# Probabilistic Graphical Models in Computer Vision (IN2329)

Csaba Domokos

Summer Semester 2015/2016



# 4. Conditional random field & Graph cut

## Agenda for today's lecture *

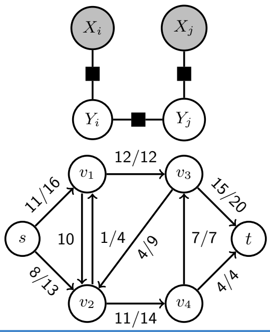


In the **previous lecture** we learnt about graphical models

- Bayesian network
- Markov random field

**Today** we are going to learn about

- Factor graph
- Conditional random field (CRF)
- Inference for graphical models
- Binary image segmentation
- Graph cut, maximum flow





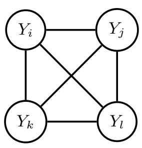



# Factor graph

## Recall: Markov random field *



- An *undirected graphical model* $G = (\mathcal{V}, \mathcal{E})$ is called **Markov random field**, if the **local Markov property** holds, i.e. two nodes are conditionally independent whenever they are not connected.




- A *probability distribution* $p(\mathbf{y})$ on an *undirected graphical model* $G = (\mathcal{V}, \mathcal{E})$ is called **Gibbs distribution** if it can be factorized into potential functions $\psi_c(\mathbf{y}_c) > 0$ defined on cliques:

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{c \in \mathcal{C}_G} \psi_c(\mathbf{y}_c) \,, \text{ where } Z = \sum_{\mathbf{y} \in \mathcal{Y}} \prod_{c \in \mathcal{C}_G} \psi_c(\mathbf{y}_c) \,,$$

and $\mathcal{C}_G$ denotes the set of all (maximal) cliques in $G$.

The Hammersley-Clifford theorem tells us that the above two definitions are equivalent.



## Factor graphs



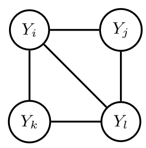
Factor graphs are *undirected graphical models* that **make the factorization explicit** of the probability function.
A factor graph $G = (\mathcal{V}, \mathcal{F}, \mathcal{E}')$ consists of

- variable nodes $V$ (◯) and factor nodes $\mathcal{F}$ (■),
- edges $\mathcal{E}' \subseteq V \times \mathcal{F}$ between variable and factor nodes
- $N : \mathcal{F} \to 2^V$ is the *scope of a factor*, defined as the **set of neighboring variables**, i.e. $N(F) = \{i \in V : (i, F) \in \mathcal{E}\}$.

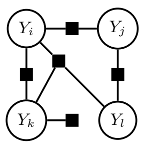
A family of distribution is defined that factorizes as:

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_{N(F)}) \quad \text{with} \quad Z = \sum_{\mathbf{y} \in \mathcal{Y}} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_{N(F)}) \,.$$

Each factor $F \in \mathcal{F}$ connects a subset of nodes, hence we write $\mathbf{y}_F = \mathbf{y}_{N(F)} = (y_{v_1}, \dots, y_{v_{|F|}})$.




MRF

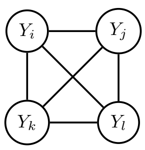



Factor graph

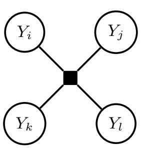


## Examples *

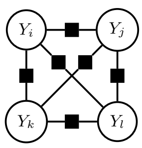





An exemplar MRF




$p_1(\mathbf{y}) = \frac{1}{Z_1} \psi_{ijkl}(y_i, y_j, y_k, y_l)$




$$p_2(\mathbf{y}) = \frac{1}{Z_2} \psi_{ij}(y_i, y_j) \psi_{ik}(y_i, y_k) \psi_{il}(y_i, y_l) \\ \psi_{jk}(y_j, y_k) \psi_{jl}(y_j, y_l) \psi_{kl}(y_k, y_l)$$

Factor graphs are universal, explicit about the factorization, hence it is easier to work with them.





