
 

Boykov and Jolly, ICCV 2001

Results – Image Segmentation


Boykov and Jolly, ICCV 2001

Results – Image Synthesis



Kwatra et al., SIGGRAPH 2003

Results – Image Synthesis

Kwatra et al., SIGGRAPH 2003

Outline

- Preliminaries
- Maximum Flow
- Algorithms
- **Energy minimization with max flow/min cut**
  - Multi-Label Energy Functions

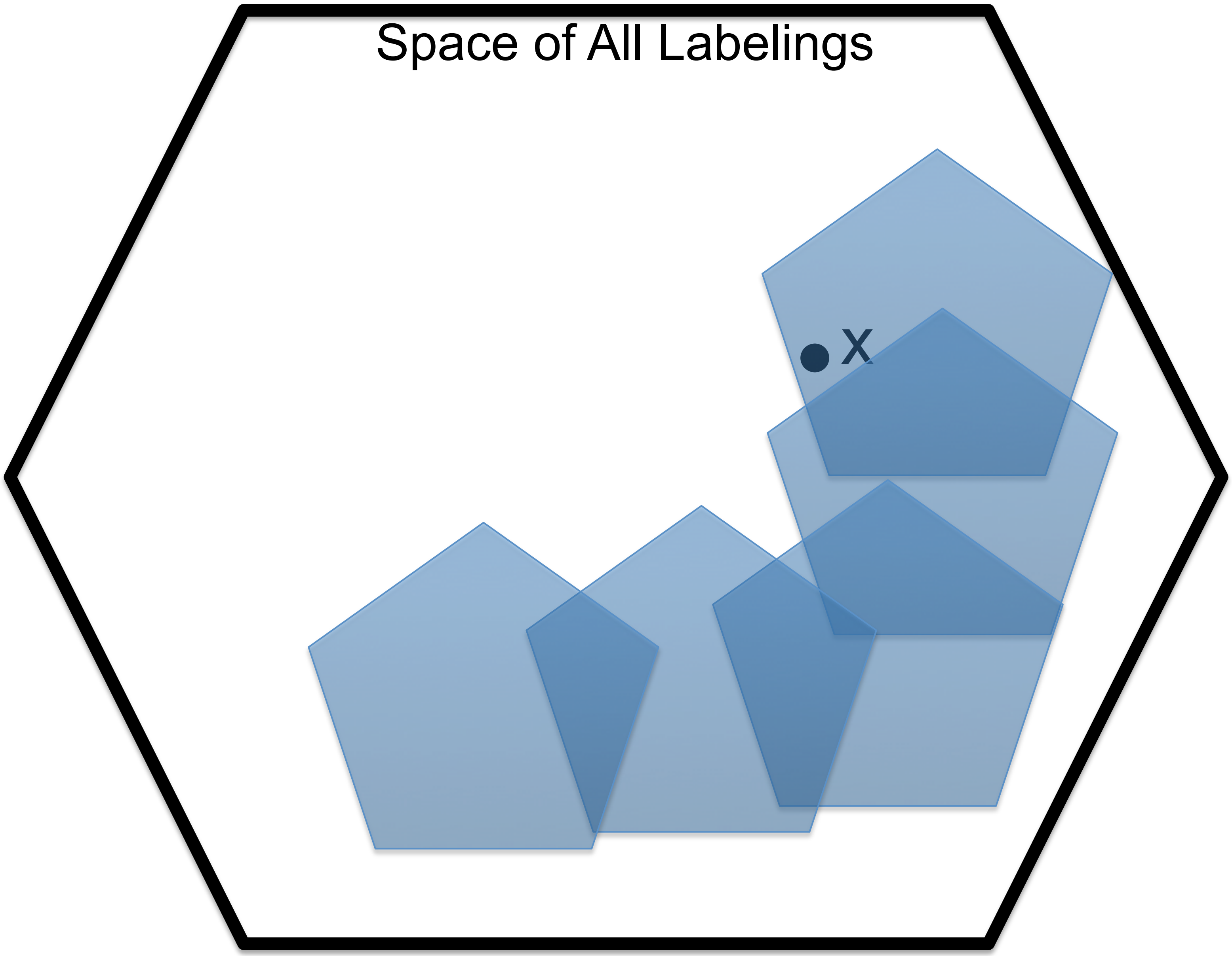
St-mincut based Move algorithms

$$E(\mathbf{x}) = \sum_i \theta_i (x_i) + \sum_{i,j} \theta_{ij} (x_i,x_j)$$

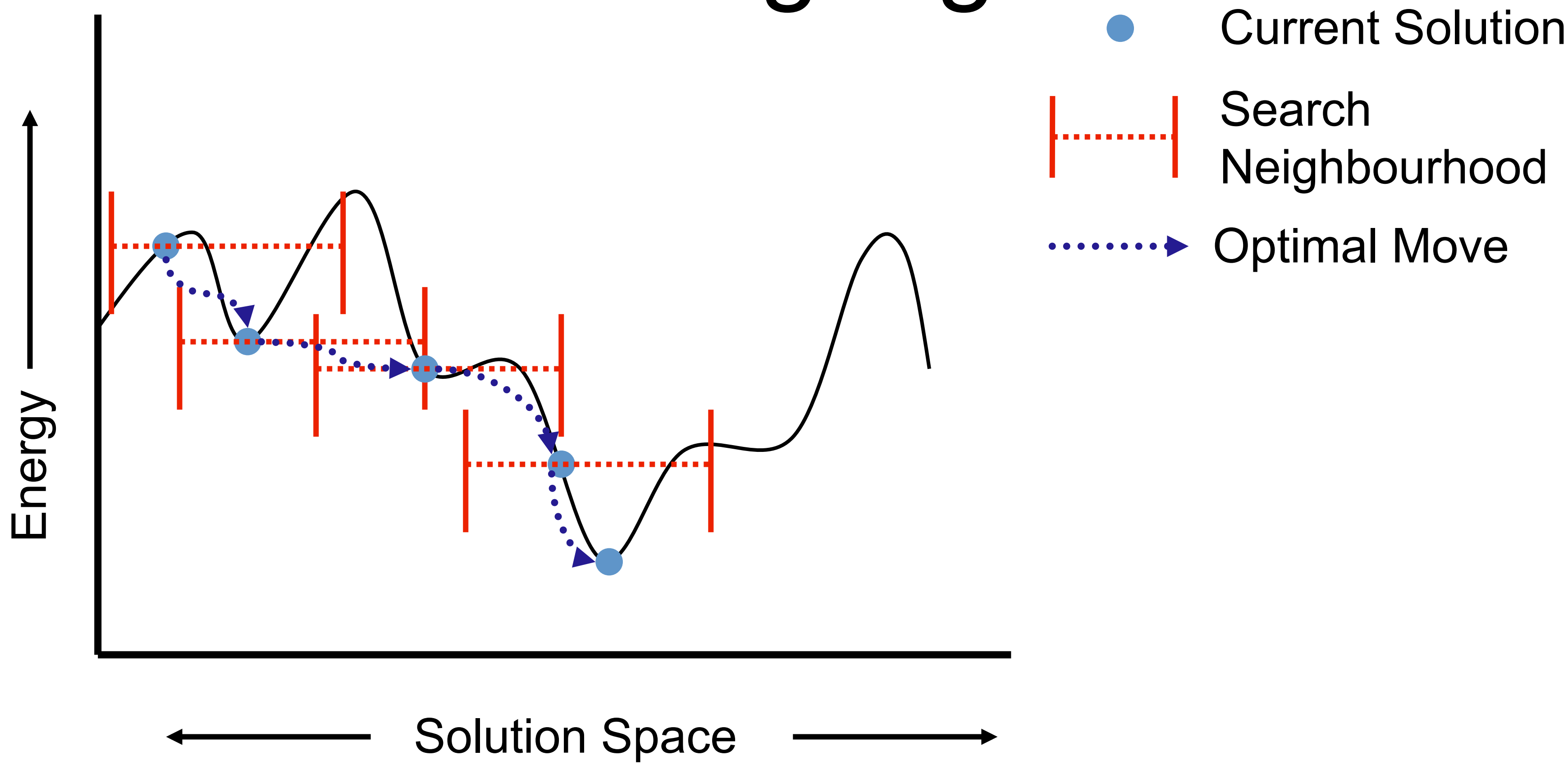
$x \in$ Labels $L = \{l_1, l_2, \ldots, l_k\}$

