Digital Image - Lecture 08

Image Segmentation
Part II

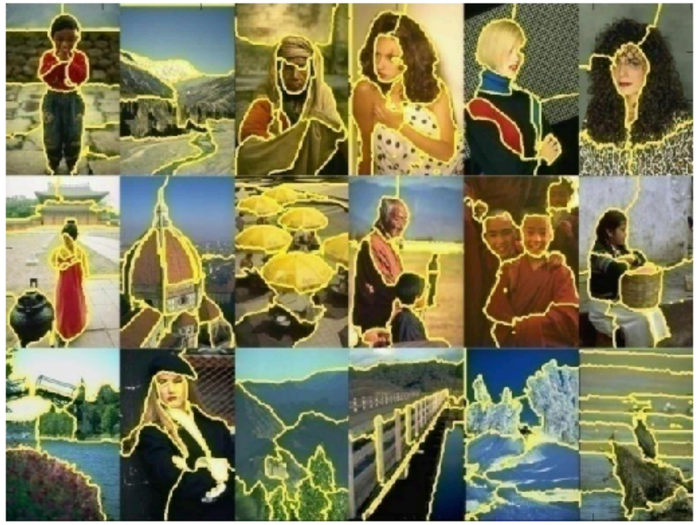
Torsten Sattler

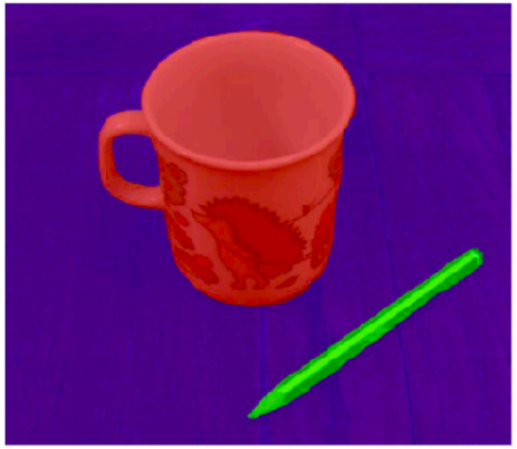
Czech Institute of Informatics, Robotics and Cybernetics
Czech Technical University in Prague

slides adapted from Václav Hlaváč and Bastian Leibe

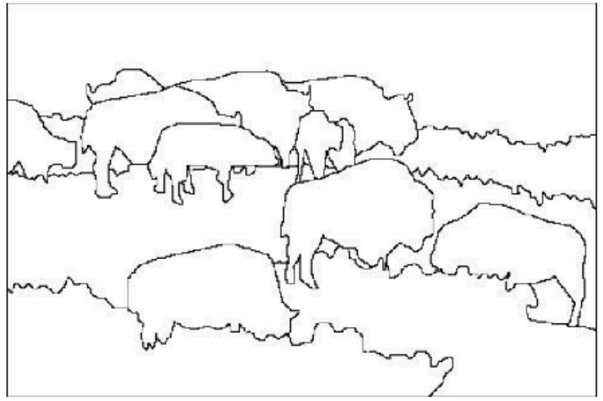
CZECH INSTITUTE OF INFORMATICS ROBOTICS AND CYBERNETICS CTU IN PRAGUE

Recap: What Is Image Segmentation?





Image, courtesy Ondřej Drbohlav



Goal: **segment** image **into** (semantically) meaningful **regions**

slide credit: Václav Hlaváč, Bastian Leibe, Kristen Grauman, Svetlana Lazebnik

Recap: Last Lecture

- A simple approach to segmentation: **(intensity) thresholding**

slide credit: Václav Hlaváč

Recap: Last Lecture

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**

slide credit: Václav Hlaváč

Recap: Last Lecture

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**

slide credit: Václav Hlaváč

Recap: Last Lecture

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**
- Segmentation as a statistical (unsupervised) learning problem: **expectation maximization (EM) algorithm**

slide credit: Václav Hlaváč

Recap: Last Lecture

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**
- Segmentation as a statistical (unsupervised) learning problem: **expectation maximization (EM) algorithm**
- Next lecture: **graph-based segmentation**, **supervised learning with neural networks** (if time and interest)

slide credit: Václav Hlaváč

Recap: Last Lecture

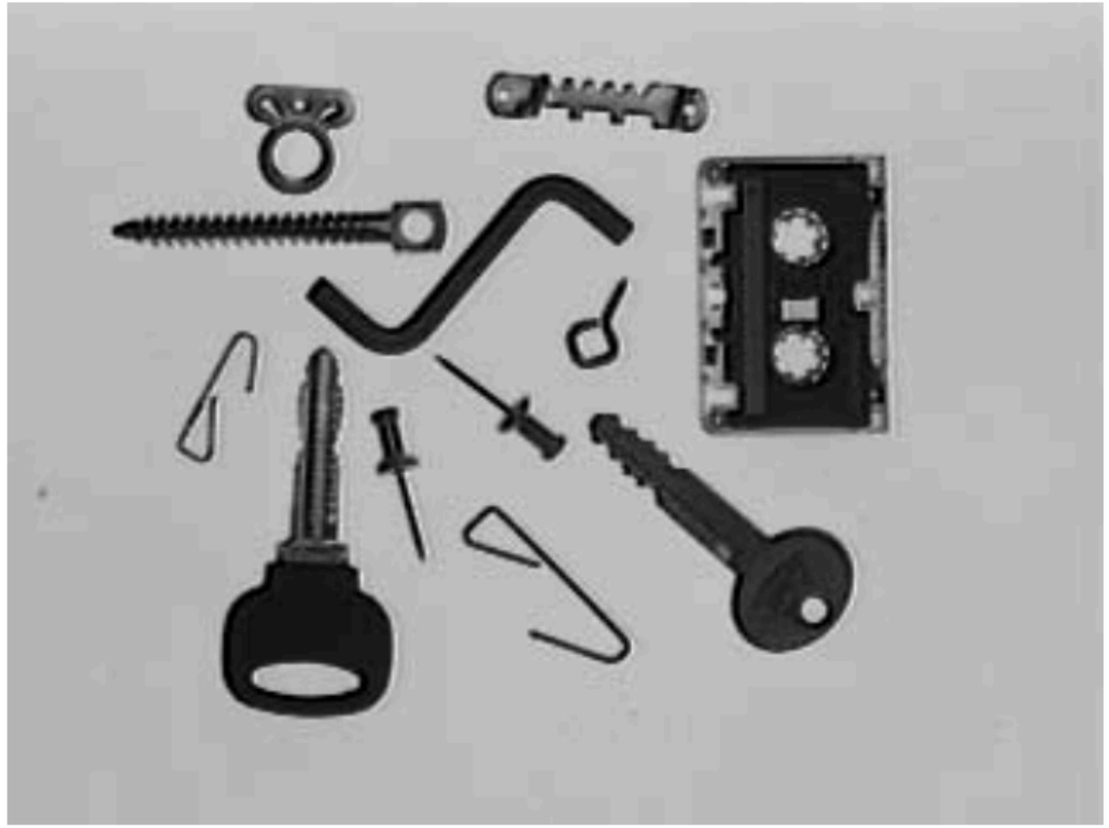
simple & heuristic

complex & principled

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**
- Segmentation as a statistical (unsupervised) learning problem: **expectation maximization (EM) algorithm**
- Next lecture: **graph-based segmentation**, **supervised learning with neural networks** (if time and interest)

slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding



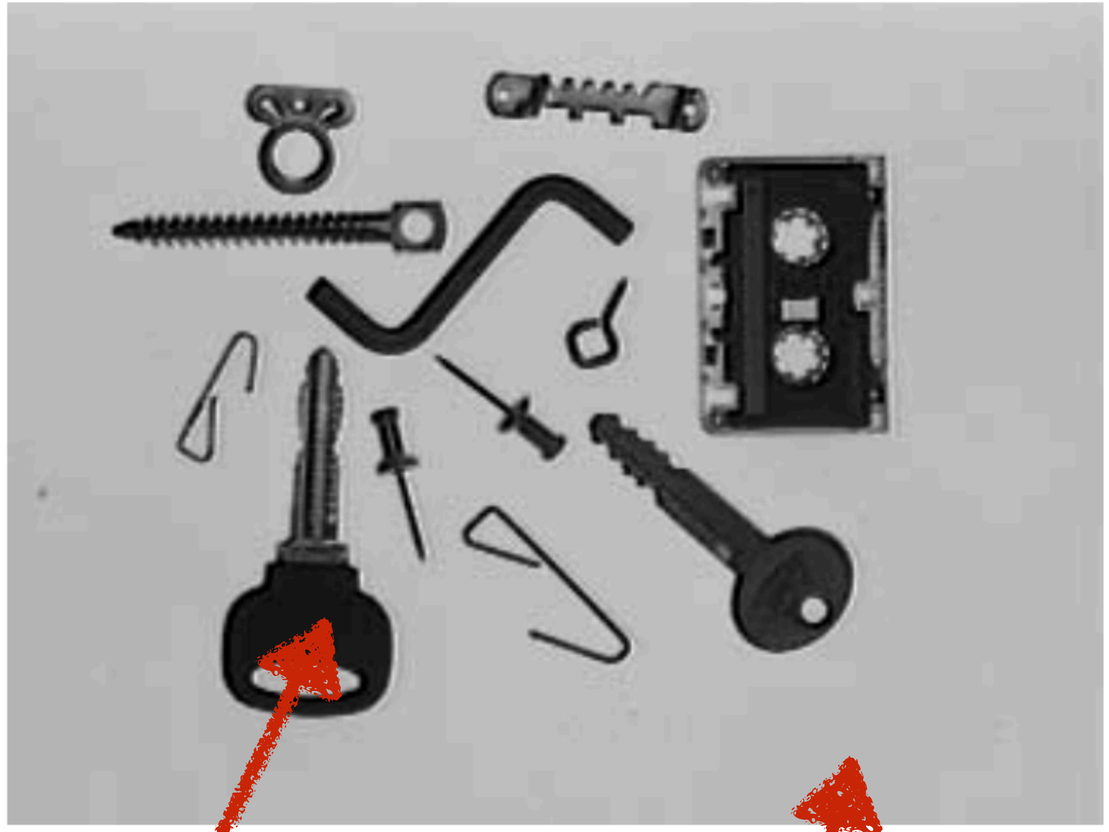
slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding

simple background

slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding

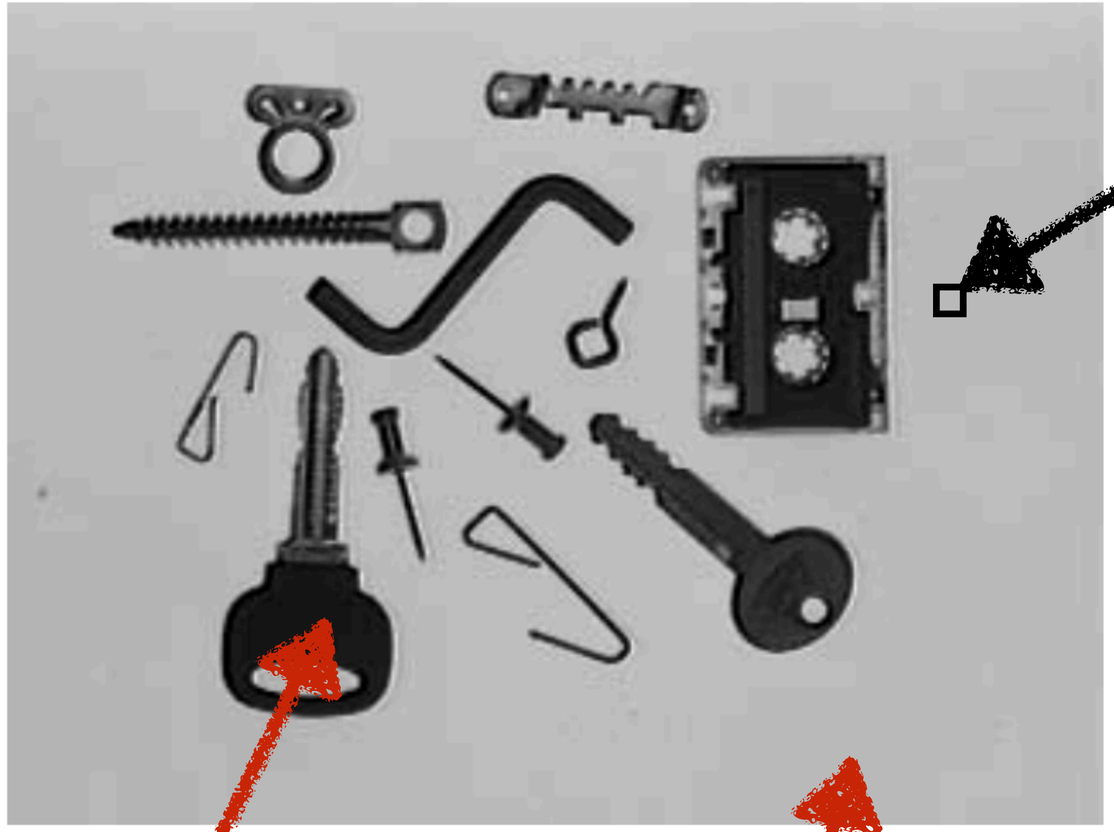


distinctly colored objects

simple background

slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding



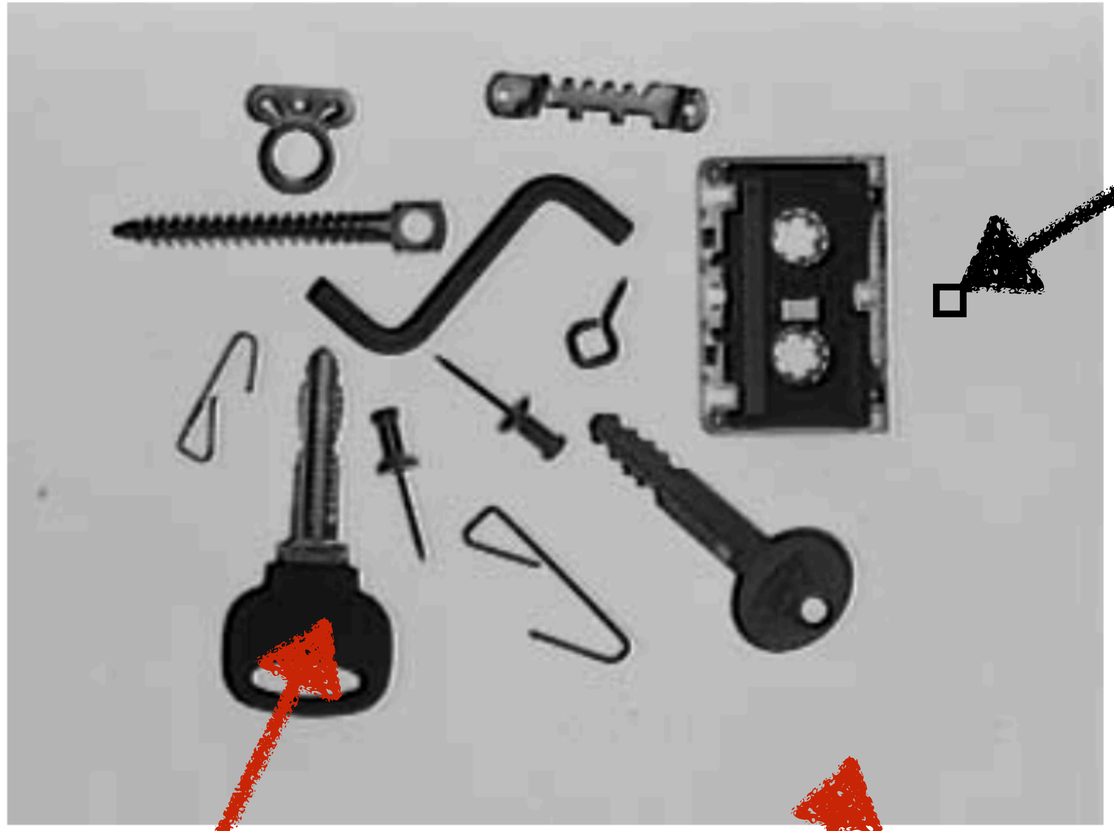
pixel (i, j) with intensity $f(i, j)$

distinctly colored objects

simple background

slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding



pixel (i, j) with intensity $f(i, j)$

generate binary image b with

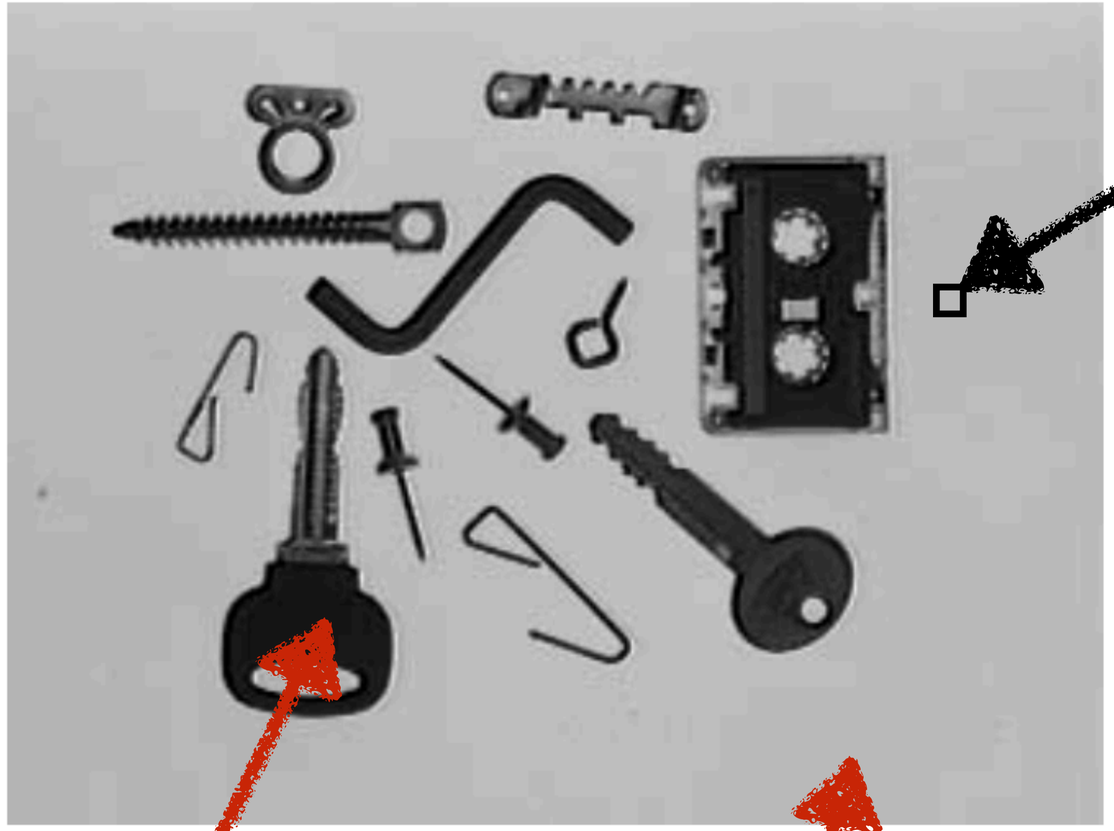
$$b(i,j) = \begin{cases} 1 & \text{if} \quad f(i,j) \geq T \\ 0 & \text{if} \quad f(i,j) < T \end{cases}$$

distinctly colored objects

simple background

slide credit: Václav Hlaváč

Recap: Image Segmentation via Thresholding



pixel (i, j) with intensity $f(i, j)$

generate binary image b with

$$b(i,j) = \begin{cases} 1 & \text{if} \quad f(i,j) \geq T \\ 0 & \text{if} \quad f(i,j) < T \end{cases}$$

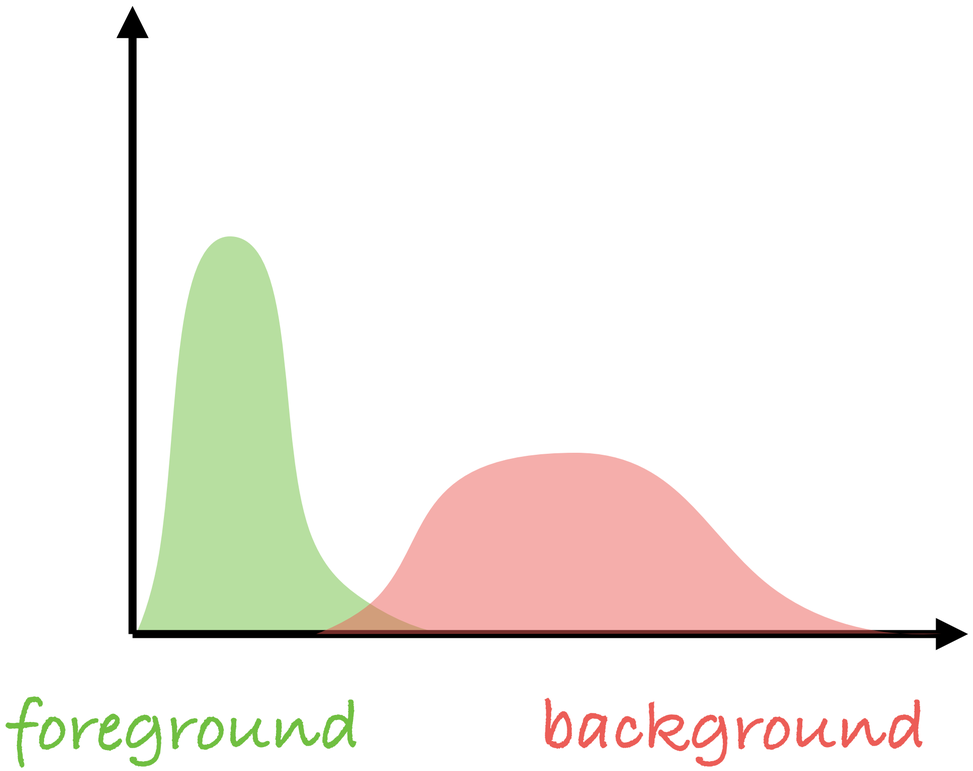
threshold

distinctly colored objects

simple background

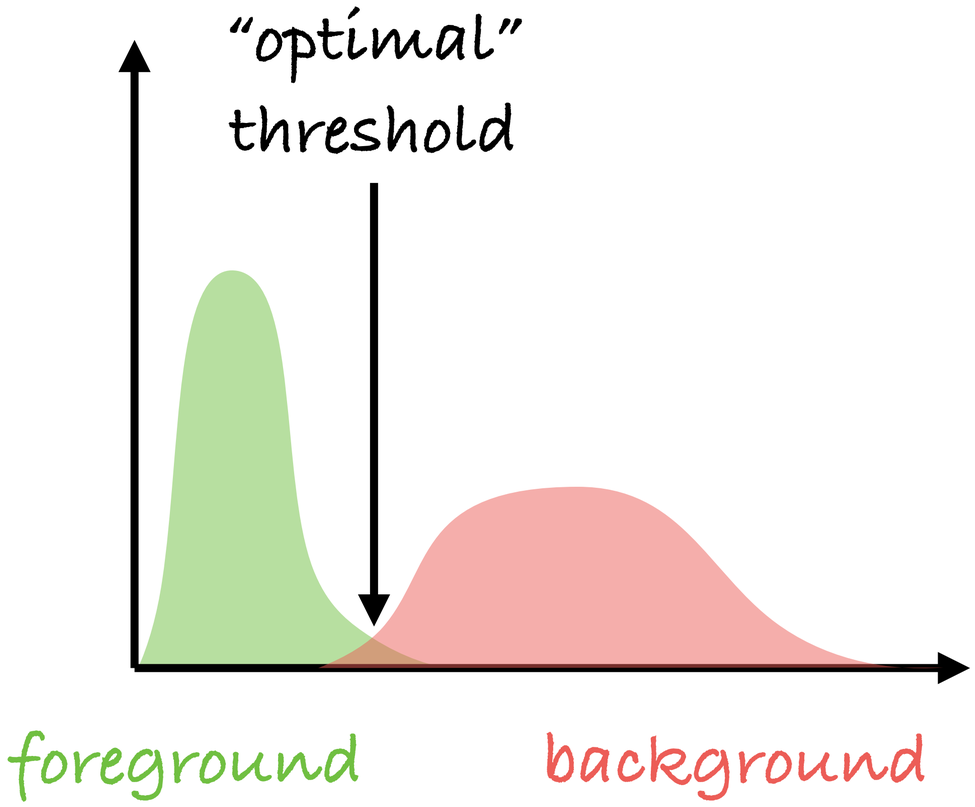
slide credit: Václav Hlaváč

Recap: “Optimal” Thresholding Via Mixture of Gaussians



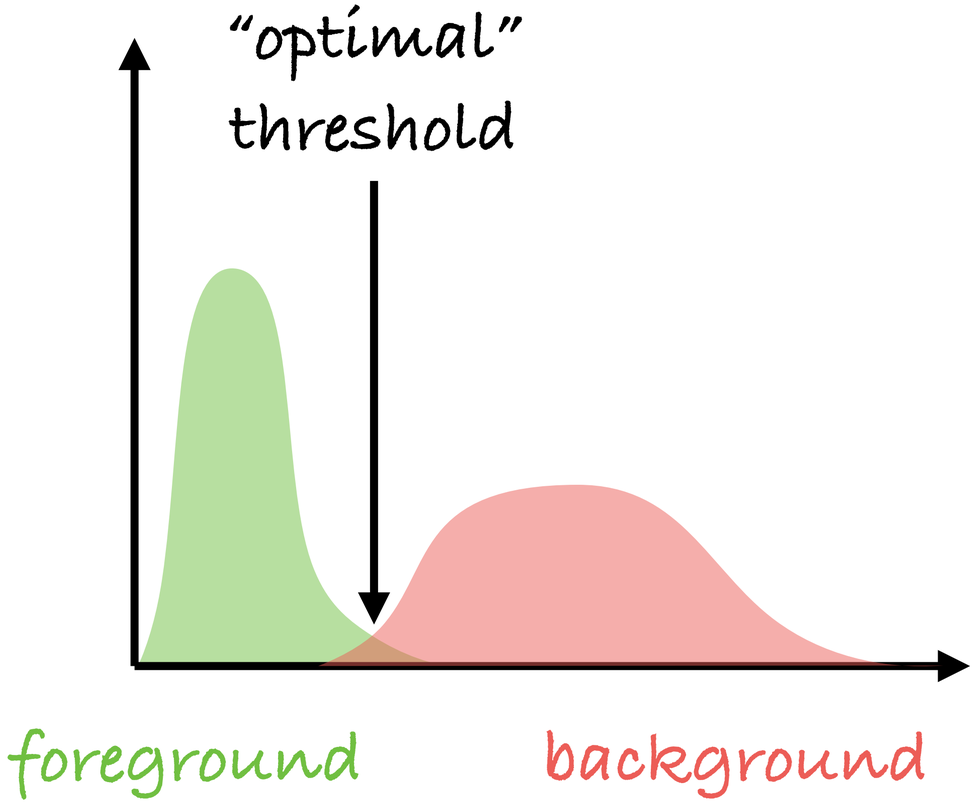
slide credit: Václav Hlaváč

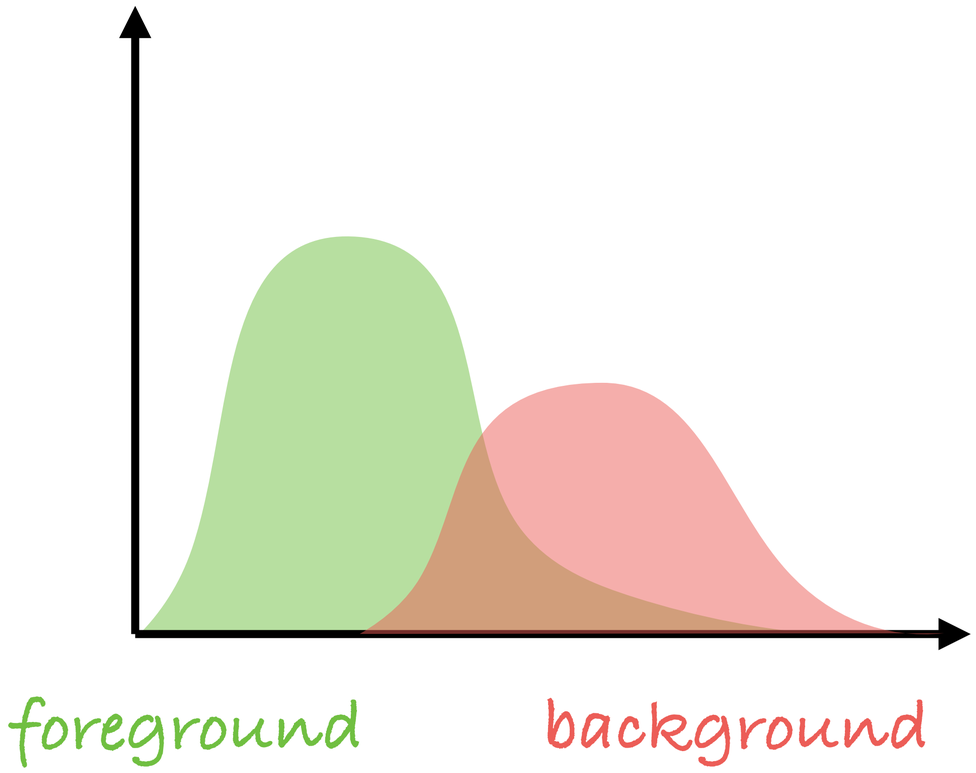
Recap: “Optimal” Thresholding Via Mixture of Gaussians



slide credit: Václav Hlaváč

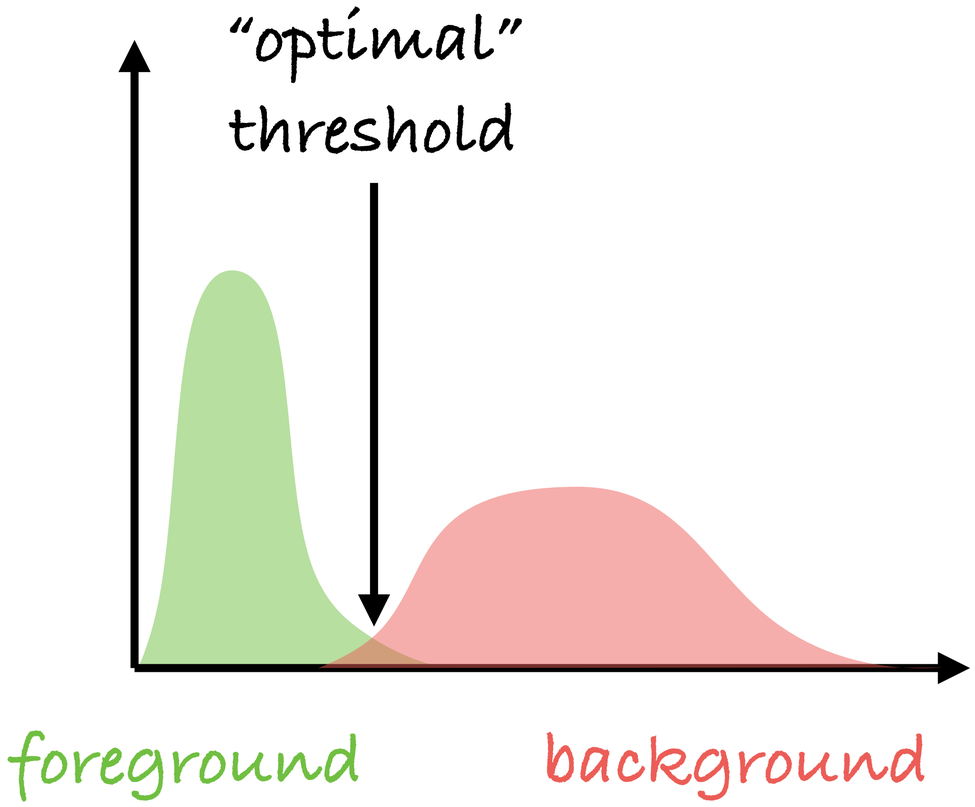
Recap: “Optimal” Thresholding Via Mixture of Gaussians

