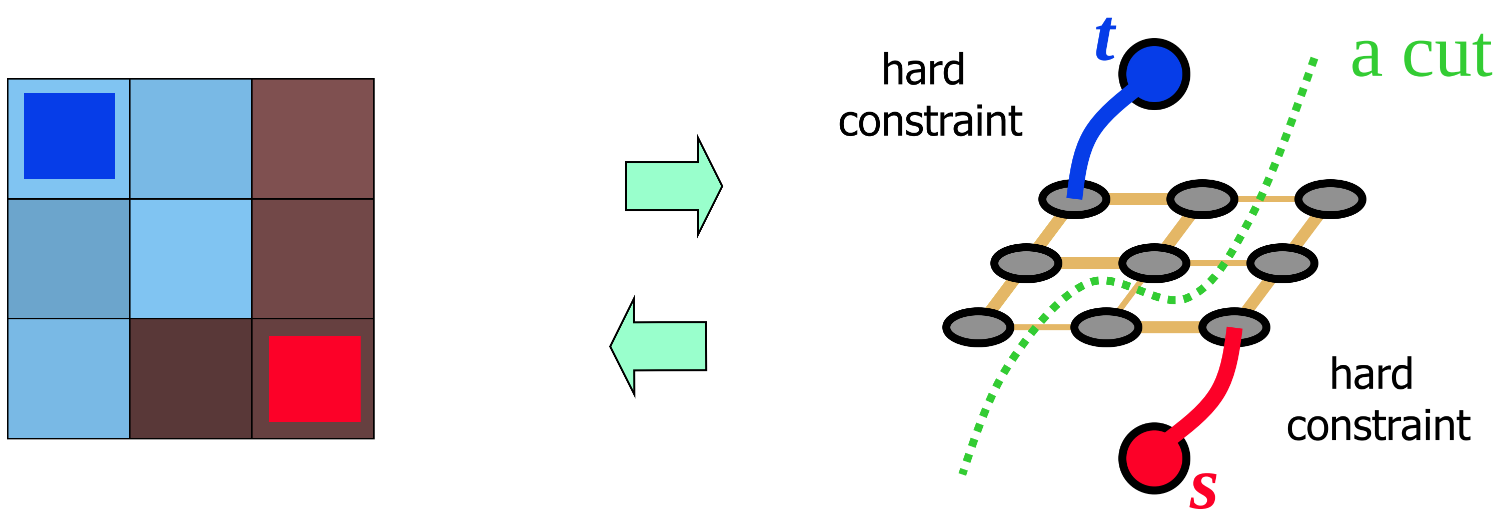
Markov Random Fields in Image Processing: Graph Cut - Part 3

Zoltan Kato

Image Processing and Computer Graphics Dept.
University of Szeged
Hungary

Graph cuts
(simple example à la Boykov&Jolly, ICCV’01)



Minimum cost cut can be computed in polynomial time

(max-flow/min-cut algorithms)

Slide credit: Yuri Boykov

Standard minimum *s-t* cuts algorithms

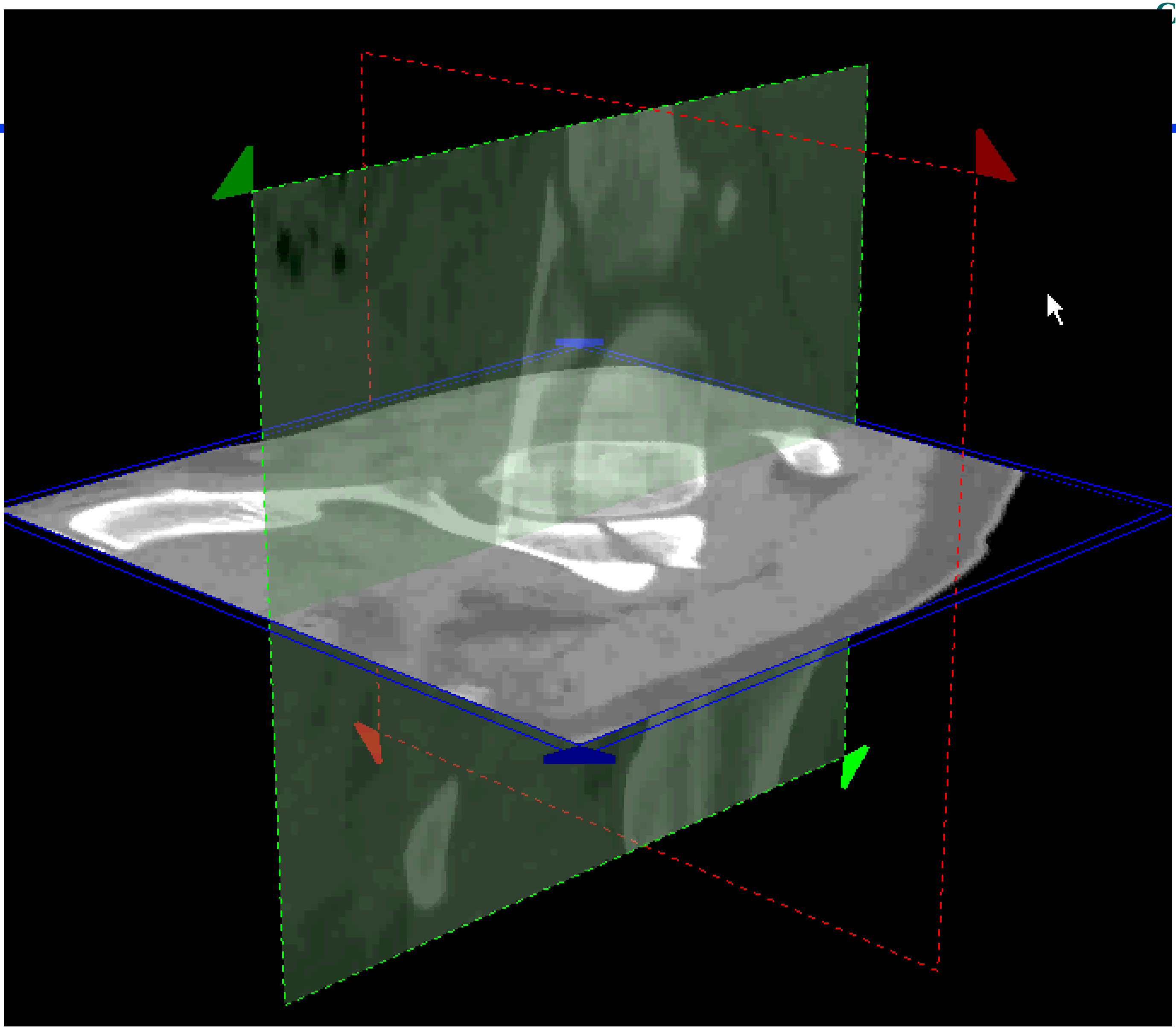
- Augmenting paths [Ford & Fulkerson, 1962]
- Push-relabel [Goldberg-Tarjan, 1986]

adapted to N-D grids used in computer vision

- Tree recycling (dynamic trees) [B&K, 2004]
- Flow recycling (*dynamic cuts*) [Kohli & Torr, 2005]
- Cut recycling (*active cuts*) [Juan & Boykov, 2006]
- Hierarchical methods
  - in search space [Lombaert et al., CVPR 2005]
  - in edge weights (*capacity scaling*) [Juan et al., ICCV07]

Slide credit: Yuri Boykov

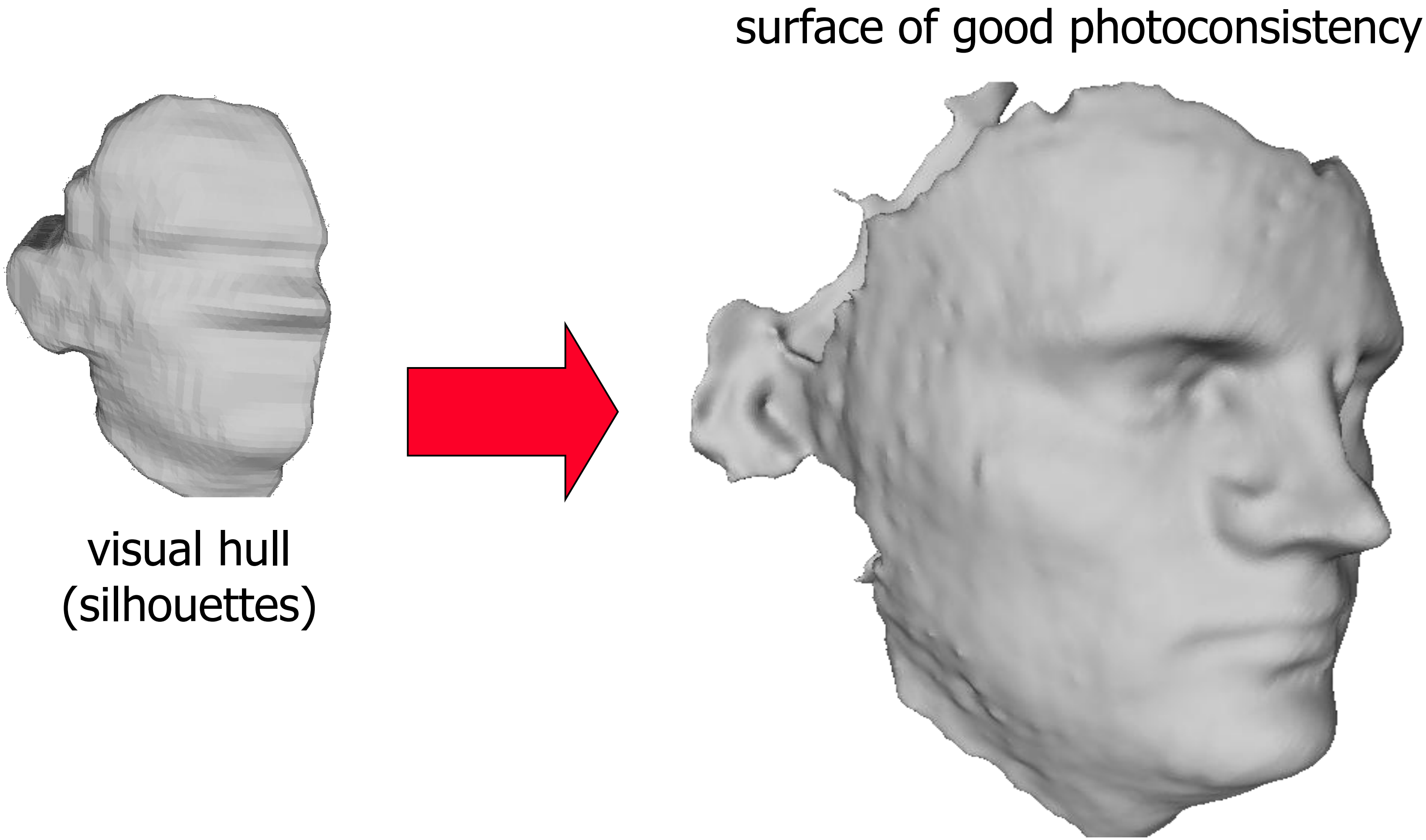
SIEMENS
Corp. Research
Princeton, NJ



3D bone segmentation (real time screen capture)

Slide credit: Yuri Boykov

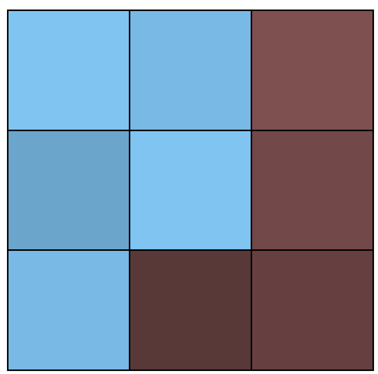
Graph cuts applied to multi-view reconstruction