- Commonly used for solving **non-submodular** multi-label problems
- Extremely efficient and produce good solutions
- Not Exact: Produce local optima

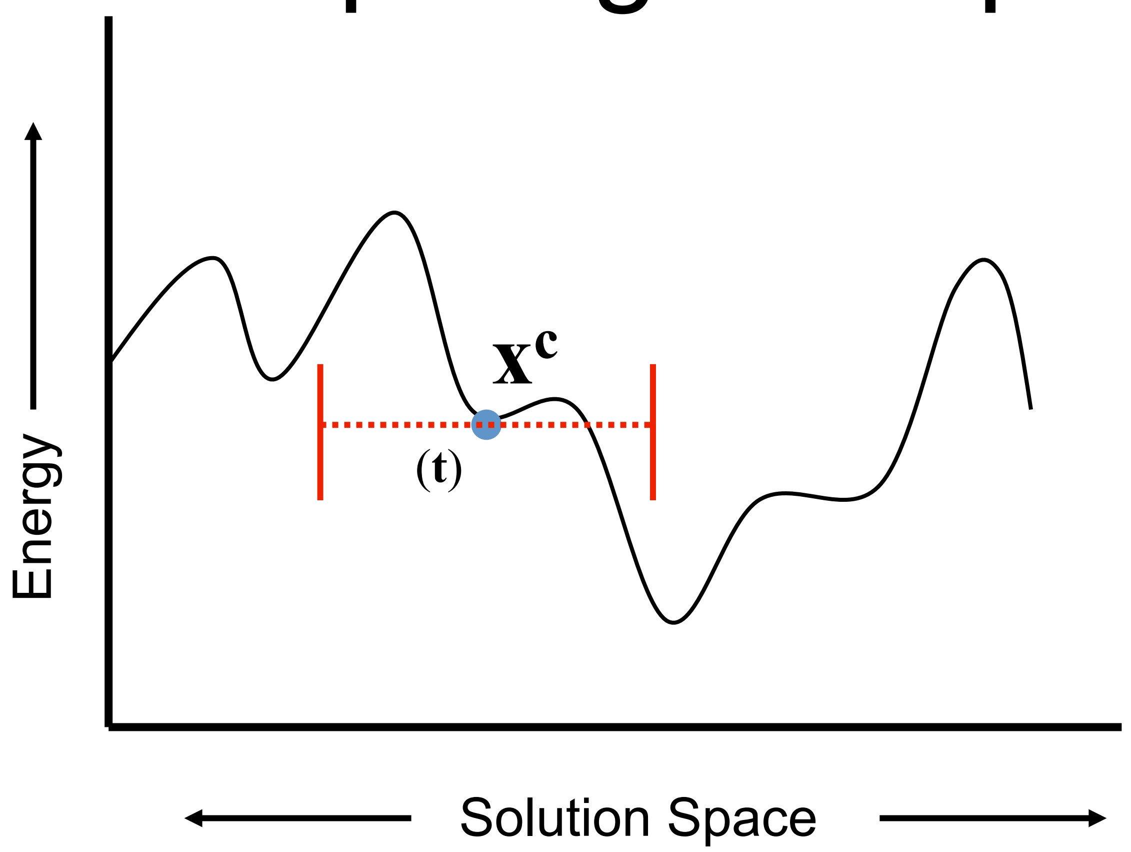
Move-Making Algorithms



Move Making Algorithms



Computing the Optimal Move



Key Property

Move Space

