Markov Random Fields for Computer Vision (Part 1)

Machine Learning Summer School (MLSS 2011)

Stephen Gould
stephen.gould@anu.edu.au

Australian National University

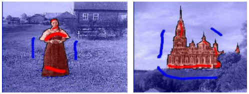
13–17 June, 2011

Pixel Labeling

Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.

Pixel Labeling

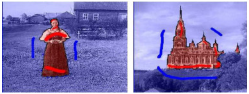
Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.



Interactive figure-ground segmentation (Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006)

Pixel Labeling

Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.

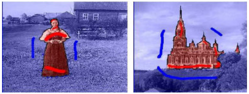


Interactive figure-ground segmentation (Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006)

Surface context (Hoiem et al., 2005)

Pixel Labeling

Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.



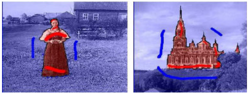
Interactive figure-ground segmentation (Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006)

Surface context (Hoiem et al., 2005)

Semantic labeling (He et al., 2004; Shotton et al., 2006; Gould et al., 2009)

Pixel Labeling

Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.



Interactive figure-ground segmentation (Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006)

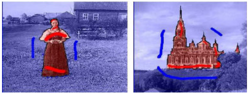
Surface context (Hoiem et al., 2005)

Semantic labeling (He et al., 2004; Shotton et al., 2006; Gould et al., 2009)

Stereo matching (Scharstein and Szeliski, 2002)

Pixel Labeling

Label every pixel in an image with a class label from some pre-defined set, i.e., $y_p \in \mathcal{L}$.



Interactive figure-ground segmentation (Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006)

Surface context (Hoiem et al., 2005)

Semantic labeling (He et al., 2004; Shotton et al., 2006; Gould et al., 2009)

Stereo matching (Scharstein and Szeliski, 2002)

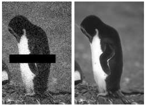


Image denoising (Felzenszwalb and Huttenlocher, 2004; Szeliski et al., 2008)

Digital Photo Montage

(Agarwala et al., 2004)

Probability Review

Bayes Rule

$$\underbrace{P(\mathbf{y} \mid \mathbf{x})}_{\text{posterior}} = \frac{\overbrace{P(\mathbf{x} \mid \mathbf{y})}^{\text{likelihood}} \cdot \overbrace{P(\mathbf{y})}^{\text{prior}}}{P(\mathbf{x})}$$

Maximum a Posteriori (MAP) inference: $\mathbf{y}^\star = \operatorname{argmax}_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x})$.

Probability Review

Bayes Rule

$$\underbrace{P(\mathbf{y} \mid \mathbf{x})}_{\text{posterior}} = \frac{\overbrace{P(\mathbf{x} \mid \mathbf{y})}^{\text{likelihood}} \cdot \overbrace{P(\mathbf{y})}^{\text{prior}}}{P(\mathbf{x})}$$

Maximum a Posteriori (MAP) inference: $\mathbf{y}^\star = \operatorname{argmax}_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x})$.

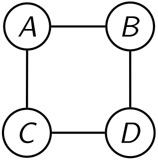
Conditional Independence

Random variables $\mathbf{y}$ and $\mathbf{x}$ are *conditionally independent* given $\mathbf{z}$ if $P(\mathbf{y}, \mathbf{x} \mid \mathbf{z}) = P(\mathbf{y} \mid \mathbf{z}) P(\mathbf{x} \mid \mathbf{z})$.

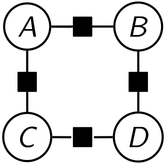
Graphical Models

We can exploit conditional independence assumptions to represent probability distributions in a way that is both *compact* and *efficient* for inference.

This tutorial is all about one particular representation, called a Markov Random Field (MRF), and the associated inference algorithms that are used in computer vision.

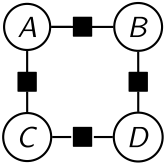


$$a \perp\!\!\!\perp d \mid b, c$$



$$\frac{1}{Z}\Psi(a,b)\Psi(b,d)\Psi(d,c)\Psi(c,a)$$

Graphical Models



$$\mathrm{P}(a, b, c, d) = \frac{1}{Z}\Psi(a, b)\Psi(b, d)\Psi(d, c)\Psi(c, a)$$
$$= \frac{1}{Z}\exp\{-\psi(a, b) - \psi(b, d) - \psi(d, c) - \psi(c, a)\}$$

where $\psi = -\log \Psi$.

Energy Functions

Let $\mathbf{x}$ be some observations (i.e., features from the image) and let $\mathbf{y} = (y_1, \ldots, y_n)$ be a vector of random variables. Then we can write the conditional probability of $\mathbf{y}$ given $\mathbf{x}$ as

$$\mathrm{P}(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\{-E(\mathbf{y}; \mathbf{x})\}$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{L}^n} \exp\{-E(\mathbf{y}; \mathbf{x})\}$ is called the *partition function*.

Energy Functions

Let $\mathbf{x}$ be some observations (i.e., features from the image) and let $\mathbf{y} = (y_1, \ldots, y_n)$ be a vector of random variables. Then we can write the conditional probability of $\mathbf{y}$ given $\mathbf{x}$ as

$$\mathrm{P}\left(\mathbf{y} \mid \mathbf{x}\right) = \frac{1}{Z(\mathbf{x})} \exp\left\{-E\left(\mathbf{y}; \mathbf{x}\right)\right\}$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{L}^n} \exp\left\{-E\left(\mathbf{y}; \mathbf{x}\right)\right\}$ is called the *partition function*.

The *energy function* $E\left(\mathbf{y}; \mathbf{x}\right)$ usually has some structured form:

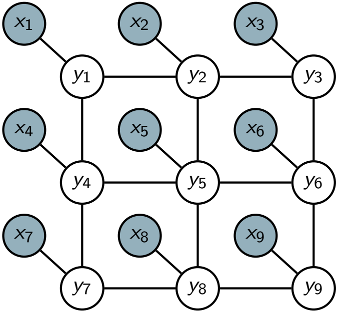
$$E\left(\mathbf{y}; \mathbf{x}\right) = \sum_c \psi_c(\mathbf{y}_c; \mathbf{x})$$

where $\psi_c(\mathbf{y}_c; \mathbf{x})$ are *clique potentials* defined over a subset of random variables $\mathbf{y}_c \subseteq \mathbf{y}$.

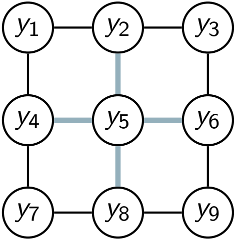
Conditional Markov Random Fields

$$E(\mathbf{y};\mathbf{x}) = \sum_c \psi_c(\mathbf{y}_c;\mathbf{x})$$

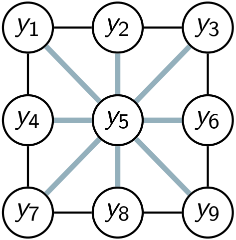
$$= \underbrace{\sum_{i\in\mathcal{V}} \psi_i^U(y_i;\mathbf{x})}_{\text{unary}} + \underbrace{\sum_{ij\in\mathcal{E}} \psi_{ij}^P(y_i, y_j;\mathbf{x})}_{\text{pairwise}} + \underbrace{\sum_{c\in\mathcal{C}} \psi_c^H(\mathbf{y}_c;\mathbf{x})}_{\text{higher-order}}.$$



Pixel Neighbourhoods



4-connected, $\mathcal{N}_4$



8-connected, $\mathcal{N}_8$

Binary MRF Example

Consider the following energy function for two binary random variables, y_1 and y_2.

0	5
1	2

0	1
1	3

	0	1
0	0	3
1	4	0

$$E(y_1, y_2) = \psi_1(y_1) + \psi_2(y_2) + \psi_{12}(y_1, y_2)$$

Binary MRF Example

Consider the following energy function for two binary random variables, y_1 and y_2.

	ψ_1
0	5
1	2

	ψ_2
0	1
1	3

	0	1
0	0	3
1	4	0

$$
\begin{aligned}
E\left(y_1, y_2\right) &= \psi_1(y_1) + \psi_2(y_2) + \psi_{12}(y_1, y_2) \\
&= \underbrace{5\bar{y}_1 + 2y_1}_{\psi_1} \\
&\quad + \underbrace{\bar{y}_2 + 3y_2}_{\psi_2} \\
&\quad + \underbrace{3\bar{y}_1 y_2 + 4y_1\bar{y}_2}_{\psi_{12}}
\end{aligned}
$$

where $\bar{y}_1 = 1 - y_1$ and $\bar{y}_2 = 1 - y_2$.

Binary MRF Example

Consider the following energy function for two binary random variables, y_1 and y_2.

0	5
1	2

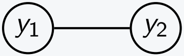
0	1
1	3

	0	1
0	0	3
1	4	0

$$E(y_1, y_2) = \psi_1(y_1) + \psi_2(y_2) + \psi_{12}(y_1, y_2)$$
$$= \underbrace{5\bar{y}_1 + 2y_1}_{\psi_1} + \underbrace{\bar{y}_2 + 3y_2}_{\psi_2} + \underbrace{3\bar{y}_1 y_2 + 4y_1\bar{y}_2}_{\psi_{12}}$$

where $\bar{y}_1 = 1 - y_1$ and $\bar{y}_2 = 1 - y_2$.

Graphical Model



Probability Table

y_1	y_2	E	P
0	0	6	0.244
0	1	11	0.002
1	0	7	0.090
1	1	5	0.664

Compactness of Representation

Consider a 1 mega-pixel image, e.g., 1000×1000 pixels. We want to annotate each pixel with a label from $\mathcal{L}$. Let $L = |\mathcal{L}|$.