# CRF

## Conditional random field

We have discussed the joint distribution

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_{N(F)}) \, ,$$

but we often have access to measurements $\mathbf{X} = \mathbf{x}$, hence the **conditional distribution** $p(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x})$ could be directly modeled, too.

This can be expressed compactly using **conditional random fields** (CRF) with the factorization

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{p(\mathbf{y}, \mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{y}, \mathbf{x})}{\sum_{\mathbf{y}' \in \mathcal{Y}} p(\mathbf{y}', \mathbf{x})} = \frac{\frac{1}{Z} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_{N(F)}; \mathbf{x}_{N(F)})}{\sum_{\mathbf{y}' \in \mathcal{Y}} \frac{1}{Z} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}'_{N(F)}; \mathbf{x}_{N(F)})}$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_{N(F)}; \mathbf{x}_{N(F)}) \, .$$



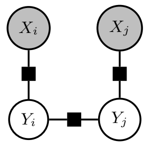


## Conditional random field

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_F; \mathbf{x}_F)$$

with the **partition function** depending on $\mathbf{x}$

$$Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_F; \mathbf{x}_F) \, .$$




Shaded variables: The observations $\mathbf{X} = \mathbf{x}$.

Note that the potentials become also functions of (part of) $\mathbf{x}$, i.e. $\psi_F(\mathbf{y}_F; \mathbf{x}_F)$ instead of just $\psi_F(\mathbf{y}_F)$. Nevertheless, $\mathbf{X}$ is **not** part of the probability model, i.e. it is not treated as random vector.





## Potentials and energy functions

We typically would like to infer marginal probabilities $p(\mathbf{Y}_F = \mathbf{y}_F \mid \mathbf{x})$ for some factors $F \in \mathcal{F}$.

Assuming $\psi_F : \mathcal{Y}_F \to \mathbb{R}^+$, where $\mathcal{Y}_F = \times_{i \in N(F)} \mathcal{Y}_i$ is the product domain of the variables adjacent to $F$, instead of *potentials*, we can also work with **energies**.

We define an **energy function** $E_F : \mathcal{Y}_F \to \mathbb{R}$ for each factor $F \in \mathcal{F}$:

$$E_F(\mathbf{y}_F; \mathbf{x}_F) = -\log(\psi_F(\mathbf{y}_F; \mathbf{x}_F)) \quad \Leftrightarrow \quad \psi_F(\mathbf{y}_F; \mathbf{x}_F) = \exp(-E_F(\mathbf{y}_F; \mathbf{x}_F)) \, .$$

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{F \in \mathcal{F}} \psi_F(\mathbf{y}_F; \mathbf{x}_F) = \frac{1}{Z(\mathbf{x})} \exp(-\sum_{F \in \mathcal{F}} E_F(\mathbf{y}_F; \mathbf{x}_F))$$

$$= \frac{1}{Z(\mathbf{x})} \exp(-E(\mathbf{y}; \mathbf{x})) \, .$$

Hence, $p(\mathbf{y} \mid \mathbf{x})$ is completely determined by $E(\mathbf{y}; \mathbf{x})$.





# Inference



## Inference

The goal is to make predictions $\mathbf{y} \in \mathcal{Y}$, *as good as possible*, about unobserved properties for a given data instance $\mathbf{x} \in \mathcal{X}$.

Suppose we are given a *graphical model* (e.g., a factor graph). The **inference** means the procedure to estimate the *probability distribution*, encoded by the *graphical model*, for a *given data* (or observation).

**Probabilistic inference**: Given a graphical model and the observation $x$, find the value of the *log partition function* and the *marginal distributions* for each factor,

$$\log Z(\mathbf{x}) = \log \sum_{\mathbf{y} \in \mathcal{Y}} \exp(-E(\mathbf{y}; \mathbf{x})) \, ,$$

$$\mu_F(y_F) = p(\mathbf{Y}_F = \mathbf{y}_F \mid \mathbf{x}) \quad \forall F \in \mathcal{F}, \ \forall \mathbf{y}_F \in \mathcal{Y}_F \, .$$

This typically includes variable marginals, i.e. $\mu_i = p(y_i \mid \mathbf{x})$, to make a single prediction $y_i$ for all variables $i \in \mathcal{V}$.