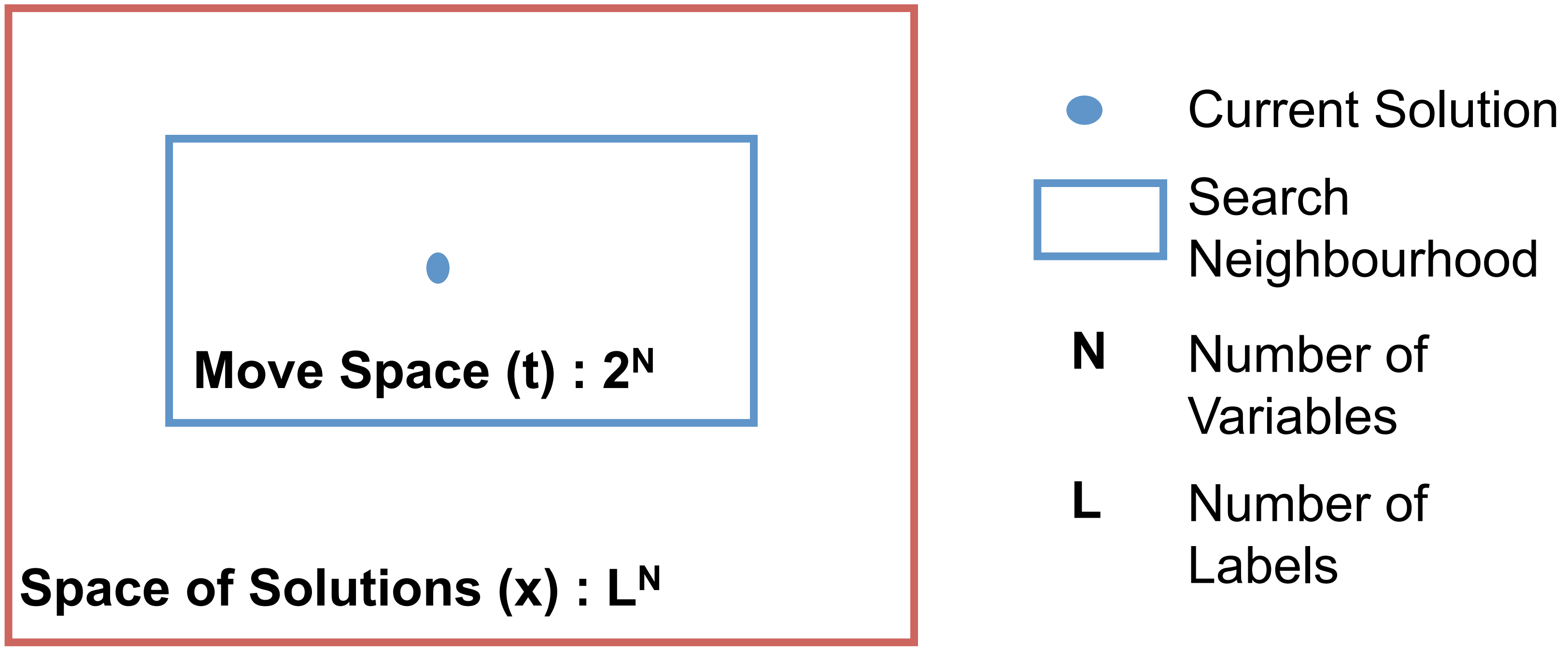
Bigger move space →
- **Better solutions**
- **Finding the optimal move hard**

Moves using Graph Cuts

Expansion and Swap move algorithms

[Boykov Veksler and Zabih, PAMI 2001]

- **Makes a series of changes to the solution (moves)**
- **Each move results in a solution with smaller energy**