- There are L^{10^6} possible ways to label such an image.
- A naive encoding—i.e., one big table—would require $L^{10^6} - 1$ parameters.
- A pairwise MRF over $\mathcal{N}_4$ requires $10^6 L$ parameters for the unary terms and $2 \times 1000 \times (1000 - 1)L^2$ parameters for the pairwise terms, i.e., $O(10^6 L^2)$. Even less are required if we share parameters.

Inference and Energy Minimization

We are usually interested in finding the most probable labeling,

$$\mathbf{y}^\star = \operatorname*{argmax}_{\mathbf{y}} \mathrm{P}\left(\mathbf{y} \mid \mathbf{x}\right) = \operatorname*{argmin}_{\mathbf{y}} E\left(\mathbf{y}; \mathbf{x}\right).$$

This is known as *maximum a posteriori* (MAP) inference or *energy minimization*.

Inference and Energy Minimization

We are usually interested in finding the most probable labeling,

$$\mathbf{y}^\star = \operatorname*{argmax}_{\mathbf{y}} \mathrm{P}\left(\mathbf{y} \mid \mathbf{x}\right) = \operatorname*{argmin}_{\mathbf{y}} E\left(\mathbf{y}; \mathbf{x}\right).$$

This is known as *maximum a posteriori* (MAP) inference or *energy minimization*.

A number of techniques can be used to find $\mathbf{y}^\star$, including:

- message-passing (dynamic programming)
- integer programming (part 3)
- graph-cuts (part 2)

However, in general, inference is NP-hard.

Characterizing Markov Random Fields

Markov random fields can be categorized via a number of different dimensions:

- **Label space:** binary vs. multi-label; homogeneous vs. heterogeneous.
- **Order:** unary vs. pairwise vs. higher-order.
- **Structure:** chain vs. tree vs. grid vs. general graph; neighbourhood size.
- **Potentials:** submodular, convex, compressible.

These all affect tractability of inference.

Markov Random Fields for Pixel Labeling

$$\mathrm{P}(\mathbf{y} \mid \mathbf{x}) \propto \mathrm{P}(\mathbf{x} \mid \mathbf{y})\, \mathrm{P}(\mathbf{y}) = \exp\{-E(\mathbf{y}; \mathbf{x})\}$$

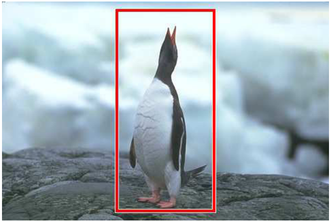
$$\overbrace{E(\mathbf{y}; \mathbf{x})}^{\text{energy}} = \underbrace{\sum_{i \in \mathcal{V}} \psi_i^U(y_i; \mathbf{x})}_{\text{unary}} + \underbrace{\lambda \sum_{ij \in \mathcal{N}_8} \psi_{ij}^P(y_i, y_j; \mathbf{x})}_{\text{pairwise}}$$

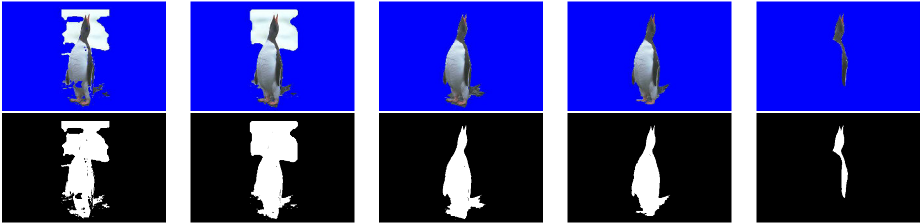
$$\psi_i^U(y_i; \mathbf{x}) = \overbrace{-\sum_{\ell \in \mathcal{L}} [\![y_i = \ell]\!] \log \mathrm{P}(x_i \mid \ell)}^{\text{likelihood}}$$

$$\psi_{ij}^P(y_i, y_j; \mathbf{x}) = \underbrace{[\![y_i \neq y_j]\!]}_{\text{Potts prior}}$$

Here the prior acts to "smooth" predictions (independent of $\mathbf{x}$).

Prior Strength



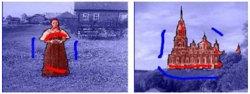


$\lambda = 1$ $\lambda = 4$ $\lambda = 16$ $\lambda = 128$ $\lambda = 1024$

Interactive Segmentation Model

- **Label space:** foreground or background

$$\mathcal{L} = \{0, 1\}$$



- **Unary term:** Gaussian mixture models for foreground and background

$$\psi_i^U(y_i; \mathbf{x}) = \sum_k \tfrac{1}{2}|\Sigma_k| + \tfrac{1}{2}\left(x_i - \mu_k\right)^T \Sigma_k^{-1}\left(x_i - \mu_k\right) - \log \lambda_k$$

- **Pairwise term:** contrast-dependent smoothness prior

$$\psi_{ij}^P(y_i, y_j; \mathbf{x}) = \begin{cases} \lambda_0 + \lambda_1 \exp\left(-\frac{\|x_i - x_j\|^2}{2\beta}\right), & \text{if } y_i \neq y_j \\ 0, & \text{otherwise} \end{cases}$$

Geometric/Semantic Labeling Model

- **Label space:** pre-defined label set, e.g.,

$$\mathcal{L} = \{\text{sky}, \text{tree}, \text{grass}, \ldots\}$$

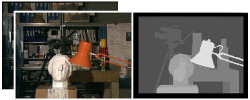
- **Unary term:** Boosted decision-tree classifiers over "texton-layout" features [Shotton et al., 2006]

$$\psi_i^U(y_i = \ell; \mathbf{x}) = \theta_\ell \log \mathrm{P}\left(\phi_i(\mathbf{x}) \mid \ell\right)$$

- **Pairwise term:** contrast-dependent smoothness prior

$$\psi_{ij}^P(y_i, y_j; \mathbf{x}) = \begin{cases} \lambda_0 + \lambda_1 \exp\left(-\frac{\|x_i - x_j\|^2}{2\beta}\right), & \text{if } y_i \neq y_j \\ 0, & \text{otherwise} \end{cases}$$

Stereo Matching Model



- **Label space:** pixel disparity

$$\mathcal{L} = \{0, 1, \ldots, 127\}$$

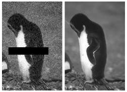
- **Unary term:** sum of absolute differences (SAD) or normalized cross-correlation (NCC)

$$\psi_i^U(y_i; \mathbf{x}) = \sum_{(u,v)\in W} |\mathbf{x}_{\text{left}}(u,v) - \mathbf{x}_{\text{right}}(u - y_i, v)|$$

- **Pairwise term:** "discontinuity preserving" prior

$$\psi_{ij}^P(y_i, y_j) = \max\{|y_i - y_j|, d_{\max}\}$$

Image Denoising Model



- **Label space:** pixel intensity or colour

$$\mathcal{L} = \{0, 1, \ldots, 255\}$$

- **Unary term:** square distance

$$\psi_i^U(y_i; \mathbf{x}) = \|y_i - x_i\|^2$$

- **Pairwise term:** truncated L_2 distance

$$\psi_{ij}^P(y_i, y_j) = \max\left\{\|y_i - y_j\|^2, d_{\max}^2\right\}$$

Digital Photo Montage Model

- **Label space:** image index

$$\mathcal{L} = \{1, 2, \ldots, K\}$$

- **Unary term:** none!
- **Pairwise term:** seem penalty

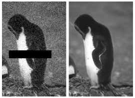
$$\psi^P_{ij}(y_i, y_j; \mathbf{x}) = \|\mathbf{x}_{y_i}(i) - \mathbf{x}_{y_j}(i)\| + \|\mathbf{x}_{y_i}(j) - \mathbf{x}_{y_j}(j)\|$$

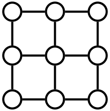
(or edge-normalized variant)

Outline of Energy Minimization via Graph-cuts

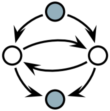
Big picture:

- Start with a pixel labeling problem
- Formulate as a (multilabel) graphical model inference problem
- Convert to a series of binary pairwise MRF inference problems
- Write MRF as a quadratic pseudo-Boolean function
- Convert pseudo-Boolean minimization to min-cut problem
- Equivalently, formulate as a max-flow problem
- Solve using augmented-path algorithm





$$\{0,1\}^n \to \mathbb{R}$$
$$\dots$$
$$\{0,1\}^n \to \mathbb{R}$$



A Note About Graphs

point of confusion:

graphs are used to represent many different things

In this talk we use graphs to...

- represent probabilistic models (or energy functions), e.g., Markov random fields and factor graphs.
- represent optimization problems, e.g., psuedo-Boolean function minimization.

Pseudo-boolean Functions [Boros and Hammer, 2001]

Pseudo-boolean Function

A mapping $f : \{0,1\}^n \rightarrow \mathbb{R}$ is called a *pseudo-Boolean function*.

Pseudo-boolean Functions [Boros and Hammer, 2001]

Pseudo-boolean Function

A mapping $f : \{0,1\}^n \to \mathbb{R}$ is called a *pseudo-Boolean function*.

- Pseudo-boolean functions can be uniquely represented as *multi-linear polynomials*, e.g., $f(y_1, y_2) = 6 + y_1 + 5y_2 - 7y_1y_2$.

Pseudo-boolean Functions [Boros and Hammer, 2001]

Pseudo-boolean Function

A mapping $f : \{0,1\}^n \to \mathbb{R}$ is called a *pseudo-Boolean function*.

- Pseudo-boolean functions can be uniquely represented as *multi-linear polynomials*, e.g., $f(y_1, y_2) = 6 + y_1 + 5y_2 - 7y_1y_2$.
- Pseudo-boolean functions can also be represented in *posiform*, e.g., $f(y_1, y_2) = 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1y_2 + 4y_1\bar{y}_2$. This representation is not unique.

Pseudo-boolean Functions [Boros and Hammer, 2001]

Pseudo-boolean Function

A mapping $f : \{0,1\}^n \to \mathbb{R}$ is called a *pseudo-Boolean function*.