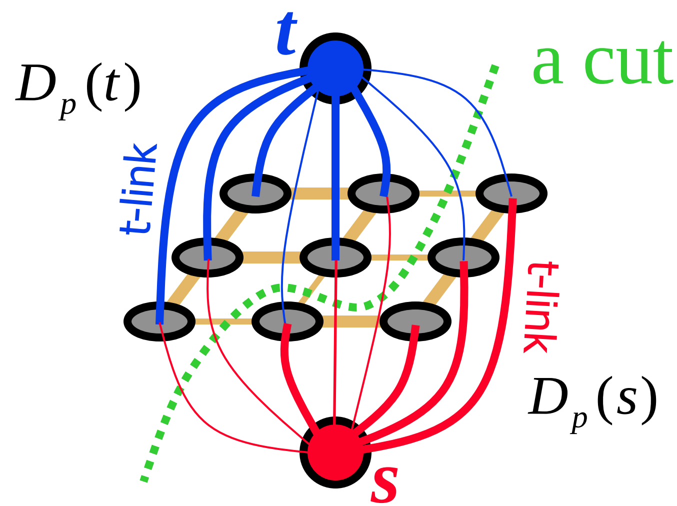


CVPR'05 slides from Vogiatzis, Torr, Cippola

Slide credit: Yuri Boykov

Adding regional properties
(B&J, ICCV'01)





regional bias example

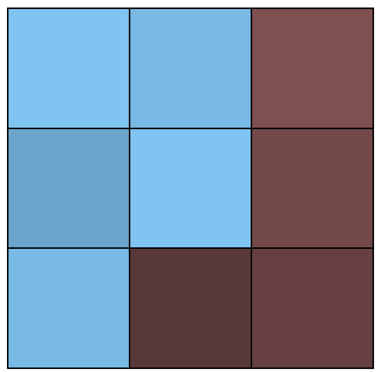
suppose I^s and I^t are given "expected" intensities of object and background

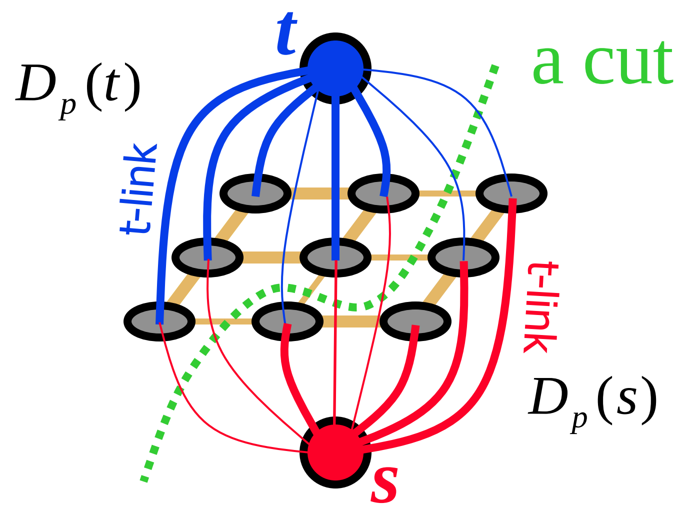
$$D_p(s) \propto \exp\left(-\|I_p - I^s\|^2 / 2\sigma^2\right)$$

$$D_p(t) \propto \exp\left(-\|I_p - I^t\|^2 / 2\sigma^2\right)$$

NOTE: hard constrains are not required, in general.

Adding regional properties
(B&J, ICCV'01)





"expected" intensities of
object and background
I^s and I^t
can be re-estimated

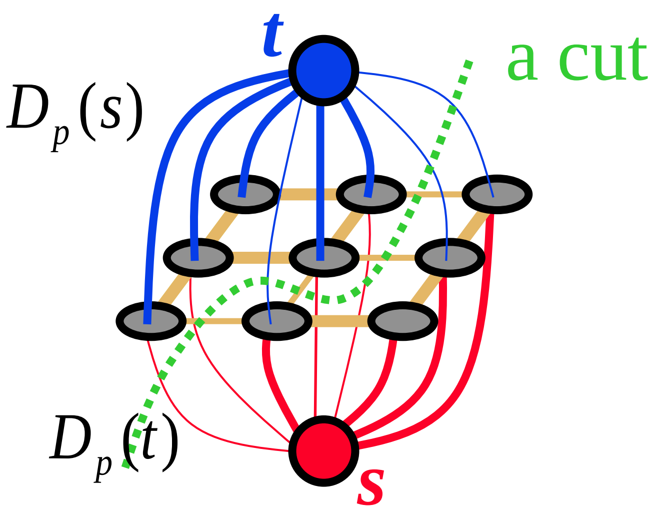
$$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$$

$$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$$

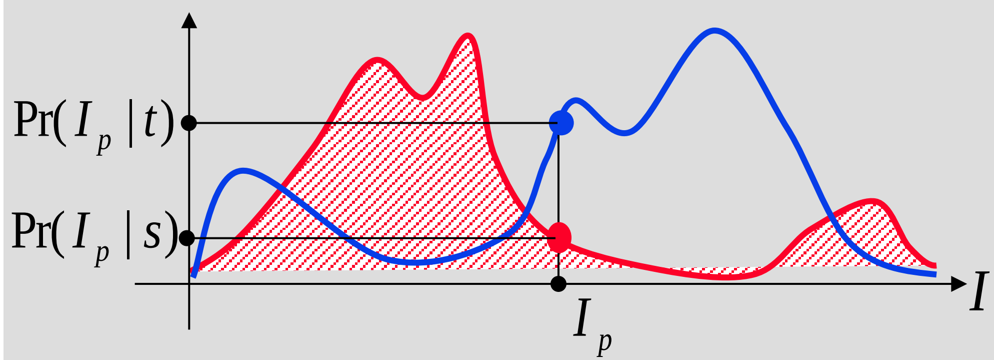
EM-style optimization of piece-vice constant *Mumford-Shah* model

Adding regional properties
(B&J, ICCV'01)

More generally, regional bias can be based on any intensity models of object and background

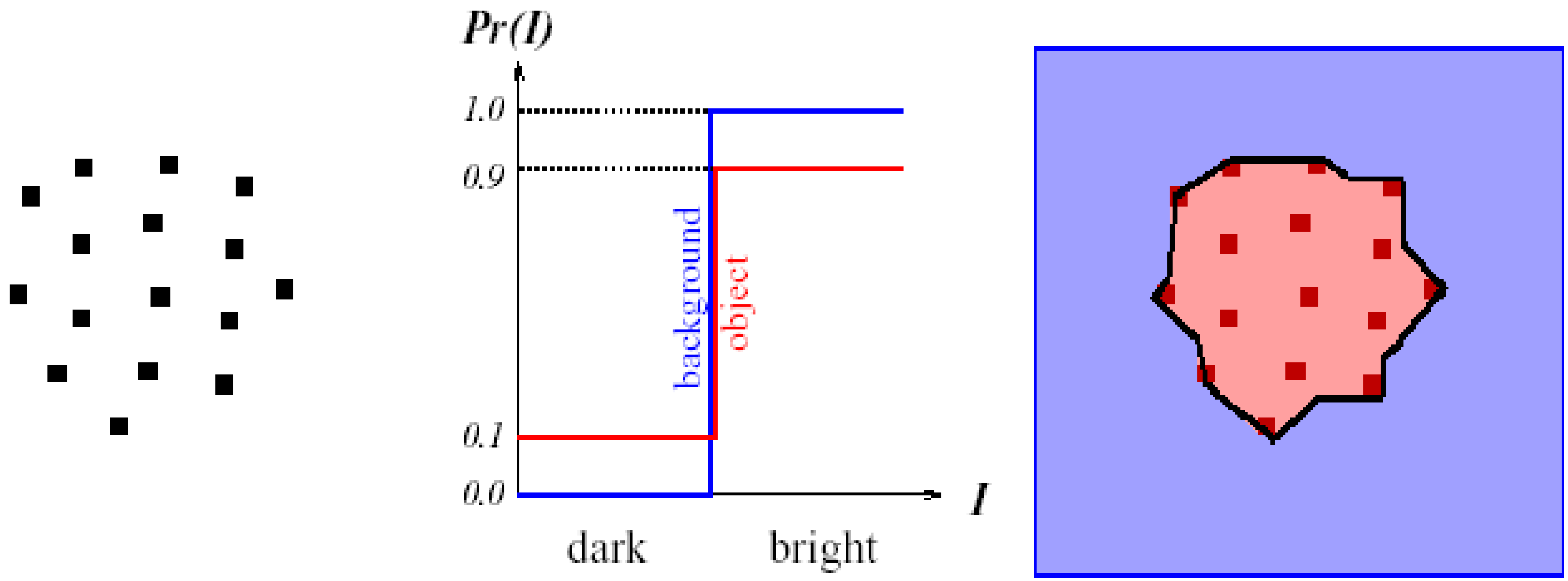


$$D_p(L_p) = -\ln \Pr(I_p \mid L_p)$$



given object and background intensity histograms

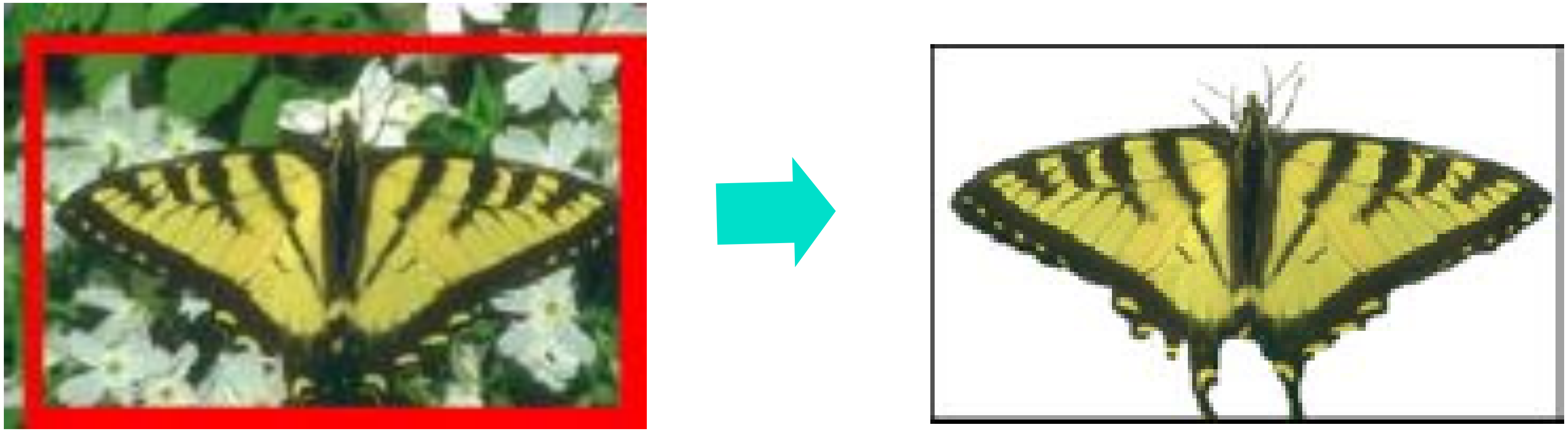
Adding regional properties
(B&J, ICCV'01)



(a) Original image (b) Intensity histograms (c) Optimal segmentation

Iterative learning of regional color-models

- GMMRF cuts (Blake et al., ECCV04)
- Grab-cut (Rother et al., SIGGRAPH 04)



parametric regional model – Gaussian Mixture (GM)
designed to guarantee convergence