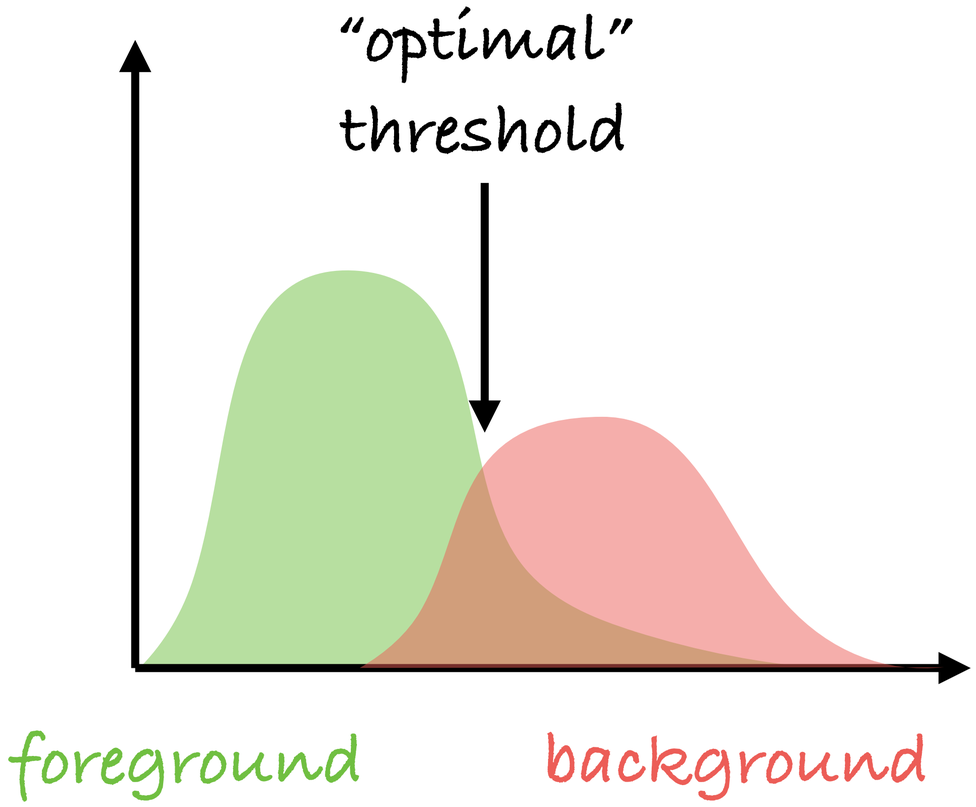




slide credit: Václav Hlaváč

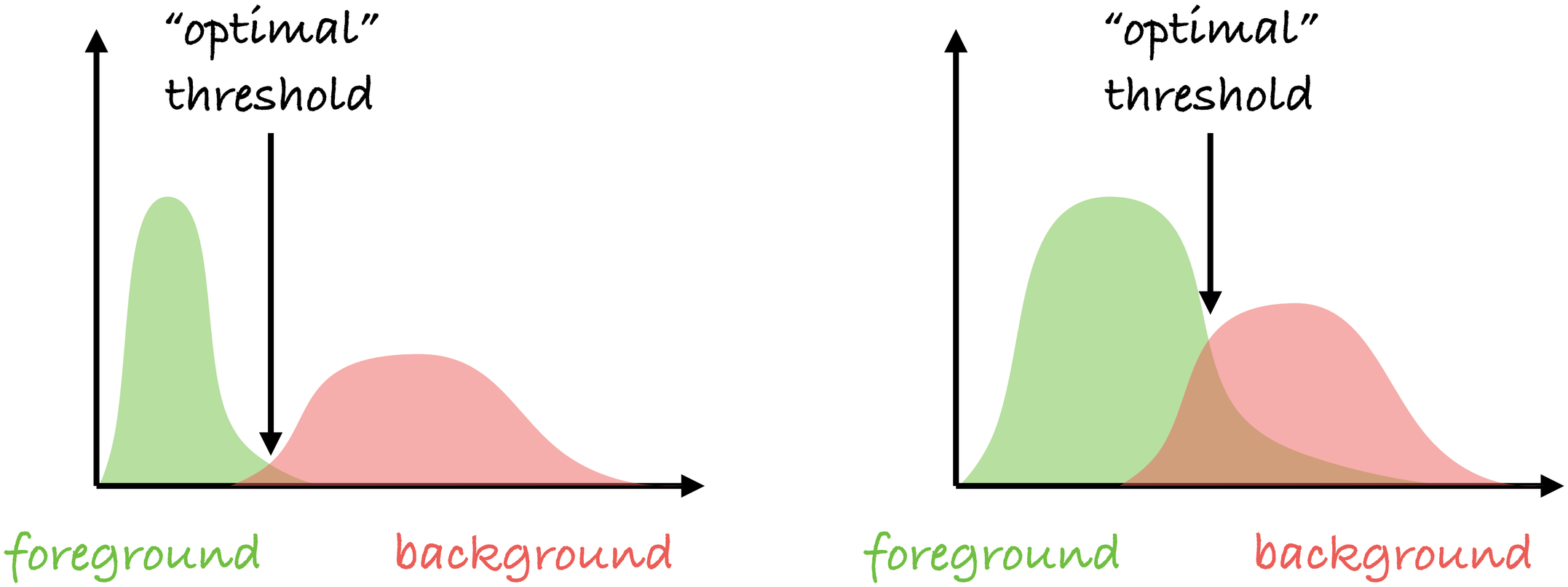
Recap: “Optimal” Thresholding Via Mixture of Gaussians





slide credit: Václav Hlaváč

Recap: “Optimal” Thresholding Via Mixture of Gaussians



Choose thresholds based on decision boundaries:

$$p(\text{foreground} \mid x) > p(\text{background} \mid x)$$

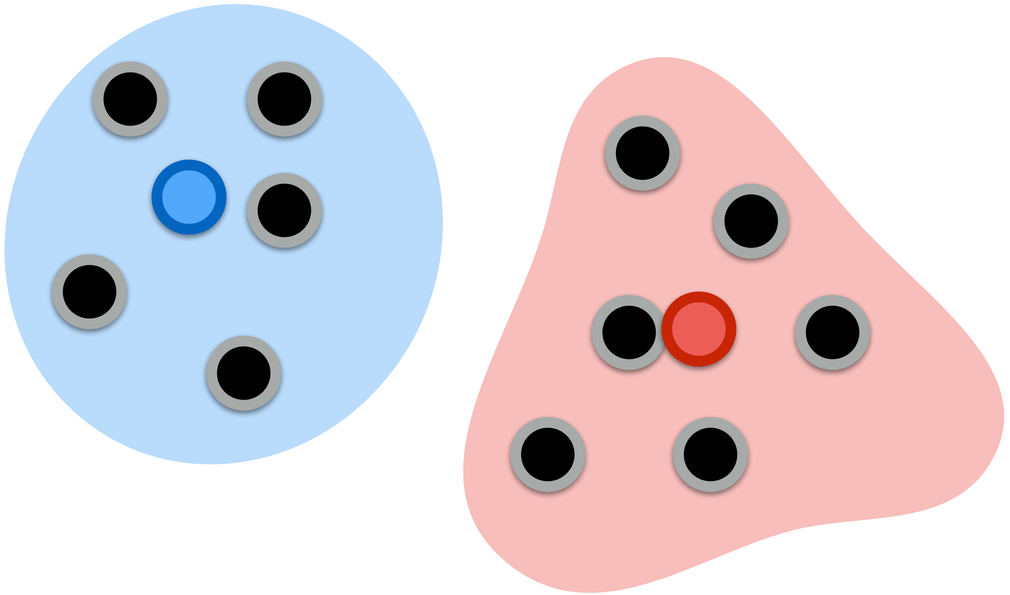
slide credit: Václav Hlaváč

Recap: k-Means Clustering

Randomly initialize k cluster centers

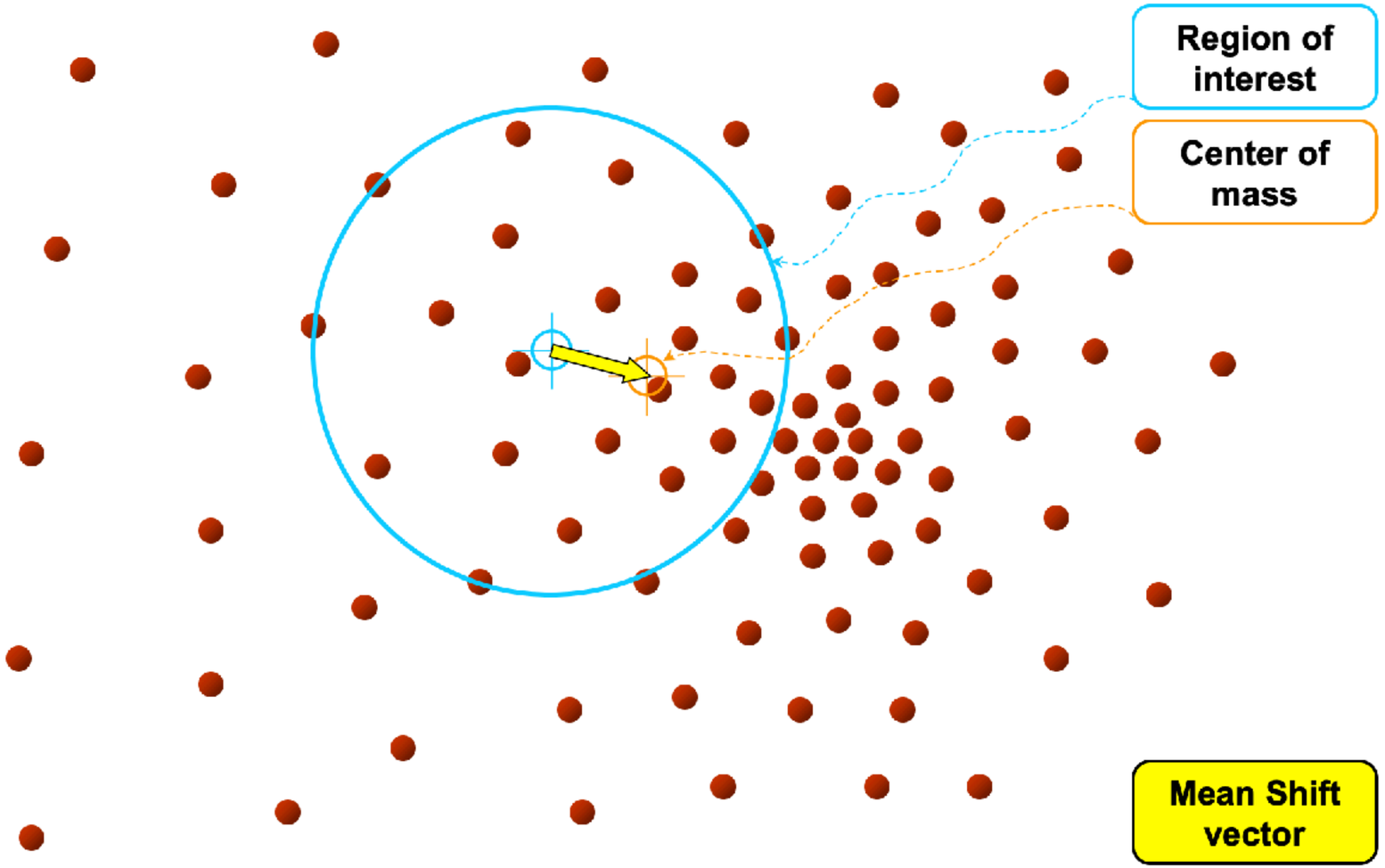
Repeat until assignments / centers do not change:

- Assign each point to the closest center
- Recompute centers as mean of all points assigned to centers



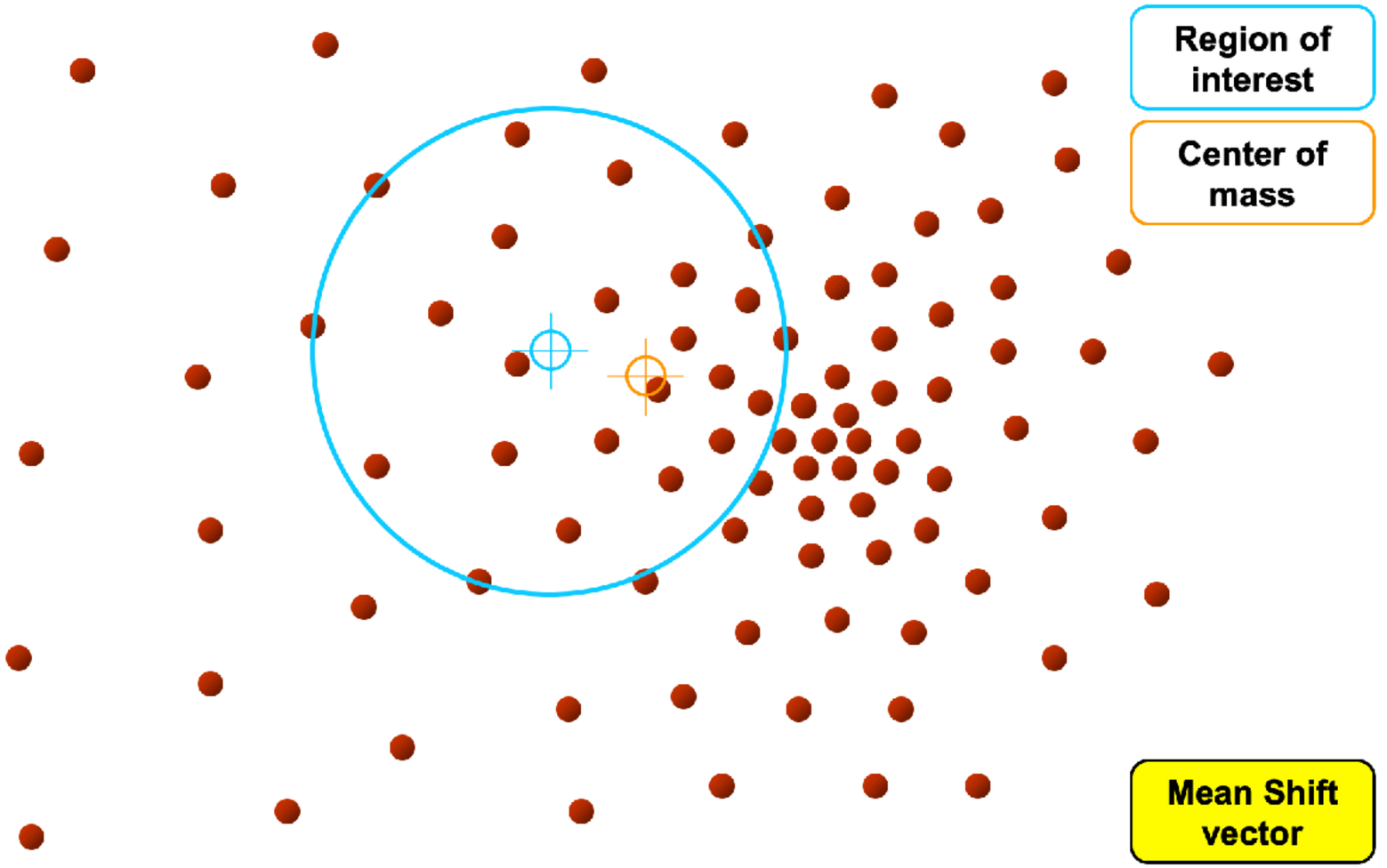
[Lloyd, Least square quantization in PCM's, Bell Telephone Laboratories Paper 1957]
[Lloyd, Least squares quantization in PCM, Spec. issue on quantiz., IEEE Trans. Inform. Theory, 28:129–137, 1982]

Recap: Mean Shift Algorithm



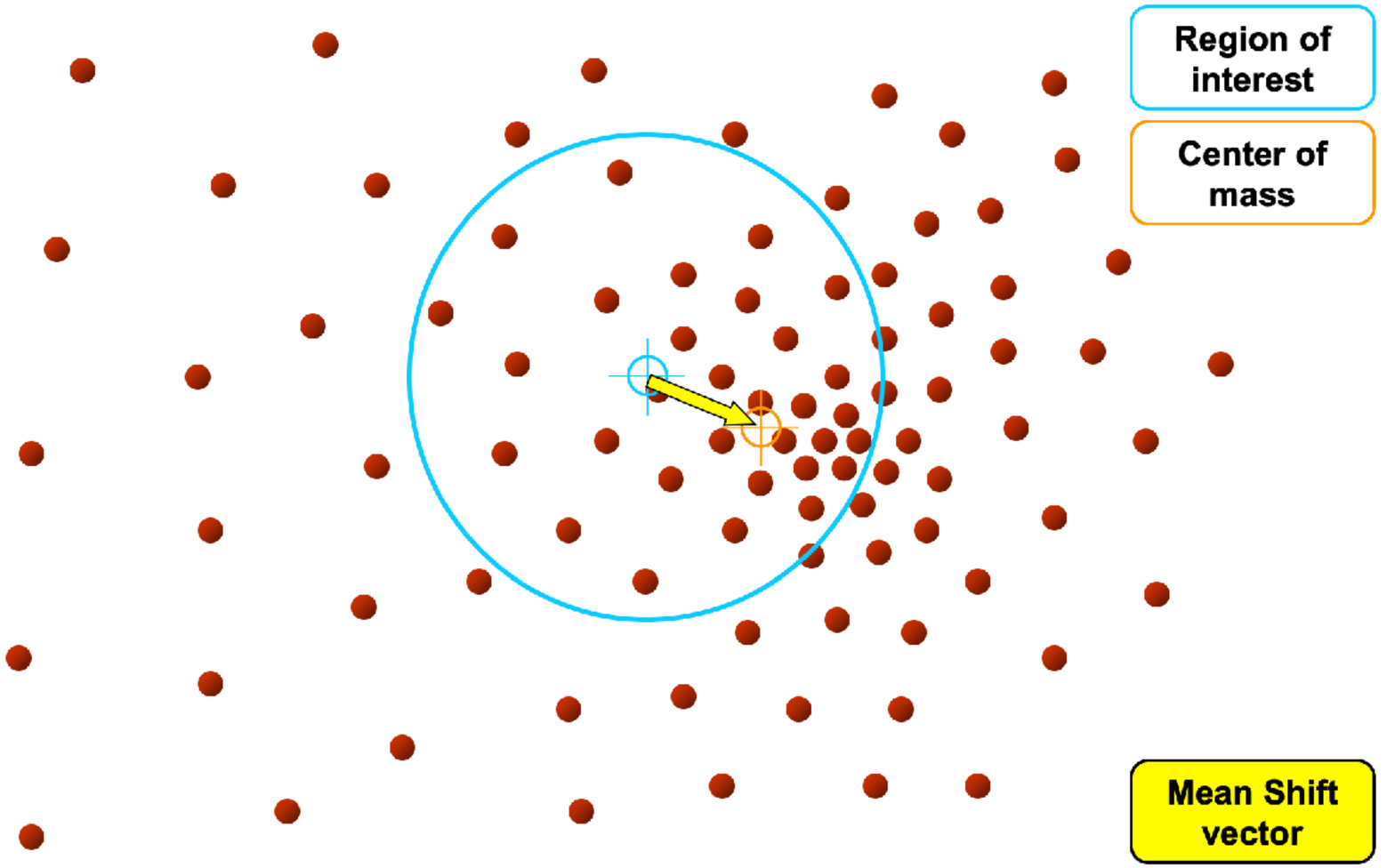
slide credit: Y. Ukrainitz & B. Sarel

Recap: Mean Shift Algorithm



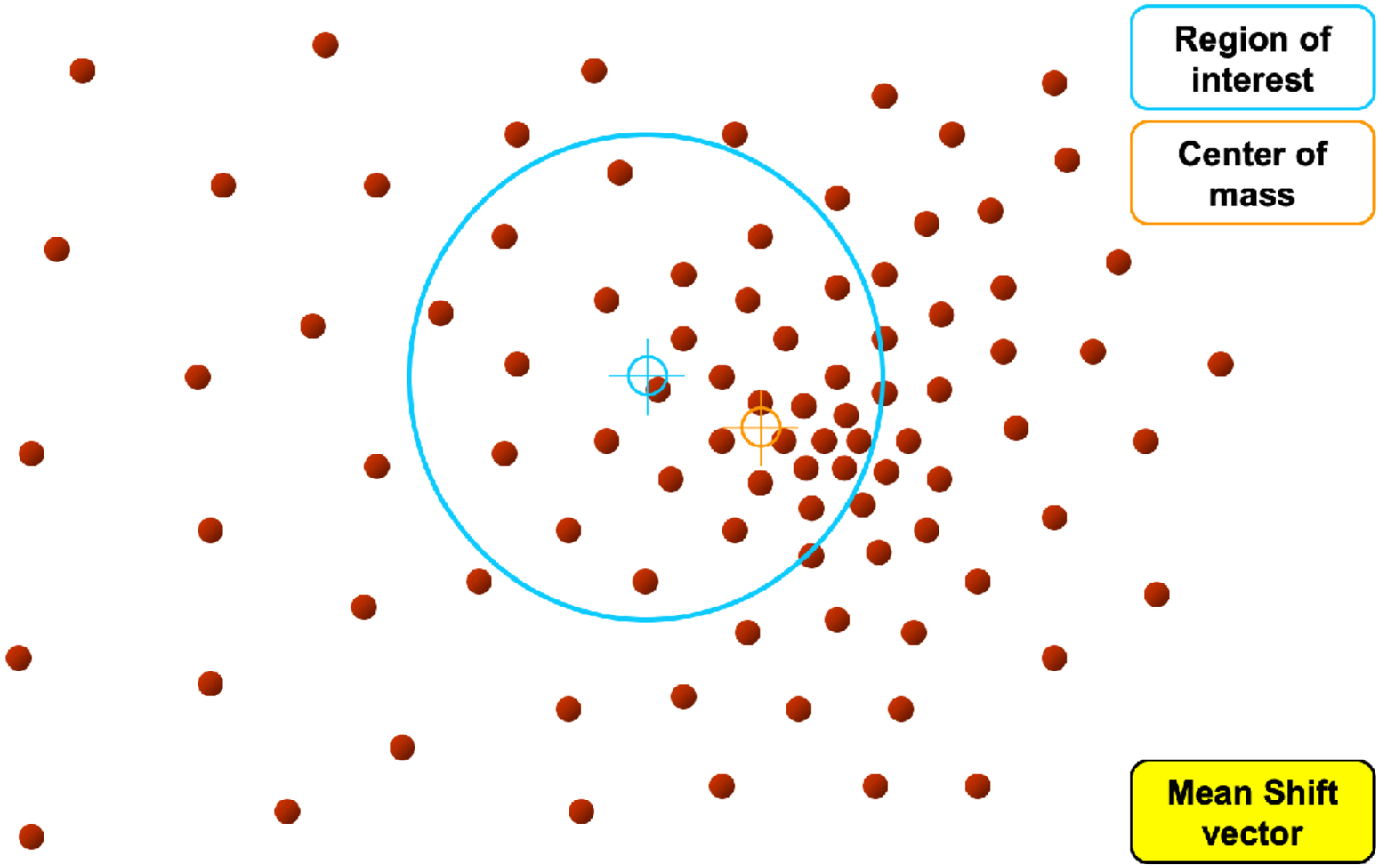
slide credit: Y. Ukrainitz & B. Sarel

Recap: Mean Shift Algorithm



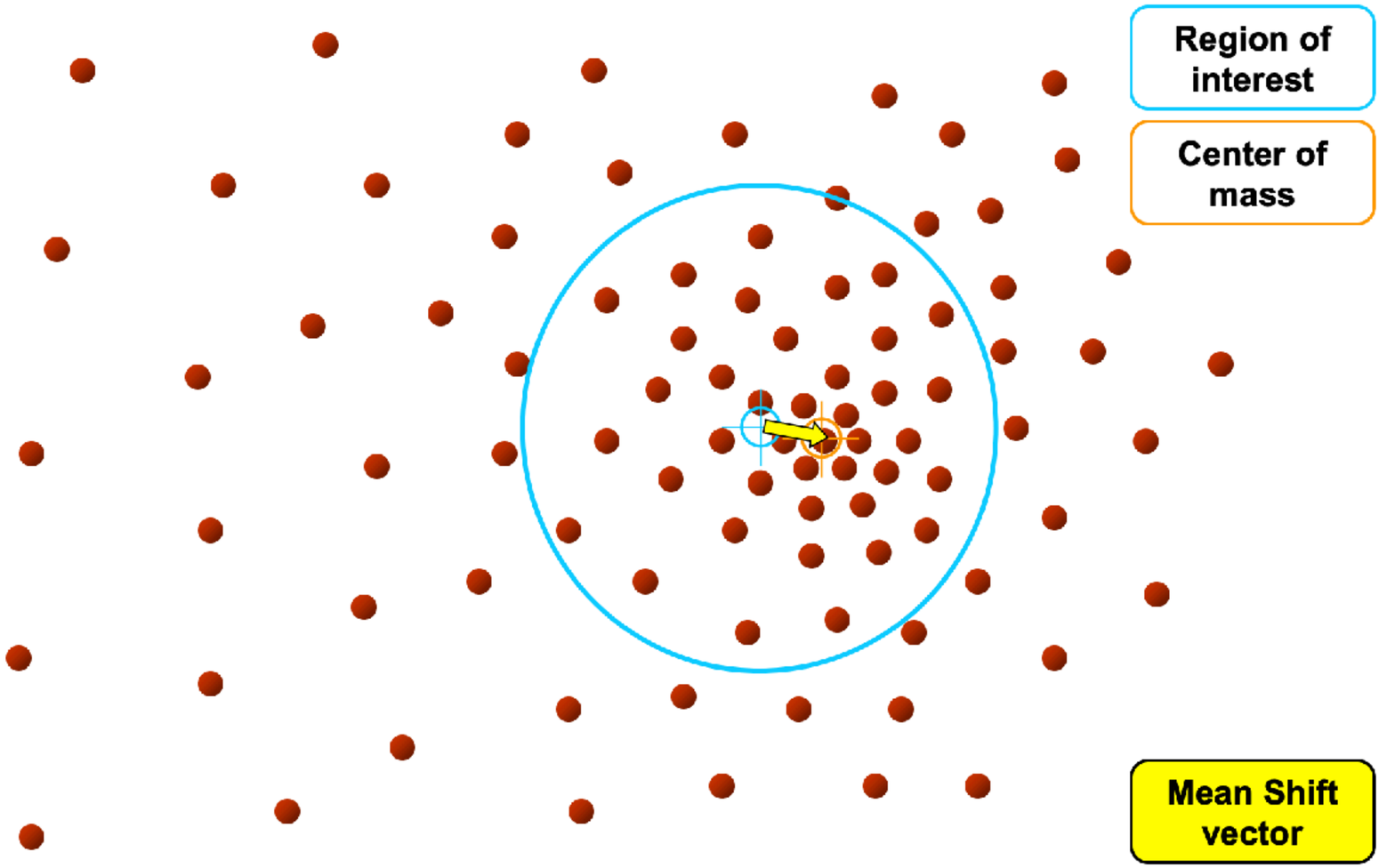
slide credit: Y. Ukrainitz & B. Sarel

Recap: Mean Shift Algorithm



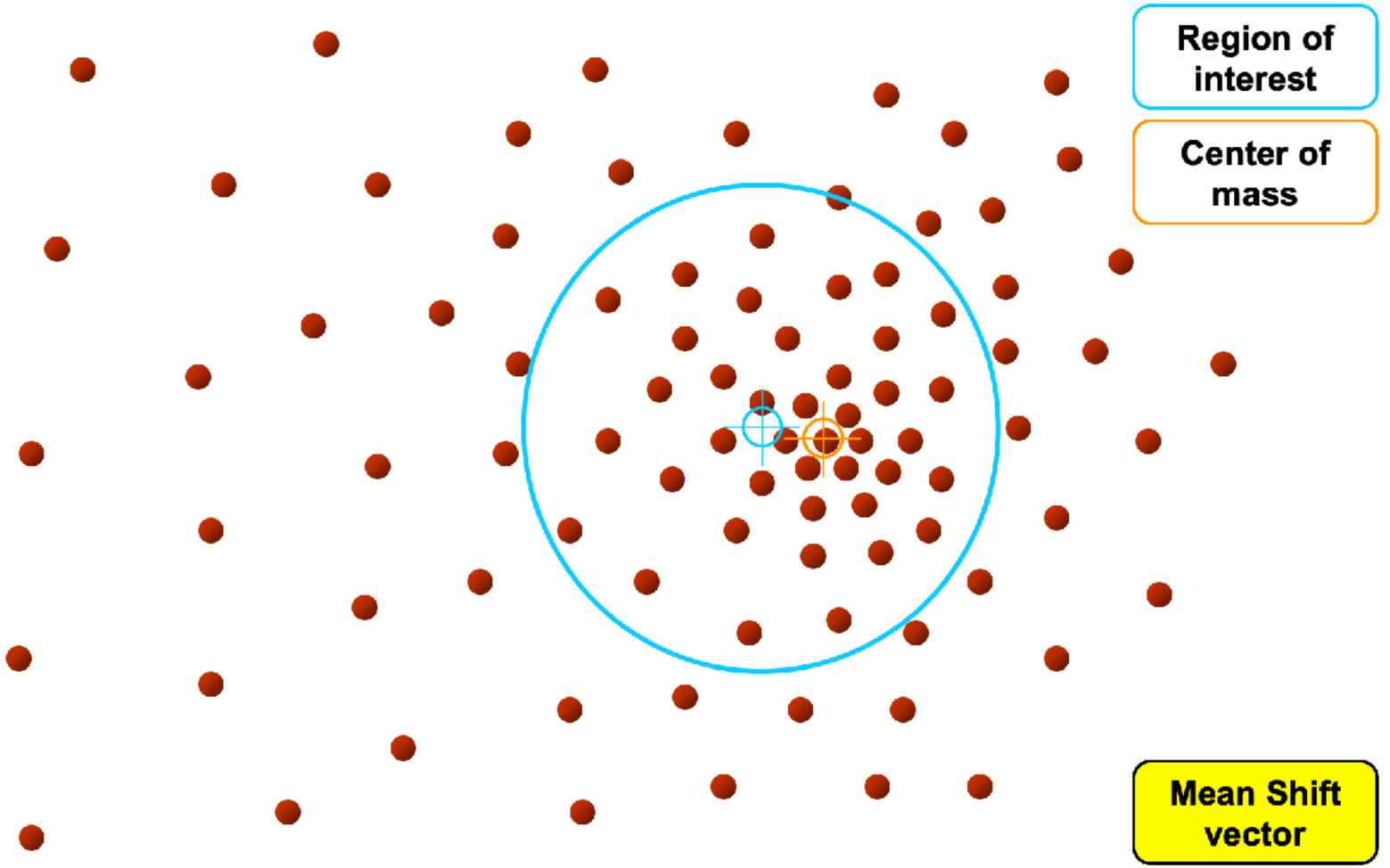
slide credit: Y. Ukrainitz & B. Sarel

Recap: Mean Shift Algorithm



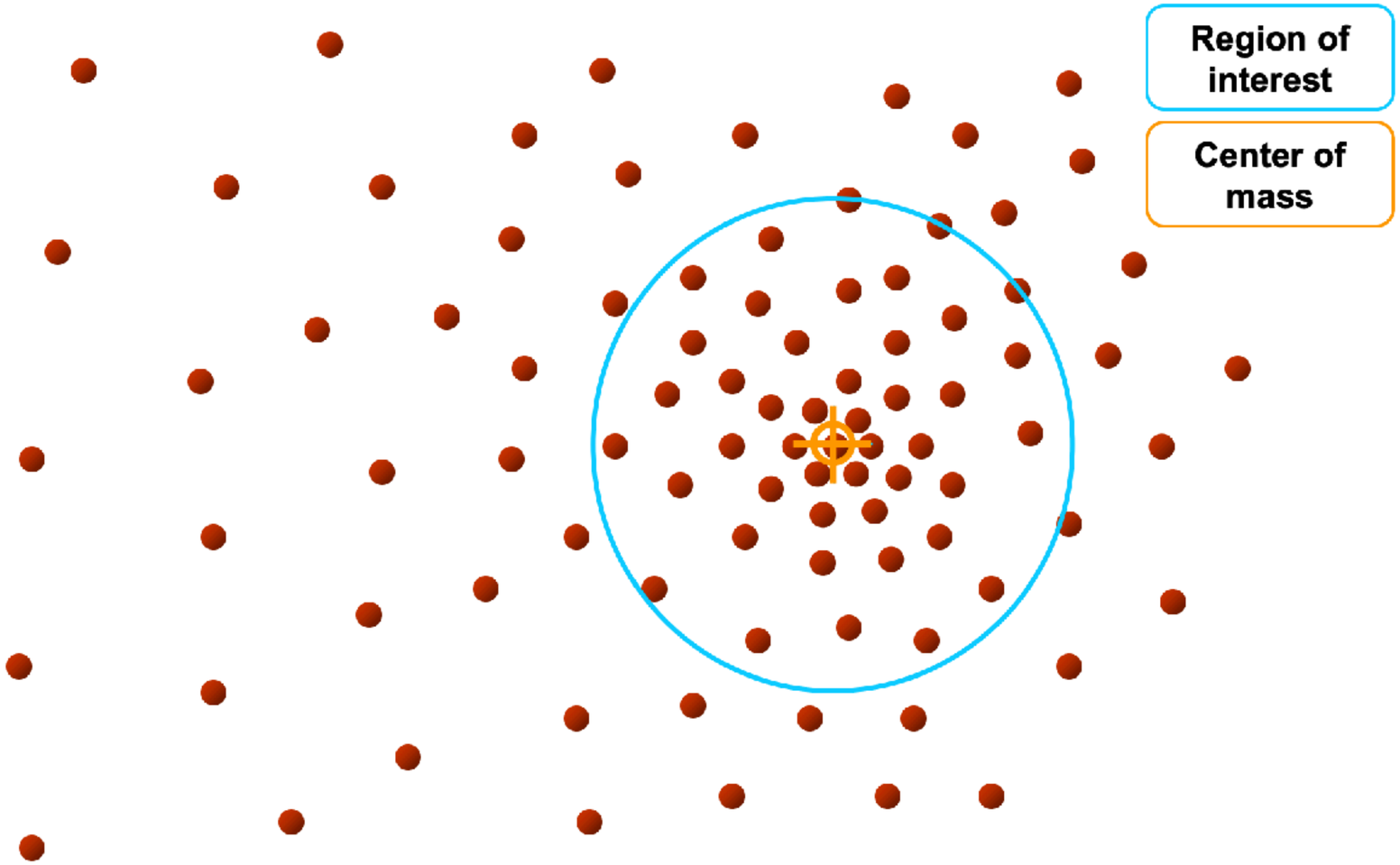
slide credit: Y. Ukrainitz & B. Sarel

Recap: Mean Shift Algorithm



slide credit: Y. Ukrainitz & B. Sarel

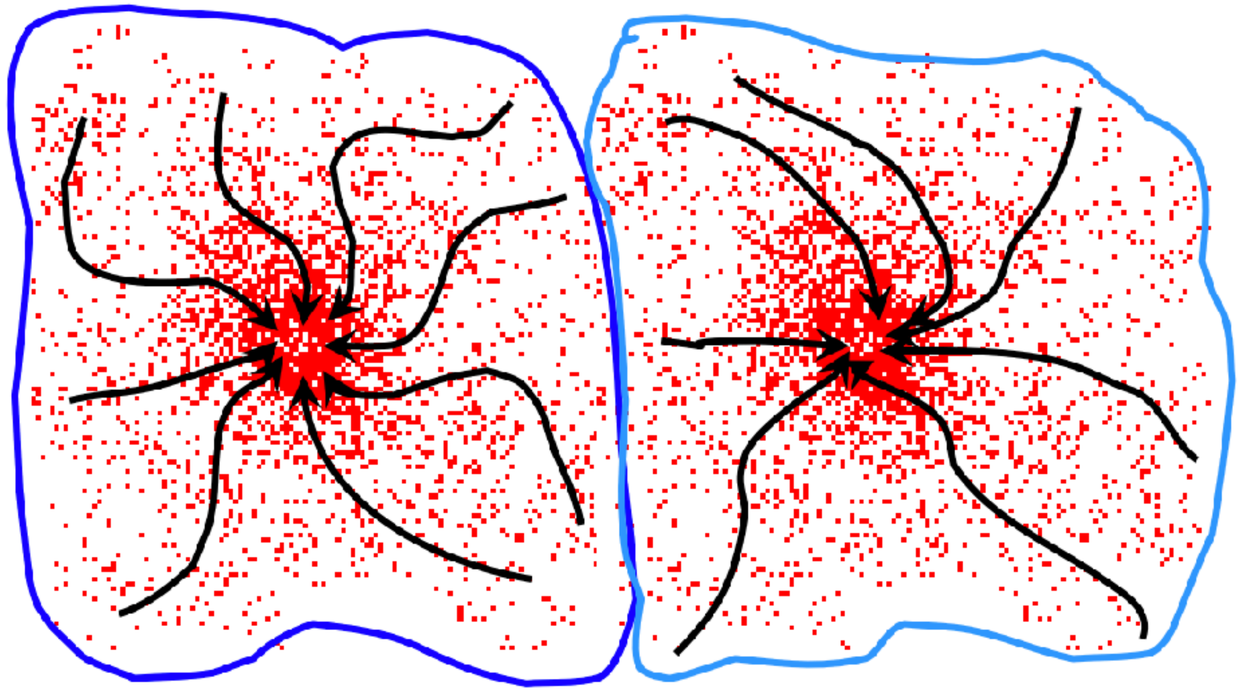
Recap: Mean Shift Algorithm



slide credit: Y. Ukrainitz & B. Sarel

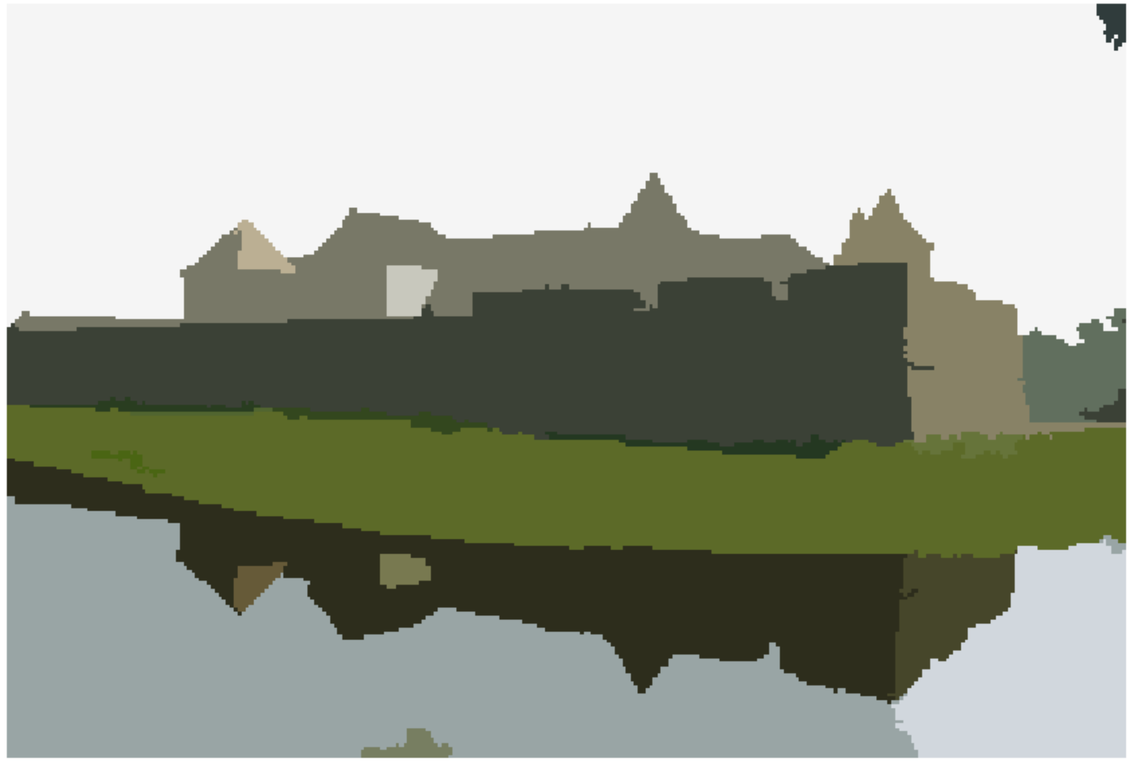
Recap: Mean Shift Clustering

- **Clusters**: all data points in attraction basin of mode
- **Attraction basin**: regions where mean shift leads to same mode



slide credit: Bastian Leibe, Y. Ukrainitz & B. Sarel

Examples



[D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002]

slide credit: Václav Hlaváč, Bastian Leibe, Svetlana Lazebnik

Lecture Overview - Today

simple & heuristic

complex & principled

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**
- Segmentation as a statistical (unsupervised) learning problem: **expectation maximization (EM) algorithm**
- **Graph-based segmentation**
- **Supervised learning with neural networks** (per video later)

slide credit: Václav Hlaváč

A Statistical Learning Perspective on Clustering

- Basic questions of practical relevance:
  - What is the shape of each cluster?
  - What is the probability a point *p* belongs to cluster *c*?