- Pseudo-boolean functions can be uniquely represented as *multi-linear polynomials*, e.g., $f(y_1, y_2) = 6 + y_1 + 5y_2 - 7y_1y_2$.
- Pseudo-boolean functions can also be represented in *posiform*, e.g., $f(y_1, y_2) = 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1y_2 + 4y_1\bar{y}_2$. This representation is not unique.
- **A binary pairwise Markov random field (MRF) is just a quadratic pseudo-Boolean function.**

Representing a Binary Pairwise MRF

Consider a binary pairwise MRF over two variables:

	0	1
0	A	B
1	C	D

Representing a Binary Pairwise MRF

Consider a binary pairwise MRF over two variables:

	0	1
0	A	B
1	C	D

$$A + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline C-A & C-A \\ \hline\end{array} + \begin{array}{|c|c|}\hline 0 & D-C \\ \hline 0 & D-C \\ \hline\end{array} + \begin{array}{|c|c|}\hline 0 & B+C-A-D \\ \hline 0 & 0 \\ \hline\end{array}$$

$$E(y_1, y_2) = A + (C - A)y_1 + (D - C)y_2 + (B + C - A - D)\bar{y}_1 y_2$$

[Kolmogorov and Zabih, 2004]

Pseudo-boolean Optimization [Boros and Hammer, 2001]

A large number of classical combinatorial optimization problems can be formulated in terms of pseudo-boolean optimization, e.g.,

- **Maximum independent set problem:** find the largest set of verticies in a graph such that no two are adjacent.

$$\alpha(G) = \max_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{i \in \mathcal{V}} x_i - \sum_{(i,j) \in \mathcal{E}} x_i x_j \right)$$

- **Minimum vertex cover:** find the smallest set of verticies such that every edge in the graph is adjacent to at least one vertex in the set.

$$\tau(G) = \min_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{i \in \mathcal{V}} x_i + \sum_{(i,j) \in \mathcal{E}} \bar{x}_i \bar{x}_j \right)$$

- **Maximum satisfiability problem:** find an assignment to a set of variables that satisfy as many clauses as possible.

$$\max_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{C \in \mathcal{C}} \left(1 - \sum_{u \in C} \bar{u} \right) \right)$$

Pseudo-boolean Optimization [Boros and Hammer, 2001]

A large number of classical combinatorial optimization problems can be formulated in terms of pseudo-boolean optimization, e.g.,

- **Maximum independent set problem:** find the largest set of verticies in a graph such that no two are adjacent.

$$\alpha(G) = \max_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{i \in \mathcal{V}} x_i - \sum_{(i,j) \in \mathcal{E}} x_i x_j \right)$$

- **Minimum vertex cover:** find the smallest set of verticies such that every edge in the graph is adjacent to at least one vertex in the set.

$$\tau(G) = \min_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{i \in \mathcal{V}} x_i + \sum_{(i,j) \in \mathcal{E}} \bar{x}_i \bar{x}_j \right)$$

- **Maximum satisfiability problem:** find an assignment to a set of variables that satisfy as many clauses as possible.

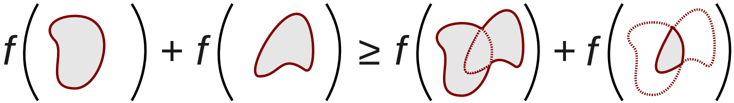
$$\max_{\mathbf{x} \in \{0,1\}^n} \left(\sum_{C \in \mathcal{C}} \left(1 - \sum_{u \in C} \bar{u} \right) \right)$$

These problems are all NP-hard.

Submodular Functions

Submodularity

Let $\mathcal{V}$ be a set. A set function $f : 2^{\mathcal{V}} \to \mathbb{R}$ is called *submodular* if $f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$ for all subsets $X, Y \subseteq \mathcal{V}$.



Submodular Binary Pairwise MRFs

Submodularity

A pseudo-Boolean function $f : \{0, 1\}^n \rightarrow \mathbb{R}$ is called *submodular* if $f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \vee \mathbf{y}) + f(\mathbf{x} \wedge \mathbf{y})$ for all vectors $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$.

Submodular Binary Pairwise MRFs

Submodularity

A pseudo-Boolean function $f : \{0,1\}^n \rightarrow \mathbb{R}$ is called *submodular* if $f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \vee \mathbf{y}) + f(\mathbf{x} \wedge \mathbf{y})$ for all vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$.

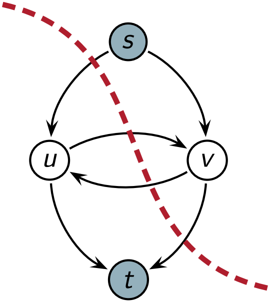
Submodularity checks for pairwise binary MRFs:

- polynomial form (of pseudo-boolean function) has negative coefficients on all bi-linear terms;
- posiform has pairwise terms of the form $u\bar{v}$;
- all pairwise potentials satisfy $\psi_{ij}^P(0,1) + \psi_{ij}^P(1,0) \geq \psi_{ij}^P(1,1) + \psi_{ij}^P(0,0)$.

Minimum-cut Problem

Graph Cut

Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ be a capacitated digraph with two distinguished vertices s and t. An st-cut is a partitioning of $\mathcal{V}$ into two disjoint sets $\mathcal{S}$ and $\mathcal{T}$ such that $s \in \mathcal{S}$ and $t \in \mathcal{T}$. The cost of the cut is the sum of edge capacities for all edges going from $\mathcal{S}$ to $\mathcal{T}$.



Quadratic Pseudo-boolean Optimization

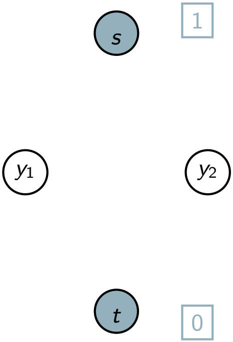
Main idea:

- construct a graph such that every *st*-cut corresponds to a joint assignment to the variables $\mathbf{y}$
- the cost of the cut should be equal to the energy of the assignment, $E(\mathbf{y}; \mathbf{x})$.*
- the minimum-cut then corresponds to the the minimum energy assignment, $\mathbf{y}^\star = \operatorname{argmin}_{\mathbf{y}} E(\mathbf{y}; \mathbf{x})$.

*Requires non-negative energies.

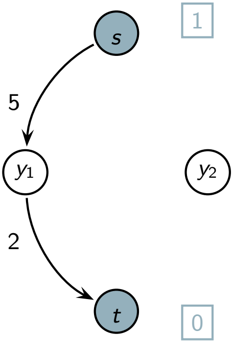
Example *st*-Graph Construction for Binary MRF

$$E(y_1, y_2) = \psi_1(y_1) + \psi_2(y_2) + \psi_{ij}(y_1, y_2)$$



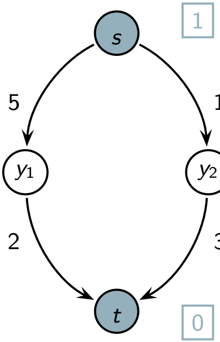
Example *st*-Graph Construction for Binary MRF

$$
\begin{aligned}
E(y_1, y_2) &= \psi_1(y_1) + \psi_2(y_2) + \psi_{ij}(y_1, y_2) \\
&= 2y_1 + 5\bar{y}_1
\end{aligned}
$$



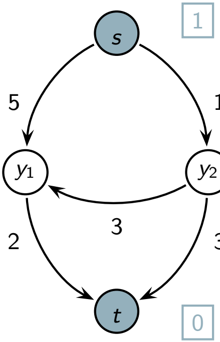
Example *st*-Graph Construction for Binary MRF

$$E(y_1, y_2) = \psi_1(y_1) + \psi_2(y_2) + \psi_{ij}(y_1, y_2)$$
$$= 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2$$



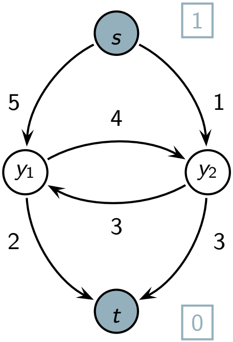
Example *st*-Graph Construction for Binary MRF

$$E(y_1, y_2) = \psi_1(y_1) + \psi_2(y_2) + \psi_{ij}(y_1, y_2)$$
$$= 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1 y_2$$



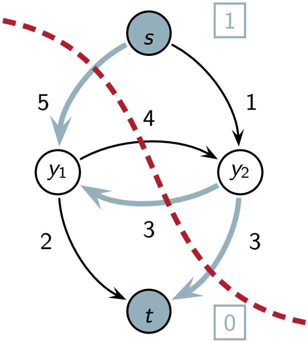
Example *st*-Graph Construction for Binary MRF

$$
\begin{aligned}
E(y_1, y_2) &= \psi_1(y_1) + \psi_2(y_2) + \psi_{ij}(y_1, y_2) \\
&= 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1 y_2 + 4y_1\bar{y}_2
\end{aligned}
$$



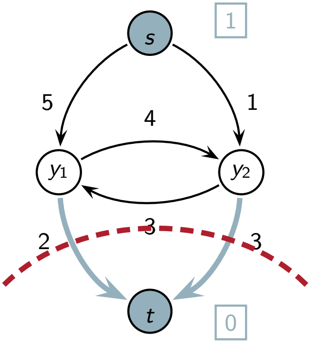
An Example *st*-Cut

$$E(0,1) = \psi_1(0) + \psi_2(1) + \psi_{ij}(0,1)$$
$$= 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1 y_2 + 4y_1\bar{y}_2$$



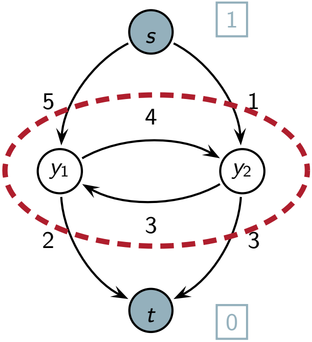
Another *st*-Cut

$$E(1,1) = \psi_1(1) + \psi_2(1) + \psi_{ij}(1,1)$$
$$= 2y_1 + 5\bar{y}_1 + 3y_2 + \bar{y}_2 + 3\bar{y}_1 y_2 + 4y_1\bar{y}_2$$



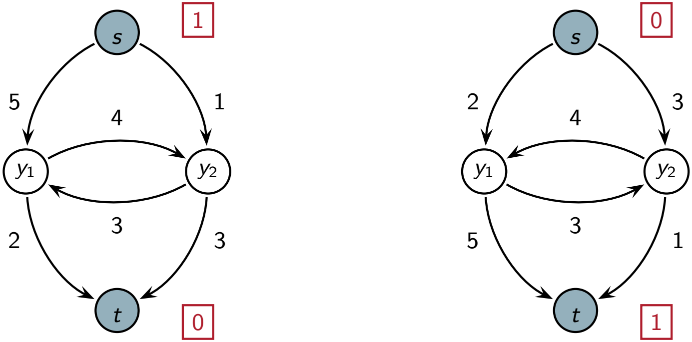
Invalid *st*-Cut

This is not a valid cut, since it does not correspond to a partitioning of the nodes into two sets—one containing s and one containing t.