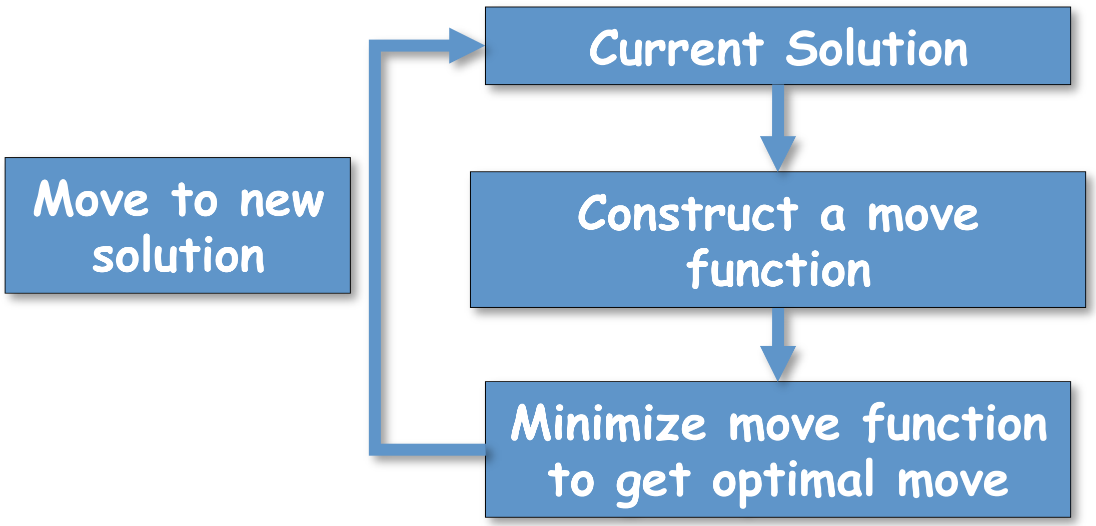


Moves using Graph Cuts

Expansion and Swap move algorithms

[Boykov Veksler and Zabih, PAMI 2001]

- **Makes a series of changes to the solution (moves)**
- **Each move results in a solution with smaller energy**



How to minimize move functions?

Expansion Algorithm

Variables take label l_α or retain current label

Tree
Ground
House
Sky

Status: **Initialize with Tree** **Expand Ground** **Expand House** **Expand Sky**

Slide courtesy Pushmeet Kohli

Expansion Algorithm

Initialize labeling $\mathbf{x} = \mathbf{x}^0$ (say $x^0_p = 0$, for all X_p)

For $\alpha = 1, 2, \ldots, h-1$

$\mathbf{x}^\alpha = \mathrm{argmin}_{\mathbf{x}'} E(\mathbf{x}')$

s.t. $x'_p \in \{x_p\} \cup \{l_\alpha\}$

Update $\mathbf{x} = \mathbf{x}^\alpha$

End

Repeat until convergence

Boykov, Veksler and Zabih, 2001

Expansion Algorithm

Restriction on pairwise potentials?

- **Move energy is submodular if:**
  - **Unary Potentials: Arbitrary**
  - **Pairwise potentials: Metric**

$$\boxed{\theta_{ij}(l_a, l_b) + \theta_{ij}(l_b, l_c) \geq \theta_{ij}(l_a, l_c)}$$

Example: Potts model

[Boykov, Veksler, Zabih]

Swap Move

- Variables labeled α, β can swap their labels

Tree
Ground
House
Sky

Swap Sky, House

[Boykov, Veksler, Zabih]

Swap Move

- Variables labeled α, β can swap their labels

- **Move energy is submodular if:**
  - **Unary Potentials: Arbitrary**
  - **Pairwise potentials: Semimetric**

$$\theta_{ij}(l_a, l_b) \geq 0$$

$$\theta_{ij}(l_a, l_b) = 0 \quad \longleftrightarrow \quad a = b$$

Example: Potts model

[Boykov, Veksler, Zabih]

General Binary Moves

$$\underbrace{x}_{\text{New solution}} = t\,\underbrace{x^1}_{\text{First solution}} + (1-t)\,\underbrace{x^2}_{\text{Second solution}}$$

Minimize over move variables t

Move Type	First Solution	Second Solution	Guarantee
Expansion	Old solution	All alpha	Metric
Fusion	Any solution	Any solution	✗