slide credit: Bastian Leibe

A Statistical Learning Perspective on Clustering

- Basic questions of practical relevance:
  - What is the shape of each cluster?
  - What is the probability a point *p* belongs to cluster *c*?
- k-means clustering cannot answer these questions

slide credit: Bastian Leibe

A Statistical Learning Perspective on Clustering

- Basic questions of practical relevance:
  - What is the shape of each cluster?
  - What is the probability a point *p* belongs to cluster *c*?
- k-means clustering cannot answer these questions

- **Statistical approach**:
  - There is a **generative model**: function relating observations $x \in X$ and their hidden state (class label) $y \in Y$
  - Described via the joint probability measure $p(x, y \mid \Theta)$ defined by parameters Θ
  - Want to **learn the parameters from data**

slide credit: Bastian Leibe

Supervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$

slide credit: Václav Hlaváč

Supervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Supervised learning**: given labelled training data $((x_1, y_1), \ldots, (x_n, y_n))$
  - Examples: random forests, deep neural networks

slide credit: Václav Hlaváč

Supervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Supervised learning**: given labelled training data $((x_1, y_1), \ldots, (x_n, y_n))$
  - Examples: random forests, deep neural networks

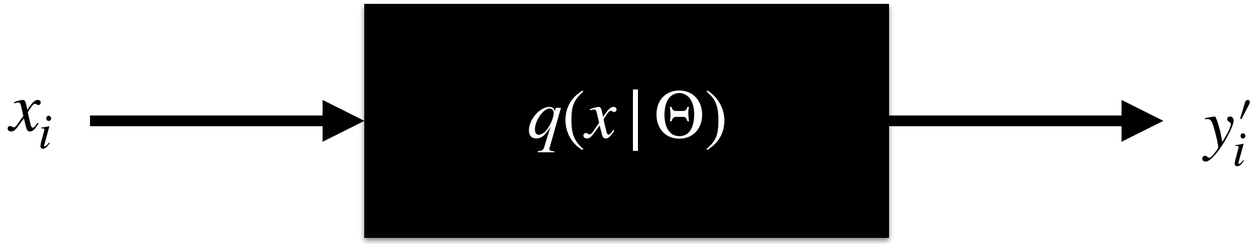
x_i

y_i

slide credit: Václav Hlaváč

Supervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Supervised learning**: given labelled training data $((x_1, y_1), \ldots, (x_n, y_n))$
  - Examples: random forests, deep neural networks

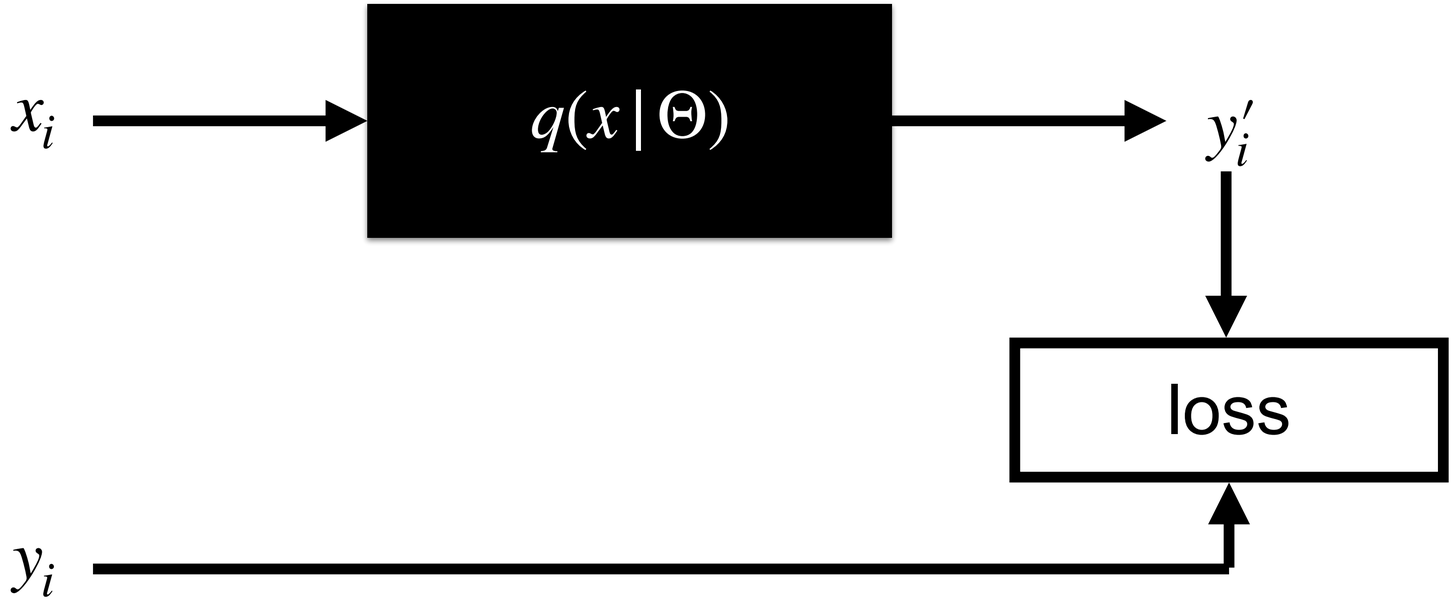


y_i

slide credit: Václav Hlaváč

Supervised Learning

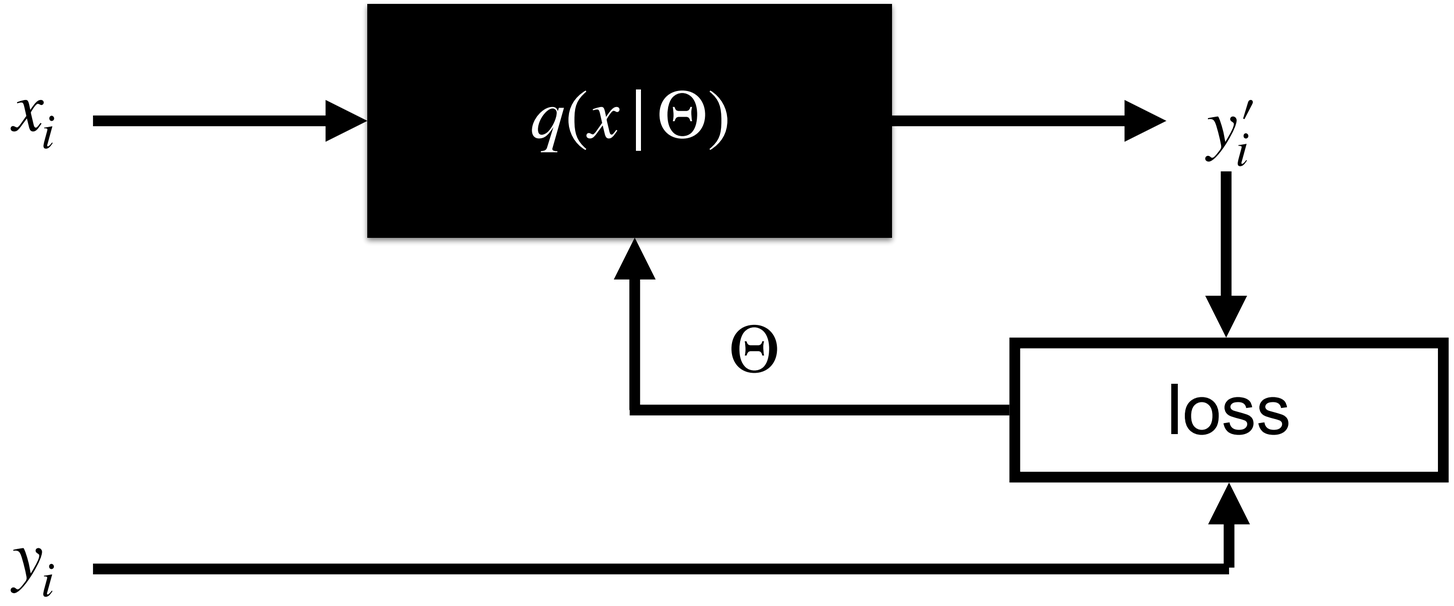
- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Supervised learning**: given labelled training data $((x_1, y_1), \ldots, (x_n, y_n))$
  - Examples: random forests, deep neural networks



slide credit: Václav Hlaváč

Supervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function
  $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Supervised learning**: given labelled training data $((x_1, y_1), \ldots, (x_n, y_n))$
  - Examples: random forests, deep neural networks



slide credit: Václav Hlaváč

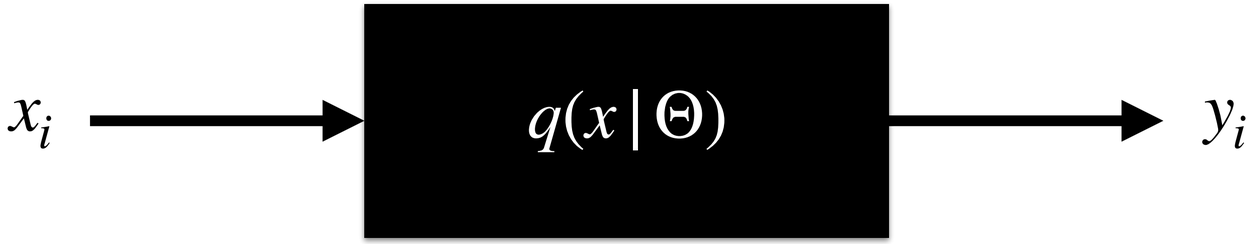
Unsupervised Learning

- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$

slide credit: Václav Hlaváč

Unsupervised Learning

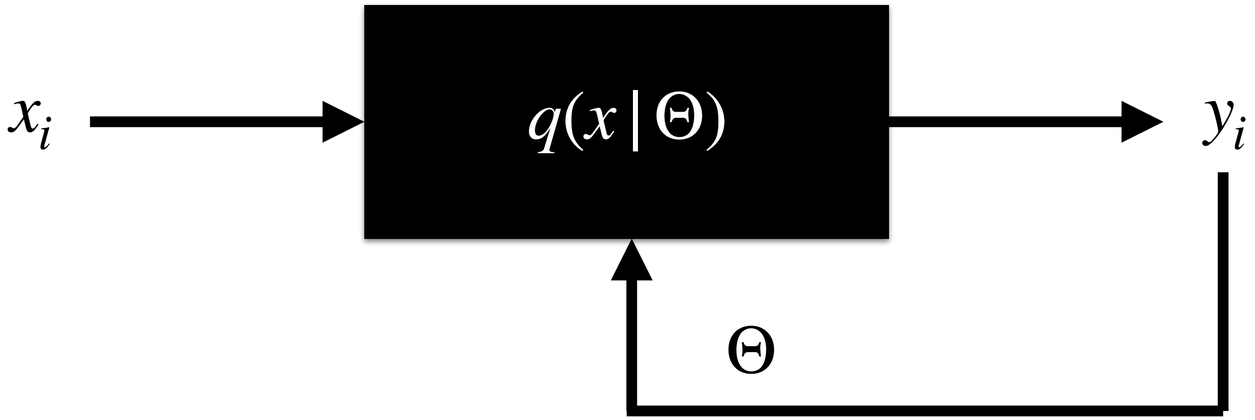
- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$



slide credit: Václav Hlaváč

Unsupervised Learning

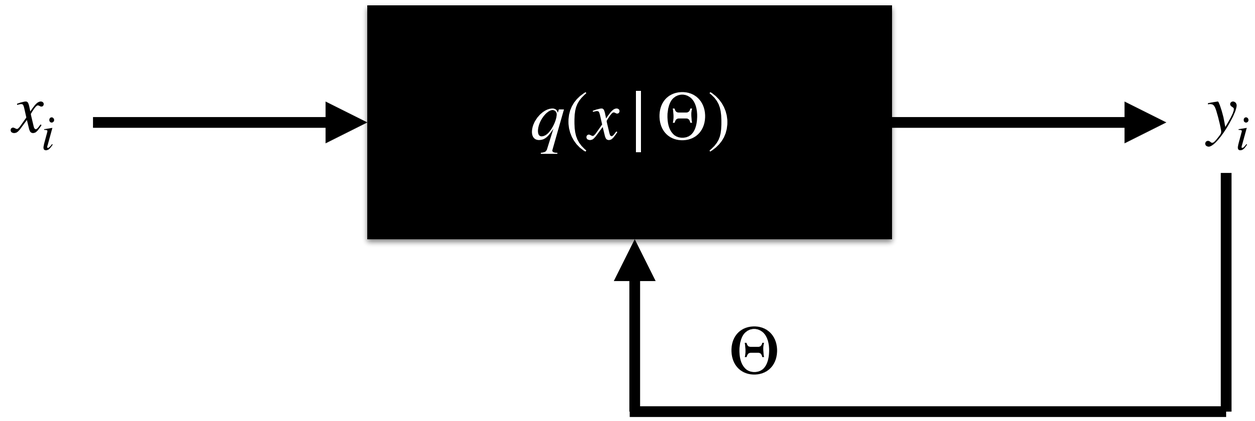
- If we have $p(x, y \mid \Theta)$, we can define classifier / decision function $y = q(x \mid \Theta) = \text{argmax}_y \, p(x, y \mid \Theta)$
- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$



slide credit: Václav Hlaváč

Unsupervised Learning

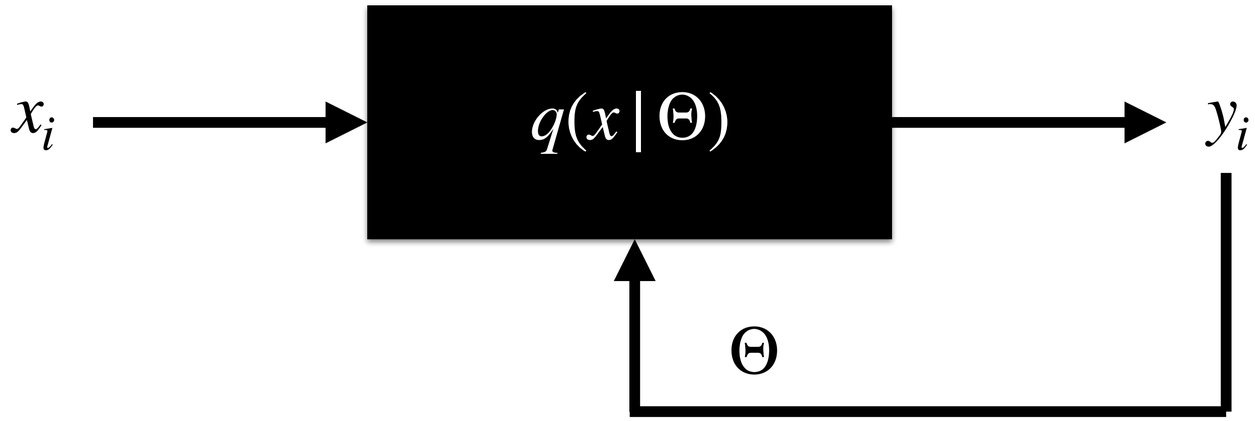
- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$



slide credit: Václav Hlaváč, Bastian Leibe

Unsupervised Learning

- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$

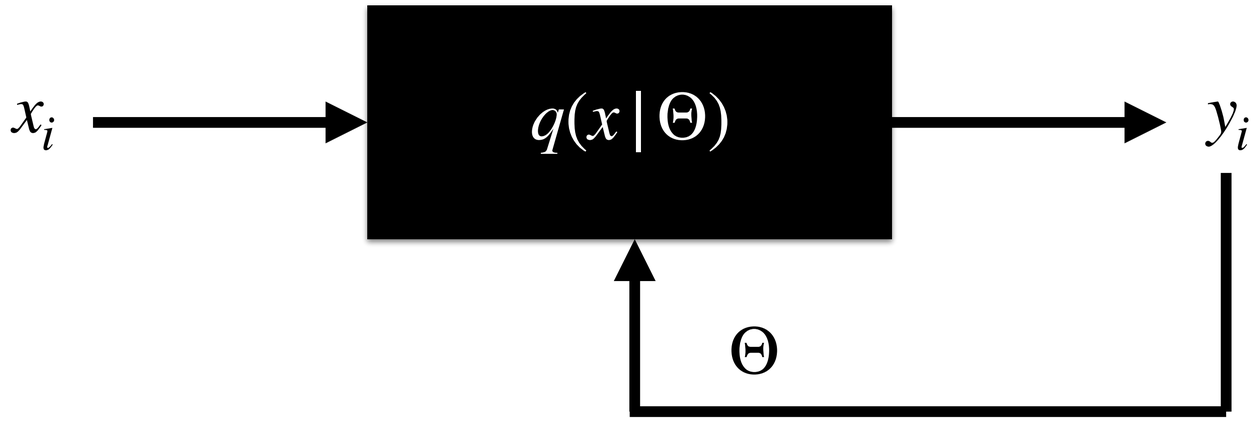


- **Chicken-and-egg problem**: if we have Θ, we can compute $y = q(x \mid \Theta)$; if we have y, we can compute Θ

slide credit: Václav Hlaváč, Bastian Leibe

Unsupervised Learning

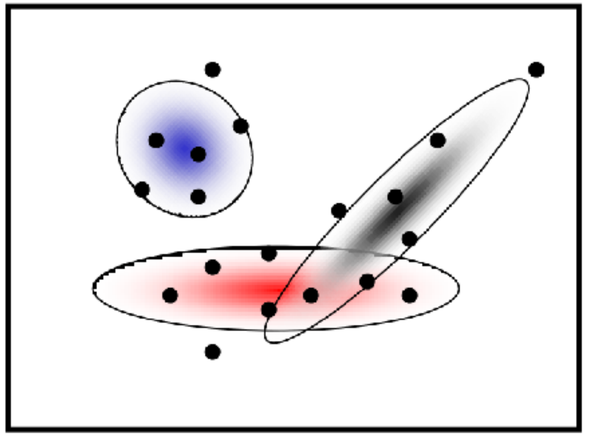
- **Unsupervised learning**: given unlabelled training data $(x_1, \ldots, x_n)$



- **Chicken-and-egg problem**: if we have Θ, we can compute $y = q(x \mid \Theta)$; if we have y, we can compute Θ
- Sounds familiar?

slide credit: Václav Hlaváč, Bastian Leibe

Mixture of Gaussians



- Mixture of Gaussians is one generative model:
  - K Gaussian blobs with means μ_j, cov. matrices Σ_j, dimensionality D

$$p(x \mid \Theta_j) = \frac{1}{(2\pi)^{D/2} \mid \Sigma_j \mid^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\right)$$

  - Gaussian j selected with probability π_j
  - Likelihood of observing data point **x** is weighted mixture of Gaussians:

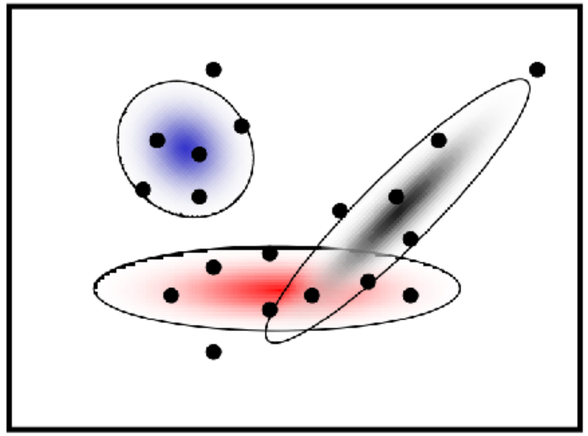
$$p(x \mid \Theta) = \sum_{j=1}^{K} \pi_j p(x \mid \Theta_j) \quad \Theta = \left(\pi_1, \mu_1, \Sigma_1, \ldots, \pi_K, \mu_K, \Sigma_K\right)$$

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

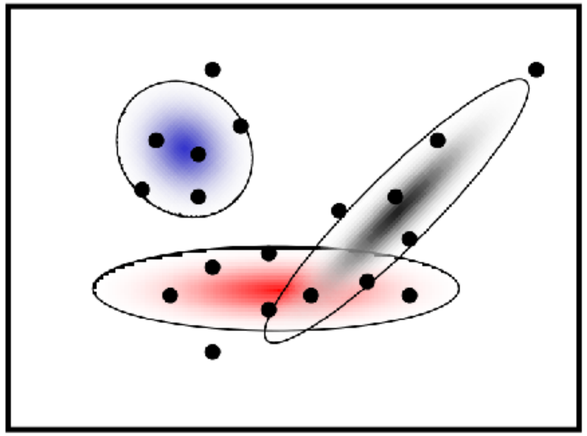
- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$



slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm



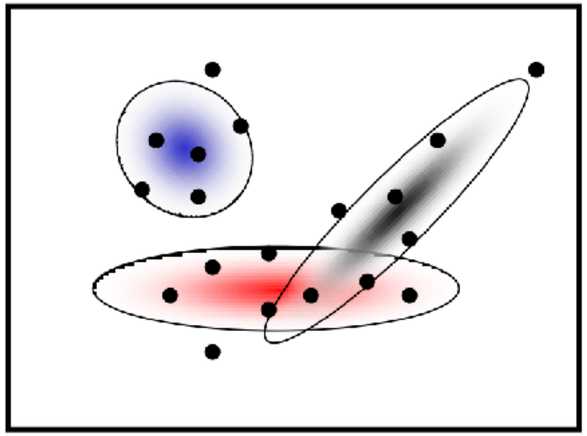
- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$

- **Expectation Maximization** (EM) approach:

slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm



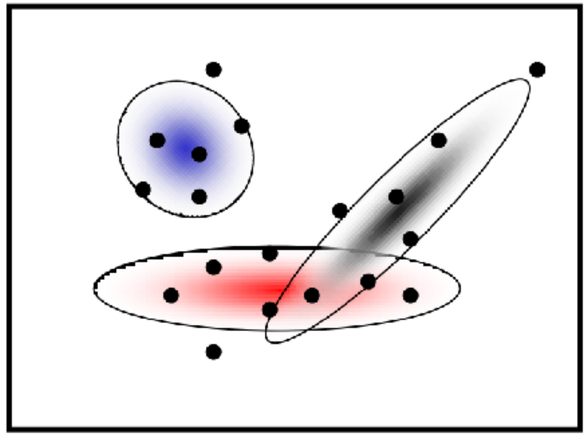
- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$

- **Expectation Maximization** (EM) approach:
  - Obtain initial estimate Θ^0 for Θ

slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm



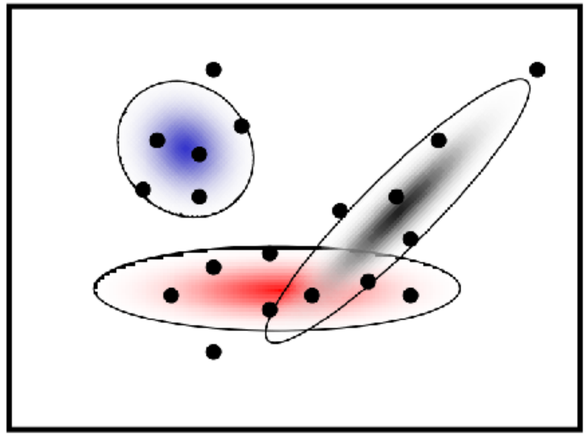
- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$

- **Expectation Maximization** (EM) approach:
  - Obtain initial estimate Θ^0 for Θ
  - Repeat:

slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm



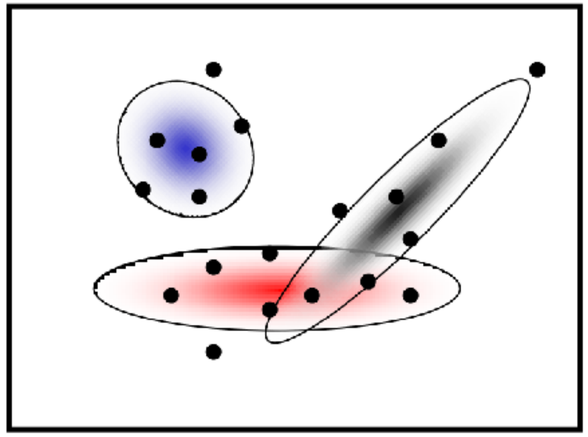
- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$

- **Expectation Maximization** (EM) approach:
  - Obtain initial estimate Θ^0 for Θ
  - Repeat:
    - **E-step**: given Θ^i assign data points to Gaussians

slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm



- Goal: find parameters Θ that maximize likelihood function:

$$p(\text{data} \mid \Theta) = \prod_{i=1}^{n} p(x_i \mid \Theta)$$

- **Expectation Maximization** (EM) approach:
  - Obtain initial estimate Θ^0 for Θ
  - Repeat:
    - **E-step**: given Θ^i assign data points to Gaussians
    - **M-step**: given assignments, estimate Θ^{i+1} by maximizing likelihood function

slide credit: Bastian Leibe, Steve Seitz

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

- **M-step**: use soft assignments to re-estimate parameters Θ_j per component:

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

- **M-step**: use soft assignments to re-estimate parameters Θ_j per component:

$$N_j = \sum_{i=1}^{n} \gamma_j(x_i) = \text{soft number of points assigned to cluster } j$$

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

- **M-step**: use soft assignments to re-estimate parameters Θ_j per component:

$$N_j = \sum_{i=1}^{n} \gamma_j(x_i) = \text{soft number of points assigned to cluster } j$$

$$\pi_j^{\text{new}} = N_j / n = \text{probability of component } j$$

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

- **M-step**: use soft assignments to re-estimate parameters Θ_j per component:

$$N_j = \sum_{i=1}^{n} \gamma_j(x_i) = \text{soft number of points assigned to cluster } j$$

$$\pi_j^{\text{new}} = N_j / n = \text{probability of component } j$$

$$\mu_j^{\text{new}} = \frac{1}{N_j} \sum_{i=1}^{n} \gamma_j(x_i) x_i = \text{new cluster center}$$

slide credit: Bastian Leibe

Expectation Maximization (EM) Algorithm

- **E-step**: compute soft assignment of data points to mixture components:

$$\gamma_j(x_i) = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

- **M-step**: use soft assignments to re-estimate parameters Θ_j per component:

$$N_j = \sum_{i=1}^{n} \gamma_j(x_i) = \text{soft number of points assigned to cluster } j$$

$$\pi_j^{\text{new}} = N_j / n = \text{probability of component } j$$

$$\mu_j^{\text{new}} = \frac{1}{N_j} \sum_{i=1}^{n} \gamma_j(x_i) x_i = \text{new cluster center} \qquad \Sigma_j^{\text{new}} = \frac{1}{N_j} \sum_{i=1}^{n} \gamma_j(x_i)(x_i - \mu_j^{\text{new}})^T (x_i - \mu_j^{\text{new}})$$

slide credit: Bastian Leibe

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i | \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2} (x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances
- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i \mid \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2}(x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

- **M-step**: update cluster centers $\Theta = (\mu_1, \ldots, \mu_K)$:

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i | \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2} (x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

- **M-step**: update cluster centers $\Theta = (\mu_1, \ldots, \mu_K)$:

$$\Theta^* = \text{argmax}_{\Theta} \sum_{i=1}^{n} \log(p(x_i | \Theta_{k_i})$$

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i | \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2} (x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

- **M-step**: update cluster centers $\Theta = (\mu_1, \ldots, \mu_K)$:

$$\Theta^* = \text{argmax}_{\Theta} \sum_{i=1}^{n} \log(p(x_i | \Theta_{k_i}) = \text{argmin}_{\mu_1, \ldots, \mu_k} \sum_{i=1}^{n} \|x_i - \mu_{k_i}\|^2$$

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i \mid \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2} (x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

- **M-step**: update cluster centers $\Theta = (\mu_1, \ldots, \mu_K)$:

$$\Theta^* = \text{argmax}_{\Theta} \sum_{i=1}^{n} \log(p(x_i \mid \Theta_{k_i}) = \text{argmin}_{\mu_1, \ldots, \mu_k} \sum_{i=1}^{n} \|x_i - \mu_{k_i}\|^2$$

$$= \text{argmin}_{\mu_1} \sum_{\{i \mid k_i = 1\}} \|x_i - \mu_1\|^2, \ldots, \text{argmin}_{\mu_K} \sum_{\{i \mid k_i = K\}} \|x_i - \mu_K\|^2$$

slide credit: Václav Hlaváč

k-Means as Expectation Maximization

- **k-means clustering** is a special case of EM algorithm
- Gaussian mixture model with unit covariances

- **E-step**: hard assignments to component:

$$k_i = \text{argmax}_{k'} p(x_i \mid \Theta_{k'}) = \text{argmax}_{k'} \log \left(\exp \left(-\frac{1}{2}(x_i - \mu_{k'})^T (x_i - \mu_{k'}) \right) \right) = \text{argmin}_{k'} \|x_i - \mu_{k'}\|^2$$

- **M-step**: update cluster centers $\Theta = (\mu_1, \ldots, \mu_K)$:

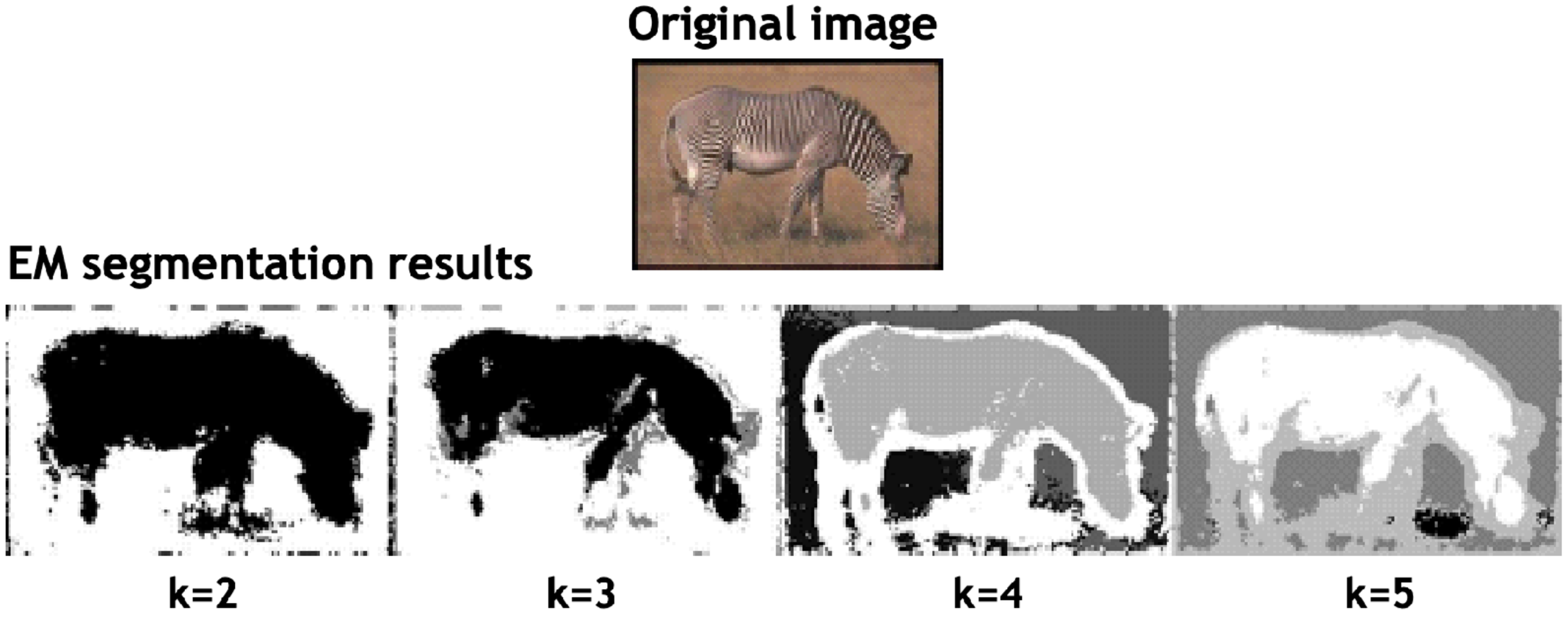
$$\Theta^* = \text{argmax}_\Theta \sum_{i=1}^{n} \log(p(x_i \mid \Theta_{k_i}) = \text{argmin}_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \|x_i - \mu_{k_i}\|^2$$

$$= \text{argmin}_{\mu_1} \sum_{\{i \mid k_i = 1\}} \|x_i - \mu_1\|^2, \ldots, \text{argmin}_{\mu_K} \sum_{\{i \mid k_i = K\}} \|x_i - \mu_K\|^2$$

$$\Rightarrow \mu_j = \frac{1}{|\{i \mid k_i = j\}|} \sum_{\{i \mid k_i = j\}} x_i, \quad j = 1, \ldots, K$$

slide credit: Václav Hlaváč

EM Algorithm for Segmentation



slide credit: Bastian Leibe

image credit: Serge Belongie

The EM Algorithm

- General statistical approach for missing data / data with hidden states

slide credit: Václav Hlaváč

The EM Algorithm

- General statistical approach for missing data / data with hidden states
- Can be used to obtain **Maximum Likelihood Estimate** (MLE) even if MLE cannot be computed directly

slide credit: Václav Hlaváč

The EM Algorithm

- General statistical approach for missing data / data with hidden states
- Can be used to obtain **Maximum Likelihood Estimate** (MLE) even if MLE cannot be computed directly
- **General concept**:
  - Marginalize over hidden states $y \in Y$: $p(x \mid \Theta) = \sum_{y} p(x, y \mid \Theta)$
  - Simplify estimation of $p(x \mid \Theta)$ by inferring hidden states

slide credit: Václav Hlaváč

The EM Algorithm

- General statistical approach for missing data / data with hidden states
- Can be used to obtain **Maximum Likelihood Estimate** (MLE) even if MLE cannot be computed directly
- **General concept**:
  - Marginalize over hidden states $y \in Y$: $p(x|\Theta) = \sum_y p(x, y|\Theta)$
  - Simplify estimation of $p(x|\Theta)$ by inferring hidden states
- **Guaranteed to converge** as cost function decreases monotonically (proof via special form of Jensen's inequality)

slide credit: Václav Hlaváč

The EM Algorithm

- General statistical approach for missing data / data with hidden states
- Can be used to obtain **Maximum Likelihood Estimate** (MLE) even if MLE cannot be computed directly
- **General concept**:
  - Marginalize over hidden states $y \in Y$: $p(x|\Theta) = \sum_y p(x, y|\Theta)$
  - Simplify estimation of $p(x|\Theta)$ by inferring hidden states
- **Guaranteed to converge** as cost function decreases monotonically (proof via special form of Jensen's inequality)
- **No general guarantees about global optimality**, EM is essentially gradient ascent

slide credit: Václav Hlaváč

The EM Algorithm

- **Pros**:
  - Probabilistic interpretation of data
  - Soft assignments instead of hard assignments
  - Generative model: can predict new datapoint

slide credit: Bastian Leibe

The EM Algorithm

- **Pros**:
  - Probabilistic interpretation of data
  - Soft assignments instead of hard assignments
  - Generative model: can predict new datapoint
- **Cons**:
  - Local optimization will lead to local minima
  - Initialization is very important (e.g., use multiple k-means runs and pick the best run as initialization)
  - Singular clusters can be a problem
  - Similar to k-means clustering: need estimate for K
  - Need to choose proper generative model
  - Numerical instabilities can be an issue

slide credit: Bastian Leibe

Lecture Overview - Today

simple & heuristic

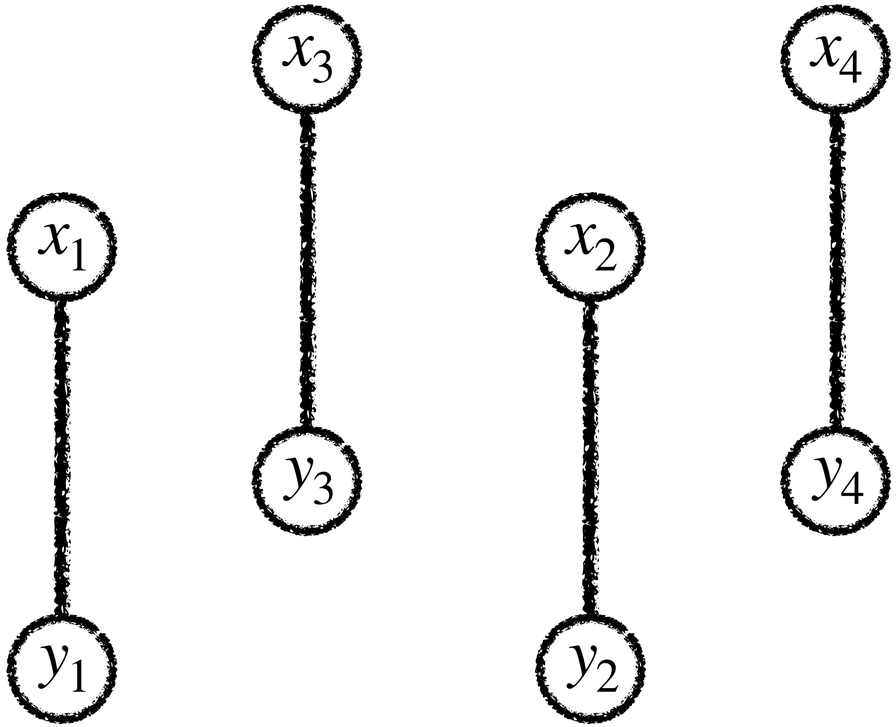
complex & principled

- A simple approach to segmentation: **(intensity) thresholding**
- Segmentation based on spatial coherence: **edge-based segmentation**, **region growing**
- Segmentation as a **clustering** problem: **k-means clustering**, **mean-shift clustering**
- Segmentation as a statistical (unsupervised) learning problem: **expectation maximization (EM) algorithm**
- **Graph-based segmentation**
- **Supervised learning with neural networks** (per video later)

slide credit: Václav Hlaváč

Graphical Models

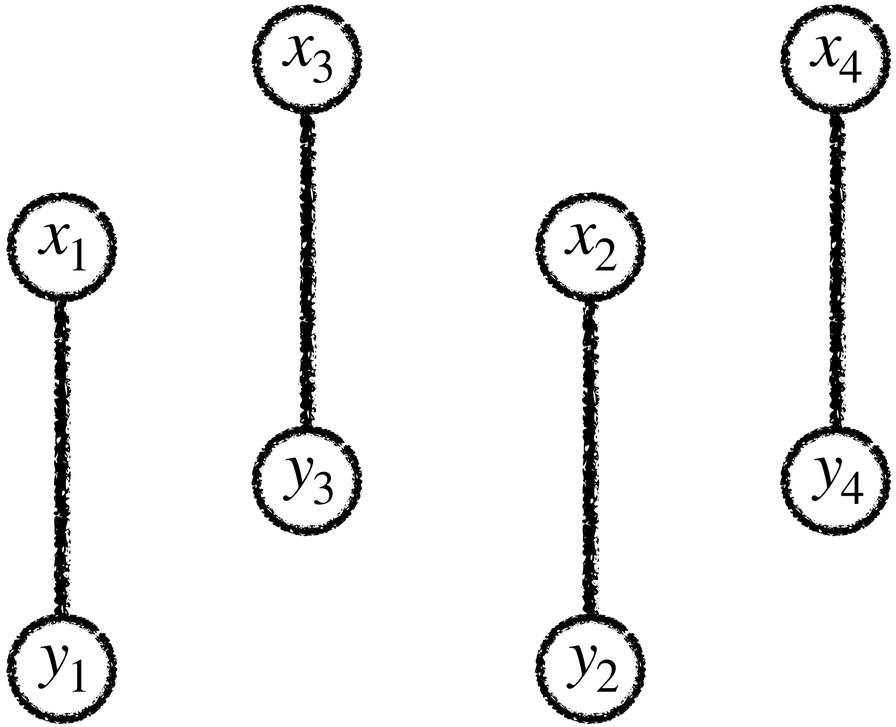
- Graphical representation of our models so far



Graphical Models

- Graphical representation of our models so far

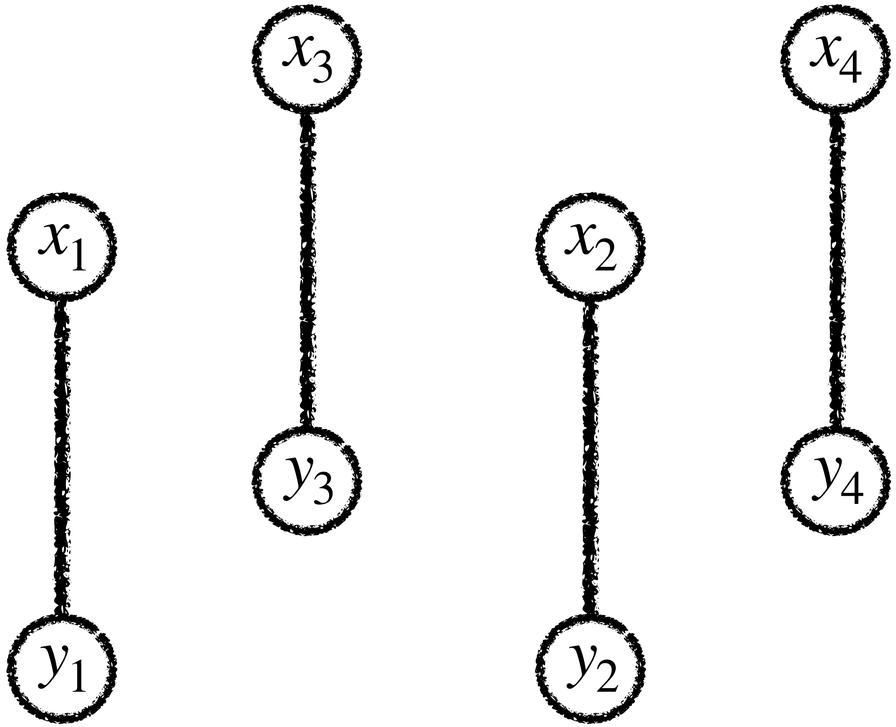
observations (e.g., colors, intensities)



Graphical Models

- Graphical representation of our models so far

observations (e.g., colors, intensities)

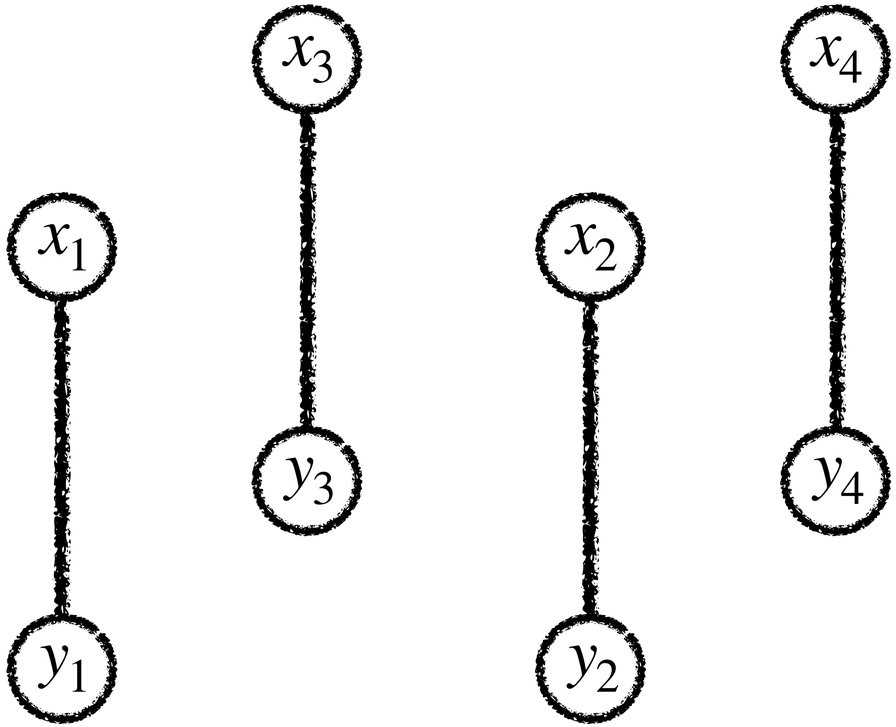


hidden state (labels)

Graphical Models

- Graphical representation of our models so far

observations (e.g., colors, intensities)



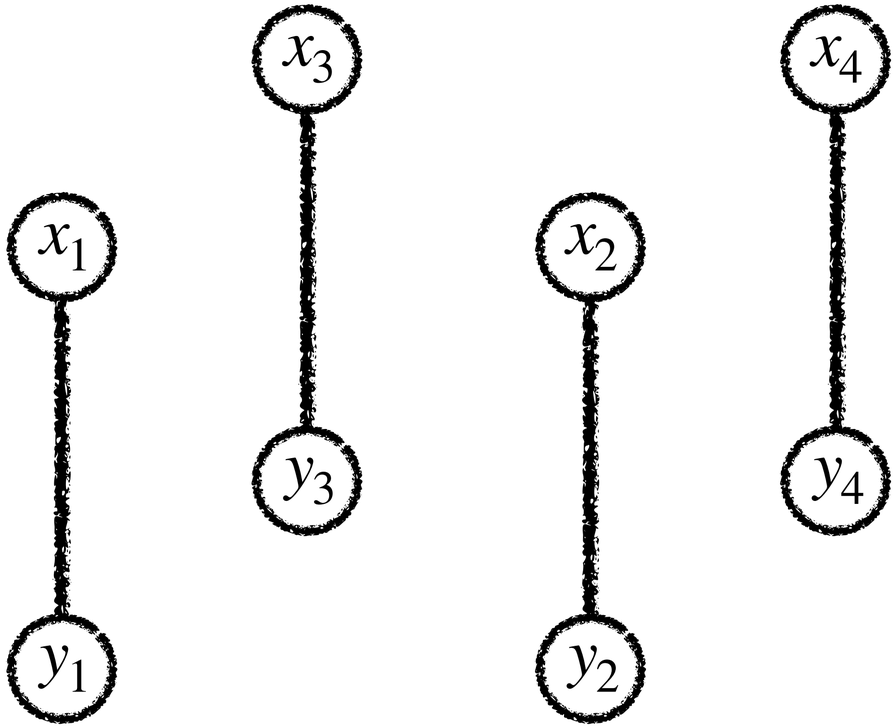
compatibility of observation and label: $\Phi(x_i, y_i)$

hidden state (labels)

Graphical Models

- Graphical representation of our models so far

observations (e.g., colors, intensities)



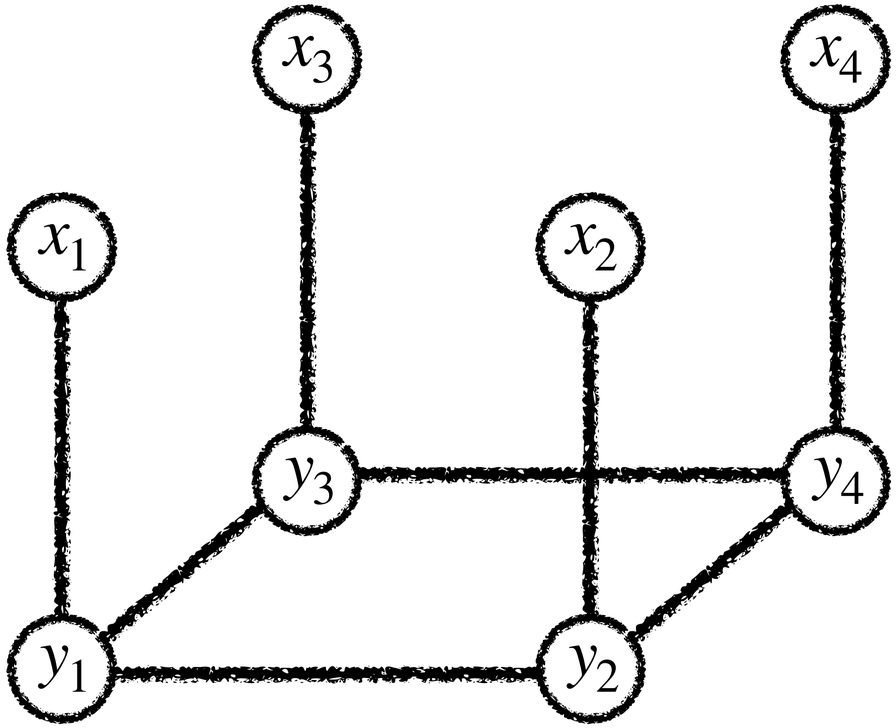
hidden state (labels)

compatibility of observation and label: $\Phi(x_i, y_i)$

find labelling y that maximizes $p(x, y) = \prod_i \Phi(x_i, y_i)$

Markov Random Fields (MRFs)

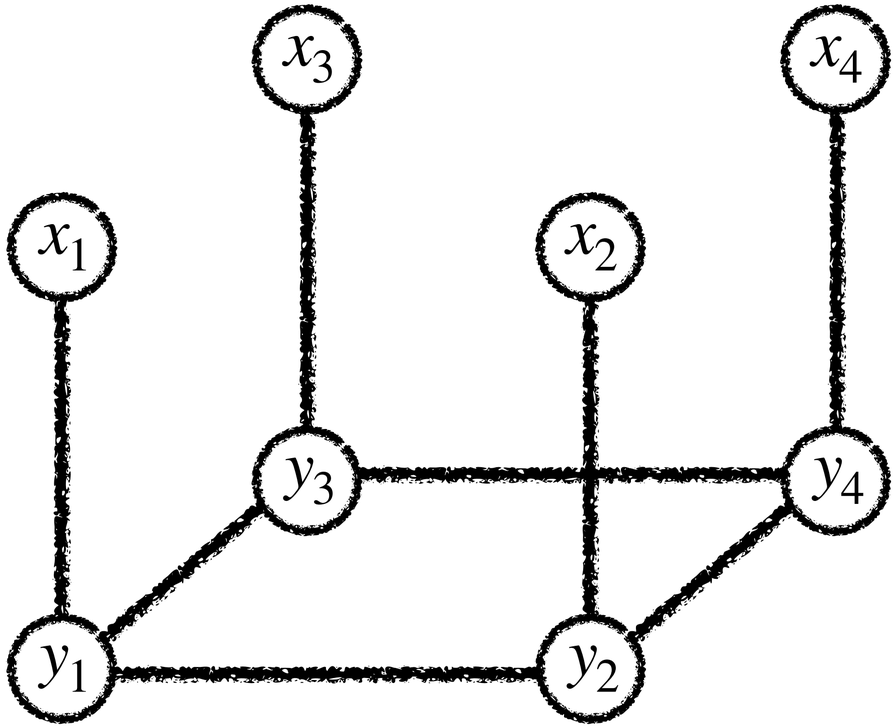
- Take also compatibility between labels into account



Markov Random Fields (MRFs)

- Take also compatibility between labels into account

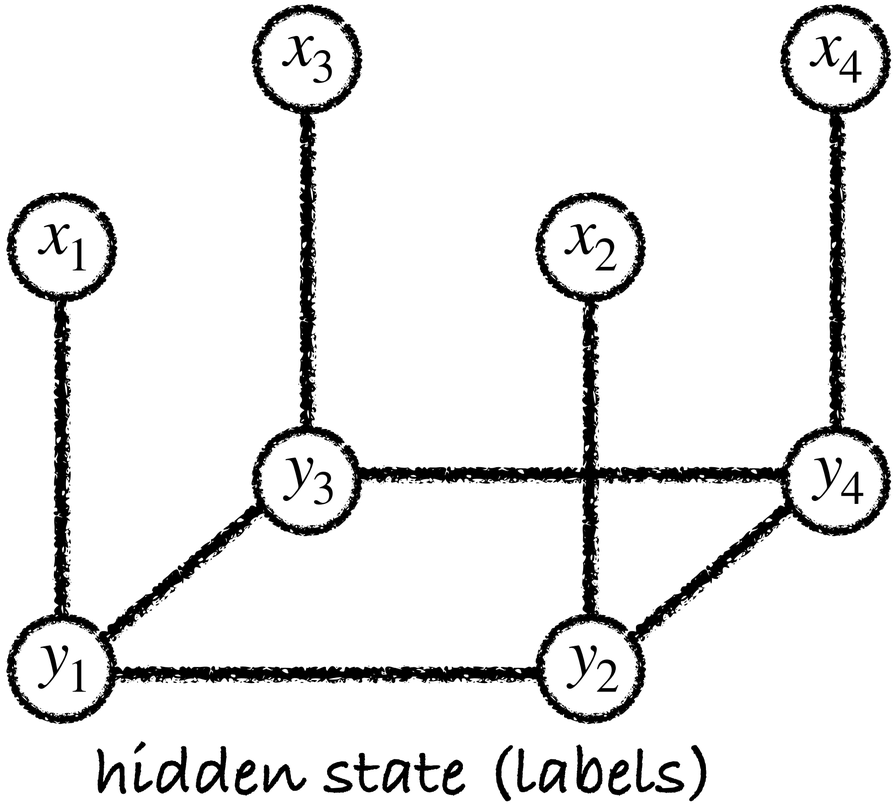
observations (e.g., colors, intensities)



Markov Random Fields (MRFs)

- Take also compatibility between labels into account

observations (e.g., colors, intensities)

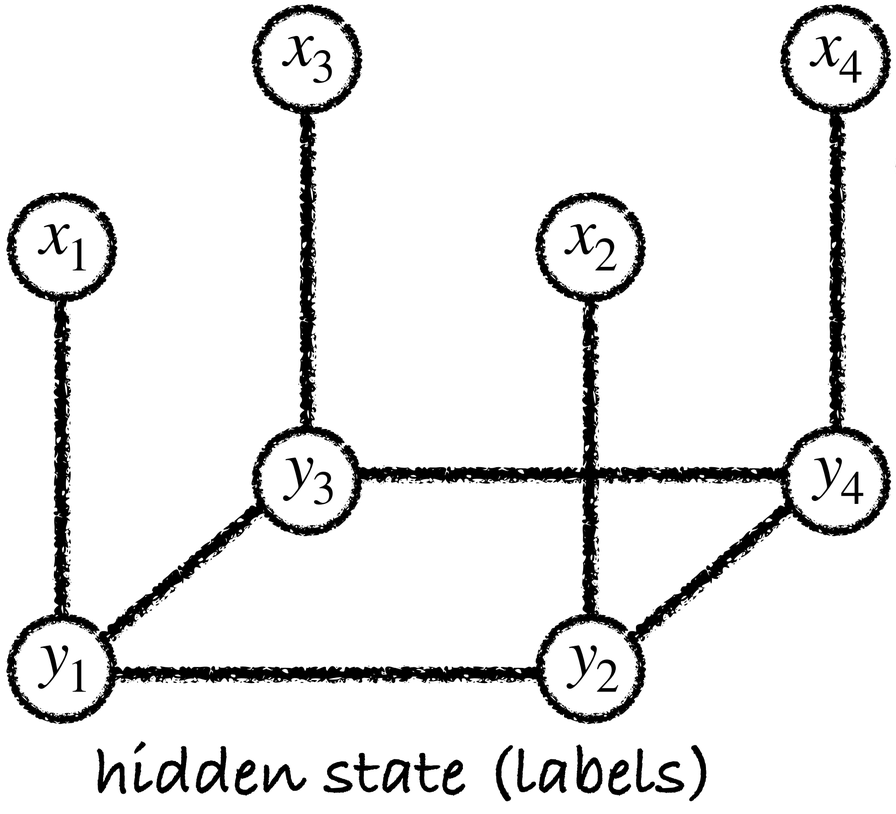


hidden state (labels)

Markov Random Fields (MRFs)

- Take also compatibility between labels into account

observations (e.g., colors, intensities)



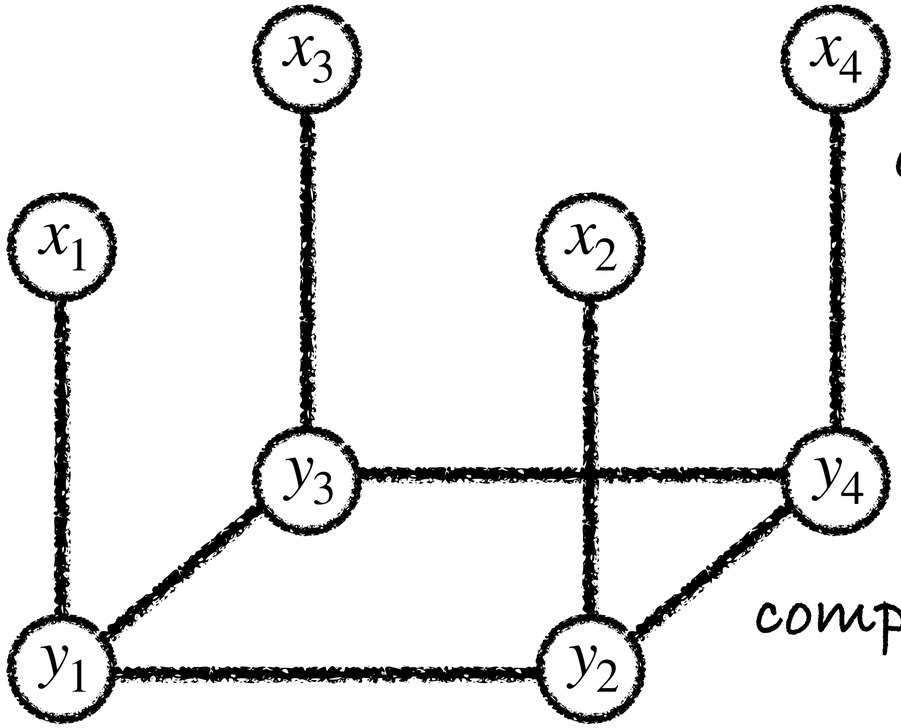
hidden state (labels)

compatibility of observation and label: $\Phi(x_i, y_i)$

Markov Random Fields (MRFs)

- Take also compatibility between labels into account

observations (e.g., colors, intensities)



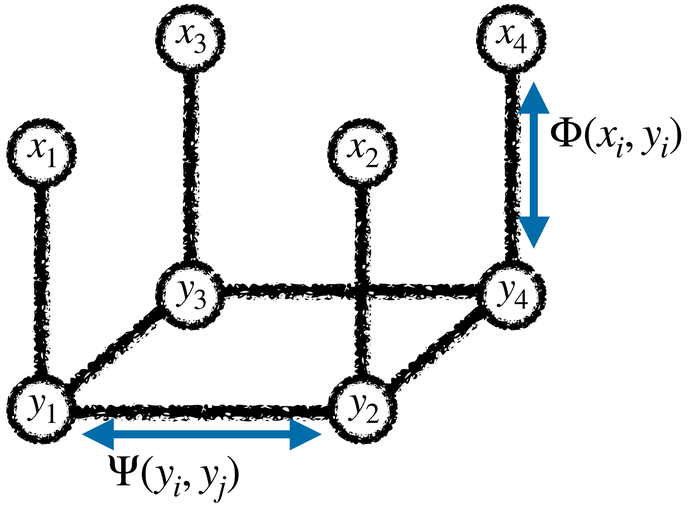
compatibility of observation and label: $\Phi(x_i, y_i)$

compatibility of neighboring labels: $\Psi(y_i, y_j)$

hidden state (labels)

Markov Random Fields (MRFs)

- Take also compatibility between labels into account



observations

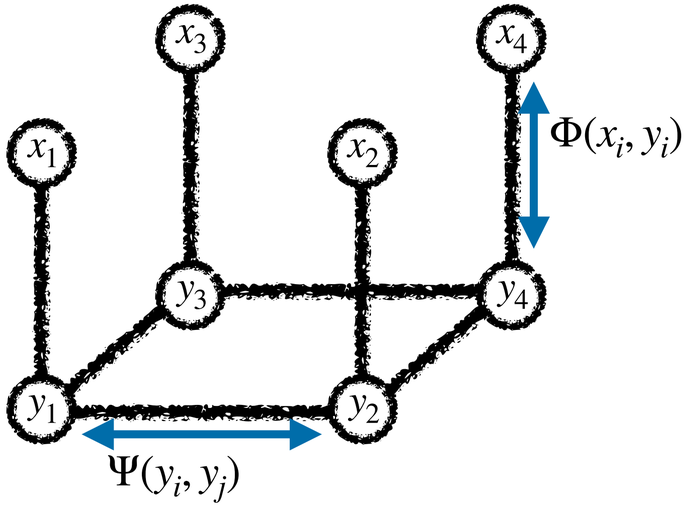
$$p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$$

hidden states

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Take also compatibility between labels into account



observations

observation-state compatibility function

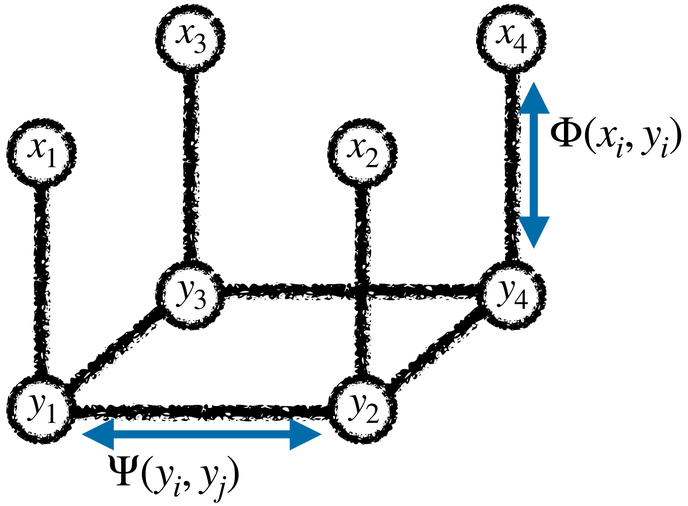
$$p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$$

hidden states

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Take also compatibility between labels into account



observations

observation-state compatibility function

$$p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$$

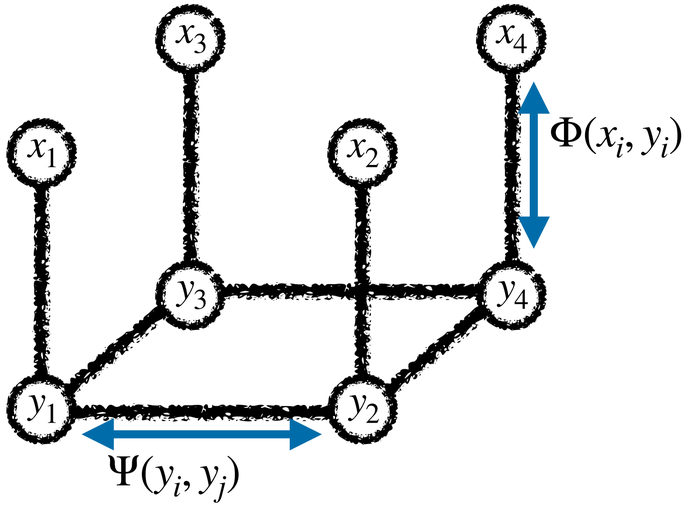
hidden states

state-state compatibility function

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Take also compatibility between labels into account



observations

observation-state compatibility function

neighboring nodes

$$p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$$

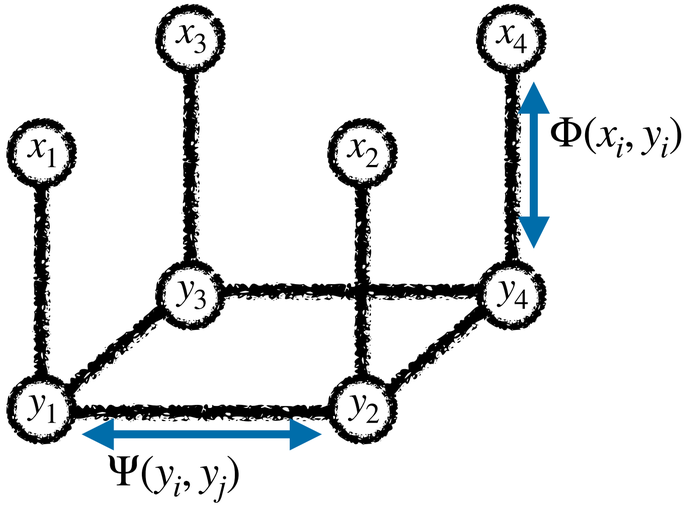
hidden states

state-state compatibility function

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Take also compatibility between labels into account



observations

observation-state compatibility function

neighboring nodes

$$p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$$

hidden states

state-state compatibility function

- Modular and local graphical model, can model global effects

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Maximizing joint probability $p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Maximizing joint probability $p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$