## MAP inference

**Maximum A Posteriori (MAP) inference**: Given a graphical model and the observation $\mathbf{x}$, find the *state* $\mathbf{y}^* \in \mathcal{Y}$ *of maximum probability*

$$\mathbf{y}^* \in \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{Y} = \mathbf{y} \mid \mathbf{x}) \, .$$

Both inference problems are known to be NP-hard for general graphs and factors, but they can be tractable if the underlying graphical model is suitably restricted.





## Energy minimization

Assuming a finite $\mathcal{X}$, the goal is to solve $\mathbf{y}^* \in \operatorname{argmax}_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{y} \mid \mathbf{x})$.

$$\begin{aligned}
\operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{y} \mid \mathbf{x}) &= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \frac{1}{Z(\mathbf{x})} \exp(-E(\mathbf{y}; \mathbf{x})) \\
&= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \exp(-E(\mathbf{y}; \mathbf{x})) \\
&= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} -E(\mathbf{y}; \mathbf{x}) \\
&= \operatorname*{argmin}_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{y}; \mathbf{x}) \, .
\end{aligned}$$

Energy minimization can be interpreted as solving for the most likely state of factor graph, i.e. MAP inference.

In practice, one typically models the energy function directly.





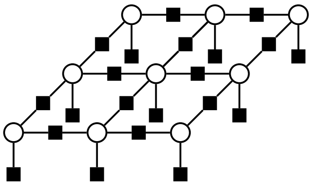
# Binary image segmentation

## Binary image segmentation

Input image

Figure–ground segmentation

*Conditional independences* are specified by a factor graph $G = (\mathcal{V}, \mathcal{F}, \mathcal{E})$, where all pixels depend only on the neighboring ones.







## Binary image segmentation

The conditional distribution factorizes (up to pairwise factors) as

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{i \in \mathcal{V}} \psi_i(y_i; x_i) \prod_{i \in \mathcal{V},\, j \in N(i)} \psi_{ij}(y_i, y_j; x_i, x_j)$$

with

$$Z(\mathbf{x}) = \sum_{\mathbf{y} \in \{0,1\}^{\mathcal{V}}} \prod_{i \in \mathcal{V}} \psi_i(y_i; x_i) \prod_{i \in \mathcal{V},\, j \in N(i)} \psi_{ij}(y_i, y_j; x_i, x_j) \,,$$

where $N(i) = \{j \in \mathcal{V} : (i, j \in \mathcal{E})\}$.

The corresponding energy function $E : \{0,1\}^{\mathcal{V}} \times \mathcal{X} \to \mathbb{R}$:

$$E(\mathbf{y}; \mathbf{x}) = \sum_{i \in \mathcal{V}} E_i(y_i; x_i) + \sum_{i \in \mathcal{V},\, j \in N(i)} E_{ij}(y_i, y_j; x_i, x_j) \,.$$





## Binary image segmentation

In order to define energy functions for unary factors, one can consider a set of functions $\phi_i : \mathcal{Y}_i \times \mathcal{X}_i \to [0; 1]$:

$$E_i(y_i; x_i) = -\log \phi_i(y_i; x_i) \quad \text{for all } i \in V \,.$$

For pairwise factor energies we use the **Potts model** here, that is

$$E_{ij}(y_i, y_j; x_i, x_j) := E_{ij}(y_i, y_j) = [\![ y_i \neq y_j ]\!] = \begin{cases} 0, & \text{if } y_i = y_j \\ 1, & \text{otherwise.} \end{cases}$$

The resulting energy function given as

$$\begin{aligned} E(\mathbf{y}; \mathbf{x}) &= \sum_{i \in \mathcal{V}} E_i(y_i; x_i) + \sum_{i \in \mathcal{V},\, j \in N(i)} E_{ij}(y_i, y_j; x_i, x_j) \\ &= \sum_{i \in \mathcal{V}} -\log \phi_i(y_i; x_i) + \sum_{i \in \mathcal{V},\, j \in N(i)} [\![ y_i \neq y_j ]\!] \,. \end{aligned}$$