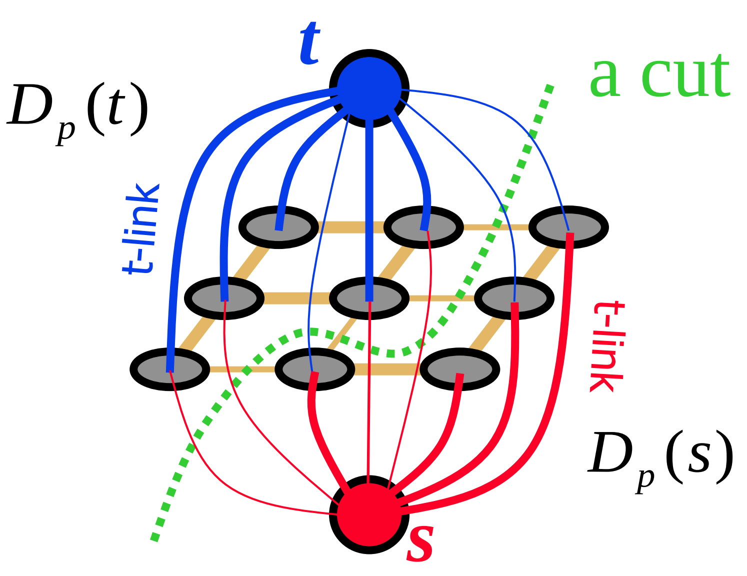
Slide credit: Yuri Boykov

Simple example of energy

Regional term Boundary term

$$E(L) = \sum_{p} D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q)$$

t-links **n-links**



$$L_p \in \{s, t\}$$

binary object segmentation

Slide credit: Yuri Boykov

Graph cuts for minimization of submodular binary energies I

Regional term — Boundary term

$$E(L) = \sum_{p} E_p(L_p) + \sum_{pq \in N} E(L_p, L_q)$$

t-links — n-links

$L_p \in \{s,t\}$

- Characterization of **binary** energies that can be globally minimized by *s-t* graph cuts [Boros&Hummer, 2002, K&Z 2004]

E(L) can be minimized by *s-t* graph cuts $\Leftrightarrow$ $E(s,s)+E(t,t) \leq E(s,t)+E(t,s)$

Submodularity ("convexity")

- **Non-submodular cases** can be addressed with some optimality guarantees, e.g. ***QPBO*** algorithm
  - (see Boros&Hummer, 2002, Tavares et al. 06, Rother et al. 07)

Slide credit: Yuri Boykov

The Problem

$$E(x) = \sum_i f_i(x_i) + \sum_{ij} g_{ij}(x_i, x_j) + \sum_c h_c(x_c)$$

Unary *Pairwise* *Higher Order*

$$\sum_i c_i x_i + \sum_{i,j} d_{ij} |x_i - x_j|$$

$E: \{0,1\}^n \rightarrow R$

n = number of pixels

Image

Segmentation

Slide credit: Pushmeet Kohli

Submodular Functions: Definition

Pseudo-boolean function $f:\{0,1\}^n \rightarrow \mathbb{R}$ is submodular if

$$f(A) + f(B) \geq f(A \vee B) + f(A \wedge B) \quad \text{for all } A,B \in \{0,1\}^n$$

(OR) (AND)

Example: $n = 2$, $A = [1,0]$, $B = [0,1]$

$$f([1,0]) + f([0,1]) \geq f([1,1]) + f([0,0])$$

Property : Sum of submodular functions is submodular

Binary Image Segmentation Energy is submodular

$$E(x) = \sum_i c_i x_i + \sum_{i,j} d_{ij} |x_i - x_j|$$

Slide credit: Pushmeet Kohli

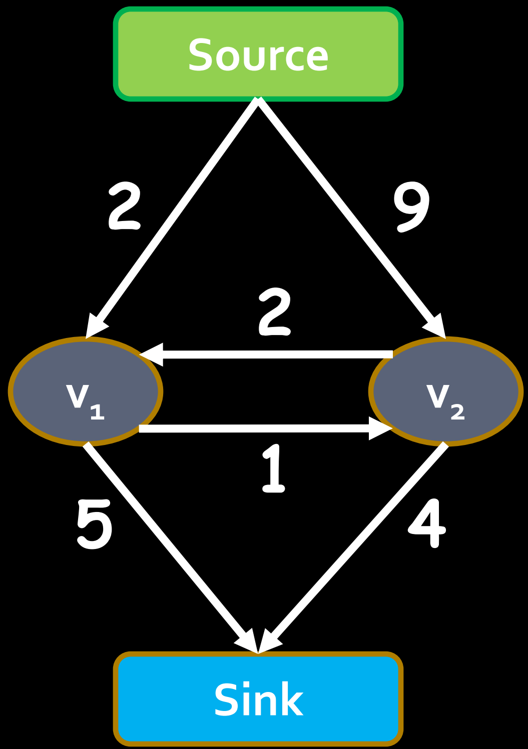
Minimizing Submodular Functions

- Polynomial time algorithms
  - Ellipsoid Algorithm: [Grotschel, Lovasz & Schrijver '81]
  - First strongly polynomial algorithm: [Iwata et al. '00] [A. Schrijver '00]
  - Current Best: $O(n^5 Q + n^6)$ [Q is function evaluation time] [Orlin '07]
- Symmetric functions: $E(x) = E(1-x)$
  - Can be minimized in $O(n^3)$
- Minimizing Pairwise submodular functions
  - Can be transformed to st-mincut/max-flow [Hammer , 1965]
  - Very low empirical running time ~ $O(n)$

$$E(X) = \sum_i f_i(x_i) + \sum_{ij} g_{ij}(x_i, x_j)$$

Slide credit: Pushmeet Kohli

The st-Mincut Problem



Graph (V, E, C)

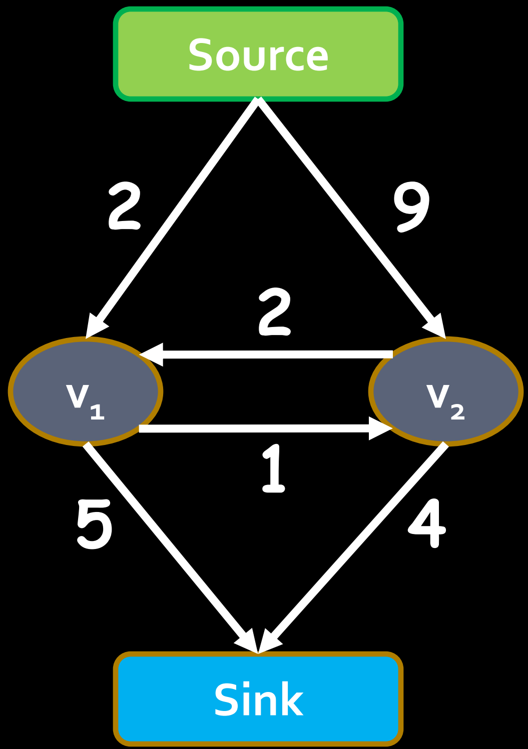
Vertices $V = \{v_1, v_2 \ldots v_n\}$

Edges $E = \{(v_1, v_2) \ldots.\}$

Costs $C = \{c_{(1, 2)} \ldots.\}$

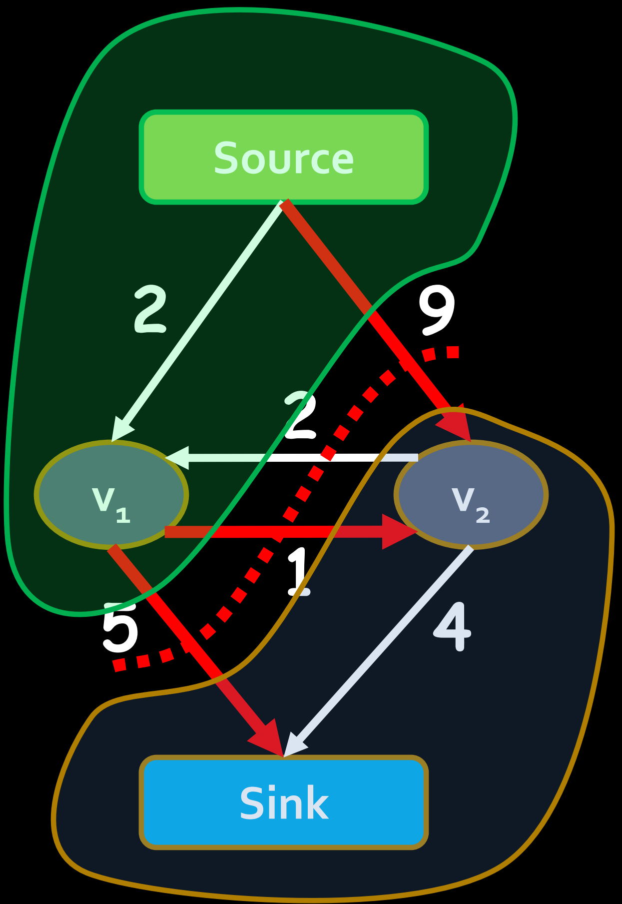
The st-Mincut Problem

What is a st-cut?



Slide credit: Pushmeet Kohli

The st-Mincut Problem



5 + 1 + 9 = 15

What is a st-cut?

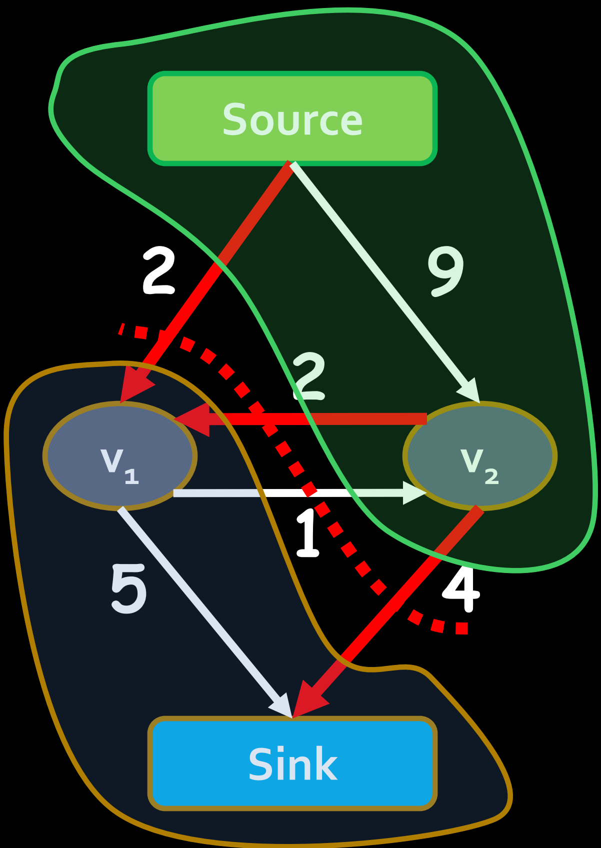
An st-cut (**S**,**T**) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

Slide credit: Pushmeet Kohli

The st-Mincut Problem



2 + 2 + 4 = 8

What is a st-cut?

An st-cut (**S**,**T**) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

What is the st-mincut?

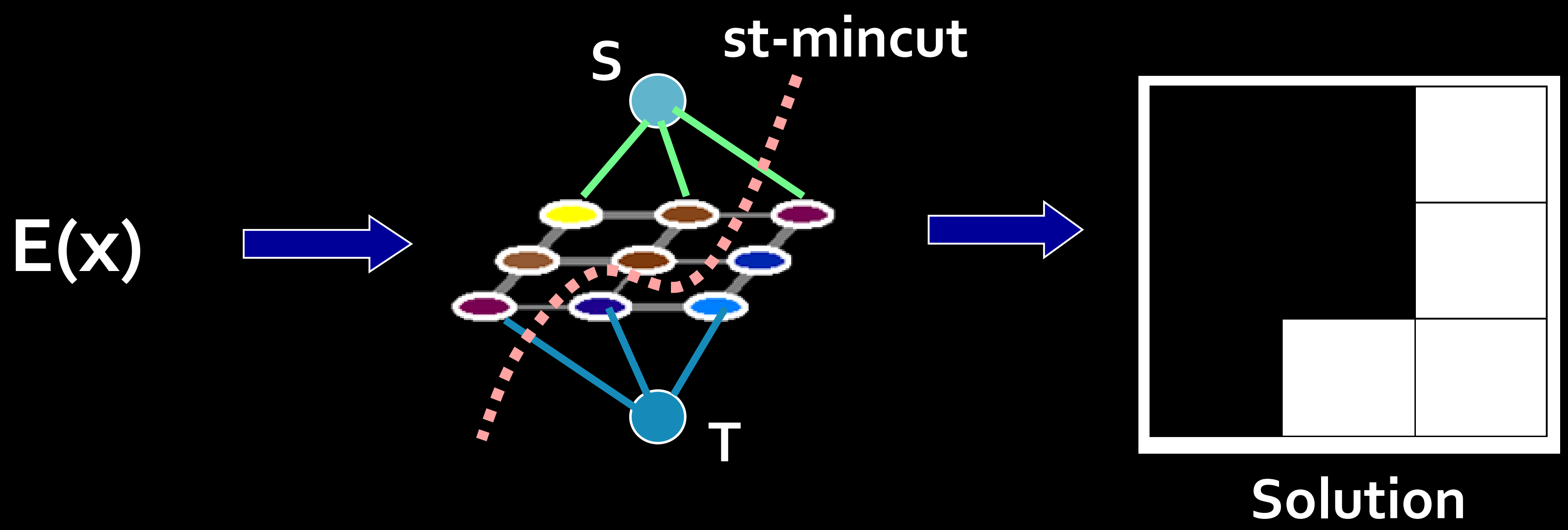
st-cut with the minimum cost

Slide credit: Pushmeet Kohli

So how does this work?