Alternative *st*-Graph Construction

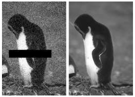
Sometimes you will see the roles of s and t switched.

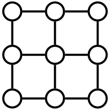


These graphs represent the same energy function.

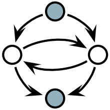
Big Picture: Where are we?

We can now formulate inference in a submodular binary pairwise MRF as a minimum-cut problem.





$\{0,1\}^n \rightarrow \mathbb{R}$

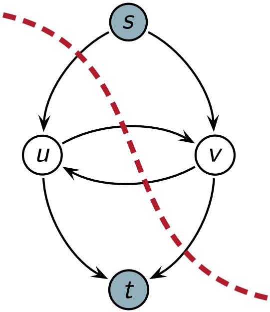


How do we solve the minimum-cut problem?

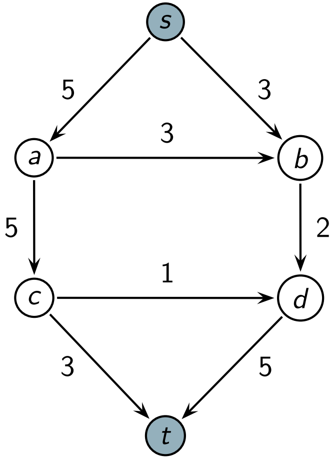
Max-flow/Min-cut Theorem

Max-flow/Min-cut Theorem [Fulkerson, 1956]

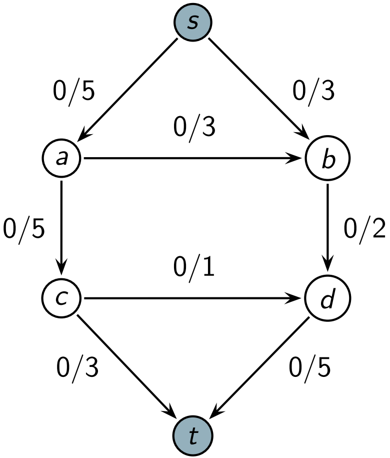
The maximum flow f from vertex s to vertex t is equal to the minimum cost st-cut.



Maximum Flow Example



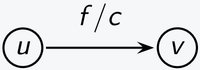
Maximum Flow Example (Augmenting Path)



flow

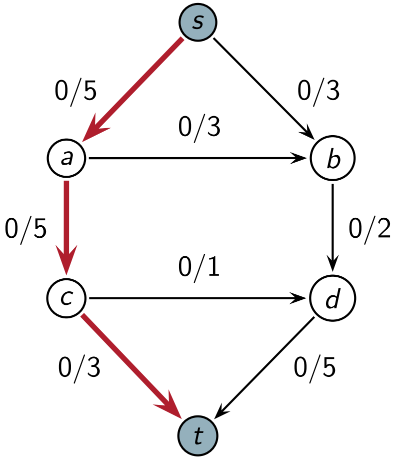
0

notation



edge with capacity c, and current flow f.

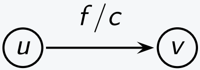
Maximum Flow Example (Augmenting Path)



flow

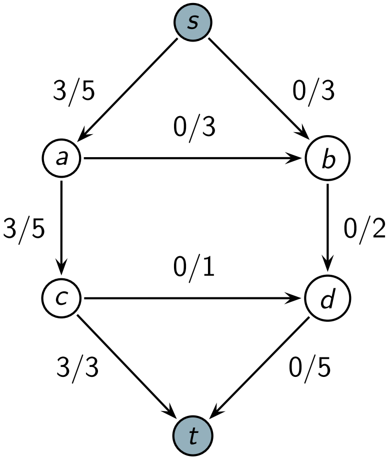
0

notation



edge with capacity c, and current flow f.

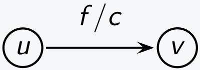
Maximum Flow Example (Augmenting Path)



flow

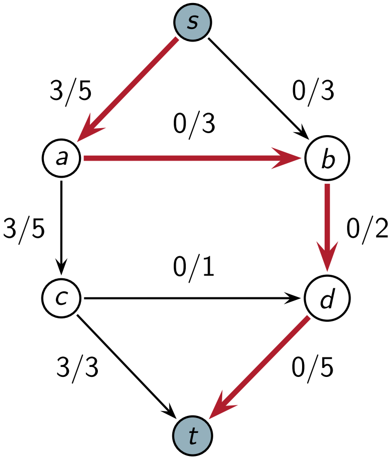
3

notation



edge with capacity c, and current flow f.

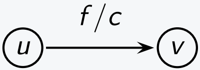
Maximum Flow Example (Augmenting Path)



flow

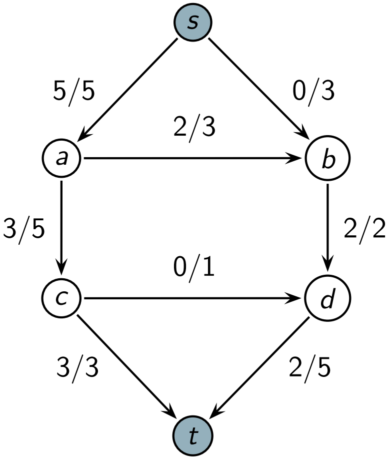
3

notation



edge with capacity c, and current flow f.

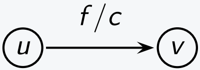
Maximum Flow Example (Augmenting Path)



flow

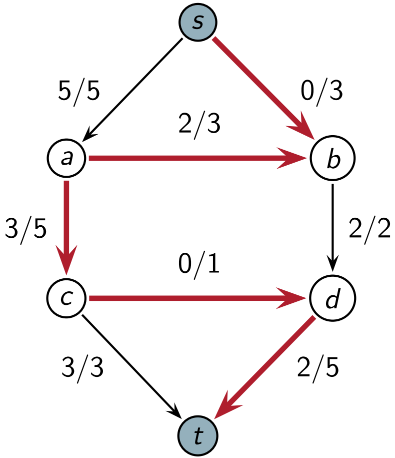
5

notation



edge with capacity c, and current flow f.

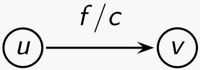
Maximum Flow Example (Augmenting Path)



flow

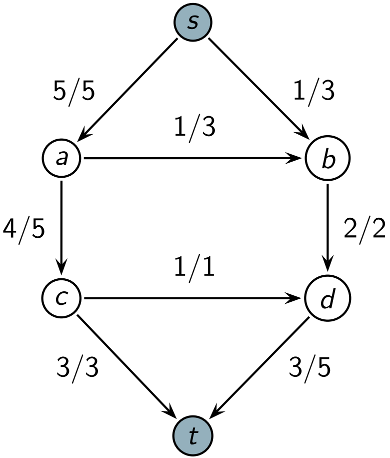
5

notation



edge with capacity c, and current flow f.

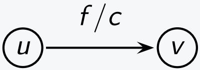
Maximum Flow Example (Augmenting Path)



flow

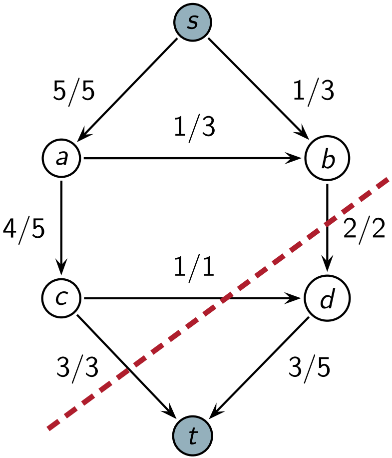
6

notation



edge with capacity c, and current flow f.

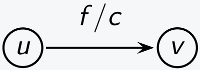
Maximum Flow Example (Augmenting Path)



flow

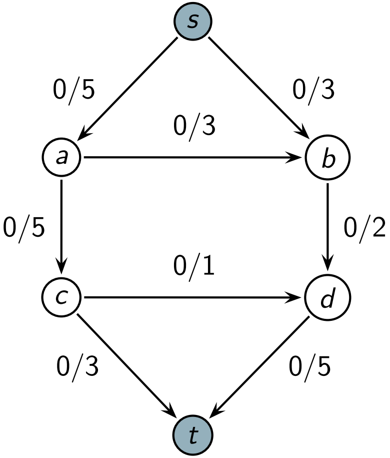
6

notation



edge with capacity c, and current flow f.