## Summary *

- **Factor graphs** are universal, explicit about the factorization, hence it is easier to work with them.
- A **Conditional random field** is an *undirected graphical model*, which expresses compactly $p(\mathbf{y} \mid \mathbf{x})$ for some observation $\mathbf{X} = \mathbf{x}$.
- The **inference** means the procedure to estimate the *probability distribution*, encoded by the *graphical model*, for a *given data*.
- Given a *graphical model* and the *observation* $\mathbf{x}$, **MAP inference** means to find the *state* $\mathbf{y}^* \in \mathcal{Y}$ *of maximum probability*

$$\mathbf{y}^* \in \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{Y} = \mathbf{y} \mid \mathbf{x}) \,.$$

- In order to solve **binary image segmentation**, one may minimize the energy function $E : \{0,1\}^{\mathcal{V}} \times \mathcal{X} \to \mathbb{R}$:

$$E(\mathbf{y}; \mathbf{x}) = \sum_{i \in \mathcal{V}} -\log \phi_i(y_i; x_i) + \sum_{i \in \mathcal{V},\, j \in N(i)} [\![ y_i \neq y_j ]\!] \,.$$





# Graph Cut



## Graph cut

Assume a **weighted directed graph** $G = (\mathcal{V}, \mathcal{E}, c)$

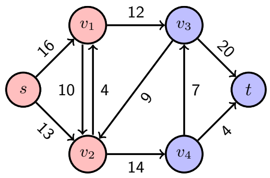
- $\mathcal{V} = \{1, \ldots, n\}$ is a finite set of nodes,
- $\mathcal{E} \subseteq \{(i, j) \in \mathcal{V} \times \mathcal{V} \mid i \neq j\}$ is the set of edges,
- $c : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ is a weight function. (For any $(i, j) \notin \mathcal{E}$, $c(i, j) = 0$.)

A **cut** $(\mathcal{S}, \mathcal{T})$ of $G$ is a *disjoint* partition of $\mathcal{V}$ into $\mathcal{S}$ and $\mathcal{T} = \mathcal{V} \backslash \mathcal{S}$.

The **capacity** of the cut $(\mathcal{S}, \mathcal{T})$ is defined as

$$\text{cut}(\mathcal{S}, \mathcal{T}) = \sum_{(i,j) \in \mathcal{S} \times \mathcal{T}} c(i, j) \,.$$

Assume distinct nodes $s, t \in \mathcal{V}$, a cut $(\mathcal{S}, \mathcal{T})$ is called **$s - t$ cut** if $s \in \mathcal{S}$ and $t \in \mathcal{T}$.




The **minimum $s - t$ cut problem** is to find an $s - t$ cut with the lowest cost.

Example: $\text{cut}(\mathcal{S}, \mathcal{T}) = c(v_1, v_3) + c(v_2, v_4) = 12 + 14 = 26$.





# Flow network

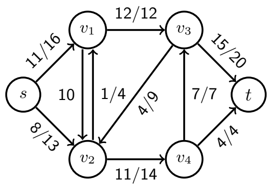


## Flow network and flow

Let $G = (\mathcal{V}, \mathcal{E}, c)$ be a *directed weighted graph* with **non-negative** edge weights. Given two distinct nodes, a **source** $s$ and a **sink** $t$, we call $(\mathcal{V}, \mathcal{E}, c, s, t)$ a **flow network**.

Let $(\mathcal{V}, \mathcal{E}, c, s, t)$ be a *flow network*. A function $f : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ is called a **flow** if it satisfies the following properties:

1. **Capacity constraint**:
$$f(i, j) \leqslant c(i, j) \text{ for all } i, j \in \mathcal{V} \,.$$
2. **Skew-symmetry**:
$$f(i, j) = -f(j, i) \text{ for all } i, j \in \mathcal{V} \,.$$
3. **Flow conservation**:
$$\sum_{j \in \mathcal{V}} f(i, j) = 0 \text{ for all } i \in \mathcal{V} \backslash \{s, t\} \,.$$




The edges are labeled by $f(i, j)/c(i, j)$.