Construct a graph such that:

1. Any st-cut corresponds to an assignment of x
2. The cost of the cut is equal to the energy of x : E(x)



[Hammer, 1965] [Kolmogorov and Zabih, 2002]

Slide credit: Pushmeet Kohli

St-mincut and Energy Minimization

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

For all ij $\theta_{ij}(0,1) + \theta_{ij}(1,0) \geq \theta_{ij}(0,0) + \theta_{ij}(1,1)$

Equivalent (transformable)

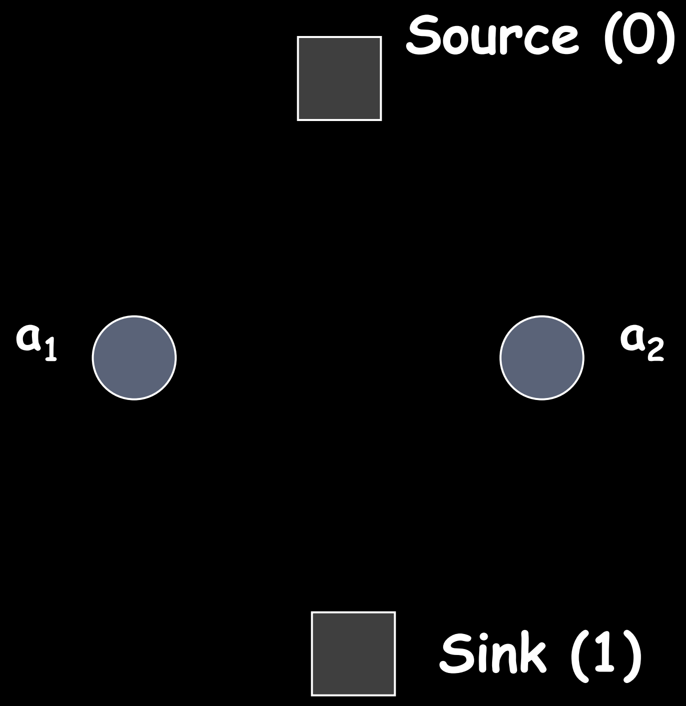
$$E(x) = \sum_i c_i x_i + \sum_{i,j} c_{ij} x_i(1-x_j)$$

$c_{ij} \geq 0$

Slide credit: Pushmeet Kohli

Graph Construction

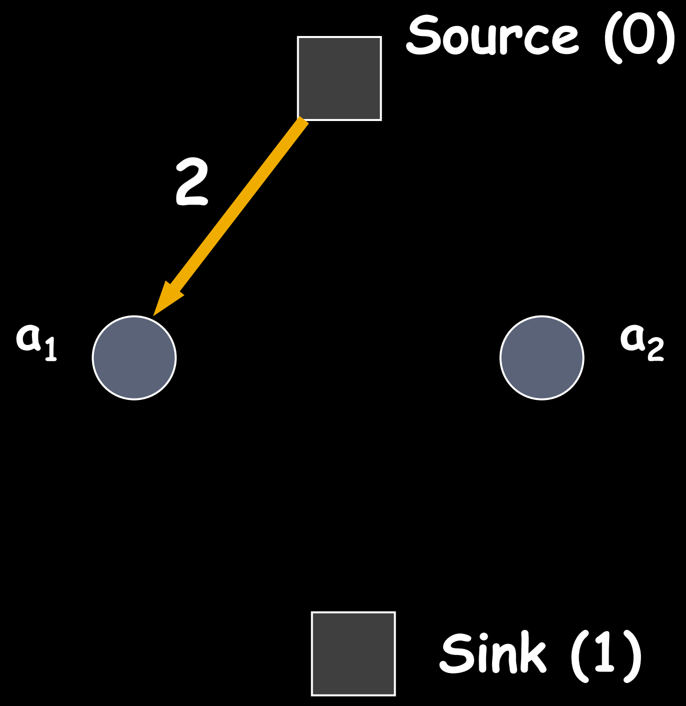
$E(a_1, a_2)$



Slide credit: Pushmeet Kohli

Graph Construction

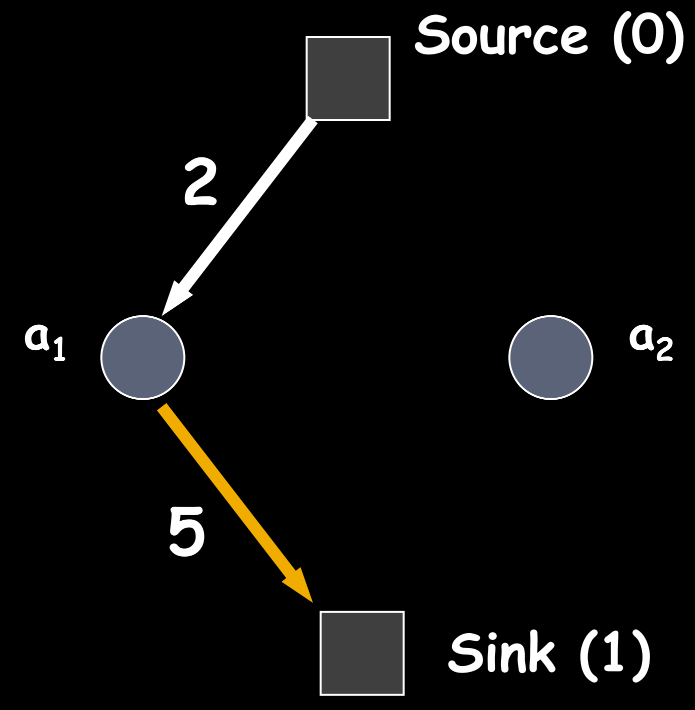
$E(a_1, a_2) = 2a_1$



Slide credit: Pushmeet Kohli

Graph Construction

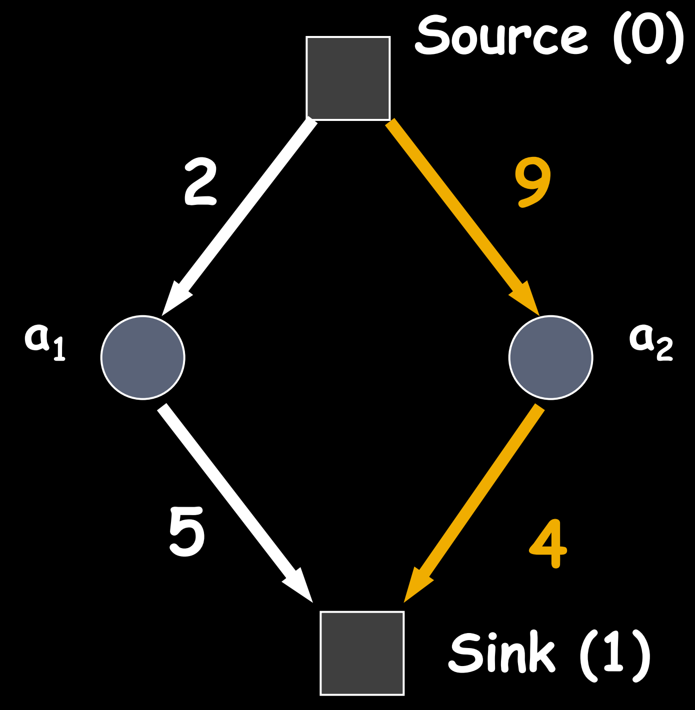
$E(a_1, a_2) = 2a_1 + 5\bar{a}_1$



Slide credit: Pushmeet Kohli

Graph Construction

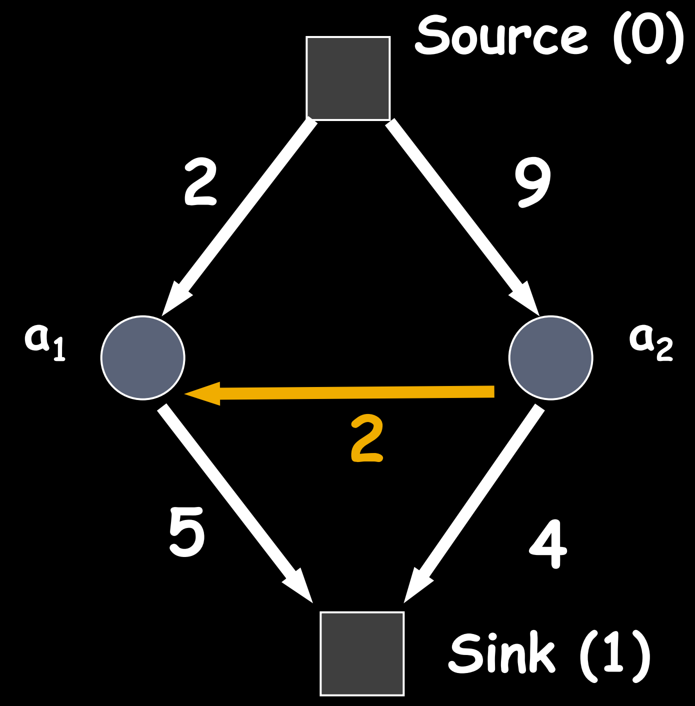
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2$$



Slide credit: Pushmeet Kohli

Graph Construction

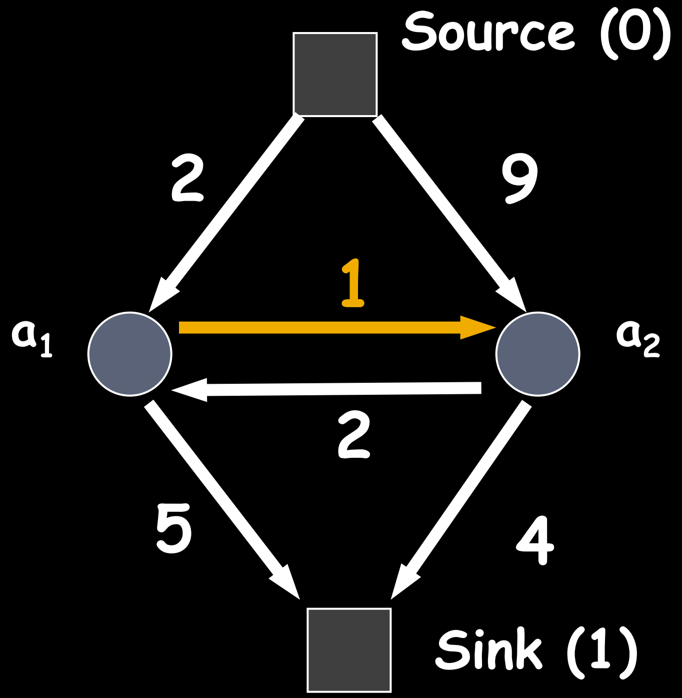
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2$$