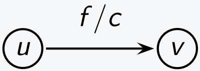
Maximum Flow Example (Push-Relabel)



state

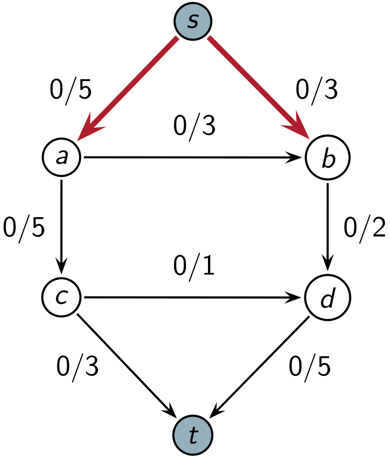
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	0	0
b	0	0
c	0	0
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

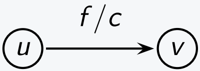
Maximum Flow Example (Push-Relabel)



state

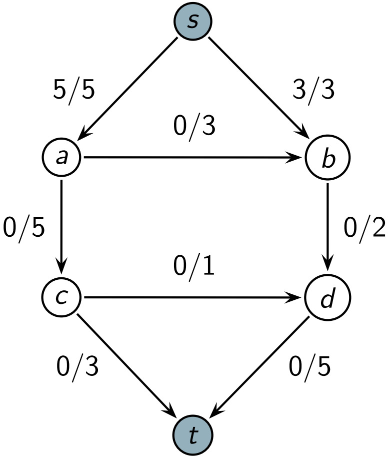
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	0	0
b	0	0
c	0	0
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

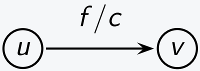
Maximum Flow Example (Push-Relabel)



state

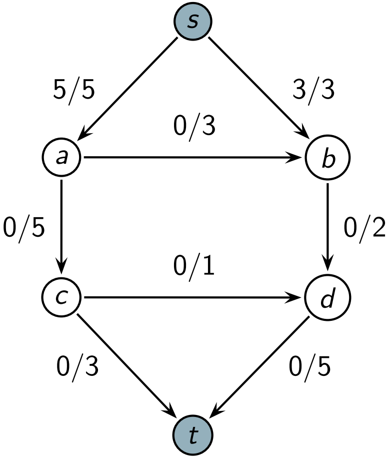
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	0	5
b	0	3
c	0	0
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

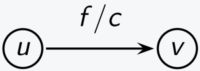
Maximum Flow Example (Push-Relabel)



state

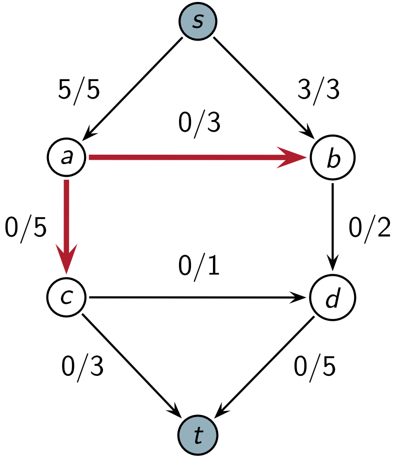
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	5
b	0	3
c	0	0
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

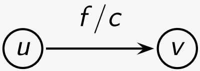
Maximum Flow Example (Push-Relabel)



state

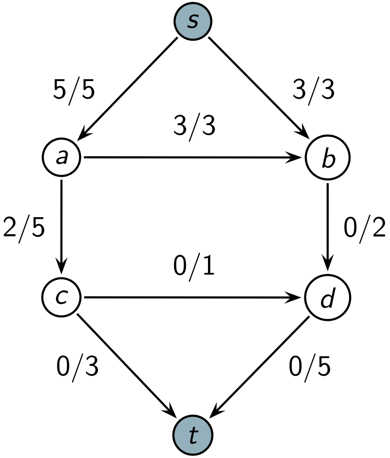
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	5
b	0	3
c	0	0
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

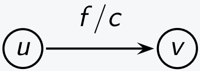
Maximum Flow Example (Push-Relabel)



state

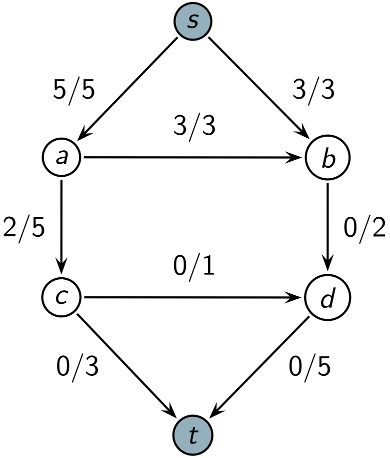
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	0	6
c	0	2
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

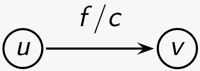
Maximum Flow Example (Push-Relabel)



state

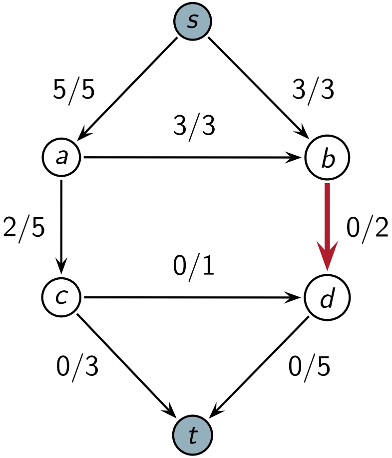
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	6
c	0	2
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

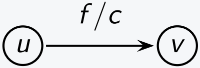
Maximum Flow Example (Push-Relabel)



state

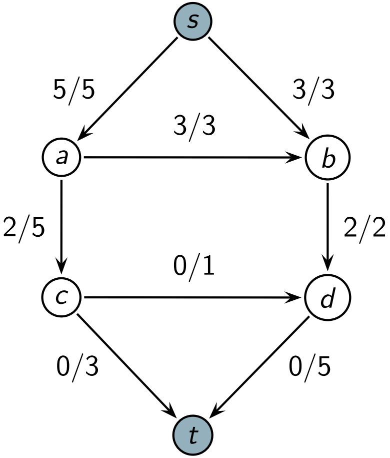
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	6
c	0	2
d	0	0
t	0	0

notation



edge with capacity c, current flow f.

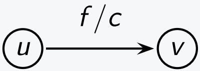
Maximum Flow Example (Push-Relabel)



state

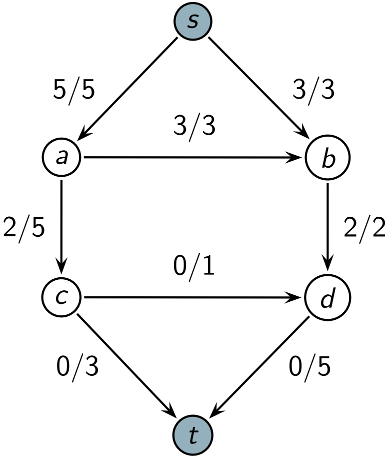
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	0	2
d	0	2
t	0	0

notation



edge with capacity c, current flow f.

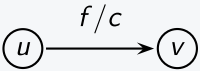
Maximum Flow Example (Push-Relabel)



state

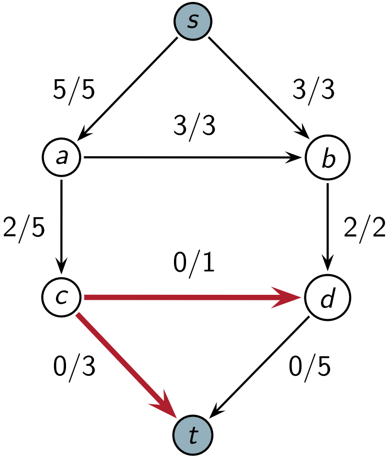
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	2
d	0	2
t	0	0

notation



edge with capacity c, current flow f.

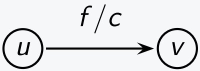
Maximum Flow Example (Push-Relabel)



state

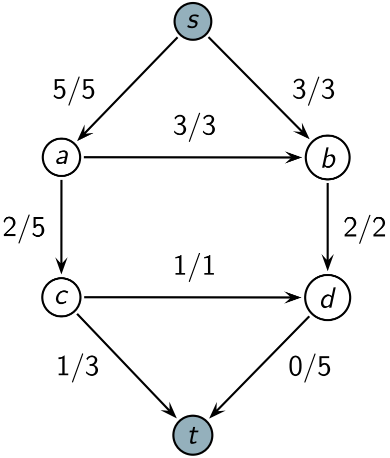
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	2
d	0	2
t	0	0

notation



edge with capacity c, current flow f.

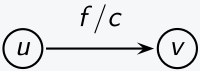
Maximum Flow Example (Push-Relabel)



state

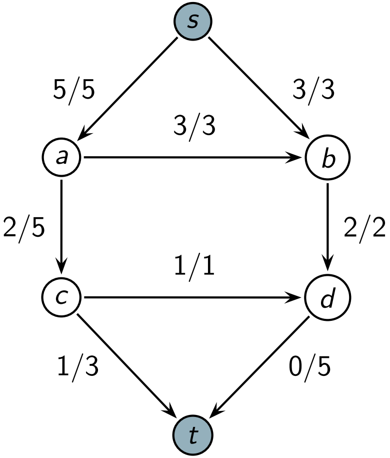
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	0
d	0	3
t	0	1

notation



edge with capacity c, current flow f.

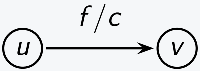
Maximum Flow Example (Push-Relabel)



state

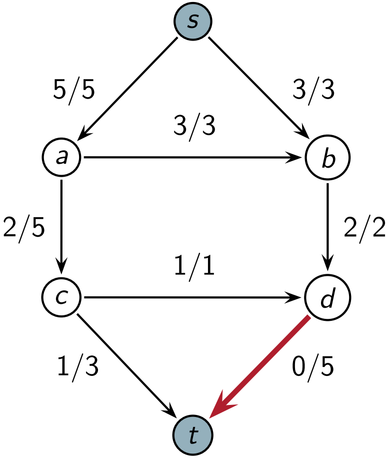
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	0
d	1	3
t	0	1

notation



edge with capacity c, current flow f.