Only positive $f(i, j)$ are shown.

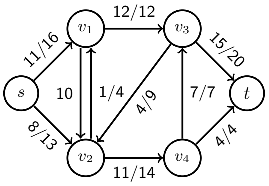
# The value of a flow *



The **value** of a flow $f$ is defined as

$$|f| \triangleq \sum_{(s,i)\in\mathcal{E}} f(s,i) = -\sum_{(i,t)\in\mathcal{E}} f(i,t) .$$

The **maximum-flow problem** is to find a *flow* $f$ with the highest cost for a given flow network $G$.




The edges are labeled by $f(i,j)/c(i,j)$.

$|f| = 19$.



# An equivalent definition of flows *

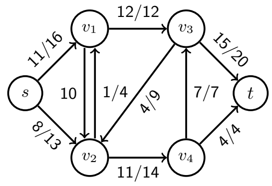


Now we give a *more intuitive definition* of *flows*. We will see that the previous definition is more helpful for the analysis of the *maximum-flow algorithm*.

Let $(\mathcal{V},\mathcal{E},c,s,t)$ be a flow network. A function $f:\mathcal{E}\to\mathbb{R}^+$ is called a **flow** if it satisfies the following two properties:

1. $f(i,j) \leqslant c(i,j)$ for all $(i,j)\in\mathcal{E}$.
2. For all $i \in \mathcal{V}\backslash\{s,t\}$

$$\sum_{(i,j)\in\mathcal{E}} f(i,j) = \sum_{(j,i)\in\mathcal{E}} f(i,j) .$$




The edges are labeled by $f(i,j)/c(i,j)$.

One can see that the two definitions of the flow are equivalent. (See Exercise)



# Working with flows *



Let $G = (\mathcal{V},\mathcal{E},c,s,t)$ be a *flow network* and let $f$ be a *flow* in $G$. We will use the following notation for $A,B\subseteq\mathcal{V}$

$$f(A,B) = \sum_{a\in A}\sum_{b\in B} f(a,b) .$$

It is easy to see that $|f| = f(\mathcal{V},\{t\})$, and $f(\{i\},\mathcal{V}) = 0$ for all $i\in\mathcal{V}\backslash\{s,t\}$ due to *flow conservation*.

Let $G = (\mathcal{V},\mathcal{E},c,s,t)$ be a *flow network* and let $f$ be a *flow* in $G$. Then the following equalities hold:

i) For all $A\subseteq\mathcal{V}$, we have $f(A,A)=0$.
ii) For all $A,B\subseteq\mathcal{V}$, we have $f(A,B) = -f(B,A)$.
iii) For all $A,B,C\subseteq\mathcal{V}$ with $A\cap B=\emptyset$, we have

$$f(A\cup B,C) = f(A,C)+f(B,C) \text{ and } f(C,A\cup B) = f(C,A)+f(C,B) .$$

*Proof.* Exercise. □

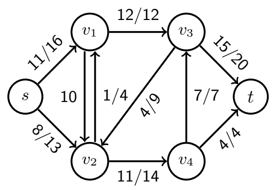


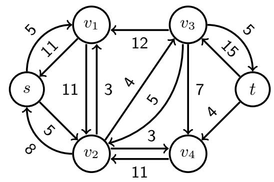
# Residual network



Let $G = (\mathcal{V},\mathcal{E},c,s,t)$ be a *flow network* and let $f$ be a *flow* in $G$. The *weighted directed graph* $G_f = (\mathcal{V},\mathcal{E}_f,c_f)$ is called **residual network** of $G$ induced by $f$, where

$$c_f(i,j) = c(i,j) - f(i,j) ,$$
$$\mathcal{E}_f = \{(i,j)\in\mathcal{V}\times\mathcal{V} : c_f(i,j) > 0\} .$$




A path $p$ from $s$ to $t$ in $G_f$ is called an **augmenting path**.