Slide credit: Pushmeet Kohli

Graph Construction

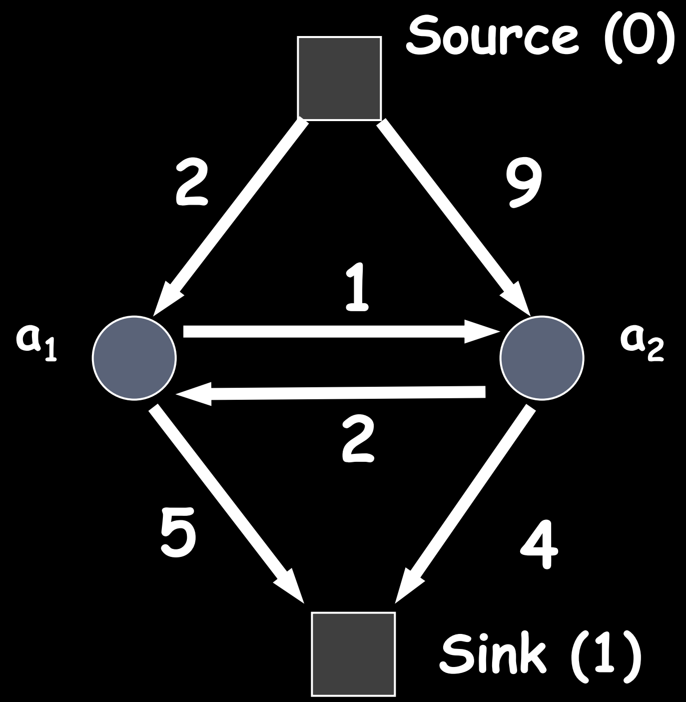
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



Slide credit: Pushmeet Kohli

Graph Construction

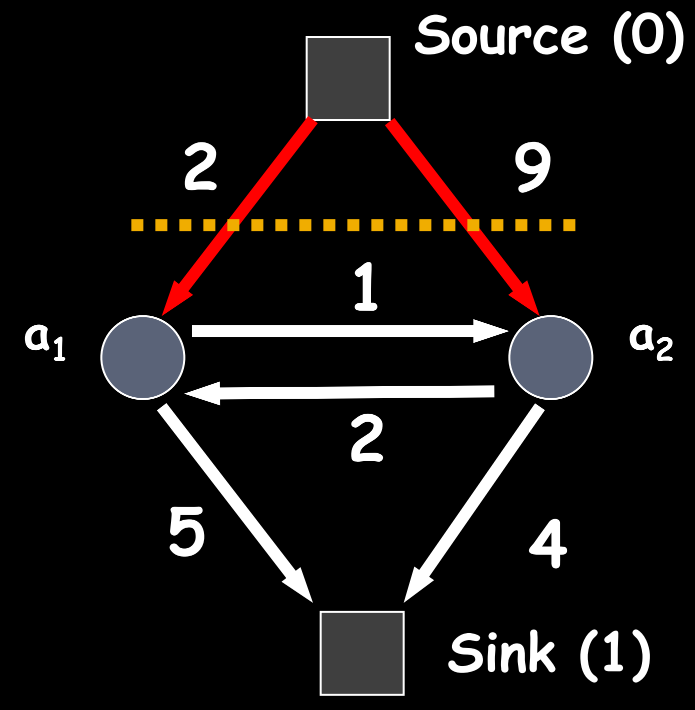
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$$



Slide credit: Pushmeet Kohli

Graph Construction

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



Cost of cut = 11

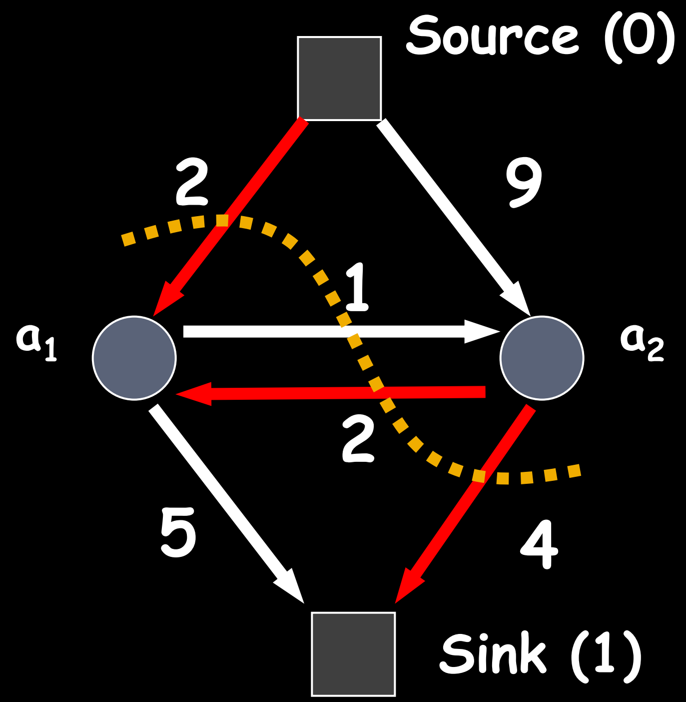
$a_1 = 1 \quad a_2 = 1$

$E(1,1) = 11$

Slide credit: Pushmeet Kohli

Graph Construction

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



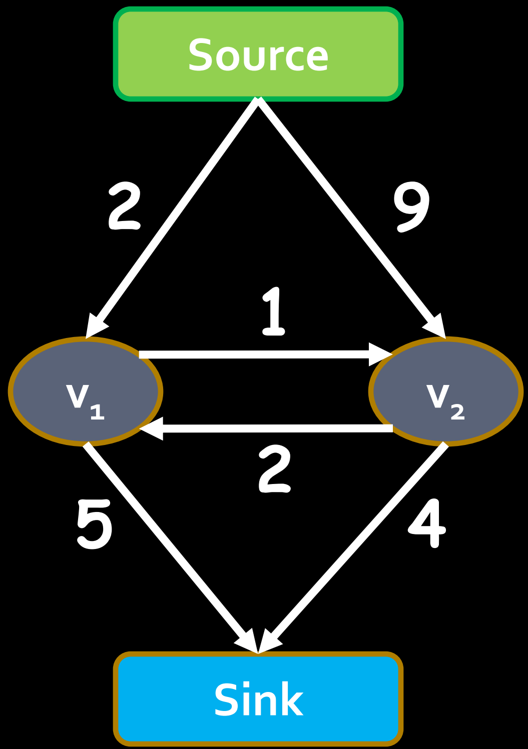
st-mincut cost = 8

$a_1 = 1 \quad a_2 = 0$

$E(1,0) = 8$

Slide credit: Pushmeet Kohli

How to compute the st-mincut?



Solve the dual **maximum flow** problem

Compute the maximum flow between Source and Sink s.t.

Edges: Flow < Capacity

Nodes: Flow in = Flow out

Min-cut\Max-flow Theorem

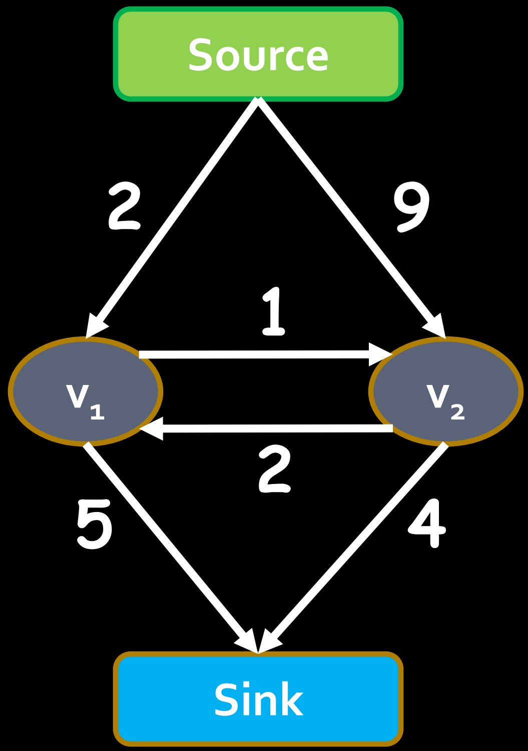
In every network, the maximum flow equals the cost of the st-mincut

Assuming non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

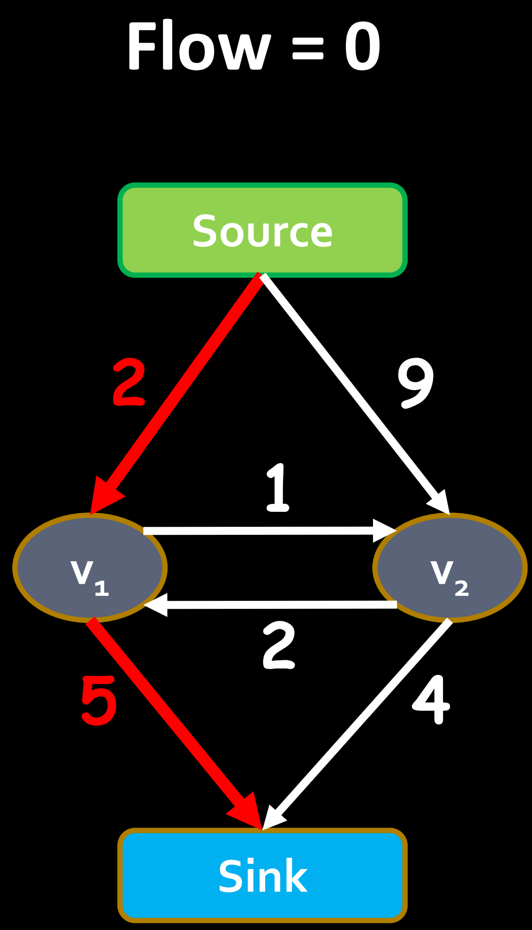
Flow = 0



Augmenting Path Based Algorithms

Slide credit: Pushmeet Kohli

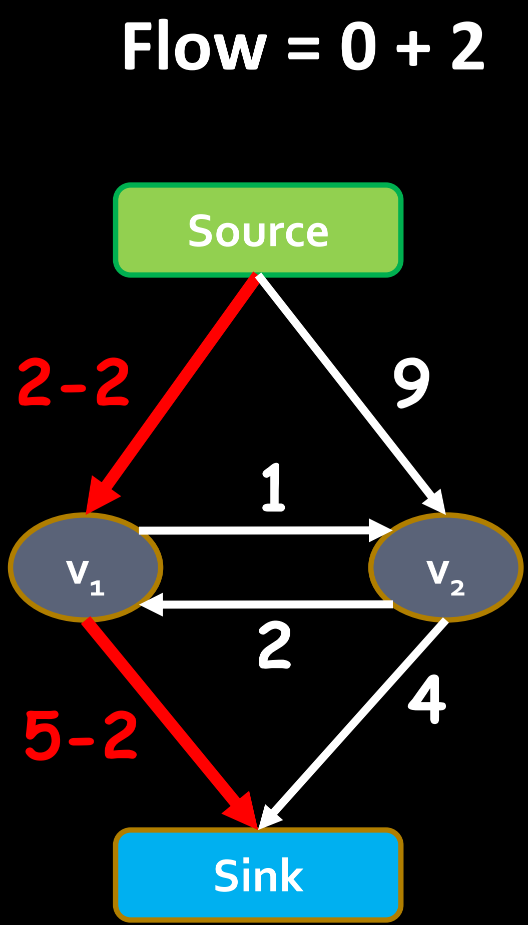
Maxflow Algorithms



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity

Maxflow Algorithms



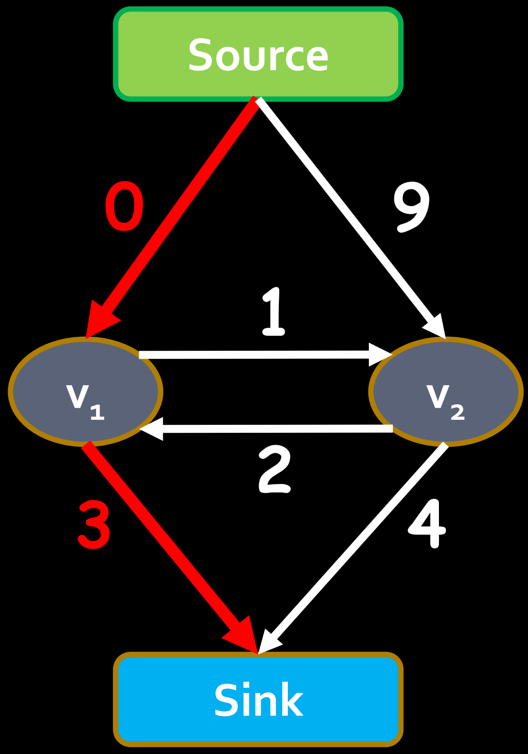
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