- Identical to minimizing negative log:

$$-\log(p(x, y)) = -\sum_i \log\left(\Phi(x_i, y_i)\right) - \sum_{i,j} \log\left(\Psi(y_i, y_j)\right)$$

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Maximizing joint probability $p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$

- Identical to minimizing negative log:

$$-\log(p(x, y)) = -\sum_i \log\left(\Phi(x_i, y_i)\right) - \sum_{i,j} \log\left(\Psi(y_i, y_j)\right)$$

$$E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(y_i, y_j)$$

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Maximizing joint probability $p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$

- Identical to minimizing negative log:

$$-\log(p(x, y)) = -\sum_i \log\left(\Phi(x_i, y_i)\right) - \sum_{i,j} \log\left(\Psi(y_i, y_j)\right)$$

$$E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(y_i, y_j)$$

- Similar to free-energy problems in statistical mechanics (spin glass theory)

slide credit: Bastian Leibe

Markov Random Fields (MRFs)

- Maximizing joint probability $p(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(y_i, y_j)$
- Identical to minimizing negative log:

$$-\log(p(x, y)) = -\sum_i \log\left(\Phi(x_i, y_i)\right) - \sum_{i,j} \log\left(\Psi(y_i, y_j)\right)$$

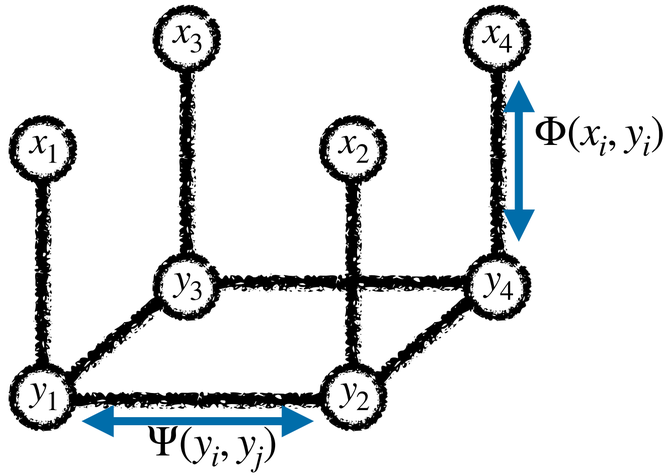
$$E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(y_i, y_j)$$

- Similar to free-energy problems in statistical mechanics (spin glass theory)
- $E(x, y)$ thus analogously referred to as an *energy function*

slide credit: Bastian Leibe

Energy Minimization

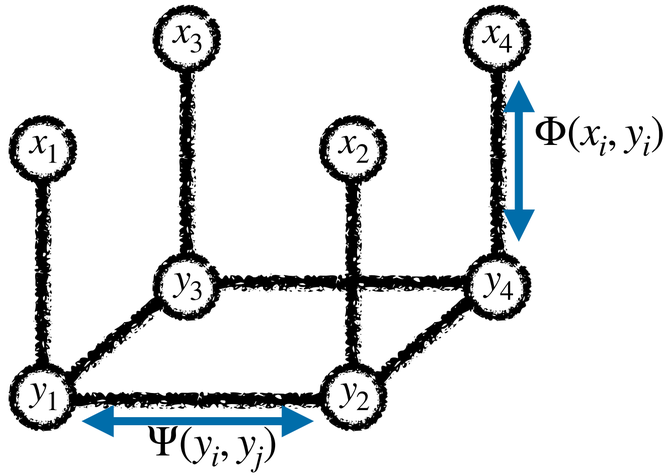
$$E(x, y) = \sum_{i} \phi(x_i, y_i) + \sum_{i,j} \psi(y_i, y_j)$$



slide credit: Bastian Leibe

Energy Minimization

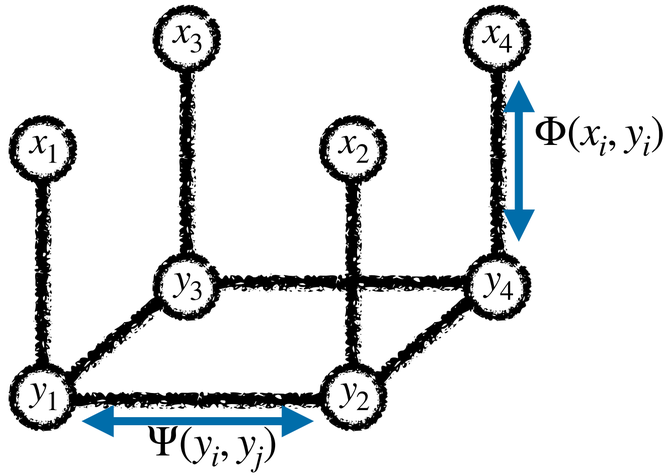
$$E(x, y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \psi(y_i, y_j)$$



slide credit: Bastian Leibe

Energy Minimization

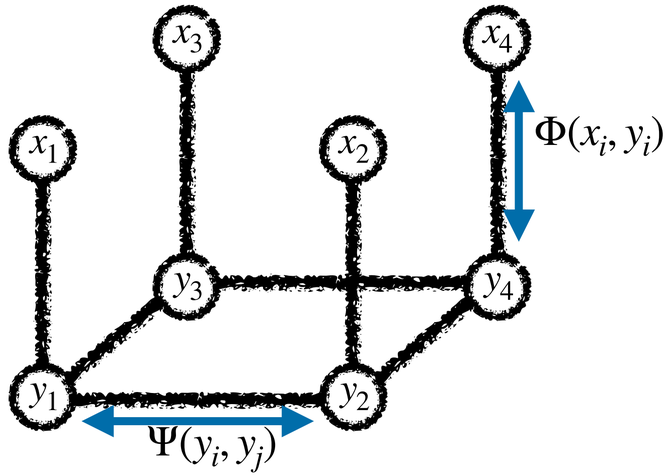
$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$



slide credit: Bastian Leibe

Energy Minimization

$$E(x, y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

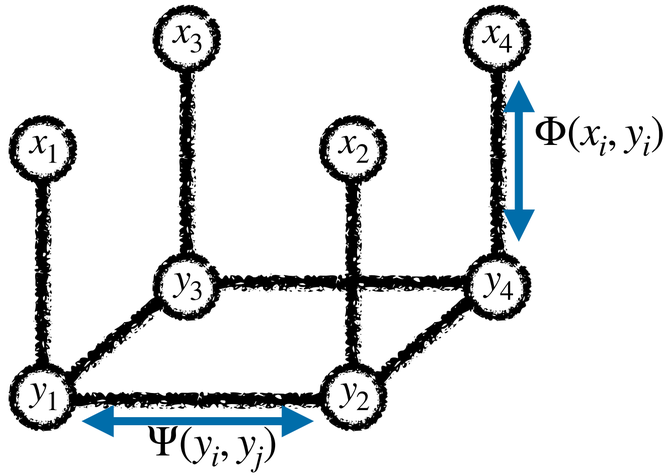


- Single-node potentials (“unary potentials”):

slide credit: Bastian Leibe

Energy Minimization

$$E(x, y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

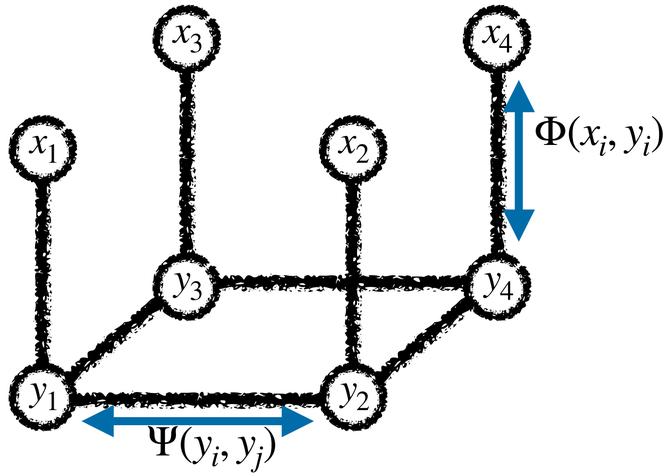


- Single-node potentials (“unary potentials”):
  - Encode local information about each pixel / patch / super pixel

slide credit: Bastian Leibe

Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

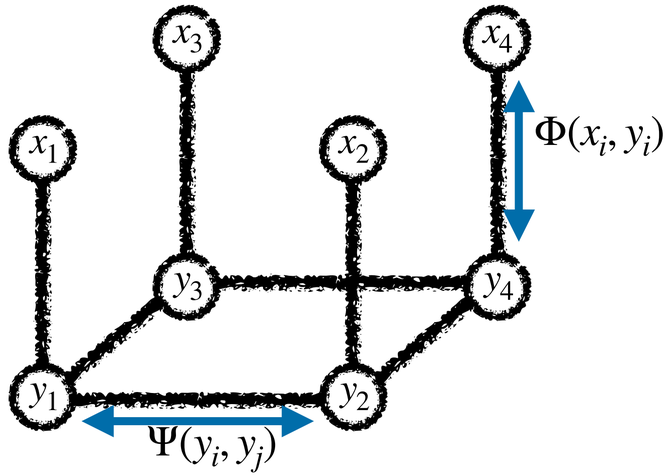


- Single-node potentials (“unary potentials”):
  - Encode local information about each pixel / patch / super pixel
  - How likely is it that pixel x_i belongs to label y_i / foreground / background, etc.

slide credit: Bastian Leibe

Energy Minimization

$$E(x, y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$



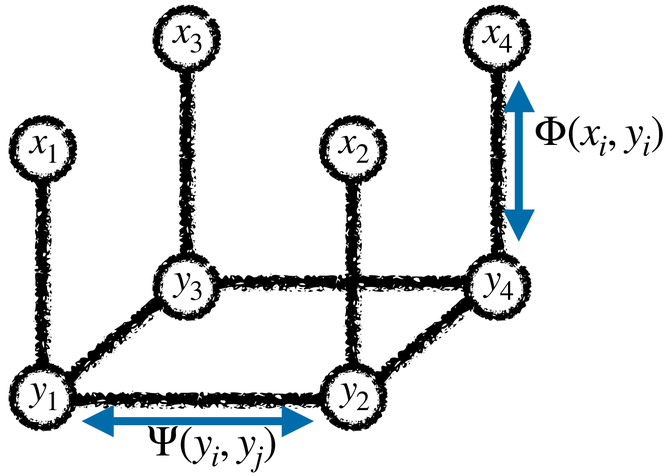
- Single-node potentials (“unary potentials”):
  - Encode local information about each pixel / patch / super pixel
  - How likely is it that pixel x_i belongs to label y_i / foreground / background, etc.
  - Example choice: Mixture of Gaussians (see lab)

$$\phi(x_i, y_i \mid \theta_\phi) = -\log \sum_k \theta_\phi(y_i, k) p(k \mid y_i) \mathcal{N}(x_i \mid \mu_k, \Sigma_k)$$

slide credit: Bastian Leibe

Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

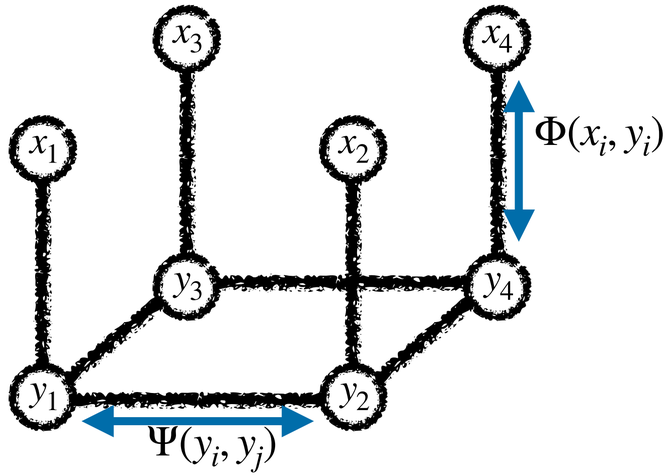


- Pairwise potentials (“binary potentials”):

slide credit: Bastian Leibe

Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

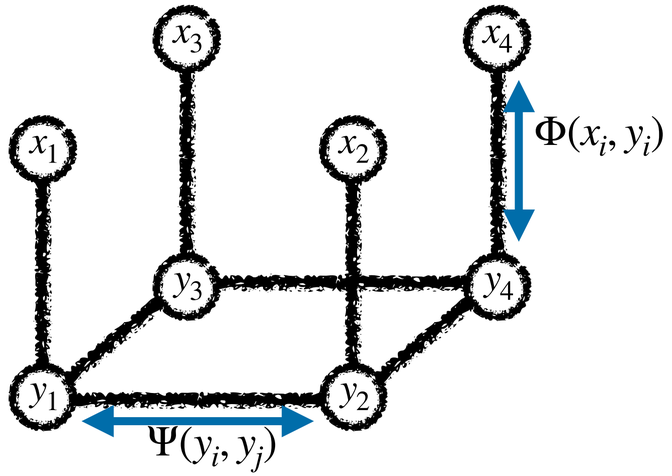


- Pairwise potentials (“binary potentials”):
  - Encode information about local neighborhoods, e.g., neighboring 4 or 8 pixels

slide credit: Bastian Leibe

Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

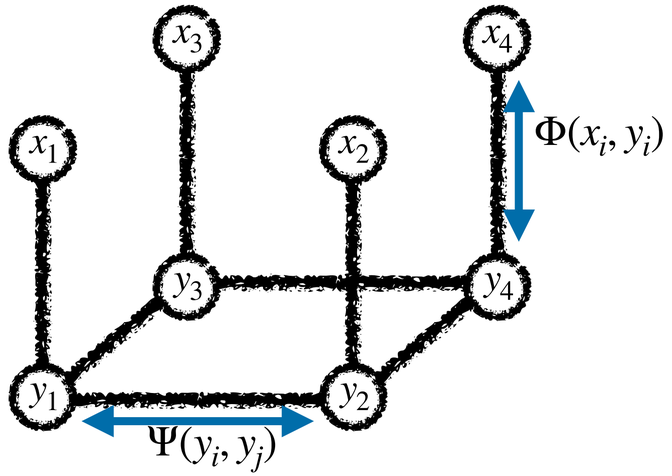


- Pairwise potentials (“binary potentials”):
  - Encode information about local neighborhoods, e.g., neighboring 4 or 8 pixels
  - How likely is it that two pixels should have the same / different labels?

slide credit: Bastian Leibe

Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$



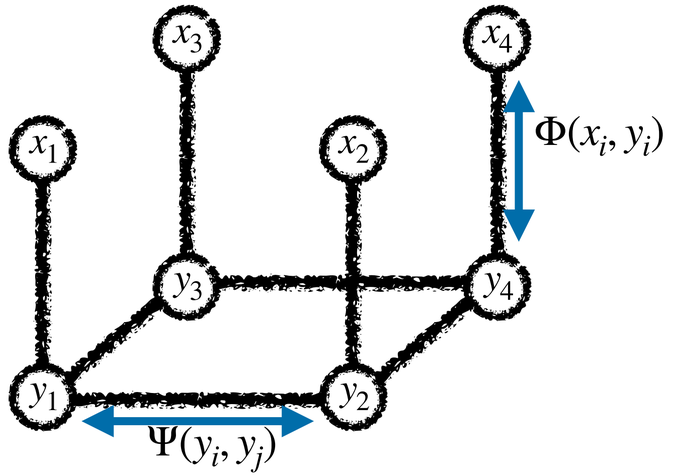
- Pairwise potentials ("binary potentials"):
  - Encode information about local neighborhoods, e.g., neighboring 4 or 8 pixels
  - How likely is it that two pixels should have the same / different labels?
  - Example choice: "contrast sensitive Potts model" (see lab)

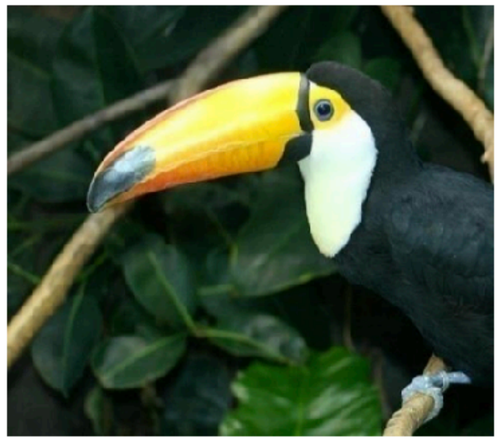
$$\psi(y_i, y_j \,|\, \theta_\psi) = -\theta_\psi \delta(y_i \neq y_j) e^{-\frac{1}{2} \frac{\|x_i - x_j\|^2}{avg(\|x_i - x_j\|^2)}}$$

slide credit: Bastian Leibe

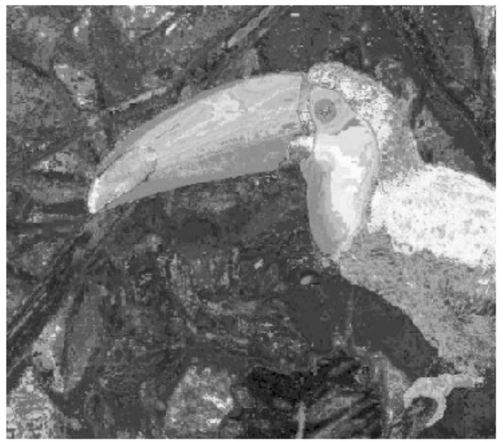
Energy Minimization

$$E(x,y) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{single-node potentials}} + \sum_{i,j} \underbrace{\psi(y_i, y_j)}_{\text{pairwise potentials}}$$

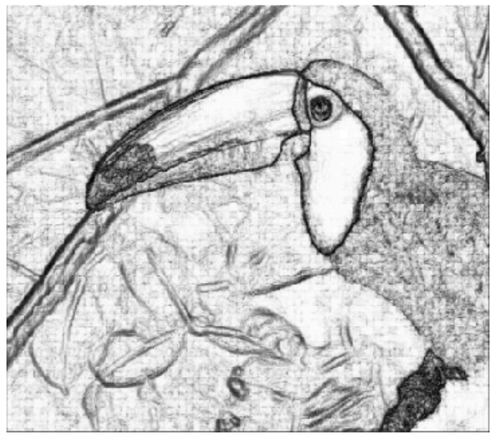




image



unary potentials



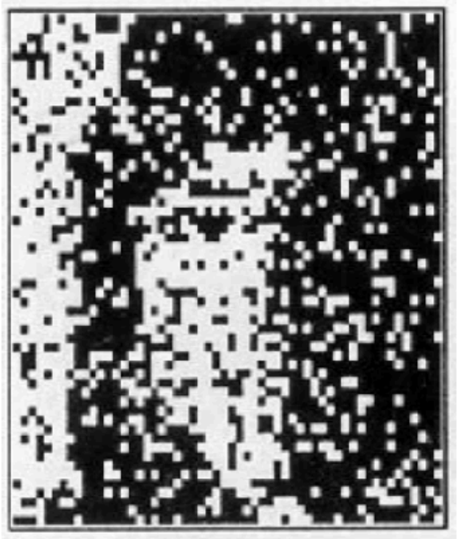
binary potentials

segmentation

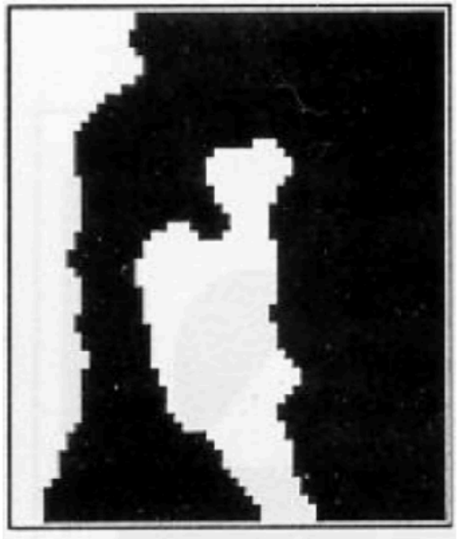
slide credit: Bastian Leibe, Phil Torr

MRF - Example

Original image



Degraded image

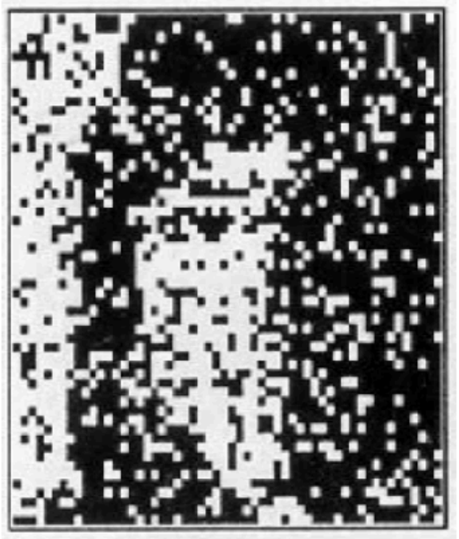


Reconstruction from MRF modeling pixel neighborhood statistics

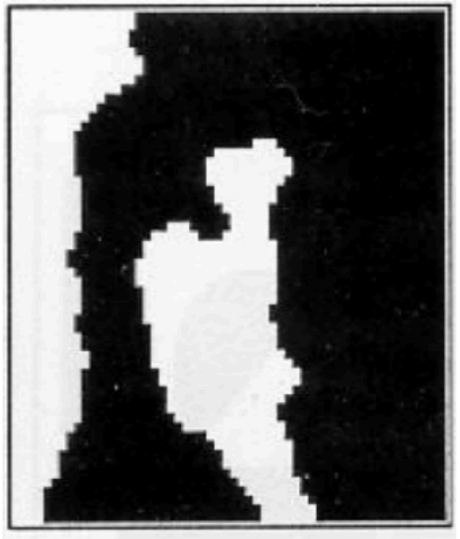
slide credit: Bastian Leibe

MRF - Example

Original image



Degraded image



Reconstruction from MRF modeling pixel neighborhood statistics

How to optimize MRFs?

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:
  - Variational methods

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:
  - Variational methods
  - Belief propagation

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:
  - Variational methods
  - Belief propagation
  - **Graph cuts**

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:
  - Variational methods
  - Belief propagation
  - **Graph cuts**
    - Very efficient for computer vision / image analysis problems where we have regular structure

slide credit: Bastian Leibe

Energy Minimization

- Want to infer the optimal labeling of pixels based on MRF energy
- Many potential approaches for inference:
  - Variational methods
  - Belief propagation
  - **Graph cuts**
    - Very efficient for computer vision / image analysis problems where we have regular structure
    - Optimality guarantees for certain class of energy functions (submodular energies)

slide credit: Bastian Leibe

The s-t Mincut Problem

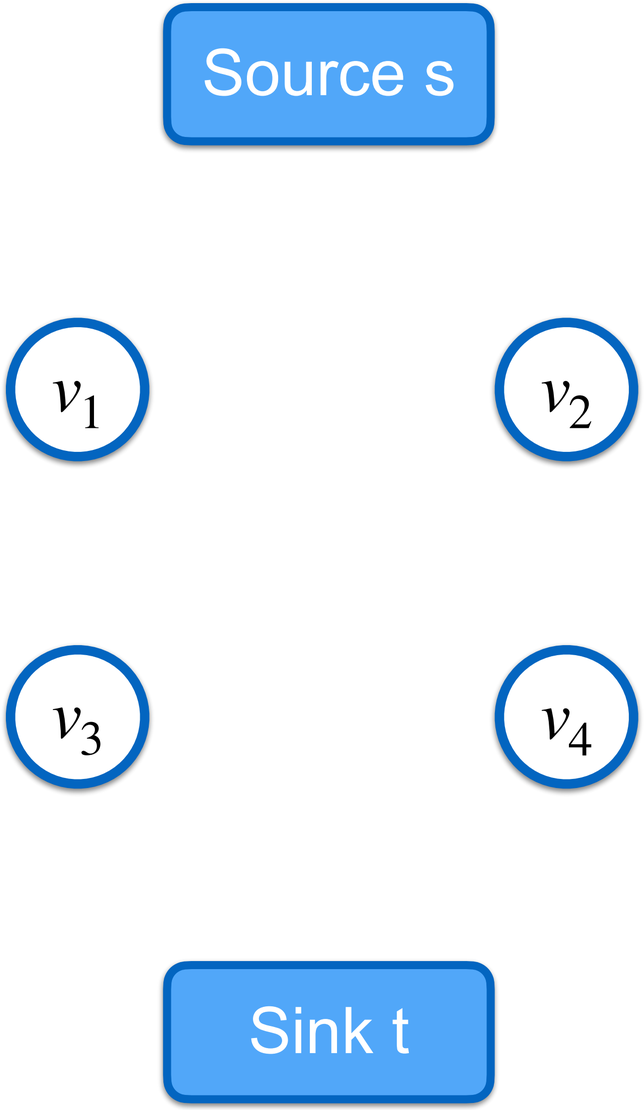
Source s

terminal nodes s, t

Sink t

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem

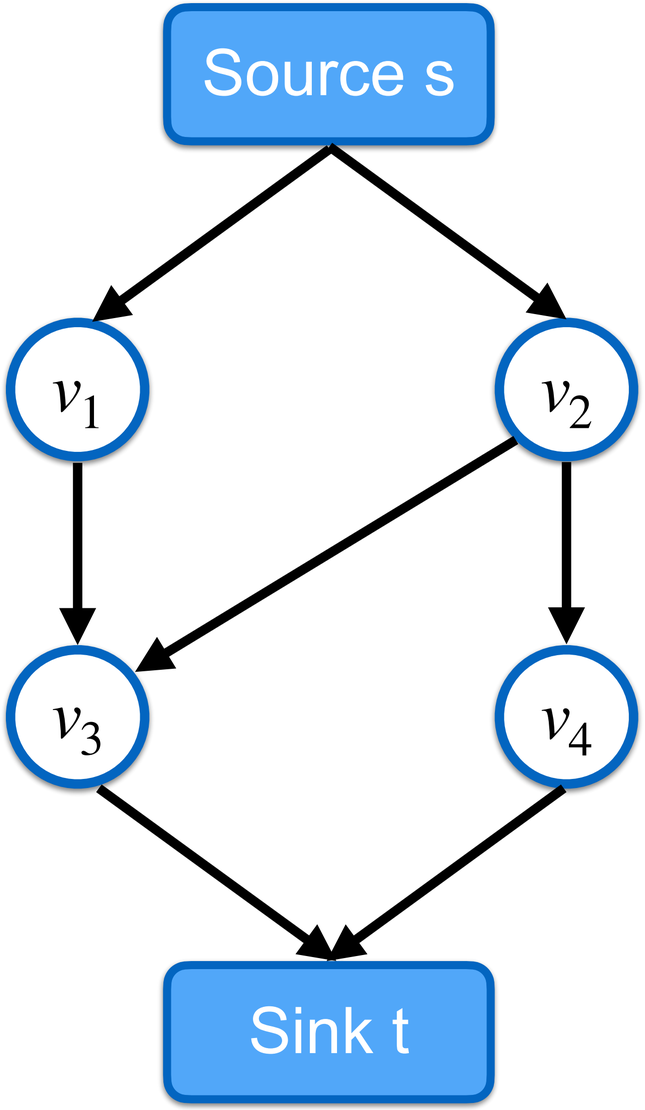


terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem



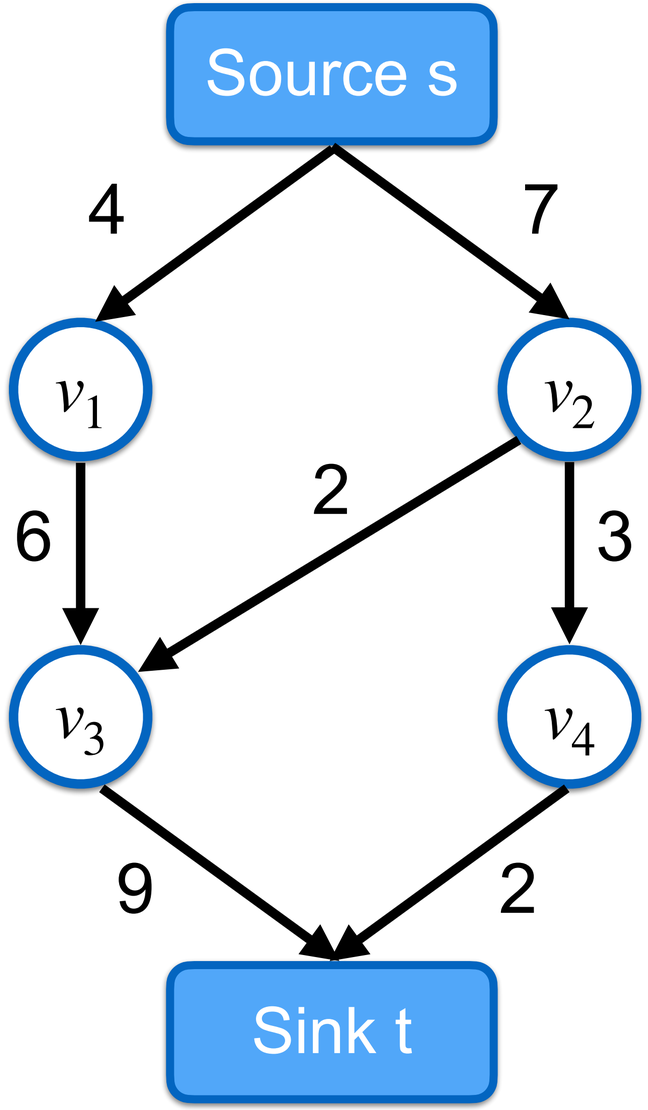
terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem



terminal nodes s, t

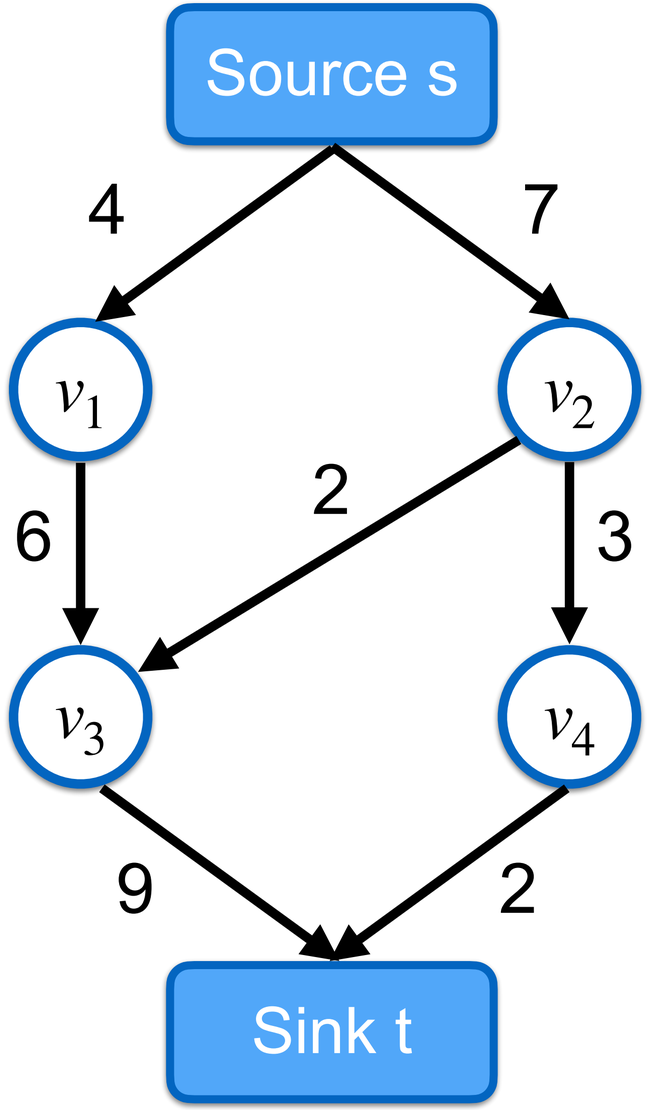
vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

costs $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem



terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

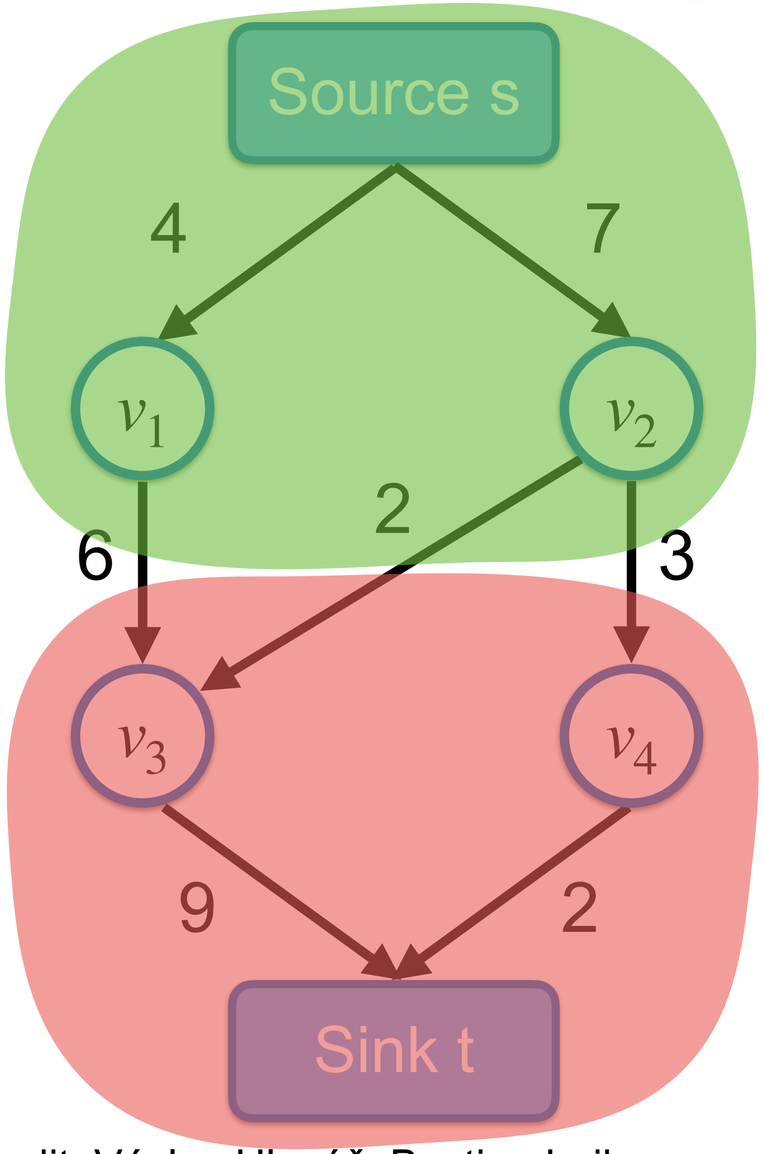
edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

costs $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

s-t cut: partition $S, T \subset V$ with $S \cap T = \emptyset, s \in S, t \in T$