# Max-flow–min-cut theorem



Let $f$ be a *flow* in a *flow network* $G = (\mathcal{V},\mathcal{E},c,s,t)$. Then the following conditions are equivalent:

1) $f$ is a maximal flow in $G$.
2) The residual graph $G_f$ contains no augmenting paths.
3) $|f| = \text{cut}(\mathcal{S},\mathcal{T})$ for some $s-t$ cut of $G$.



# Proof of Max-flow–min-cut theorem $1)\Rightarrow 2)$ *



Suppose that $f$ is *maximum flow* in $G$, but and there exists an *augmenting path* $p$ in the *residual graph* $G_f$.

The maximum amount by which we can **increase** the flow in $p$ is the **residual capacity** of $p$, given by

$$c_f(p) = \min\{c_f(i,j) : (i,j) \text{ is on } p\} .$$

Furthermore, let us define $f_p:\mathcal{E}\to\mathbb{R}$ as follows:

$$f_p(i,j) = \begin{cases} c_f(p) & \text{if } (i,j) \text{ is on } p \\ -c_f(p) & \text{if } (j,i) \text{ is on } p \\ 0 & \text{otherwise.} \end{cases}$$

One can see that $f_p$ is a flow in $G_f$ with value $|f_p| = c_f(p) > 0$. Therefore the flow $f+f_p$ has the value $|f|+|f_p| > |f|$, which contradicts the optimality of $f$.



# Proof of Max-flow–min-cut theorem $2)\Rightarrow 3)$ *



Suppose that $G_f$ *has no augmenting path*, i.e. $s$ and $t$ are *disconnected* in $G_f$. Define

$$\mathcal{S} := \{v\in\mathcal{V} : \text{ there exists a path from } s \text{ to } v \text{ in } G_f\} .$$

Obviously, $(\mathcal{S},\mathcal{T})$ is a *cut* of $G$, where $\mathcal{T} = \mathcal{V}\backslash\mathcal{S}$.

For each pair of $(i,j)\in\mathcal{S}\times\mathcal{T}$, we have $f(i,j) = c(i,j)$, since otherwise $(i,j)\in\mathcal{E}_f$, which would mean $j\in\mathcal{S}$.

One can see that the flow across $(\mathcal{S},\mathcal{T})$ is $|f|$:

$$f(\mathcal{S},\mathcal{T}) \overset{\text{iii)}}{=} f(\mathcal{S},\mathcal{V}) - f(\mathcal{S},\mathcal{S}) \overset{\text{i)}}{=} f(\mathcal{S},\mathcal{V}) \overset{\text{iii)}}{=} f(\{s\},\mathcal{V}) + f(\mathcal{S}\backslash\{s\},\mathcal{V})$$
$$= f(\{s\},\mathcal{V}) = |f| .$$

Therefore $|f| = f(\mathcal{S},\mathcal{T}) = \text{cut}(\mathcal{S},\mathcal{T})$.



# Proof of Max-flow–min-cut theorem $3)\Rightarrow 1)$ *



Let $f$ be a *flow* in $G$ such that $|f| = \text{cut}(\mathcal{S},\mathcal{T})$. In general, for **any** flow $f$ in $G$ the following holds:

$$|f| = f(\mathcal{S},\mathcal{T}) = \sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{T}} f(i,j) \leqslant \sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{T}} c(i,j) = \text{cut}(\mathcal{S},\mathcal{T}) .$$

Hence $|f| = \text{cut}(\mathcal{S},\mathcal{T})$ is maximal (equivalently $\text{cut}(\mathcal{S},\mathcal{T})$ is minimal). □