Augmenting Path Based Algorithms

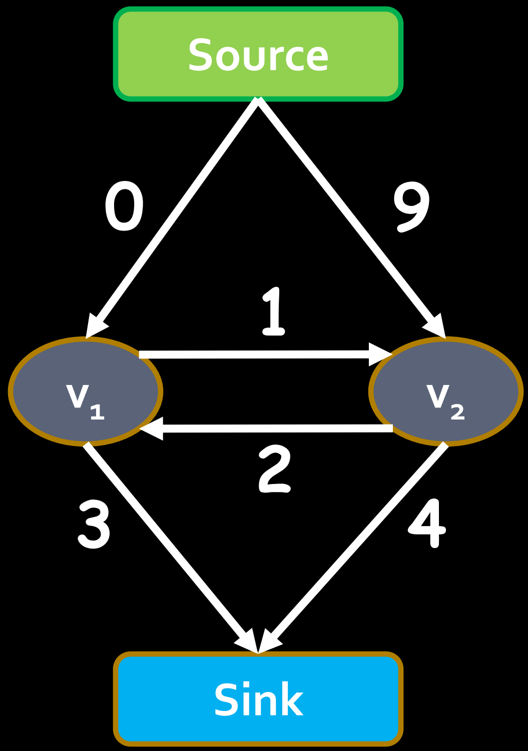
1. Find path from source to sink with positive capacity

2. Push maximum possible flow through this path

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



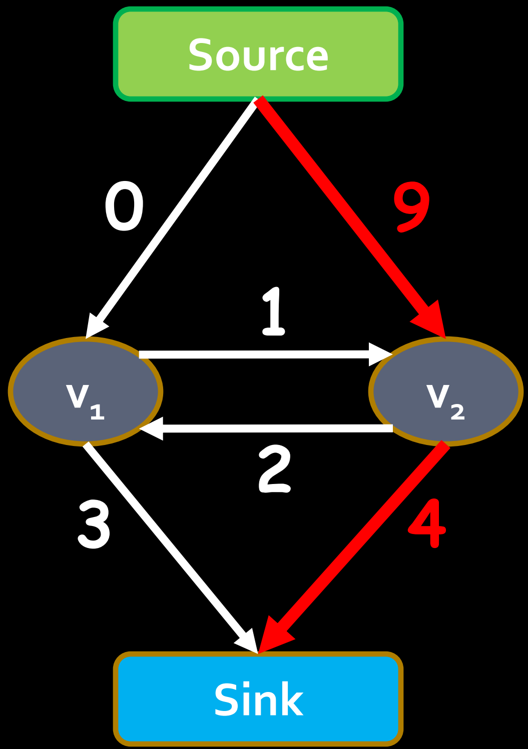
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity

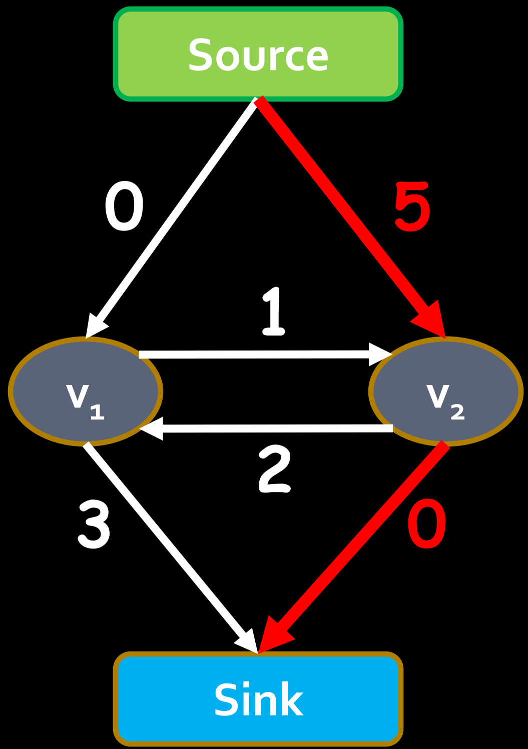
2. Push maximum possible flow through this path

3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2 + 4



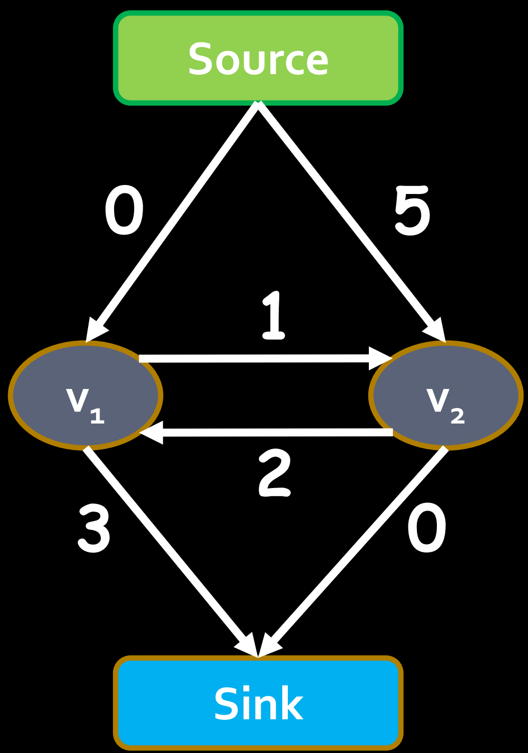
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6



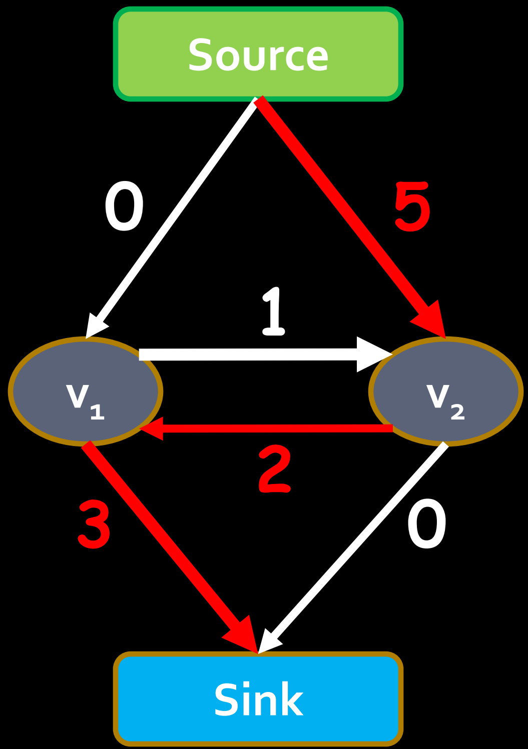
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6



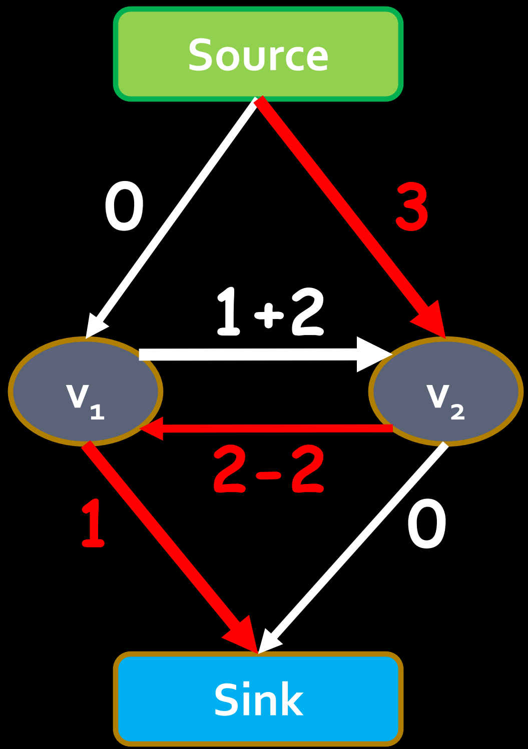
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6 + 2

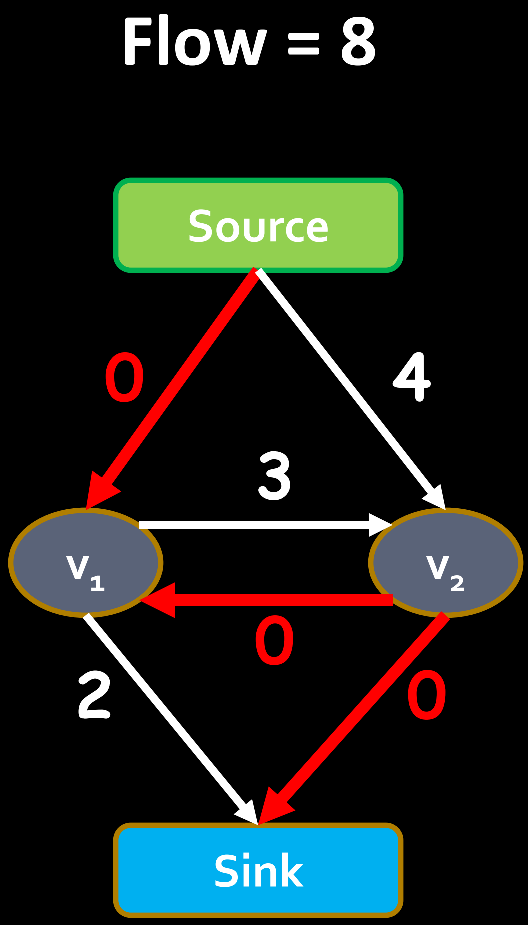


Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms

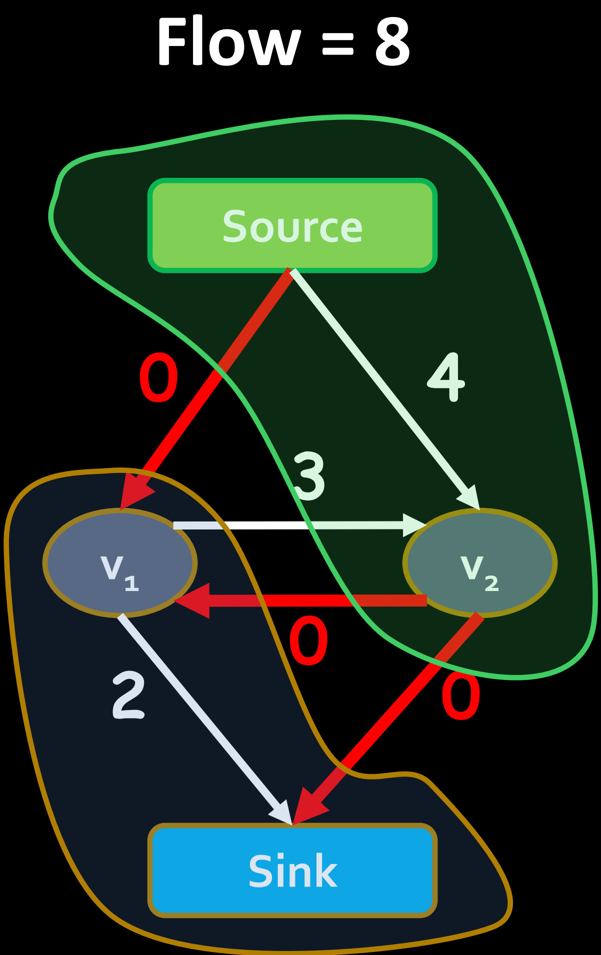


Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Maxflow Algorithms



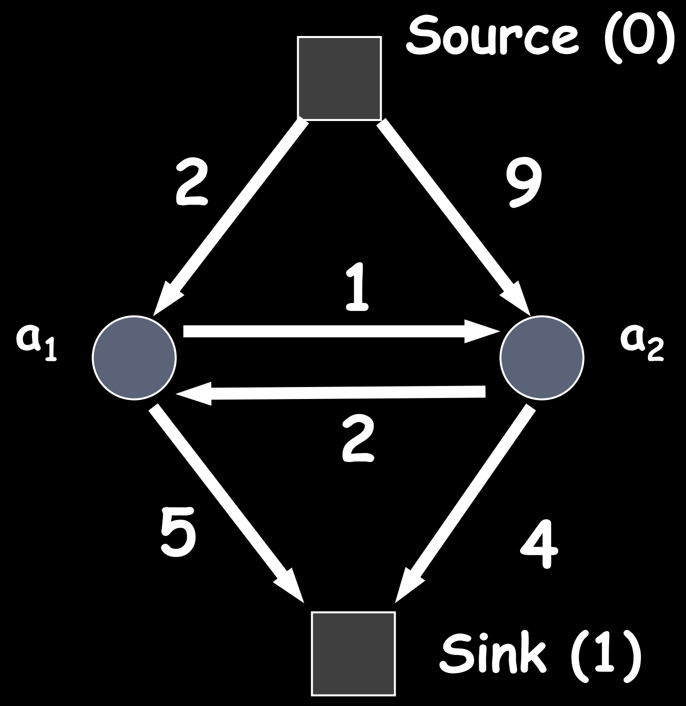
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Slide credit: Pushmeet Kohli