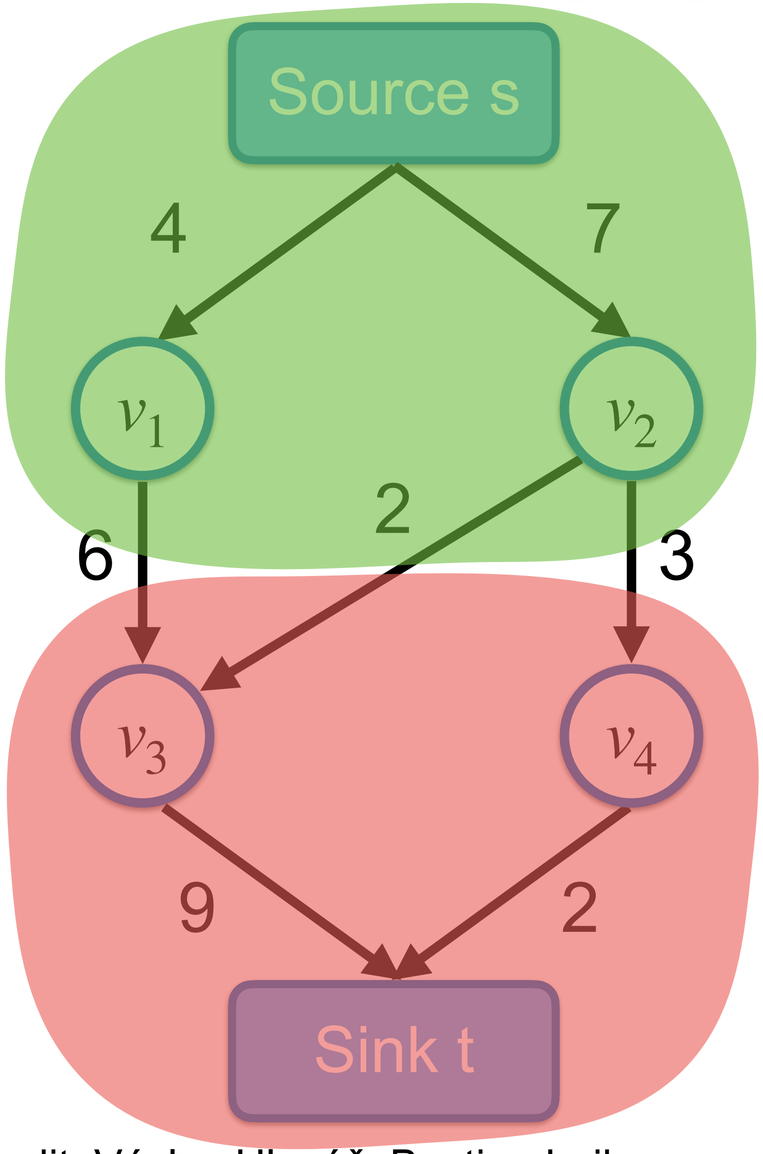
slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem



slide credit: Václav Hlaváč, Bastian Leibe

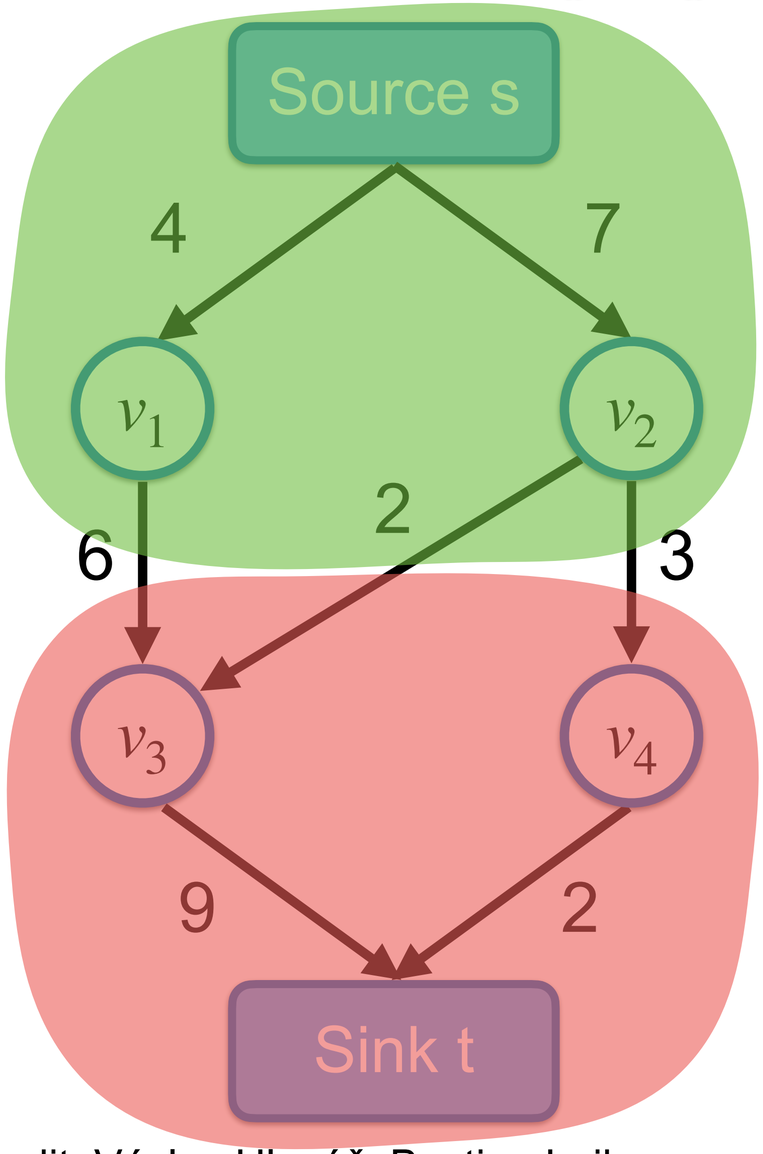
The s-t Mincut Problem



Cost of s-t cut:
cost of edges from S to T

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem

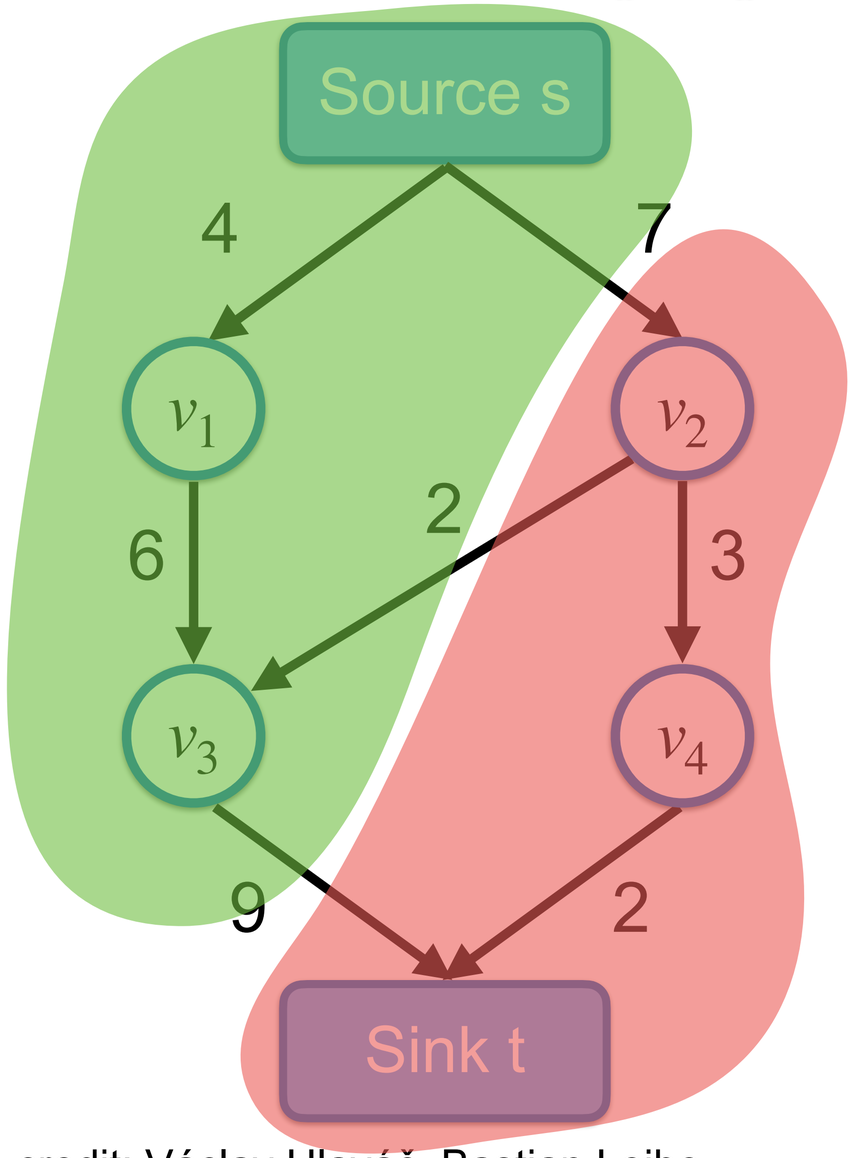


Cost of s-t cut:
cost of edges from S to T

cost: 6 + 2 + 3 = 11

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem

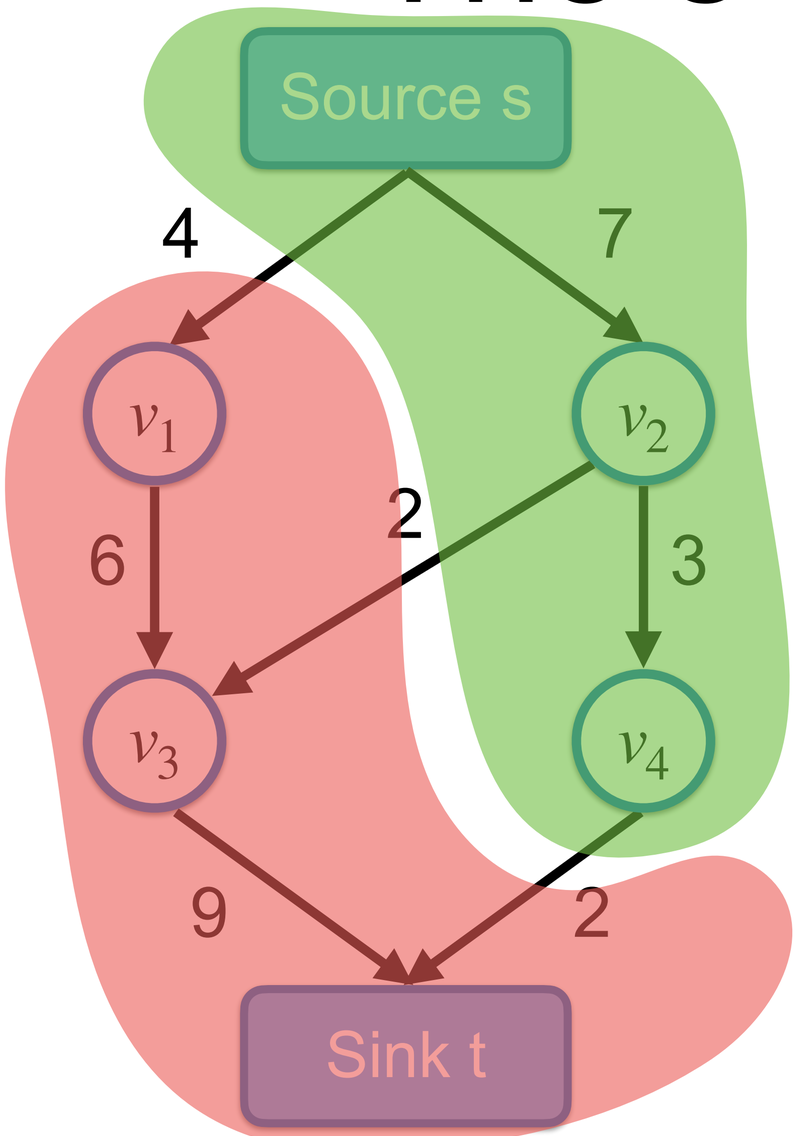


Cost of s-t cut:
cost of edges from S to T

cost: $7 + 9 = 16$

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem

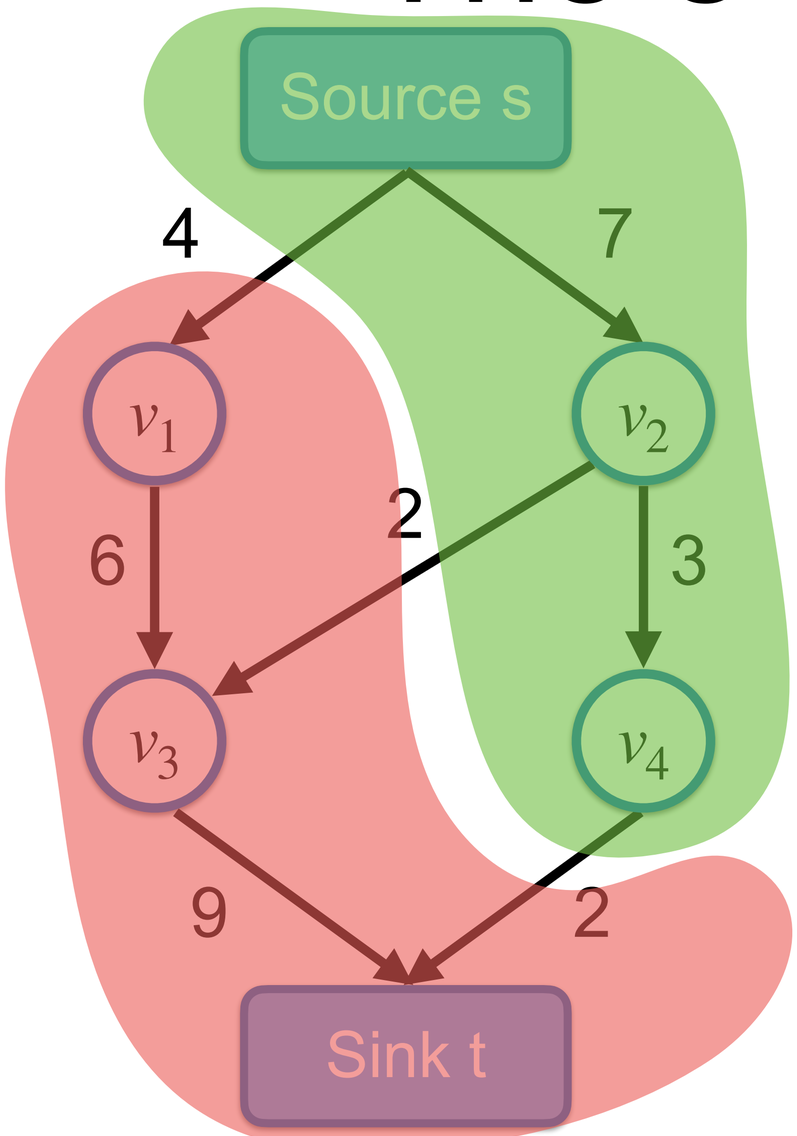


Cost of s-t cut:
cost of edges from S to T

cost: 4 + 2 + 2 = 8

slide credit: Václav Hlaváč, Bastian Leibe

The s-t Mincut Problem



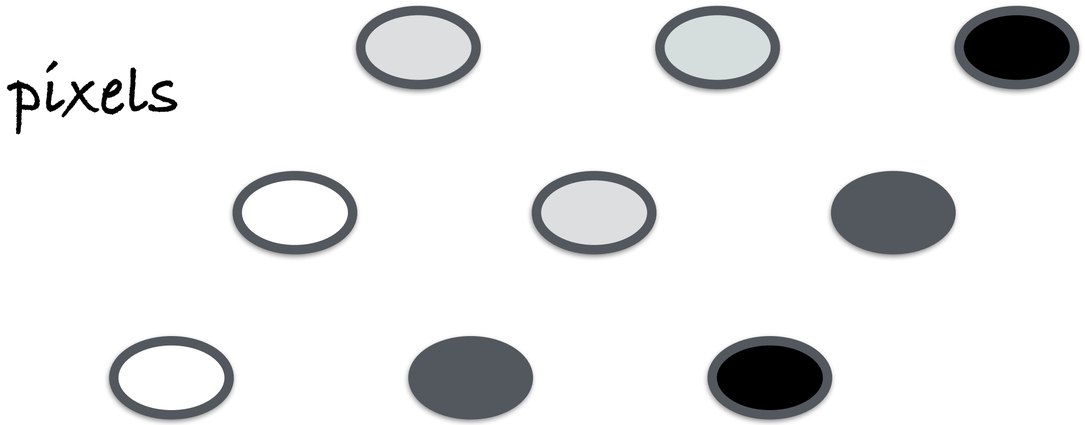
Cost of s-t cut:
cost of edges from S to T

cost: 4 + 2 + 2 = 8

s-t mincut:
s-t cut with minimum cost

slide credit: Václav Hlaváč, Bastian Leibe

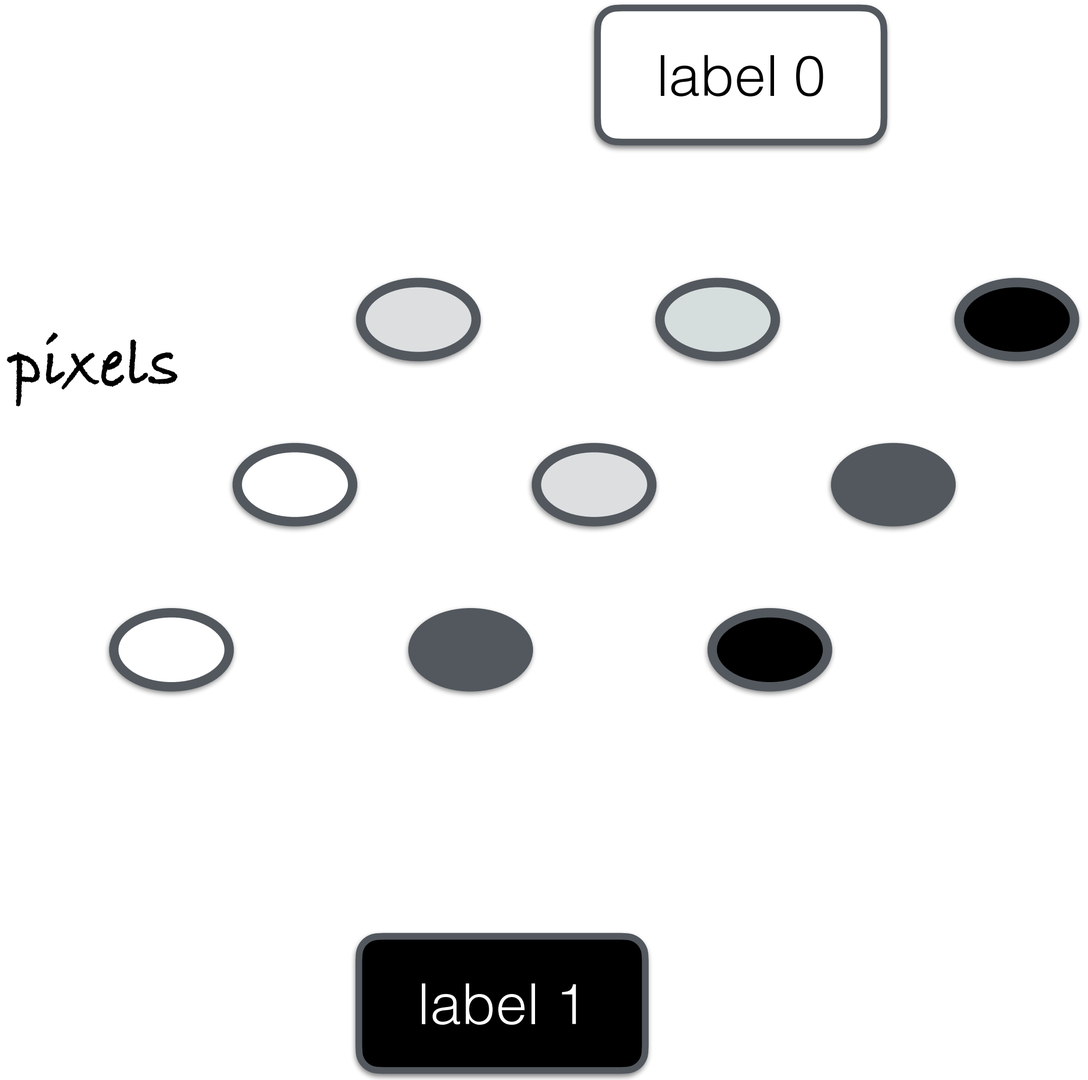
s-t Mincuts and MRFs



$$\text{minimize } E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

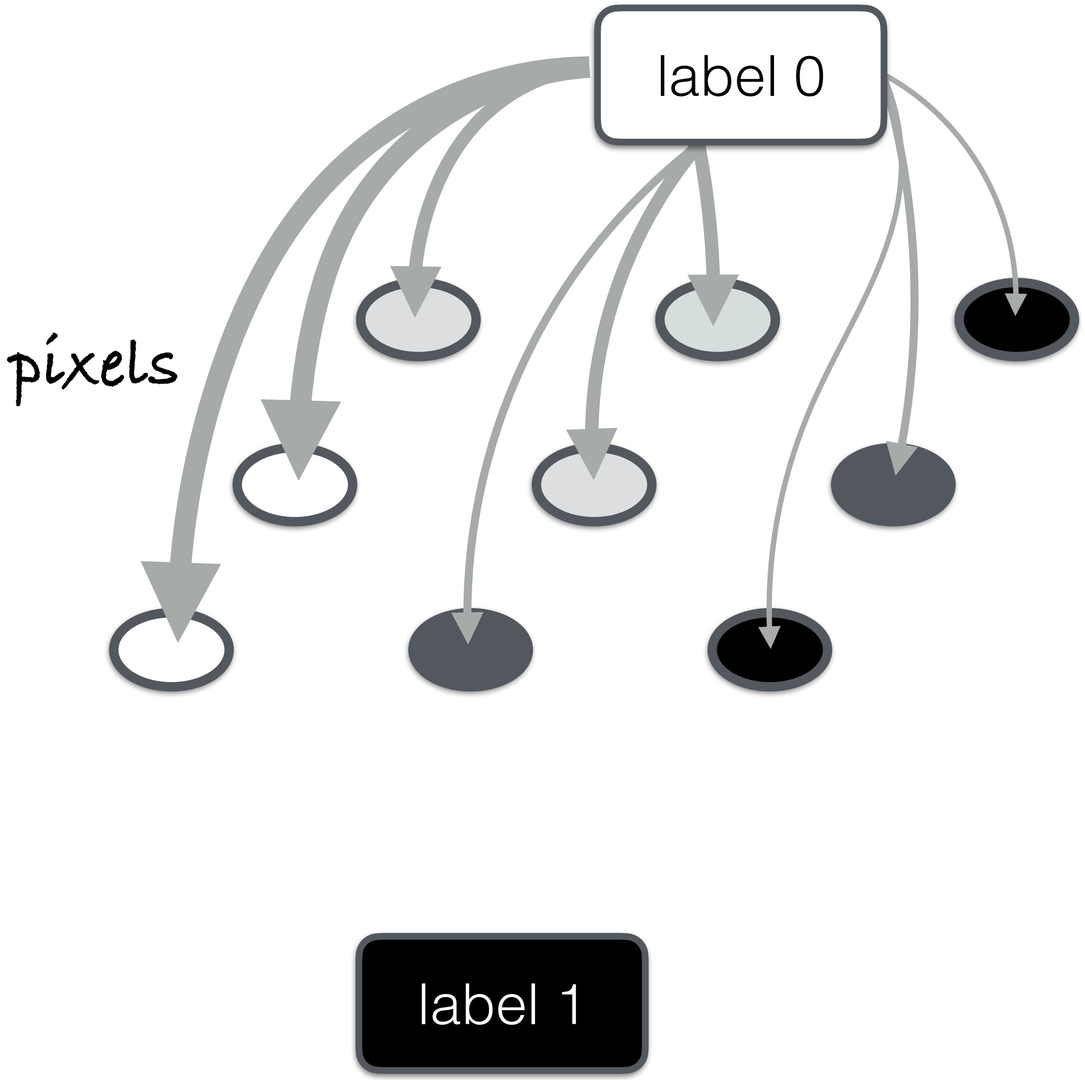
s-t Mincuts and MRFs



minimize $E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

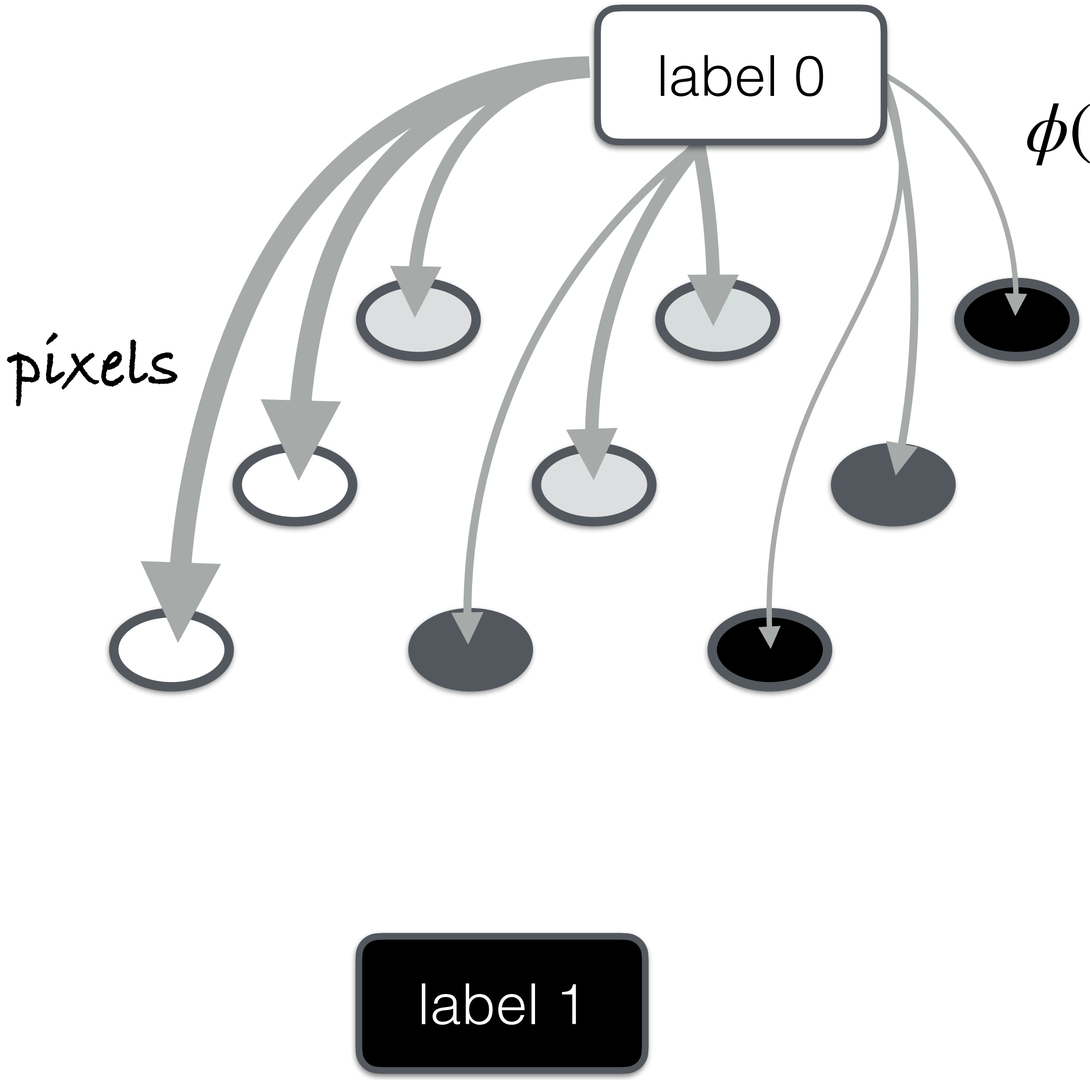
s-t Mincuts and MRFs



minimize $E(x, y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\, \delta(y_i \neq y_j)$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs

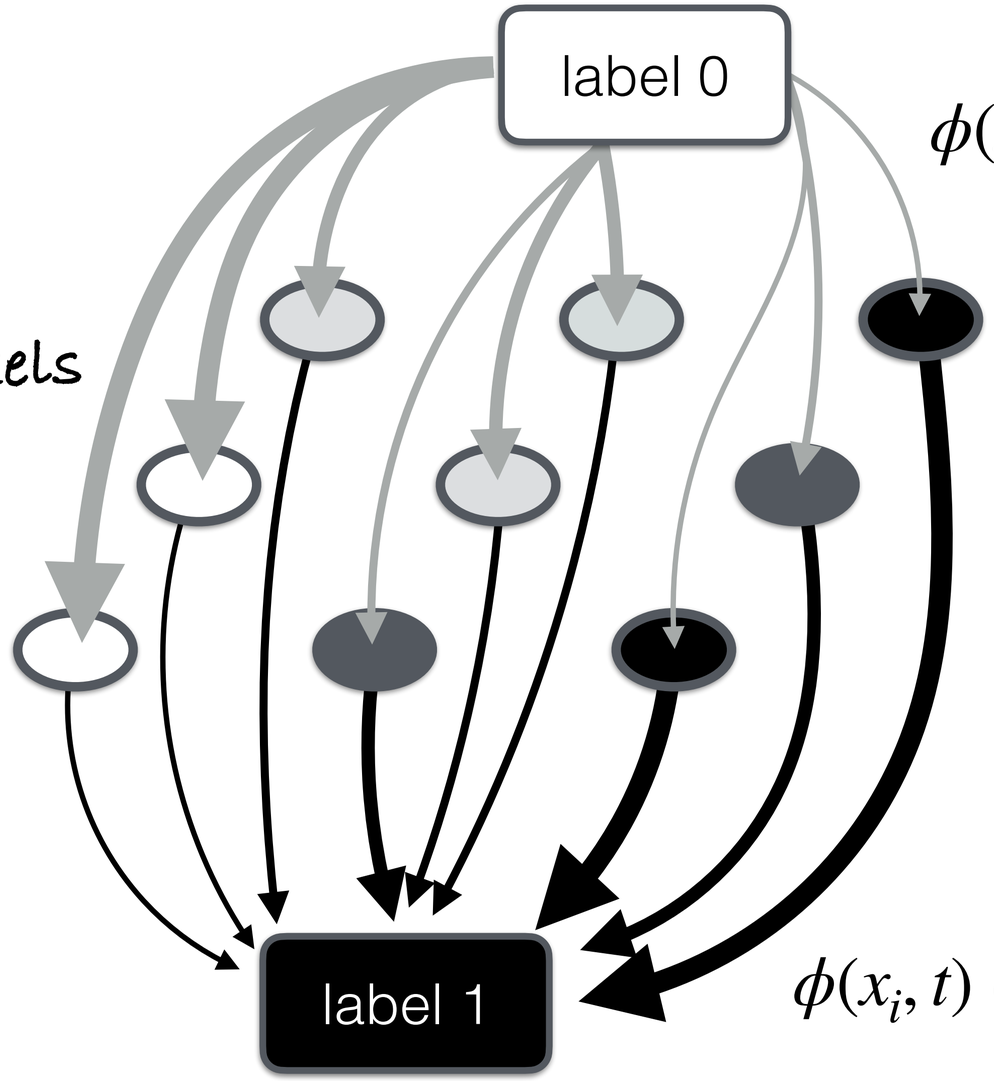


$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

$$\text{minimize } E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs



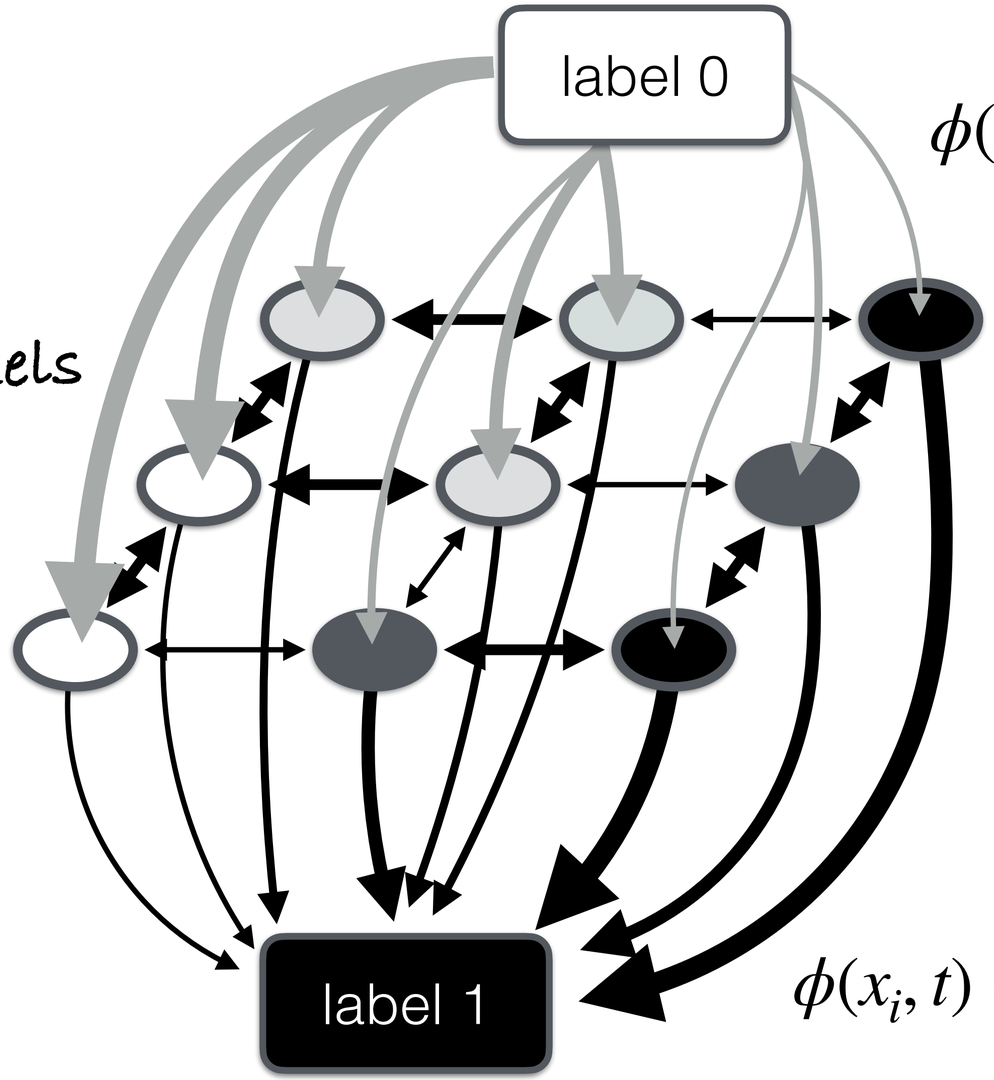
$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

$$\text{minimize } E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$$

$$\phi(x_i, t) \propto exp(-\|I_i - I_t\|^2/2\sigma^2)$$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs



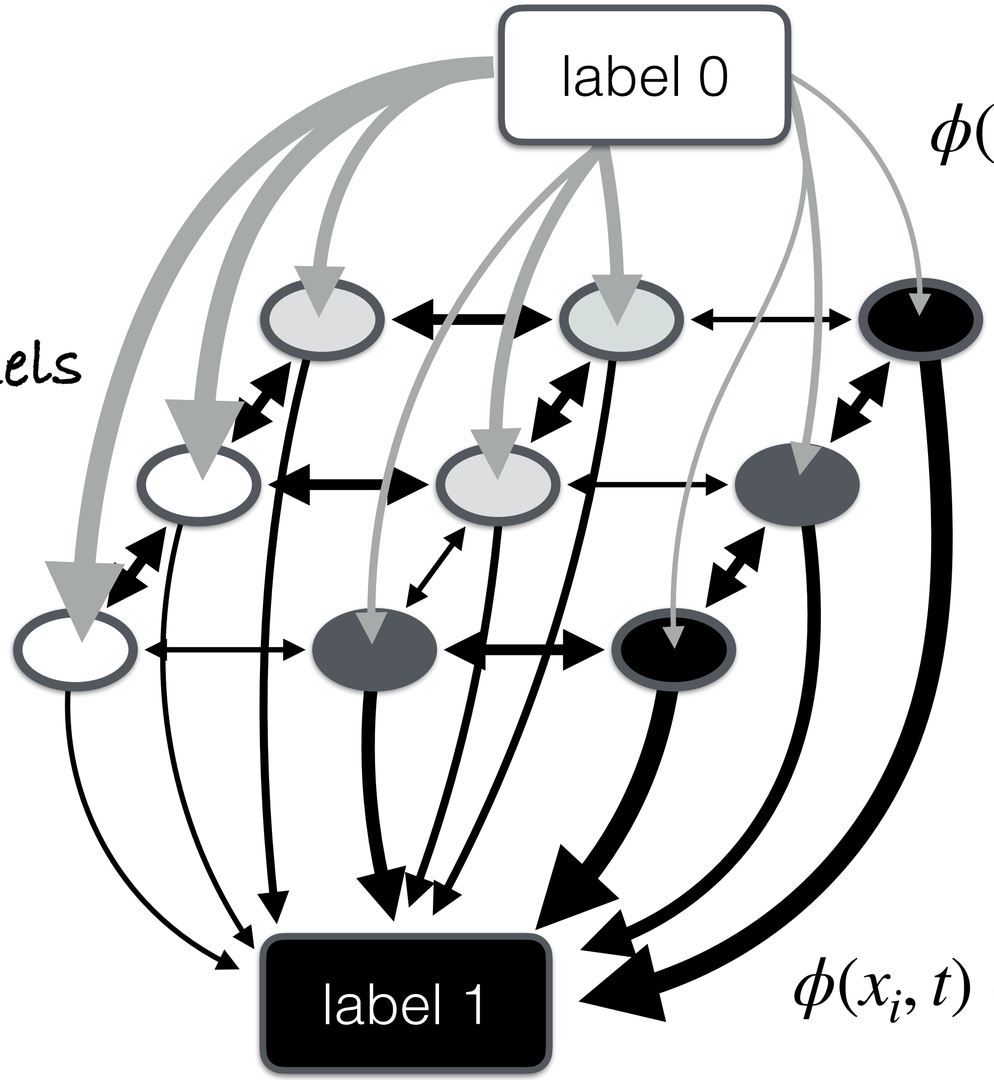
$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

$$\text{minimize } E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$$

$$\phi(x_i, t) \propto exp(-\|I_i - I_t\|^2/2\sigma^2)$$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs



$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

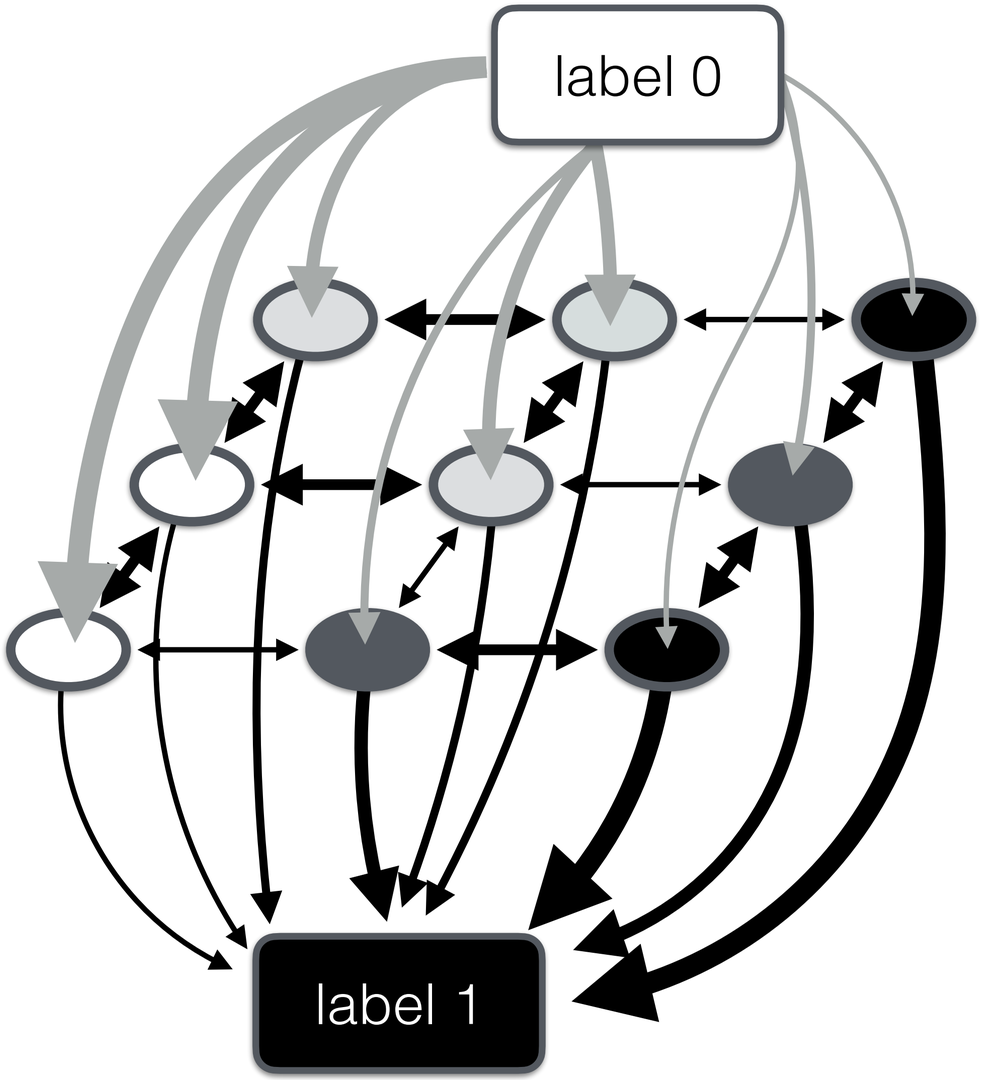
$$\text{minimize } E(x,y) = \sum_i \phi(x_i, y_i) + \sum_{i,j} w_{ij}\,\delta(y_i \neq y_j)$$

$$w_{ij} = exp(-\|I_i - I_j\|^2/2\sigma^2)$$

$$\phi(x_i, t) \propto exp(-\|I_i - I_t\|^2/2\sigma^2)$$

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

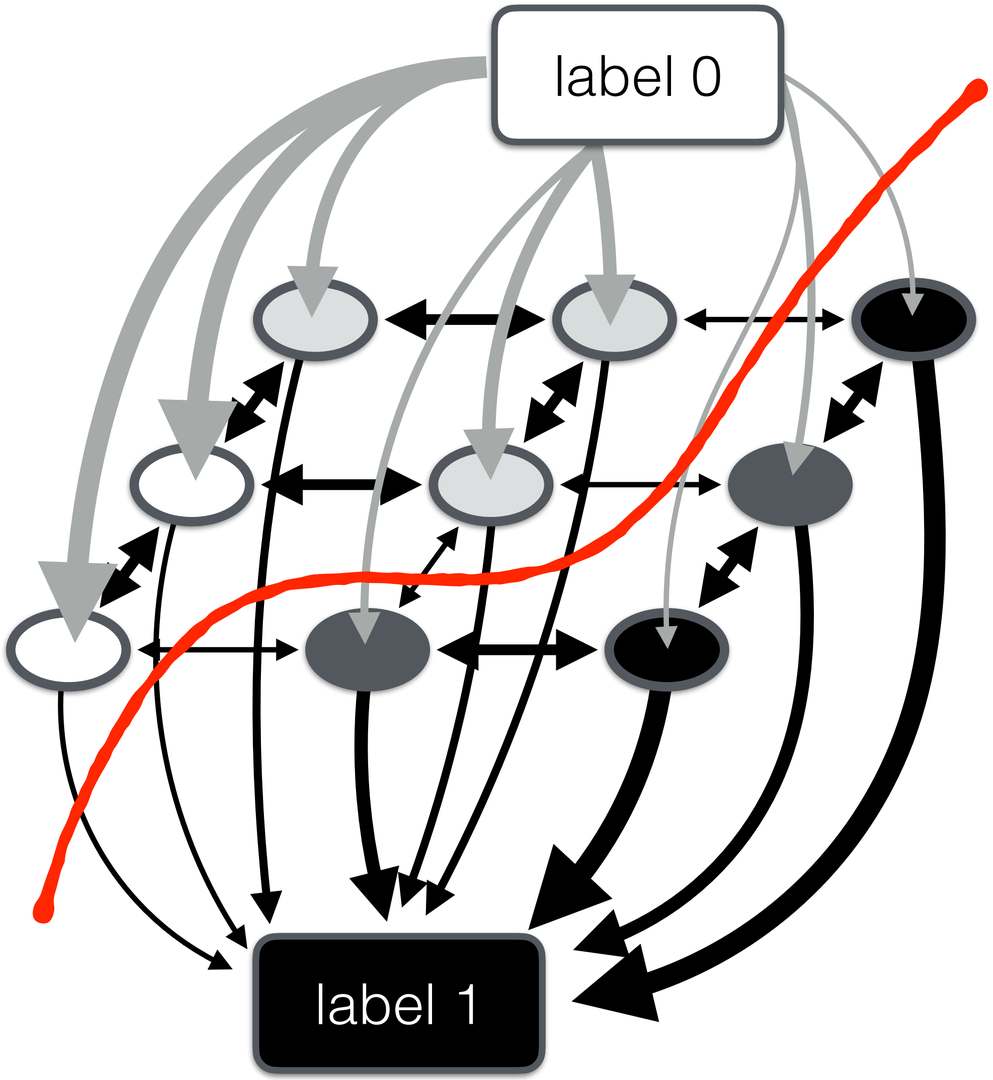
s-t Mincuts and MRFs



cut edges with low costs

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

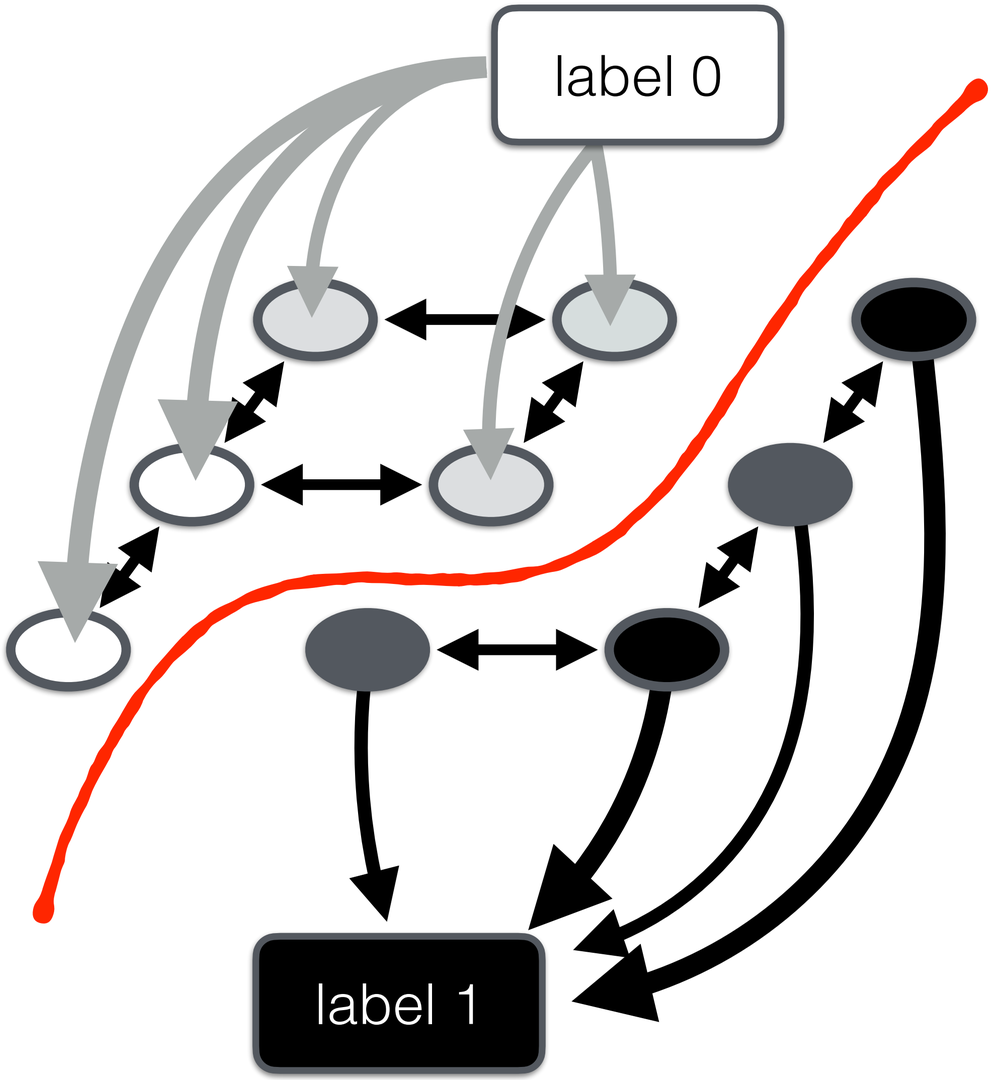
s-t Mincuts and MRFs



cut edges with low costs

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

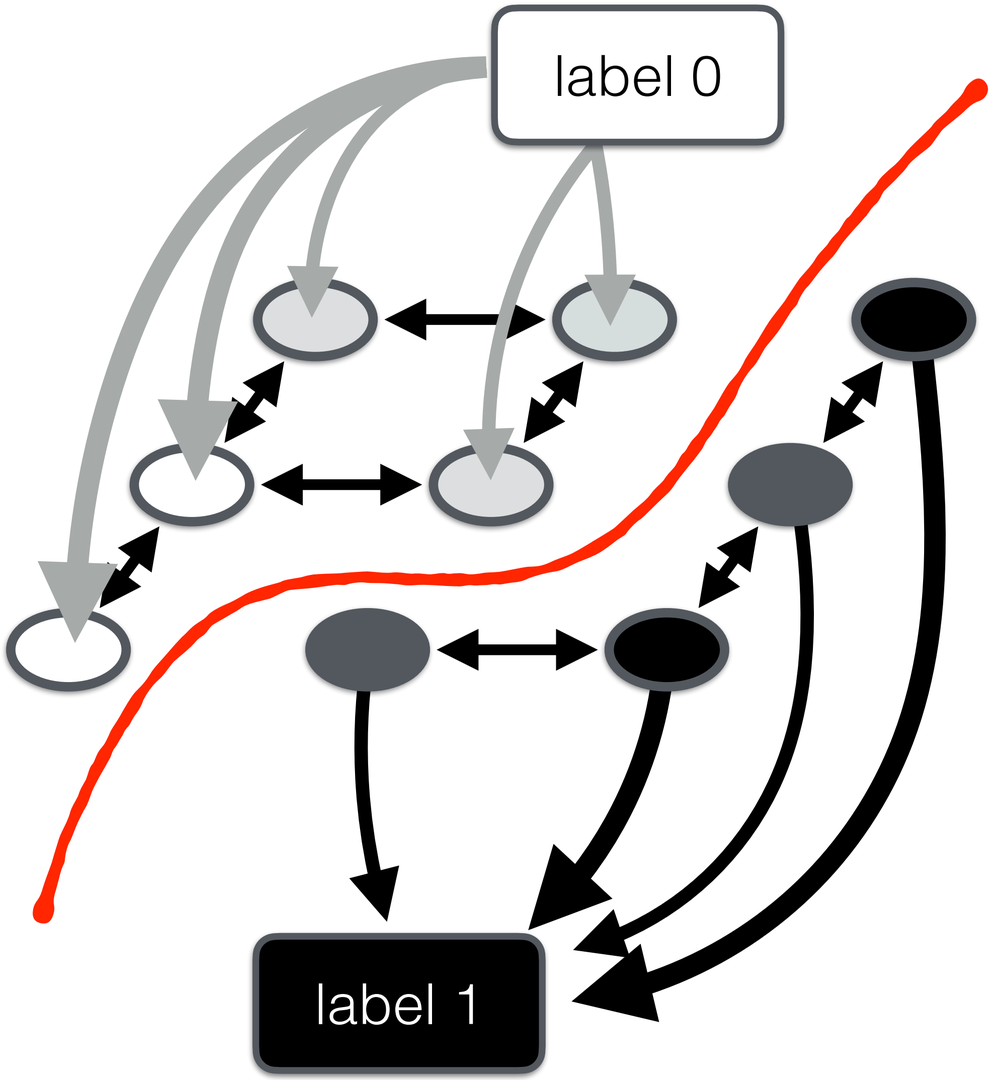
s-t Mincuts and MRFs



cut edges with low costs

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs

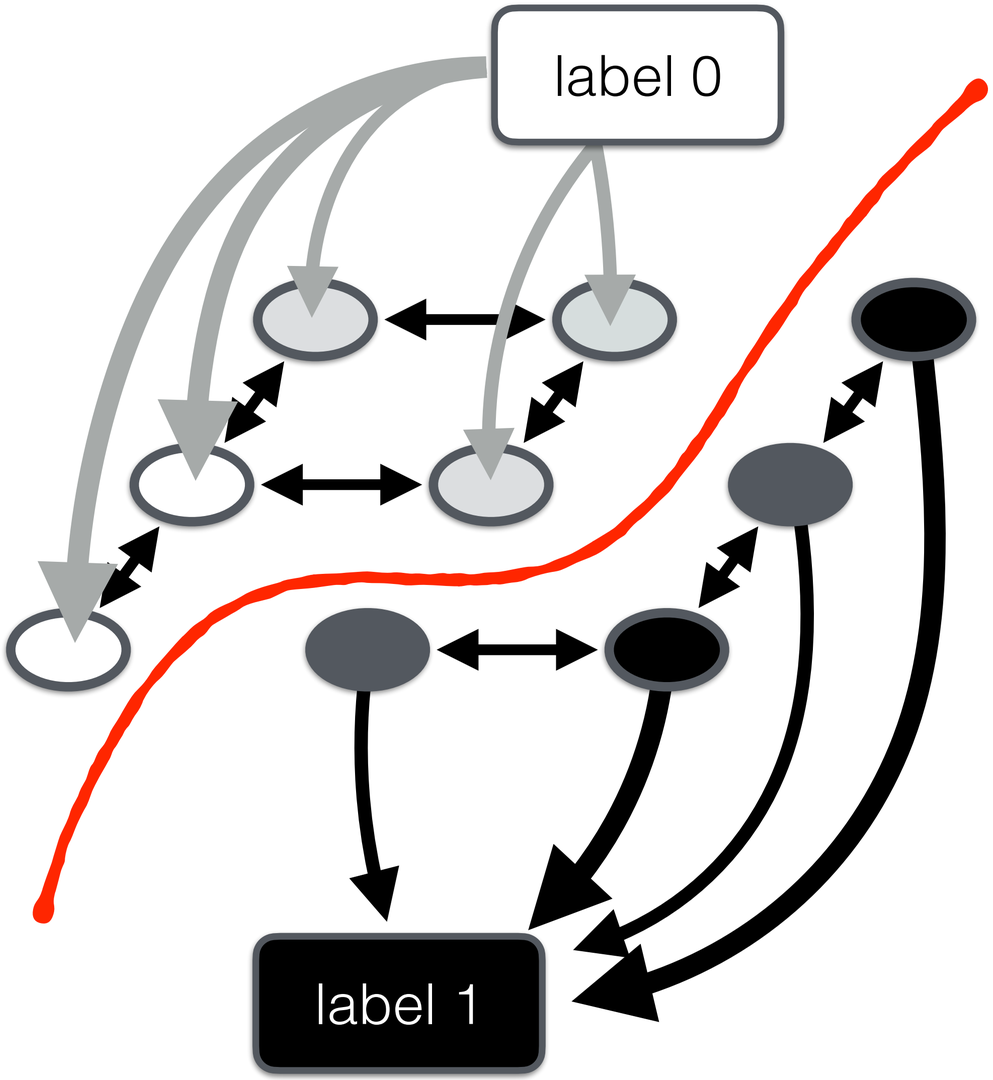


cut edges with low costs

resulting components = labelling

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Mincuts and MRFs



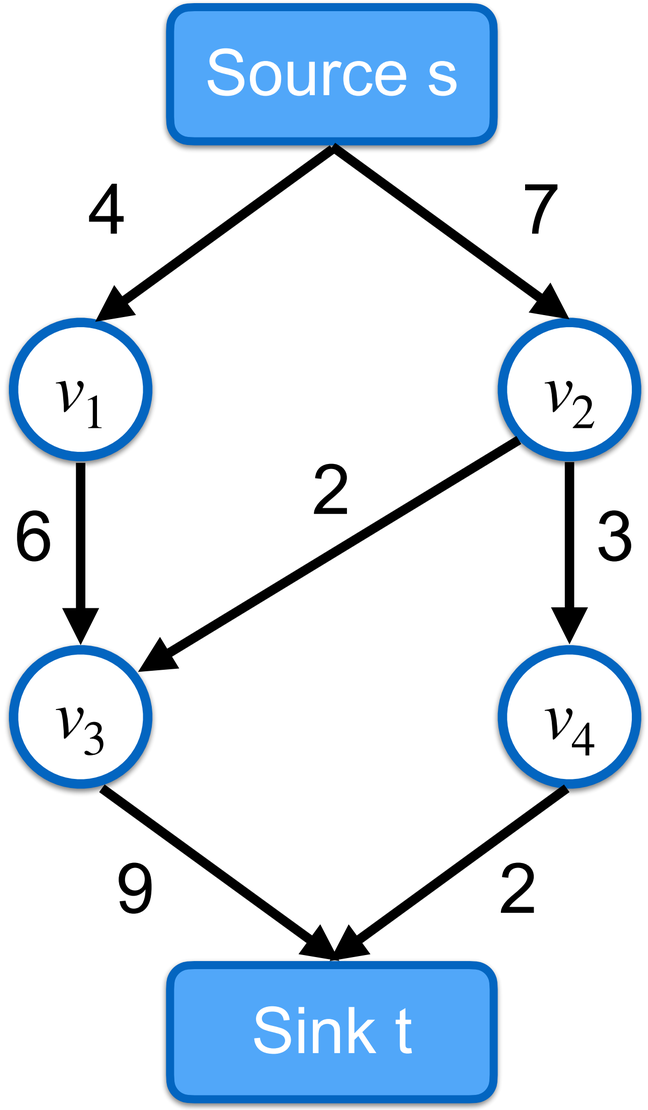
cut edges with low costs

resulting components = labelling

How to compute the s-t mincut?

slide credit: Václav Hlaváč, Bastian Leibe, Yuri Boykov

s-t Maxflow



terminal nodes s, t

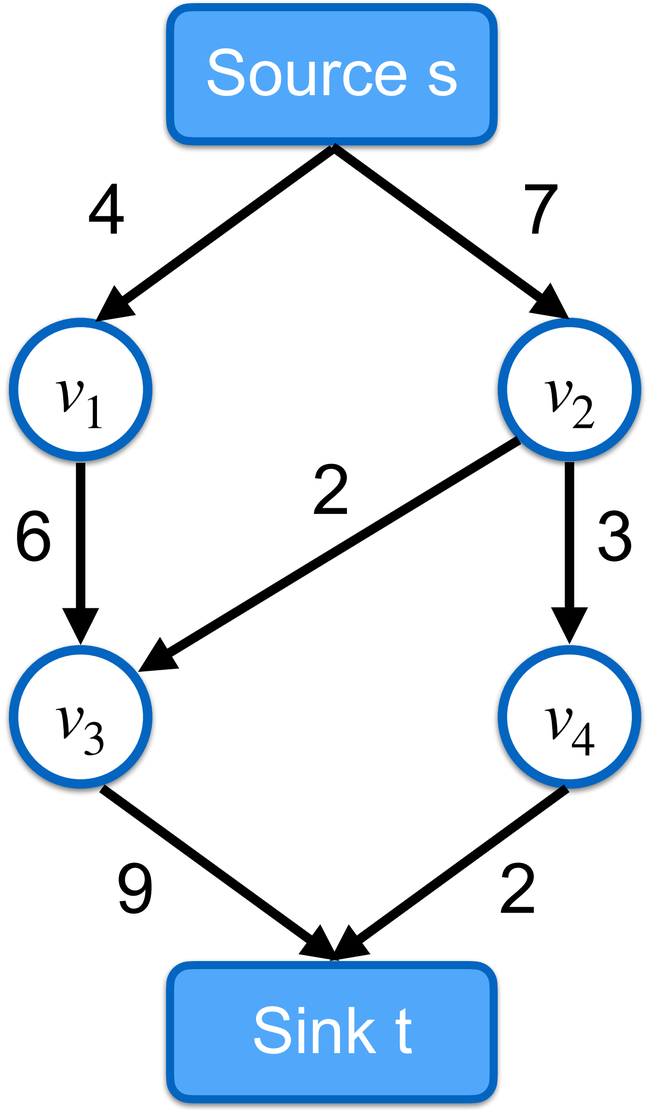
vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

capacities $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow



terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

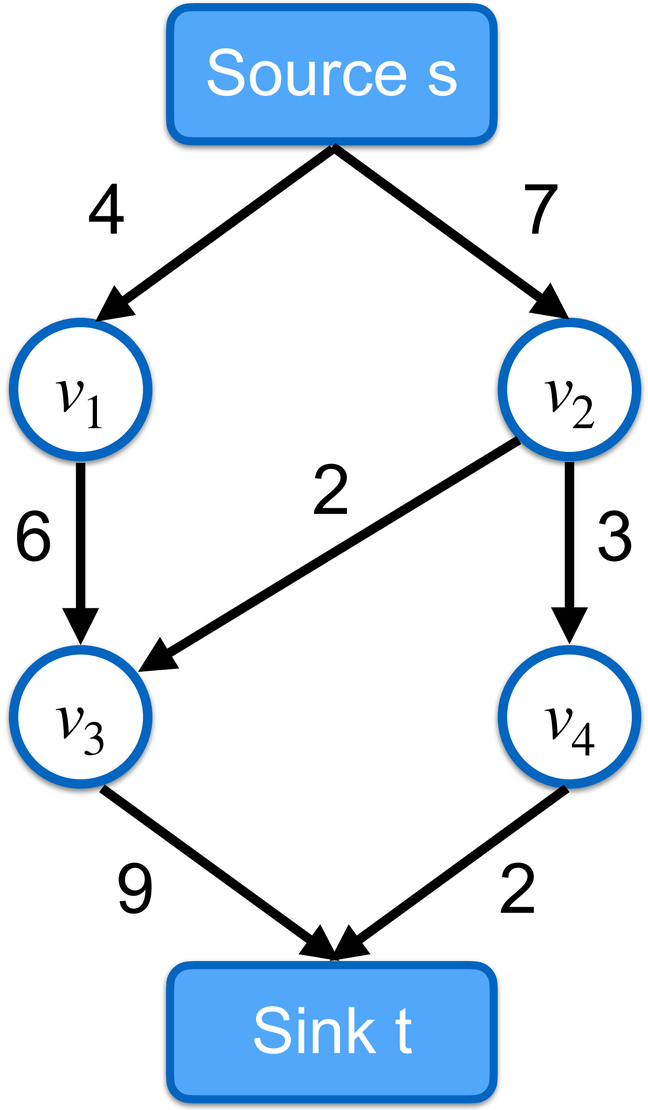
edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

capacities $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

s-t flow: function $f : E \rightarrow \mathbb{R}$ s.t.

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow



terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

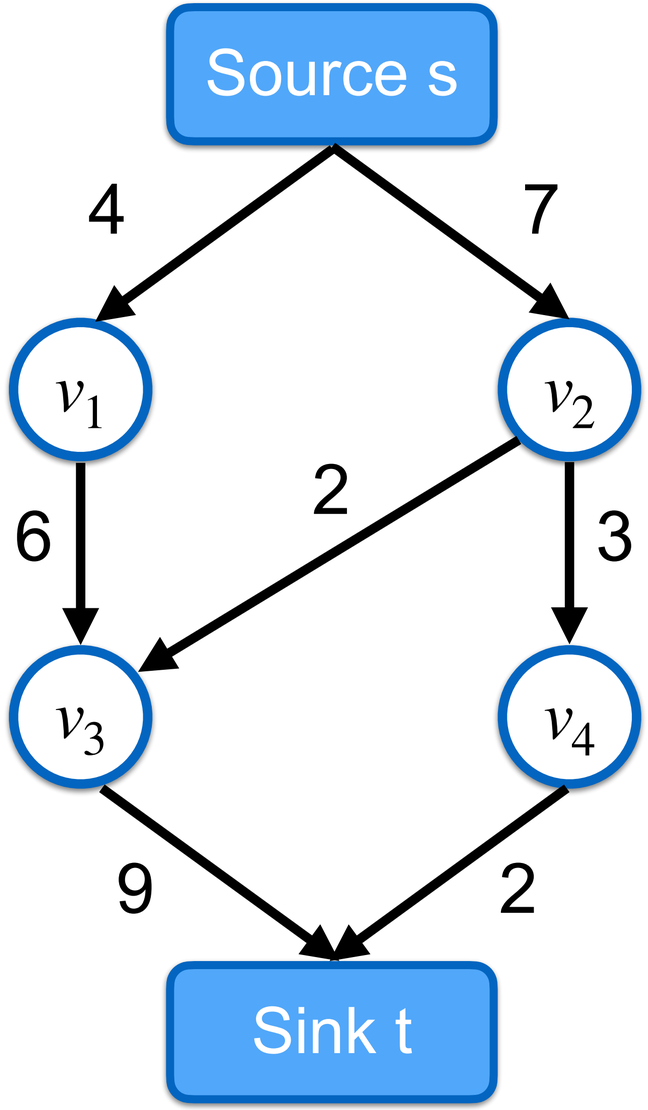
capacities $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

s-t flow: function $f : E \rightarrow \mathbb{R}$ s.t.