# Ford-Fulkerson algorithm *



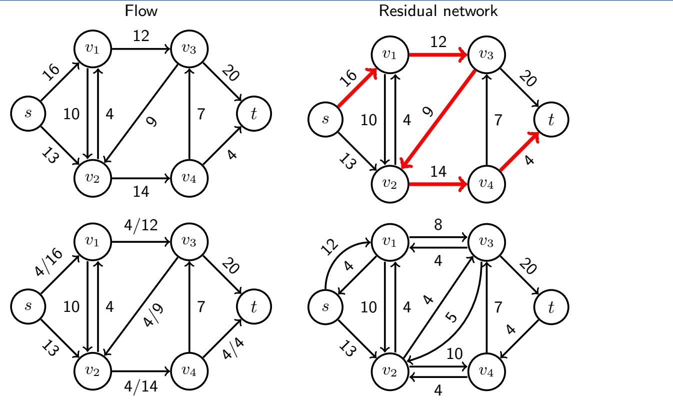
**Input:** A flow network $G = (\mathcal{V}, \mathcal{E}, c, s, t)$
**Output:** A minumum $s-t$ cut $(\mathcal{S}, \mathcal{T})$ of $G$

1: **for all** $(i,j) \in \mathcal{E}$ **do**
2: $f(i,j) \leftarrow 0$ and $f(j,i) \leftarrow 0$
3: **end for**
4: **while** there exists a path $p$ from $s$ to $t$ in the residual network $G_f$ **do**
5: $c_f(p) \leftarrow \min\{c_f(i,j) : (i,j) \text{ is in } p\}$
6: **for all** $(i,j)$ in $p$ **do**
7: $f(i,j) \leftarrow f(i,j) + c_f(p)$
8: $f(j,i) \leftarrow -f(i,j)$
9: **end for**
10: **end while**
11: $\mathcal{S} \leftarrow \{v \in \mathcal{V} :$ there exists a path from $s$ to $v$ in $G_f\}$ and $\mathcal{T} \leftarrow \mathcal{V} \backslash \mathcal{S}$