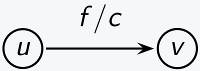
Maximum Flow Example (Push-Relabel)



state

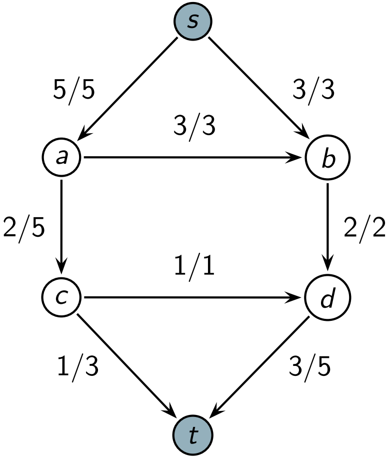
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	0
d	1	3
t	0	1

notation



edge with capacity c, current flow f.

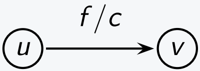
Maximum Flow Example (Push-Relabel)



state

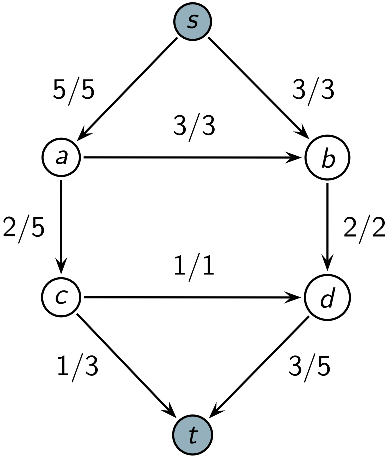
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	1	4
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

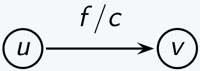
Maximum Flow Example (Push-Relabel)



state

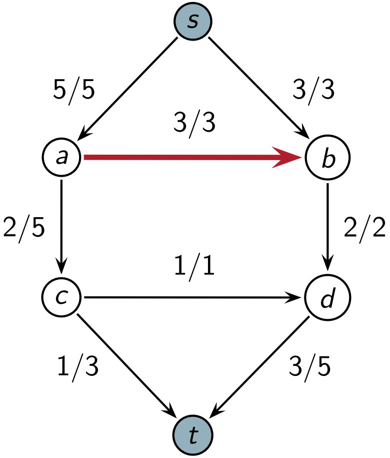
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	2	4
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

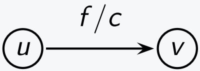
Maximum Flow Example (Push-Relabel)



state

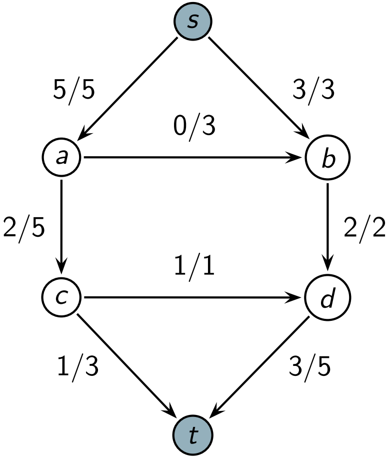
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	0
b	2	4
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

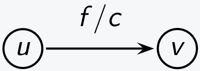
Maximum Flow Example (Push-Relabel)



state

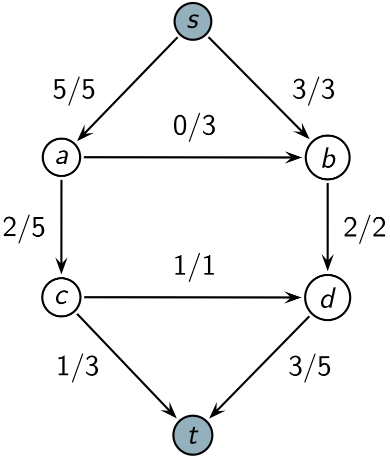
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	1	3
b	2	1
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

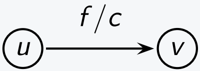
Maximum Flow Example (Push-Relabel)



state

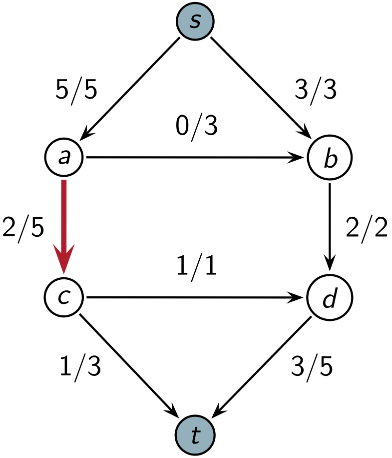
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	3
b	2	1
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

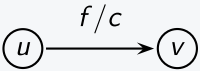
Maximum Flow Example (Push-Relabel)



state

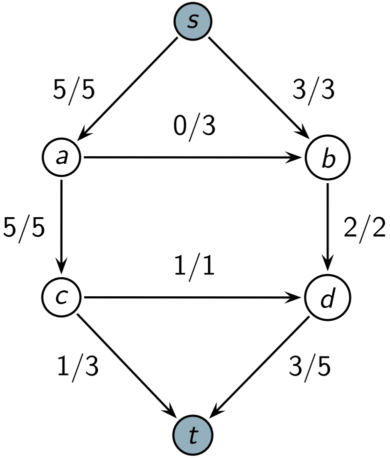
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	3
b	2	1
c	1	0
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

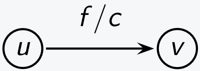
Maximum Flow Example (Push-Relabel)



state

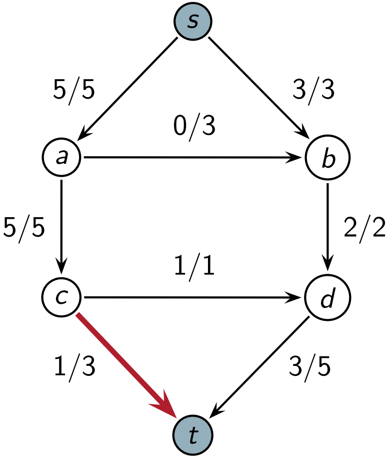
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	2	1
c	1	3
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

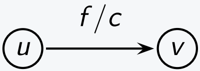
Maximum Flow Example (Push-Relabel)



state

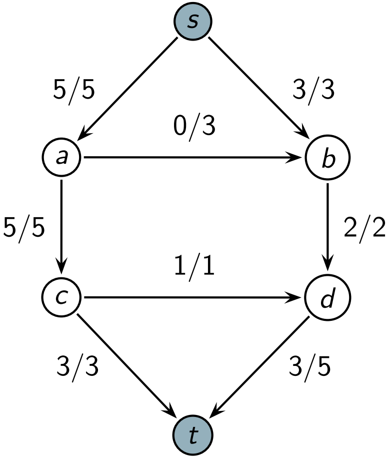
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	2	1
c	1	3
d	1	0
t	0	4

notation



edge with capacity c, current flow f.

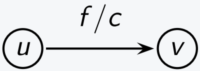
Maximum Flow Example (Push-Relabel)



state

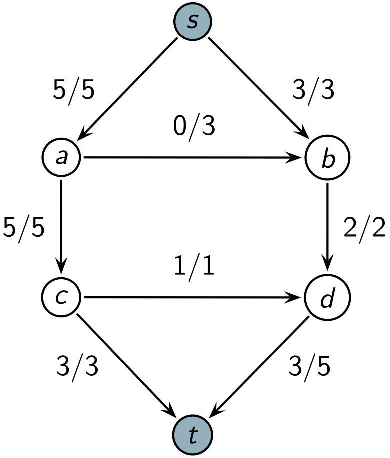
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	2	1
c	1	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

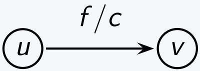
Maximum Flow Example (Push-Relabel)



state

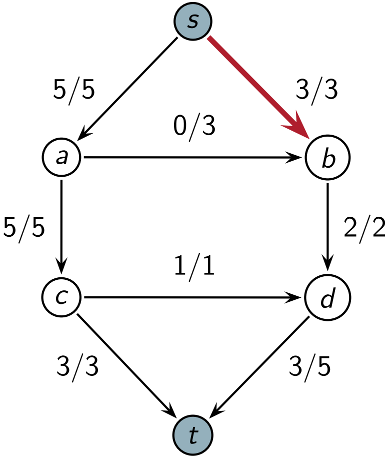
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	7	1
c	1	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

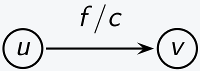
Maximum Flow Example (Push-Relabel)



state

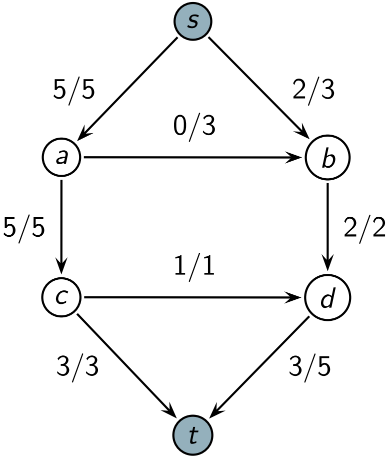
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	7	1
c	1	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

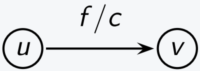
Maximum Flow Example (Push-Relabel)



state

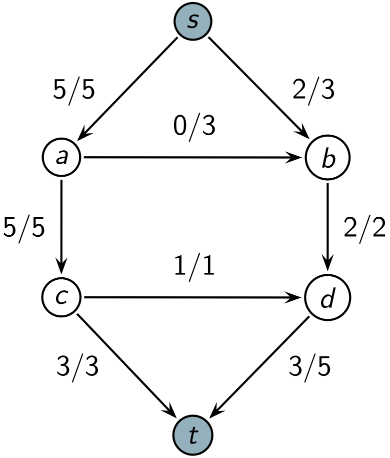
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	7	0
c	1	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