Flow and Reparametrization

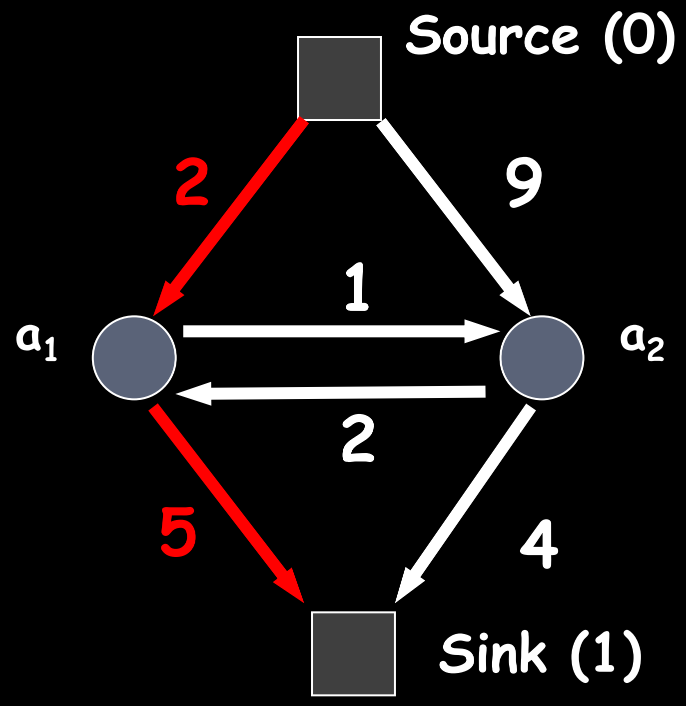
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



Slide credit: Pushmeet Kohli

Flow and Reparametrization

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

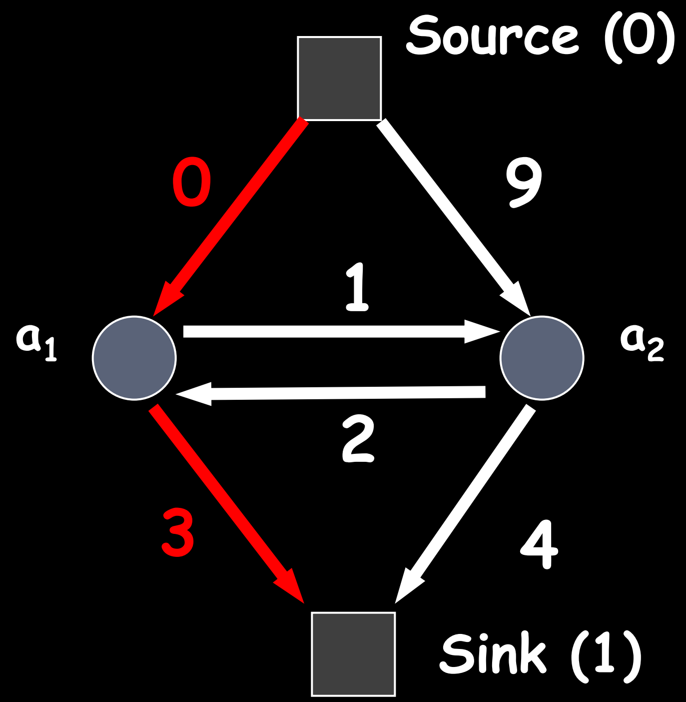


$$2a_1 + 5\bar{a}_1$$
$$= 2(a_1+\bar{a}_1) + 3\bar{a}_1$$
$$= 2 + 3\bar{a}_1$$

Slide credit: Pushmeet Kohli

Flow and Reparametrization

$E(a_1,a_2) = 2 + 3\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$



$2a_1 + 5\bar{a}_1$

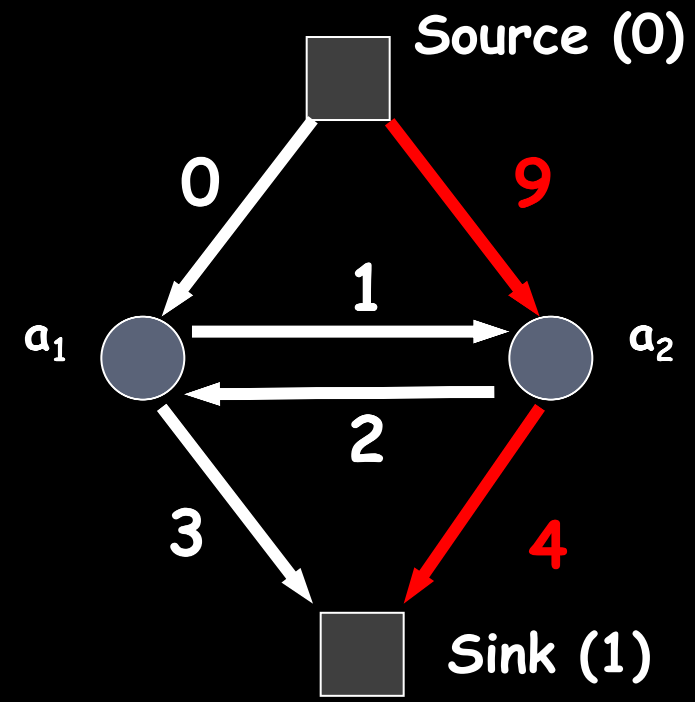
$= 2(a_1+\bar{a}_1) + 3\bar{a}_1$

$= 2 + 3\bar{a}_1$

Slide credit: Pushmeet Kohli

Flow and Reparametrization

$$E(a_1,a_2) = 2 + 3\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

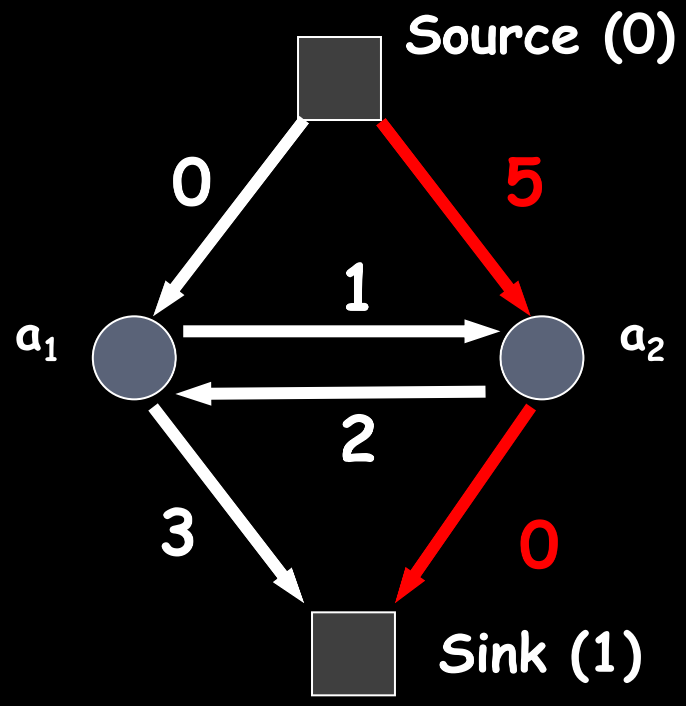


$$9a_2 + 4\bar{a}_2$$
$$= 4(a_2+\bar{a}_2) + 5\bar{a}_2$$
$$= 4 + 5\bar{a}_2$$

Slide credit: Pushmeet Kohli

Flow and Reparametrization

$$E(a_1,a_2) = 2 + 3\bar{a}_1 + 5a_2 + 4 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

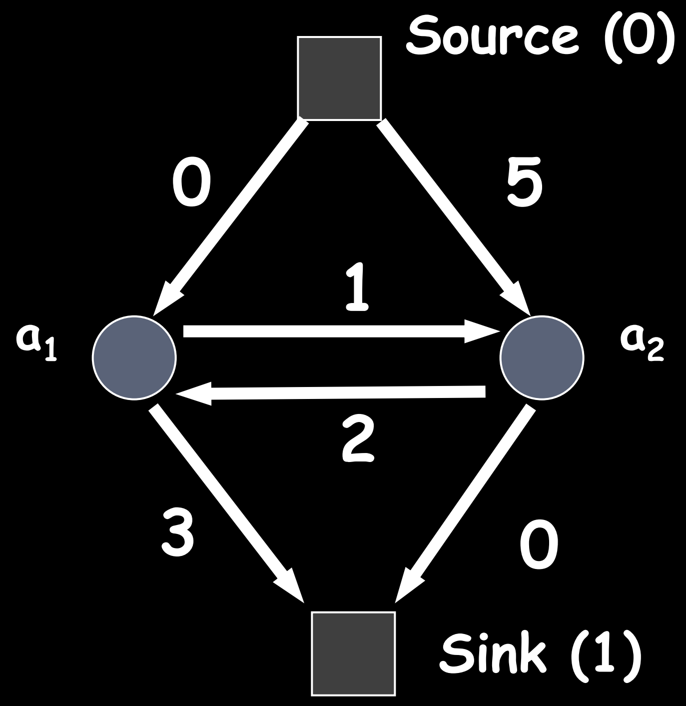


$$9a_2 + 4\bar{a}_2$$
$$= 4(a_2+\bar{a}_2) + 5\bar{a}_2$$
$$= 4 + 5\bar{a}_2$$

Slide credit: Pushmeet Kohli

Flow and Reparametrization

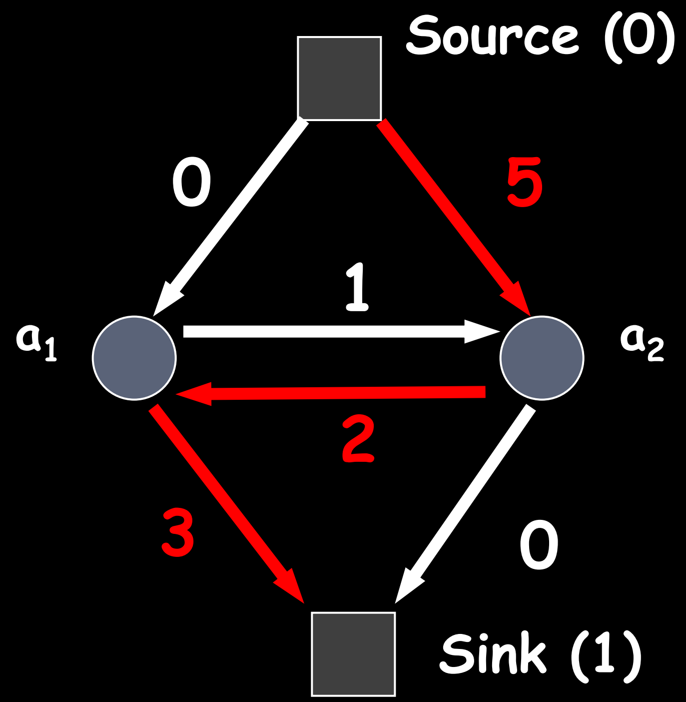
$E(a_1,a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$