# Example: iteration 1 *

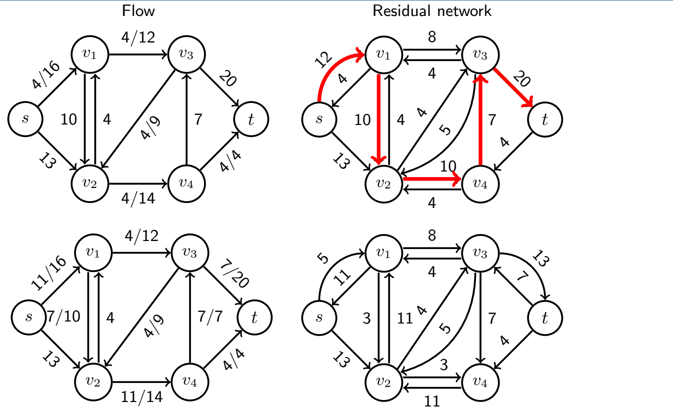







# Example: iteration 2 *

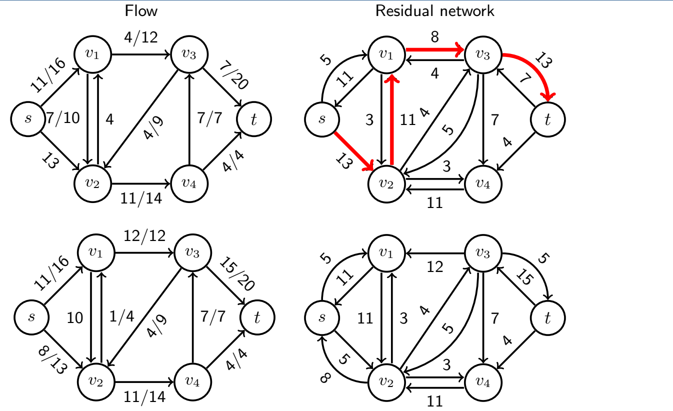







# Example: iteration 3 *

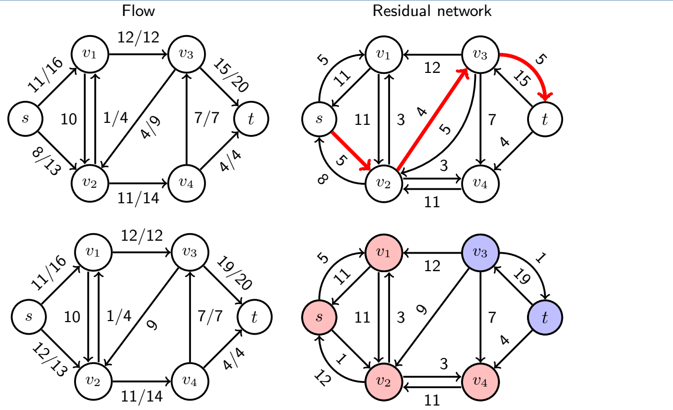







# Example: iteration 4 *

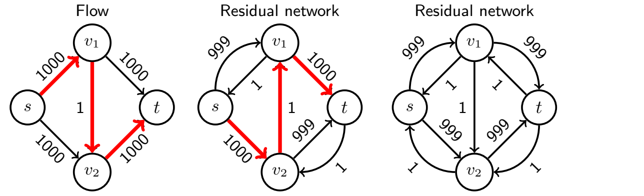







# A "bad" example *






Note that there exists an example, where the flow, computed by the Ford–Fulkerson algorithm, does not even converge to the maximum flow.

More precisely, if a flow network has integer ($\mathbb{N}_0$) or rational ($\mathbb{Q}_0^+$) capacities, then the Ford–Fulkerson algorithm terminates and it computes a maximum flow.



# Edmonds–Karp algorithm



**Input:** A flow network $G = (\mathcal{V}, \mathcal{E}, c, s, t)$
**Output:** A minumum $s-t$ cut $(\mathcal{S}, \mathcal{T})$ of $G$

1: **for all** $(i,j) \in \mathcal{E}$ **do**
2: $f(i,j) \leftarrow 0$ and $f(j,i) \leftarrow 0$
3: **end for**
4: **while** there exists a path $p$ from $s$ to $t$ in the residual network $G_f$ **do**
5: $p \leftarrow$ `shorthestPath`$(G_f, s, t)$
6: $c_f(p) \leftarrow \min\{c_f(i,j) : (i,j) \text{ is in } p\}$
7: **for all** $(i,j)$ in $p$ **do**
8: $f(i,j) \leftarrow f(i,j) + c_f(p)$
9: $f(j,i) \leftarrow -f(i,j)$
10: **end for**
11: **end while**
12: $\mathcal{S} \leftarrow \{v \in \mathcal{V} :$ there exists a path from $s$ to $v$ in $G_f\}$ and $\mathcal{T} \leftarrow \mathcal{V} \backslash \mathcal{S}$

The complexity of this algorithm is $\mathcal{O}(\mathcal{V}\mathcal{E}^2)$. There exists more efficient algorithms for maximum flow calculation with complexity $\mathcal{O}(\mathcal{V}^2\mathcal{E})$ and $\mathcal{O}(\mathcal{V}^3)$.



# Summary *



- **Max-flow–min-cut theorem** tells us that the minimum cut problem can be solved via maximum flow.
  These two problems are dual to each other, moreover strong duality holds.
- `Edmonds--Karp algorithm`: The `Ford-Fulkerson algorithm` becomes polynomial, if the shortest path is used as augmented path.

In the **next lecture** we will learn about

- Exact solution for *binary image segmentation* via *graph cut*
- Multi-label problem

# Literature *

**Conditional random field**


1. Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009
2. Sebastian Nowozin and Christoph H. Lampert. Structured prediction and learning in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4), 2010
3. Andrew Blake, Pushmeet Kohli, and Carsten Rother, editors. *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011
4. Charles Sutton and Andrew McCallum. An introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012


**Graph cut, Maximum flow**


5. Ronald L. Rivest Thomas H. Cormen, Charles E. Leiserson and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009
6. Tanmay Verma and Dhruv Batra. MaxFlow revisited: An empirical comparison of MaxFlow algorithms for dense vision problems. In Richard Bowden, John Collomosse, and Krystian Mikolajczyk, editors, *In Proceedings of British Machine Vision Conference*, pages 61.1–61.12, Surrey, UK, September 2012. BMVA Press