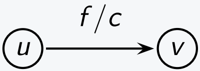
Maximum Flow Example (Push-Relabel)



state

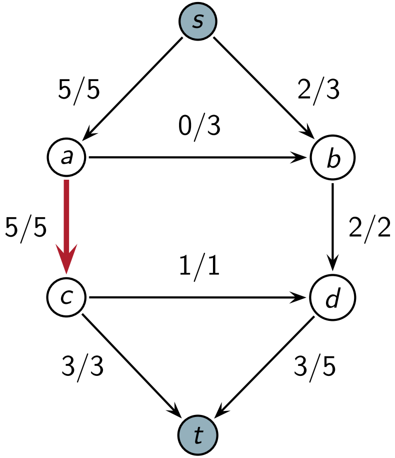
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	7	0
c	3	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

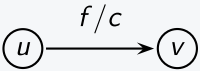
Maximum Flow Example (Push-Relabel)



state

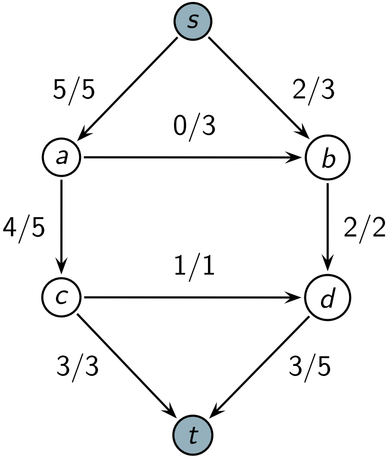
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	0
b	7	0
c	3	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

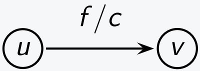
Maximum Flow Example (Push-Relabel)



state

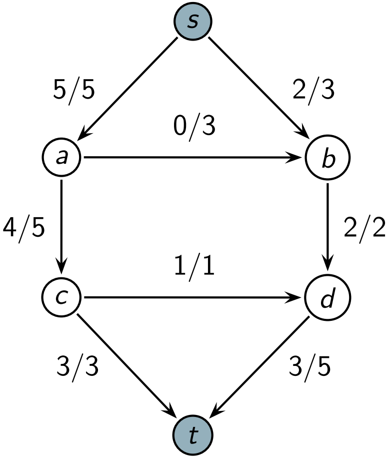
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	2	1
b	7	0
c	3	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

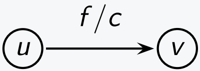
Maximum Flow Example (Push-Relabel)



state

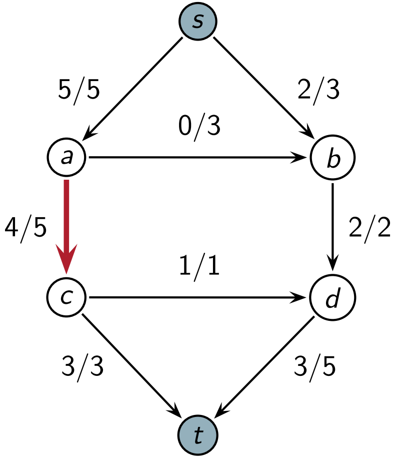
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	1
b	7	0
c	3	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

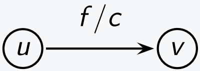
Maximum Flow Example (Push-Relabel)



state

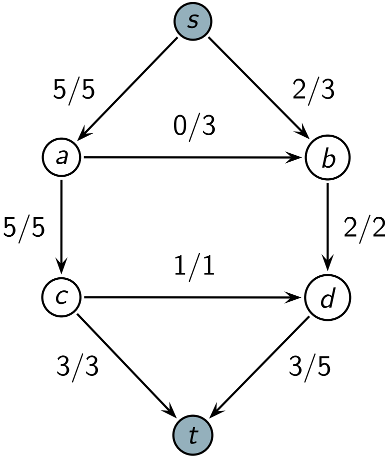
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	1
b	7	0
c	3	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

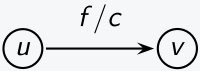
Maximum Flow Example (Push-Relabel)



state

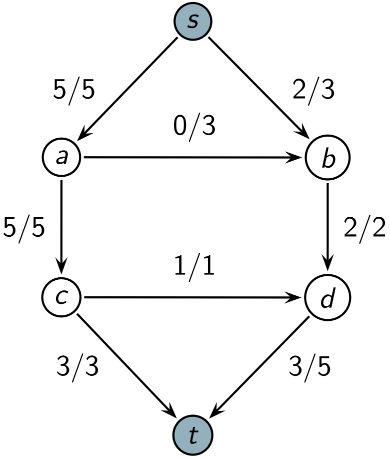
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	0
b	7	0
c	3	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

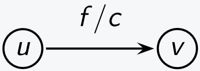
Maximum Flow Example (Push-Relabel)



state

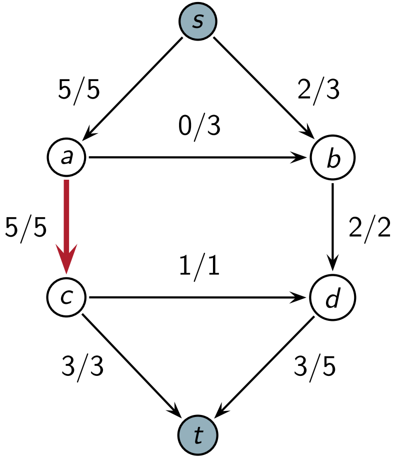
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	0
b	7	0
c	5	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

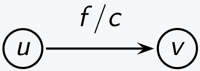
Maximum Flow Example (Push-Relabel)



state

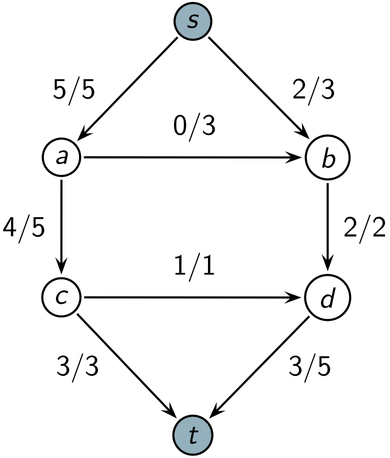
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	0
b	7	0
c	5	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

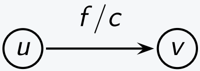
Maximum Flow Example (Push-Relabel)



state

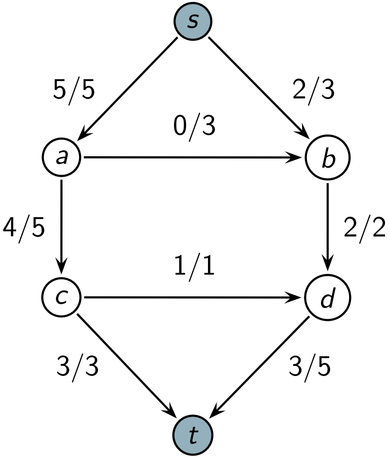
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	4	1
b	7	0
c	5	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

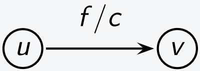
Maximum Flow Example (Push-Relabel)



state

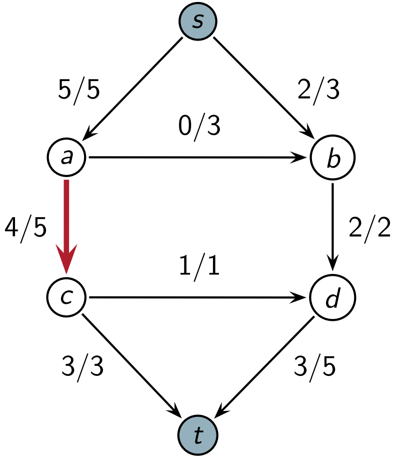
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	1
b	7	0
c	5	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

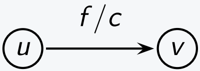
Maximum Flow Example (Push-Relabel)



state

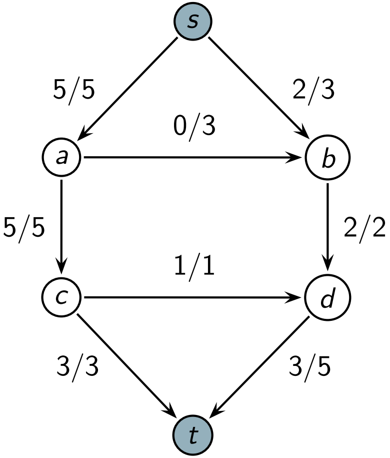
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	1
b	7	0
c	5	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

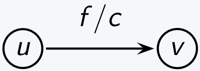
Maximum Flow Example (Push-Relabel)



state

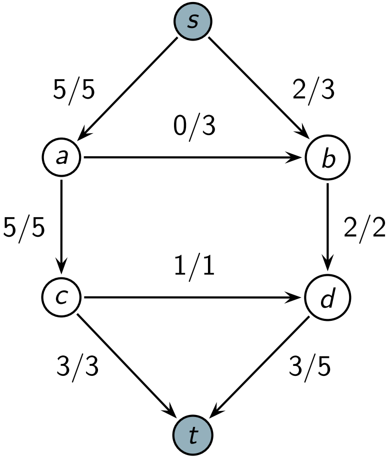
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	0
b	7	0
c	5	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

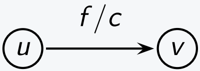
Maximum Flow Example (Push-Relabel)



state

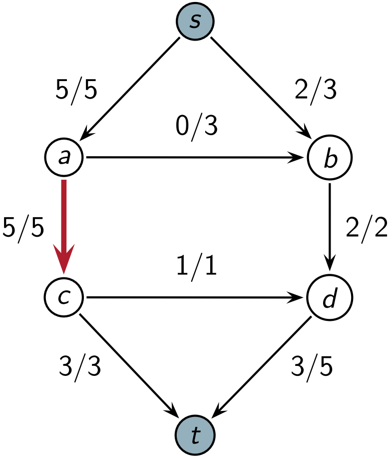
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	0
b	7	0
c	7	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