Slide credit: Pushmeet Kohli

Flow and Reparametrization

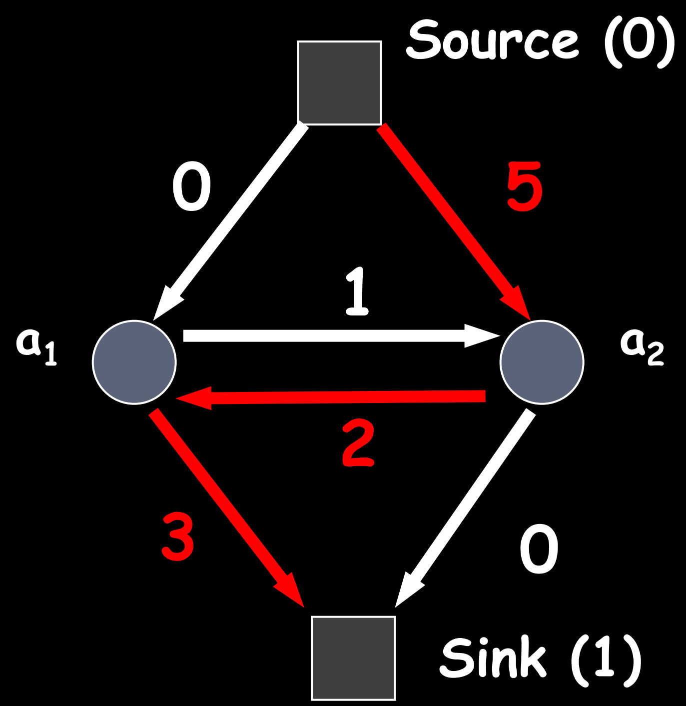
$E(a_1,a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$



Slide credit: Pushmeet Kohli

Flow and Reparametrization

$$E(a_1,a_2) = 6 + 3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$$



$$3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2$$
$$= 2(\bar{a}_1 + a_2 + a_1\bar{a}_2) + \bar{a}_1 + 3a_2$$
$$= 2(1 + \bar{a}_1a_2) + \bar{a}_1 + 3a_2$$

$F1 = \bar{a}_1 + a_2 + a_1\bar{a}_2$

$F2 = 1 + \bar{a}_1a_2$

a_1	a_2	F1	F2
0	0	1	1
0	1	2	2
1	0	1	1
1	1	1	1

Slide credit: Pushmeet Kohli

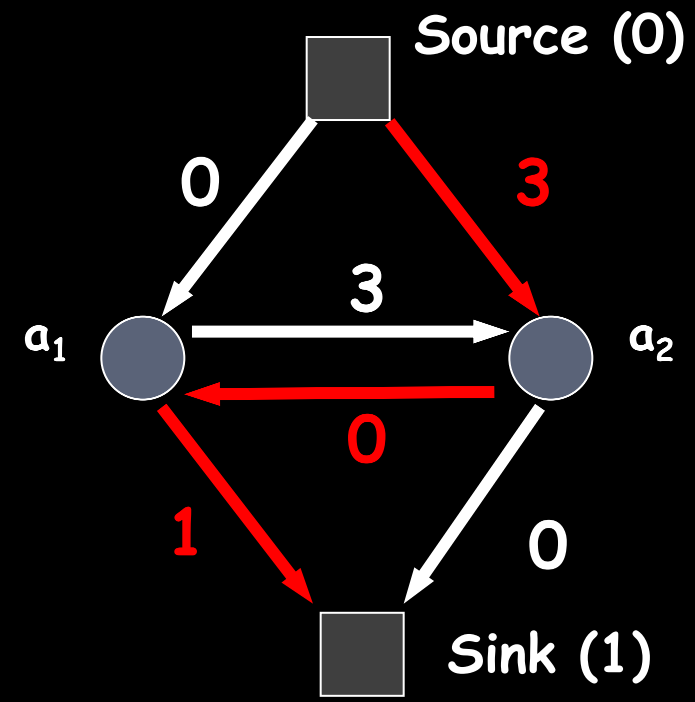
Flow and Reparametrization

$E(a_1,a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1 a_2$

$3\bar{a}_1 + 5a_2 + 2a_1\bar{a}_2$

$= 2(\bar{a}_1 + a_2 + a_1\bar{a}_2) + \bar{a}_1 + 3a_2$

$= 2(1 + \bar{a}_1 a_2) + \bar{a}_1 + 3a_2$



$F1 = \bar{a}_1 + a_2 + a_1\bar{a}_2$

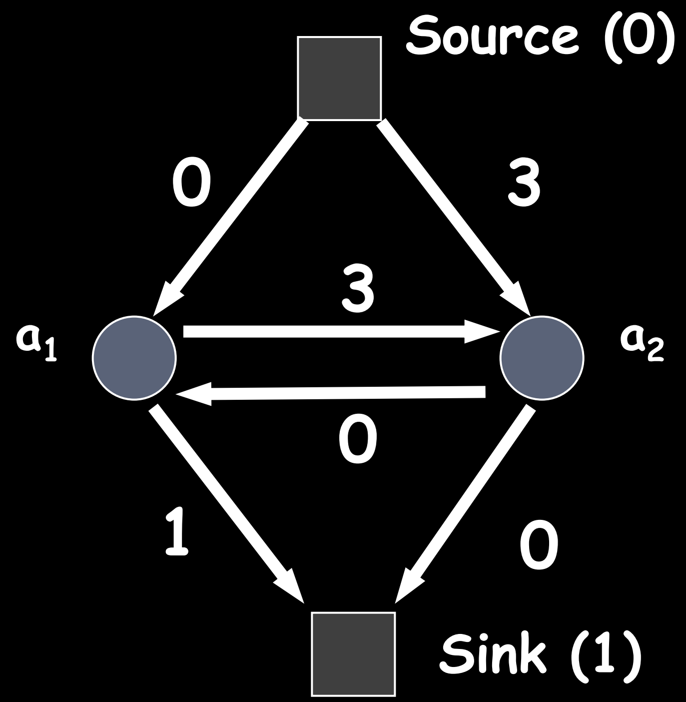
$F2 = 1 + \bar{a}_1 a_2$

a_1	a_2	F1	F2
0	0	1	1
0	1	2	2
1	0	1	1
1	1	1	1

Slide credit: Pushmeet Kohli

Flow and Reparametrization

$E(a_1,a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1 a_2$



No more augmenting paths possible

Slide credit: Pushmeet Kohli

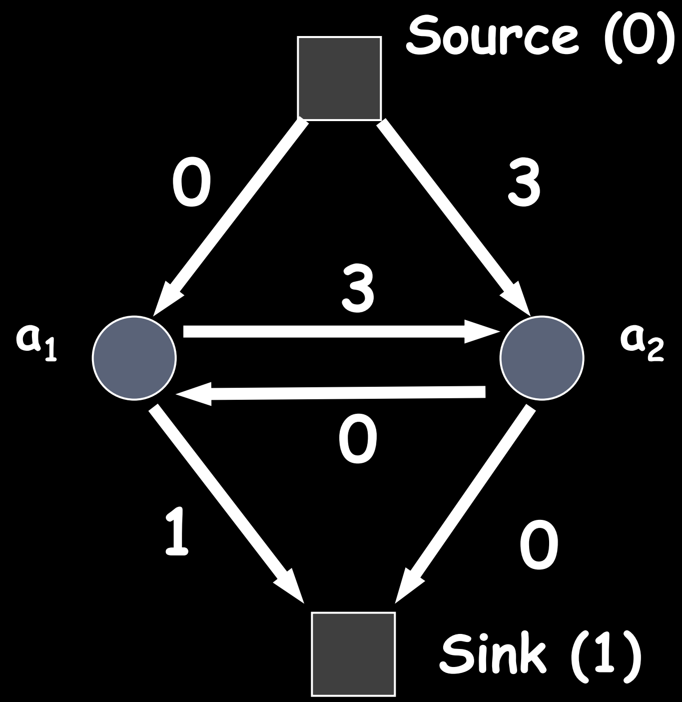
Flow and Reparametrization

$$E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1 a_2$$

Residual Graph (positive coefficients)

Total Flow

bound on the optimal solution



Tight Bound --> Inference of the optimal solution becomes trivial

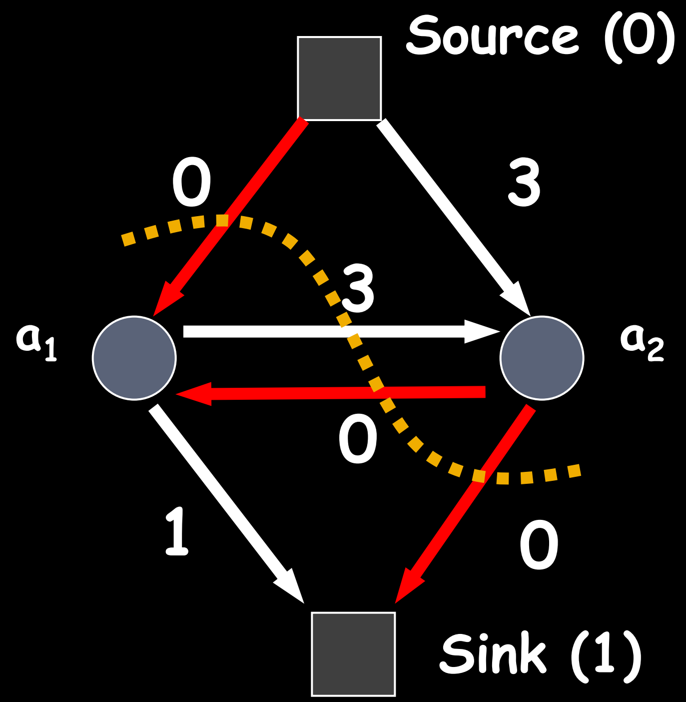
Slide credit: Pushmeet Kohli

Flow and Reparametrization

$E(a_1, a_2) = 8 + \bar{a}_1 + 3a_2 + 3\bar{a}_1 a_2$ → Residual Graph (positive coefficients)

Total Flow

bound on the optimal solution



st-mincut cost = 8

$a_1 = 1 \quad a_2 = 0$

$E(1,0) = 8$

Tight Bound --> Inference of the optimal solution becomes trivial

Slide credit: Pushmeet Kohli

History of Maxflow Algorithms

Augmenting Path and Push-Relabel

year	discoverer(s)	bound
1951	Dantzig	$O(n^2mU)$
1955	Ford & Fulkerson	$O(m^2U)$
1970	Dinitz	$O(n^2m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2m^{1/2})$
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: #edges

U: maximum edge weight

Algorithms assume non-negative edge weights

Slide credit: Pushmeet Kohli

[Slide credit: Andrew Goldberg]

History of Maxflow Algorithms

Augmenting Path and Push-Relabel

year	discoverer(s)	bound
1951	Dantzig	$O(n^2mU)$
1955	Ford & Fulkerson	$O(m^2U)$
1970	Dinitz	$O(n^2m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2m^{1/2})$
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: #edges

U: maximum edge weight

Algorithms assume non-negative edge weights

Slide credit: Pushmeet Kohli

[Slide credit: Andrew Goldberg]

References

- Visit http://www.inf.u-szeged.hu/~kato/
- Additional slides adopted from:
  - **Yuri Boykov**: *Computing geodesics and minimal surfaces via graph cuts* (ICCV 2003) http://www.csd.uwo.ca/~yuri/Presentations/iccv03.ppt
  - **Pushmeet Kohli**: *MAP Inference in Discrete Models* (ICCV 2009 tutorial) http://research.microsoft.com/en-us/um/cambridge/projects/tutorial/