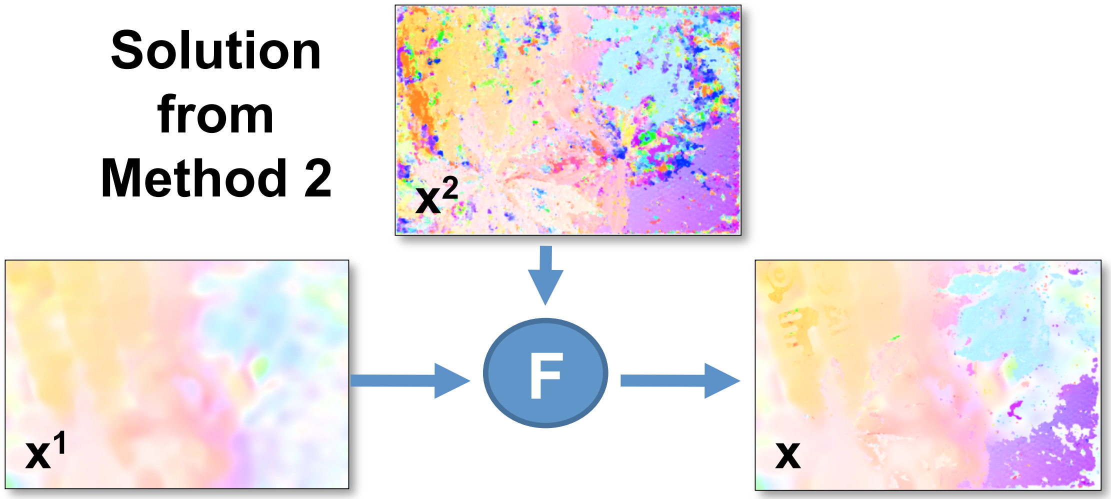
Solving Continuous Problems using Fusion Move

$$x = t\,x^1 + (1-t)\,x^2$$

Optical Flow Example

Solution from Method 2

Solution from Method 1

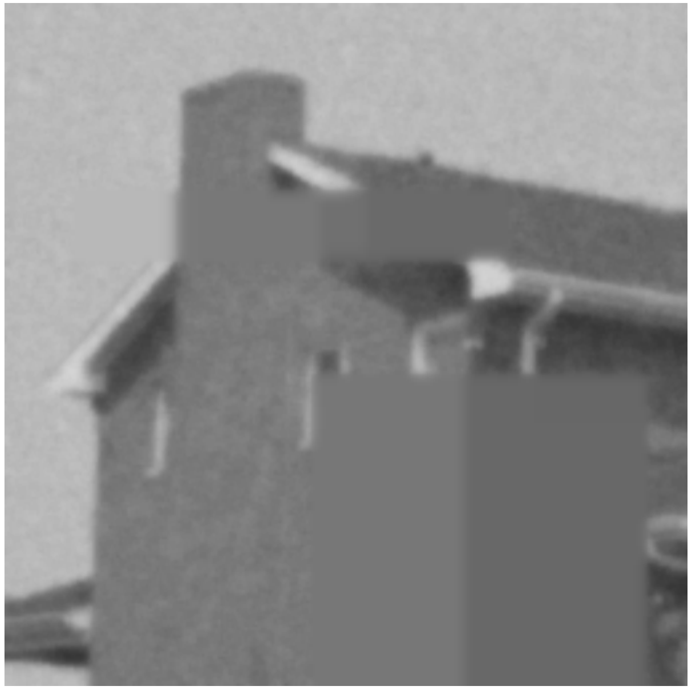


Final Solution

(Lempitsky et al. CVPR08, Woodford et al. CVPR08)

Results – Denoising + Inpainting

Results – Denoising + Inpainting



Results – Denoising + Inpainting

Results – Denoising + Inpainting

Paper presentation

- Tarabalka et al., “Spatio-temporal video segmentation with shape growth or shrinkage constraint”, Transactions on Image Processing 2014.