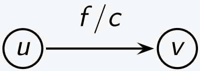
Maximum Flow Example (Push-Relabel)



state

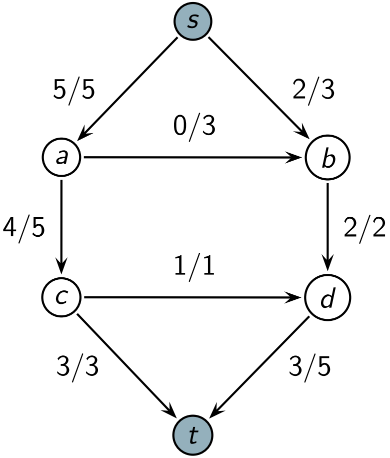
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	0
b	7	0
c	7	1
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

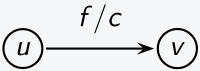
Maximum Flow Example (Push-Relabel)



state

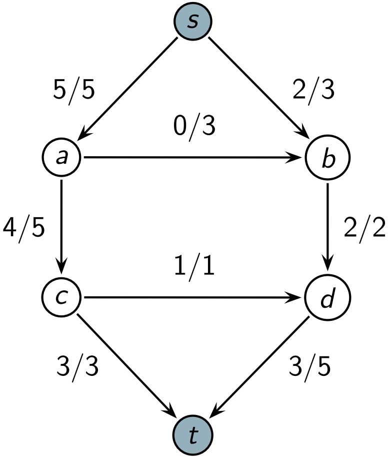
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	6	1
b	7	0
c	7	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

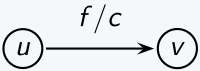
Maximum Flow Example (Push-Relabel)



state

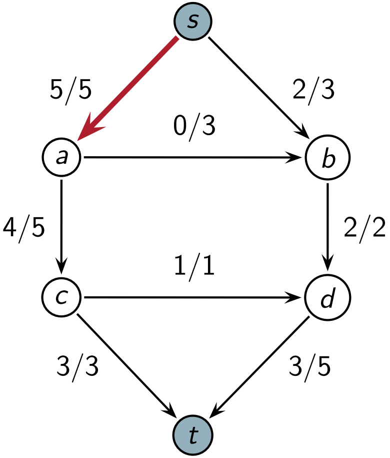
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	7	1
b	7	0
c	7	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

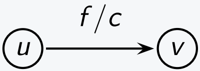
Maximum Flow Example (Push-Relabel)



state

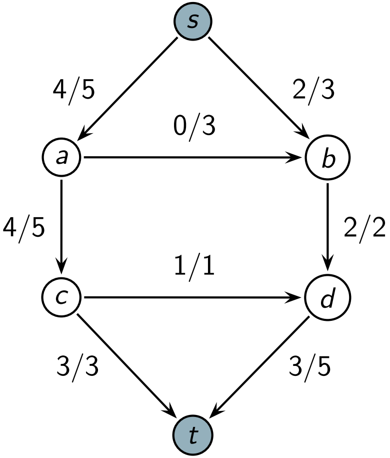
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	7	1
b	7	0
c	7	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

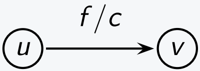
Maximum Flow Example (Push-Relabel)



state

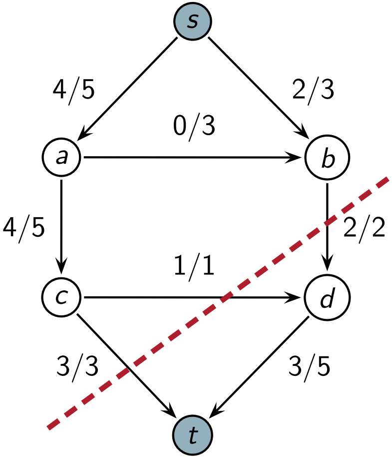
	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	7	0
b	7	0
c	7	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

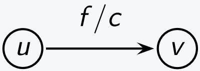
Maximum Flow Example (Push-Relabel)



state

	$h(\cdot)$	$e(\cdot)$
s	6	∞
a	7	0
b	7	0
c	7	0
d	1	0
t	0	6

notation



edge with capacity c, current flow f.

Comparison of Maximum Flow Algorithms

Current state-of-the-art algorithm for exact minimization of general submodular pseudo-Boolean functions is $O(n^5 T + n^6)$, where T is the time taken to evaluate the function [Orlin, 2007].

† assumes integer capacities

Comparison of Maximum Flow Algorithms

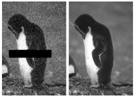
Current state-of-the-art algorithm for exact minimization of general submodular pseudo-Boolean functions is $O(n^5 T + n^6)$, where T is the time taken to evaluate the function [Orlin, 2007].

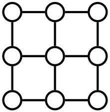
Algorithm	**Complexity**
Ford-Fulkerson	$O(E \max f)^\dagger$
Edmonds-Karp (BFS)	$O(VE^2)$
Push-relabel	$O(V^3)$
Boykov-Kolmogorov	$O(V^2 E \max f)$ ($\sim O(V)$ in practice)

† assumes integer capacities

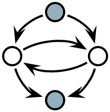
Big Picture: Where are we now?

We can perform inference in submodular binary pairwise Markov random fields exactly.





$\{0,1\}^n \rightarrow \mathbb{R}$



What about...

- non-submodular binary pairwise Markov random fields?
- multi-label Markov random fields?
- higher-order Markov random fields? (part 3)

Multi-label Markov Random Fields

The quadratic pseudo-Boolean optimization techniques described above cannot be applied directly to multi-label MRFs.

However...

- ...for certain MRFs we can transform the multi-label problem into a binary one exactly.
- ...we can project the multi-label problem onto a series of binary problems in a so-called *move-making* algorithm.

The "Battleship" Transform [Ishikawa, 2003]

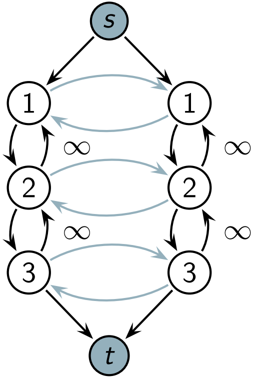
If the multi-label MRFs has pairwise potentials that are convex functions over the label differences, i.e., $\psi^P_{ij}(y_i, y_j) = g(|y_i - y_j|)$ where $g(\cdot)$ is convex, then we can transform the energy function into an equivalent binary one.

$$y = 1 \Leftrightarrow \mathbf{z} = (0, 0, 0)$$
$$y = 2 \Leftrightarrow \mathbf{z} = (1, 0, 0)$$
$$y = 3 \Leftrightarrow \mathbf{z} = (1, 1, 0)$$
$$y = 4 \Leftrightarrow \mathbf{z} = (1, 1, 1)$$

The "Battleship" Transform [Ishikawa, 2003]

If the multi-label MRFs has pairwise potentials that are convex functions over the label differences, i.e., $\psi^P_{ij}(y_i, y_j) = g(|y_i - y_j|)$ where $g(\cdot)$ is convex, then we can transform the energy function into an equivalent binary one.

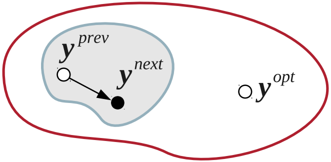
$$y = 1 \Leftrightarrow \mathbf{z} = (0, 0, 0)$$
$$y = 2 \Leftrightarrow \mathbf{z} = (1, 0, 0)$$
$$y = 3 \Leftrightarrow \mathbf{z} = (1, 1, 0)$$
$$y = 4 \Leftrightarrow \mathbf{z} = (1, 1, 1)$$



Move-making Inference

Idea:

- initialize $\mathbf{y}^{\text{prev}}$ to any valid assignment
- restrict the label-space of each variable y_i from $\mathcal{L}$ to $\mathcal{Y}_i \subseteq \mathcal{L}$ (with $y_i^{\text{prev}} \in \mathcal{Y}_i$)
- transform $E : \mathcal{L}^n \to \mathbb{R}$ to $\hat{E} : \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_n \to \mathbb{R}$
- find the optimal assignment $\hat{\mathbf{y}}$ for $\hat{E}$ and repeat



each move results in an assignment with lower energy

Iterated Conditional Modes [Besag, 1986]

Reduce multi-variate inference to solving a series of univariate inference problems.

ICM move

For one of the variables y_i, set $\mathcal{Y}_i = \mathcal{L}$. Set $\mathcal{Y}_j = \{y_j^{\text{prev}}\}$ for all $j \neq i$ (i.e., hold all other variables fixed).

Iterated Conditional Modes [Besag, 1986]

Reduce multi-variate inference to solving a series of univariate inference problems.

ICM move

For one of the variables y_i, set $\mathcal{Y}_i = \mathcal{L}$. Set $\mathcal{Y}_j = \{y_j^{\text{prev}}\}$ for all $j \neq i$ (i.e., hold all other variables fixed).

Can be used for arbitrary energy functions.

Alpha Expansion and Alpha-Beta Swap [Boykov et al., 2001]

Reduce multi-label inference to solving a series of binary (submodular) inference problems.

α-expansion move

Choose some $\alpha \in \mathcal{L}$. Then for all variables, set $\mathcal{Y}_i = \{\alpha, y_i^{\text{prev}}\}$.

$\alpha\beta$-swap move

Choose two labels $\alpha, \beta \in \mathcal{L}$. Then for each variable y_i such that $y_i^{\text{prev}} \in \{\alpha, \beta\}$, set $\mathcal{Y}_i = \{\alpha, \beta\}$. Otherwise set $\mathcal{Y}_i = \{y_i^{\text{prev}}\}$.