- capacity constraint: $f(e) \leq c(e) \quad \forall e \in E$

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow



terminal nodes s, t

vertices $V = \{v_1, \ldots, v_n\} \cup \{s, t\}$

edges $E = \{(s, v_1), (s, v_2), (v_1, v_3), \ldots\}$

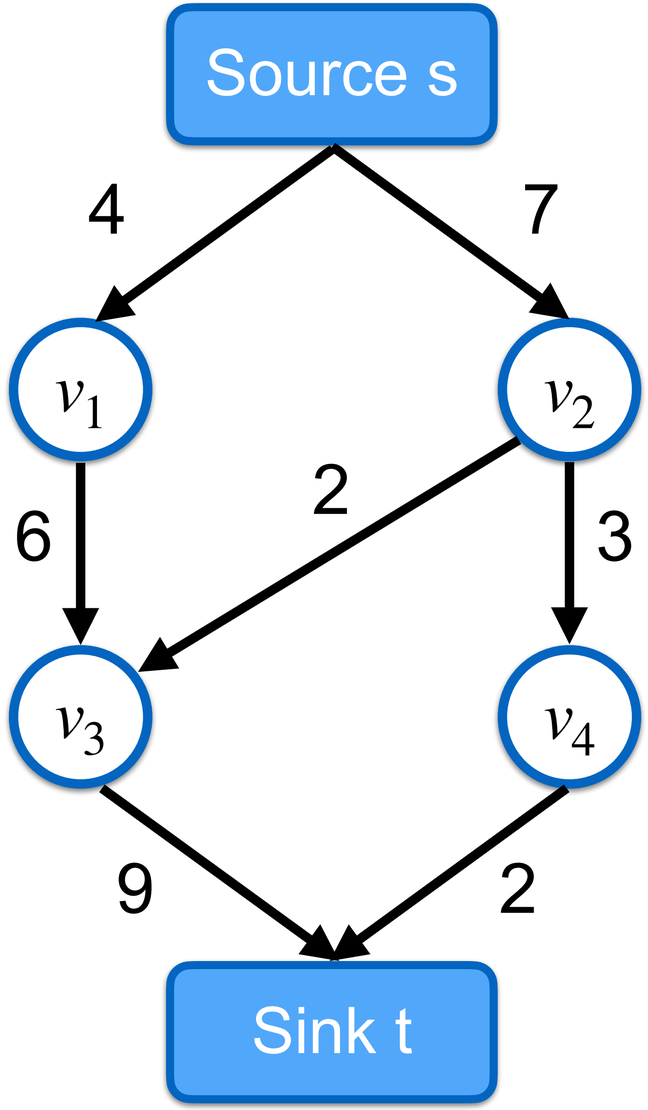
capacities $C = \{c(s, v_1), c(s, v_2), c(v_1, v_3), \ldots\}$

s-t flow: function $f : E \rightarrow \mathbb{R}$ s.t.

- capacity constraint: $f(e) \leq c(e) \quad \forall e \in E$
- flow conservation:
$$\sum_u f((u, v)) = \sum_u f((v, u)) \quad \forall v \in V \backslash \{s, t\}$$

slide credit: Václav Hlaváč, Bastian Leibe

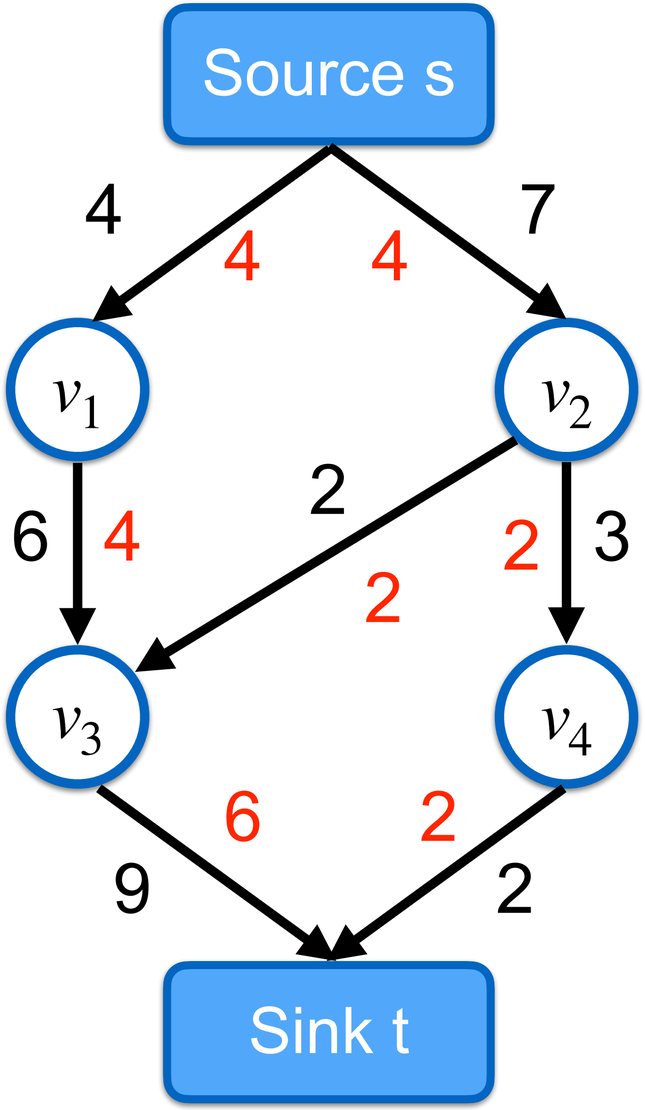
s-t Maxflow



s-t maxflow:
largest outgoing flow at s
= largest incoming flow at t

slide credit: Václav Hlaváč, Bastian Leibe

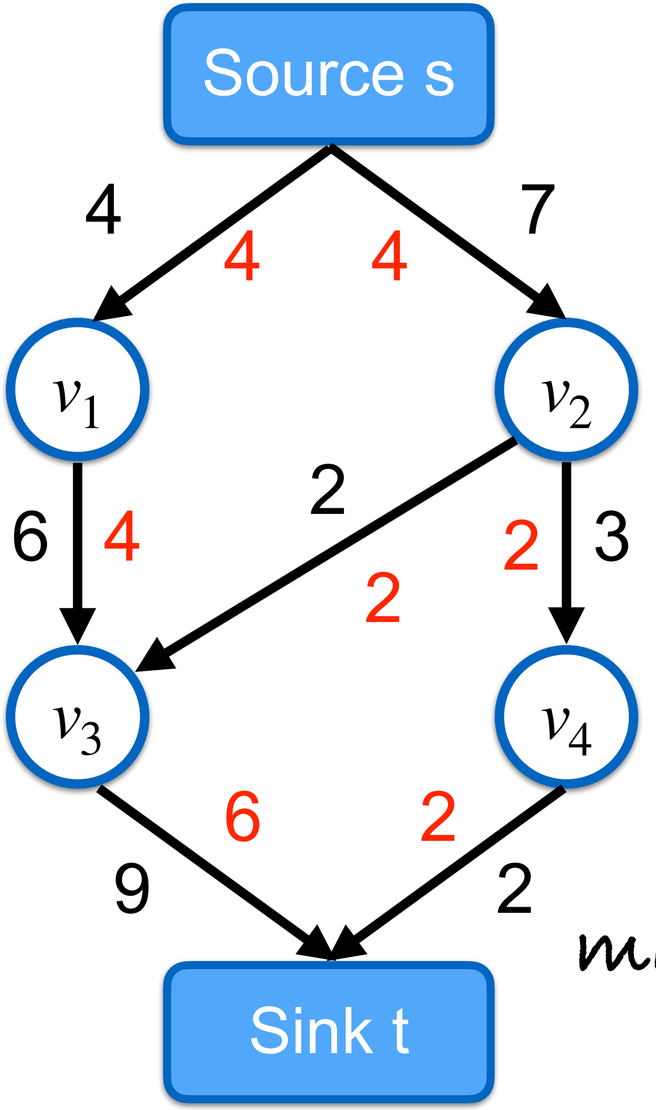
s-t Maxflow



s-t maxflow:
largest outgoing flow at s
= largest incoming flow at t

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow

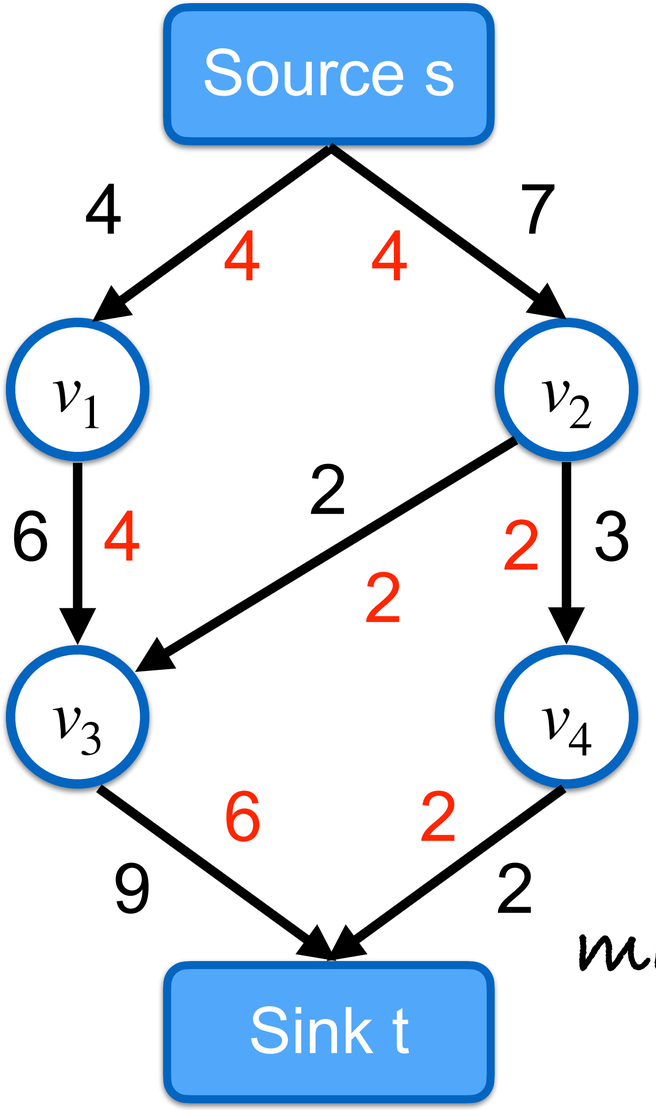


s-t maxflow:
largest outgoing flow at s
= largest incoming flow at t

$max\ flow = 8$

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow

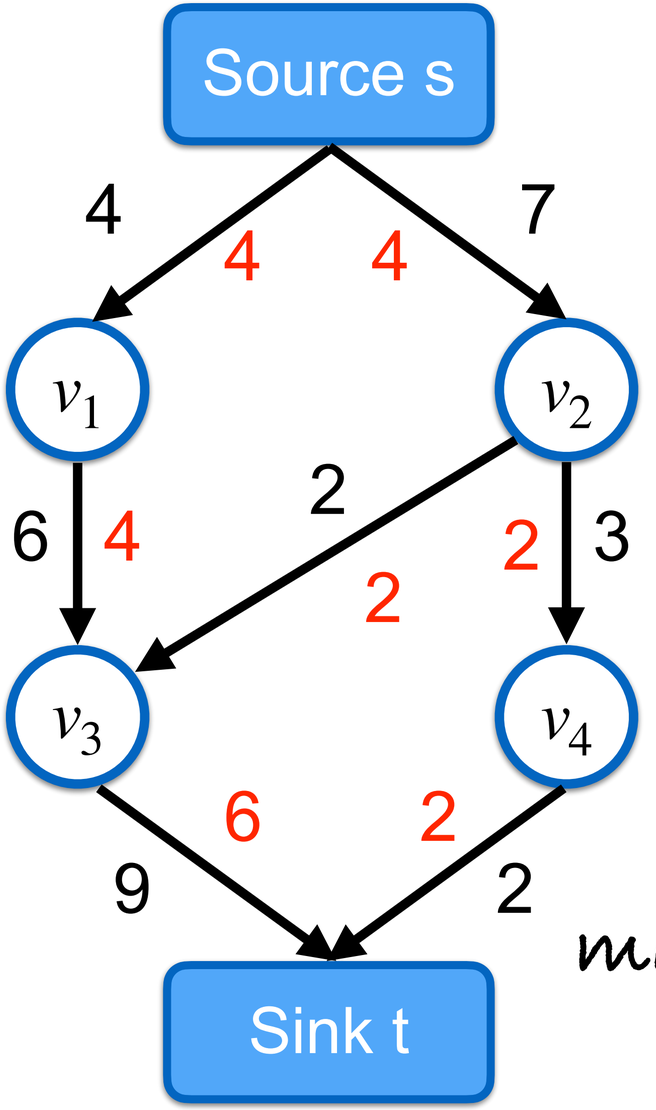


s-t maxflow:
largest outgoing flow at s
= largest incoming flow at t

$$max\ flow = 8 = min\ cut$$

slide credit: Václav Hlaváč, Bastian Leibe

s-t Maxflow



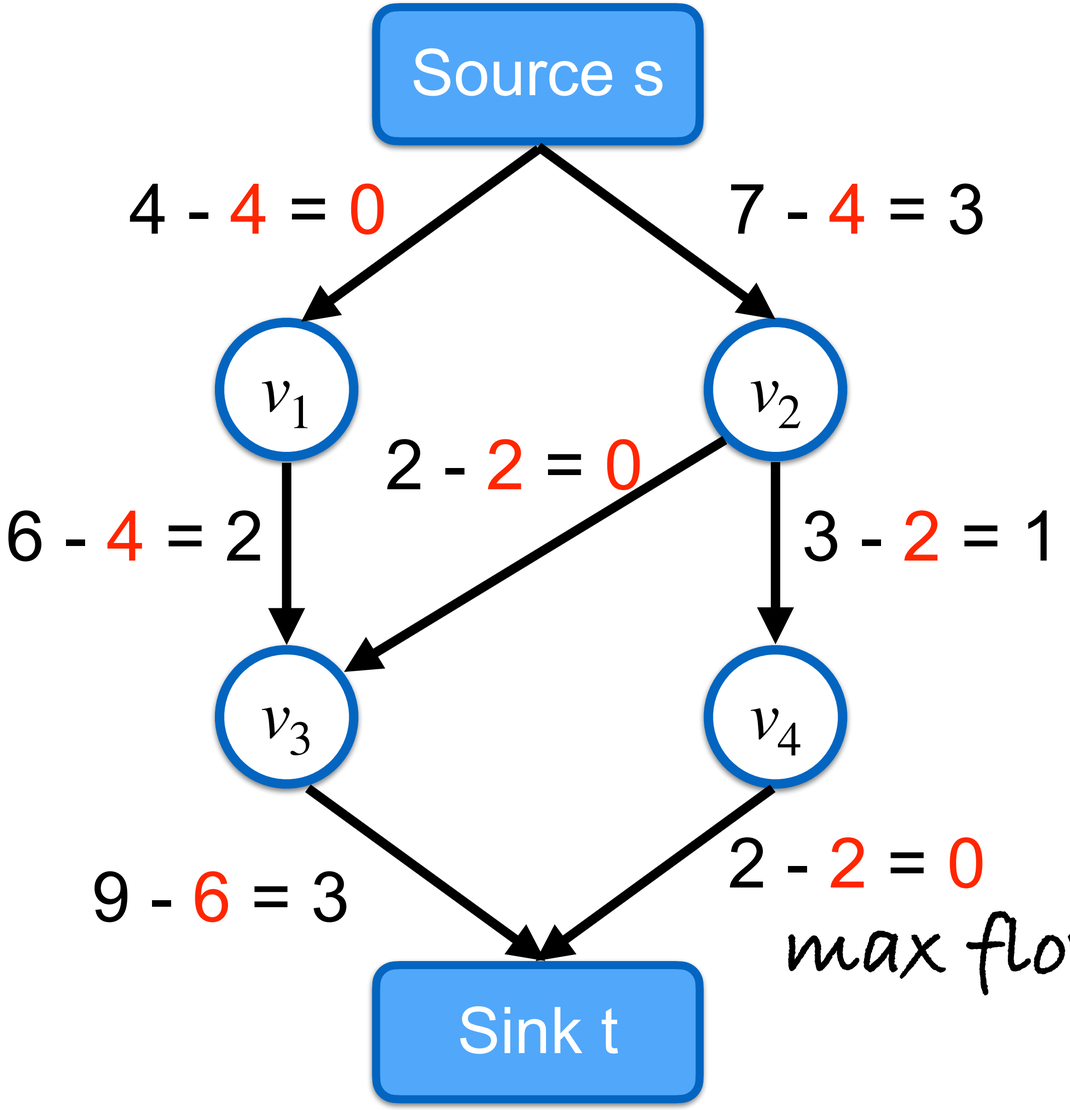
s-t maxflow:
largest outgoing flow at s
= largest incoming flow at t

Theorem:
maxflow = mincut

$$max\ flow = 8 = min\ cut$$

slide credit: Václav Hlaváč, Bastian Leibe

Maxflow = Mincut

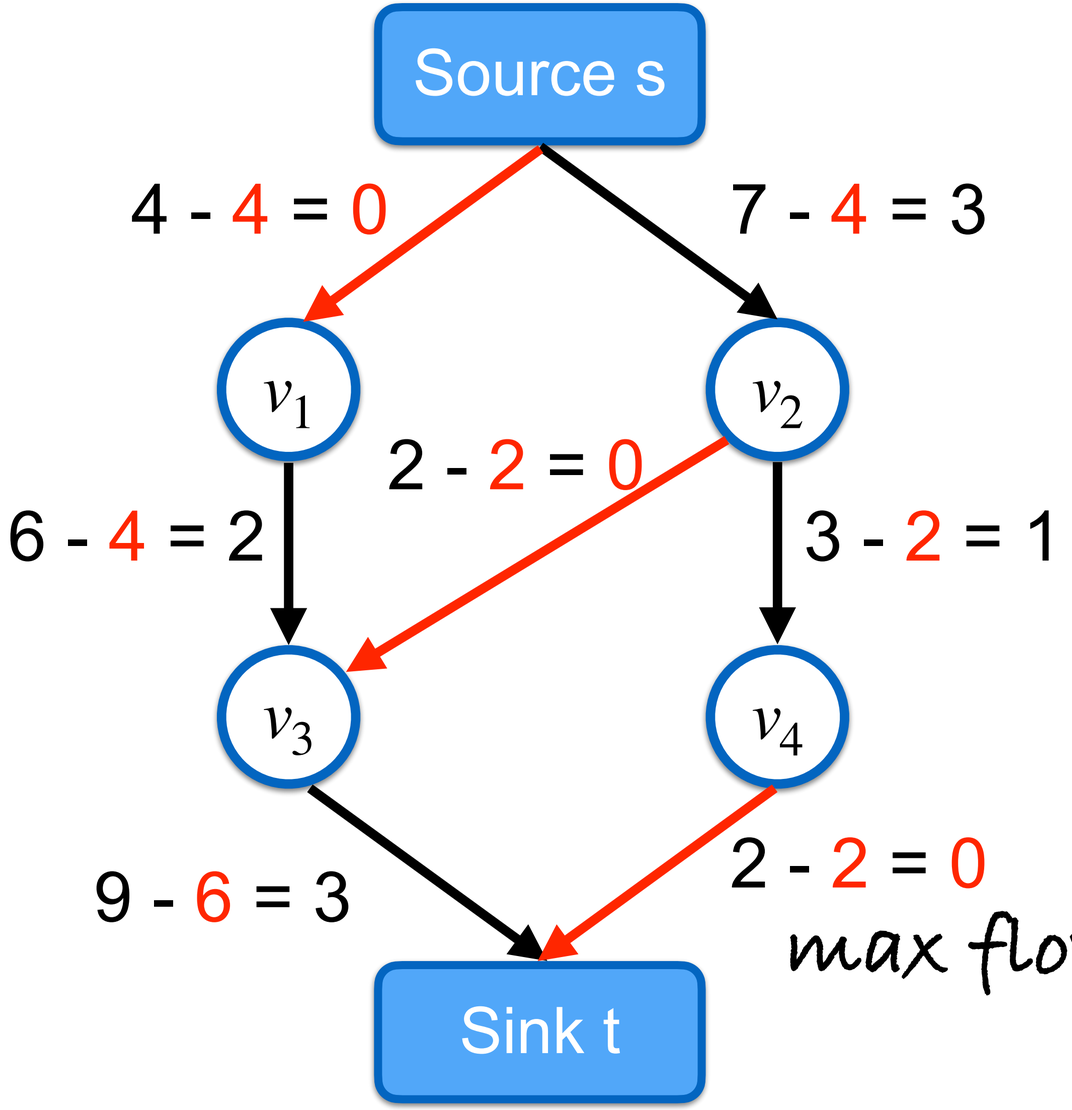


Theorem:
maxflow = mincut

max flow = 8 = min cut

slide credit: Václav Hlaváč, Bastian Leibe

Maxflow = Mincut



Theorem:
maxflow = mincut

max flow = 8 = min cut

slide credit: Václav Hlaváč, Bastian Leibe

Computing Maxflow

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

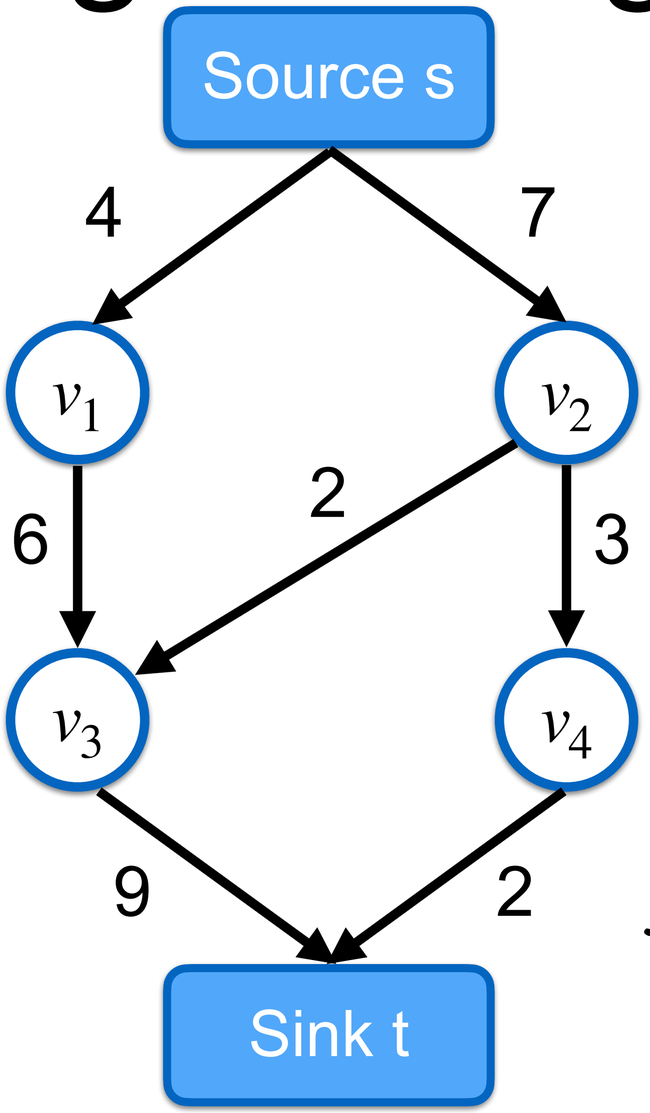
m: #edges

U: maximum edge weight

Algorithms assume non-negative edge weights

slide credit: Andrew Goldberg, Bastian Leibe

Augmenting Path-Based Algorithms

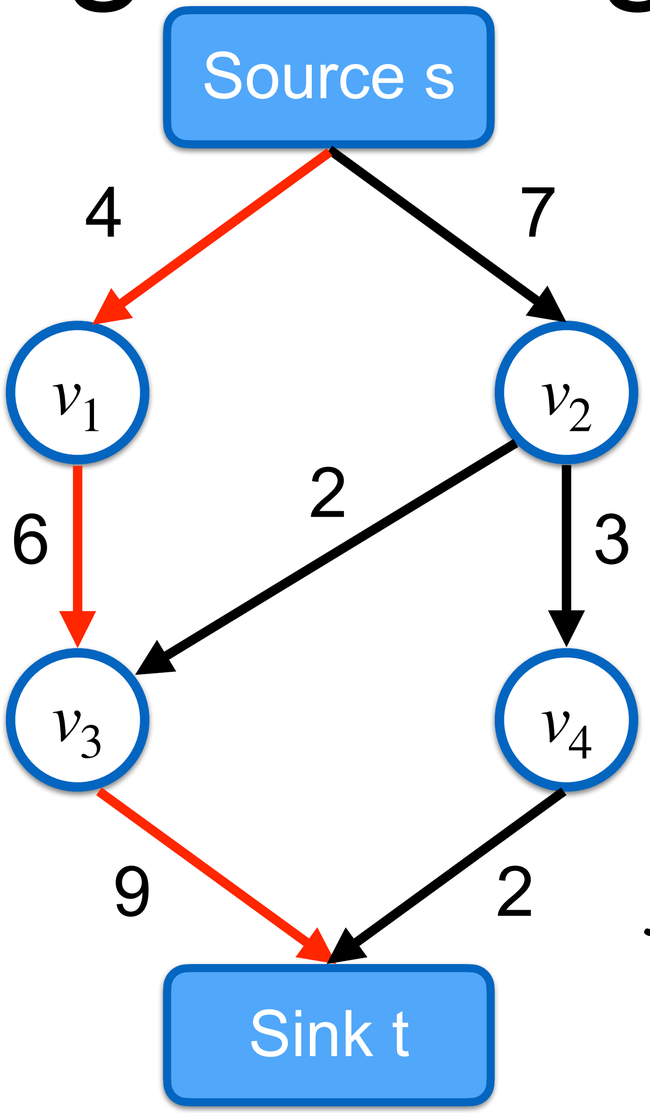


1. Find path with non-zero capacity

flow = 0

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

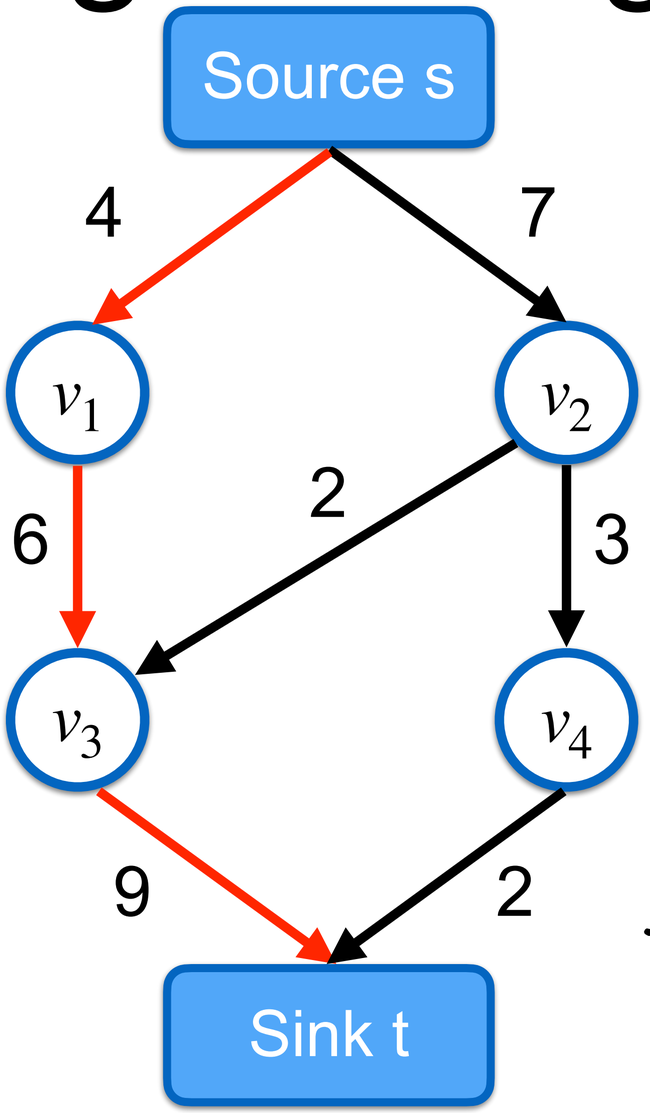


1. Find path with non-zero capacity

flow = 0

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

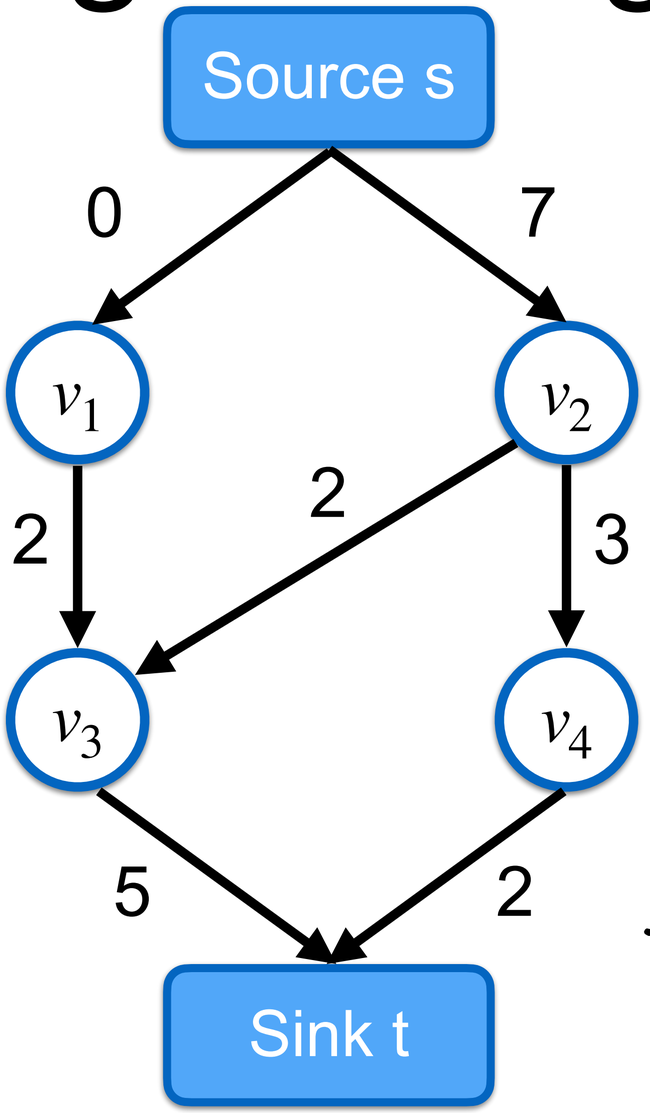


1. Find path with non-zero capacity
2. Maximize flow on path

$flow = 4$

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

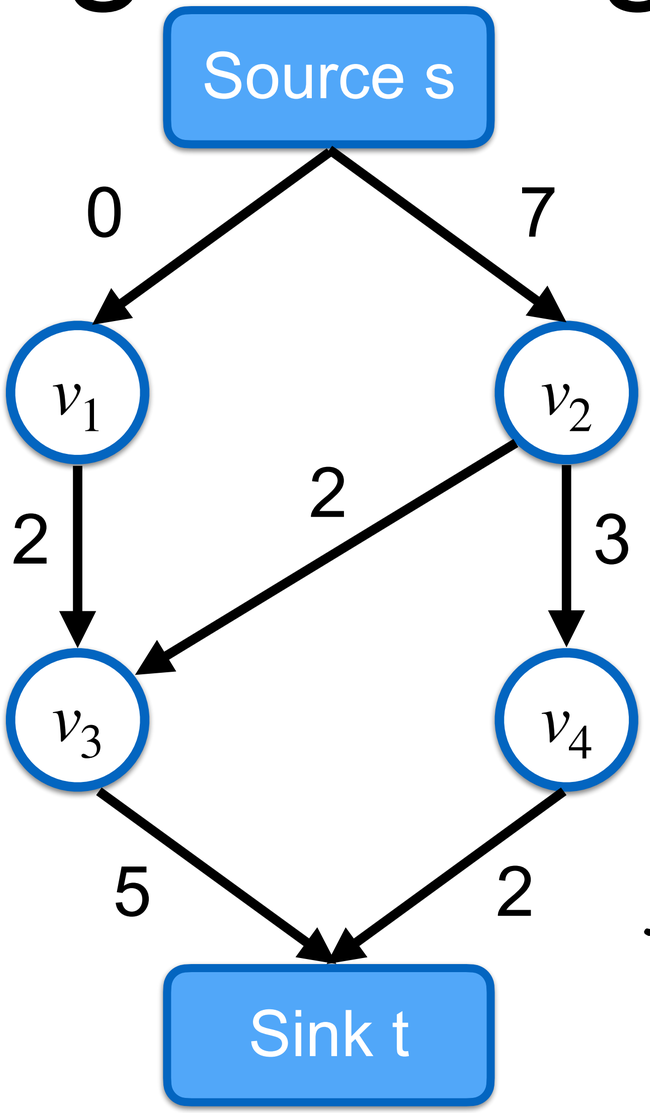


1. Find path with non-zero capacity
2. Maximize flow on path

$flow = 4$

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

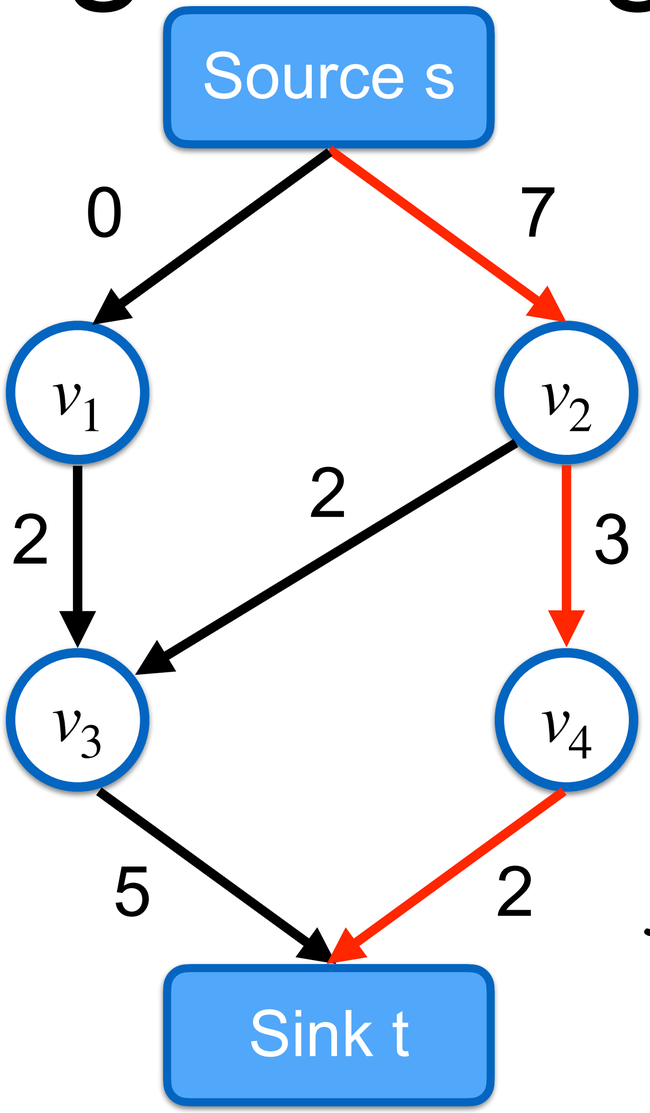


1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

$flow = 4$

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

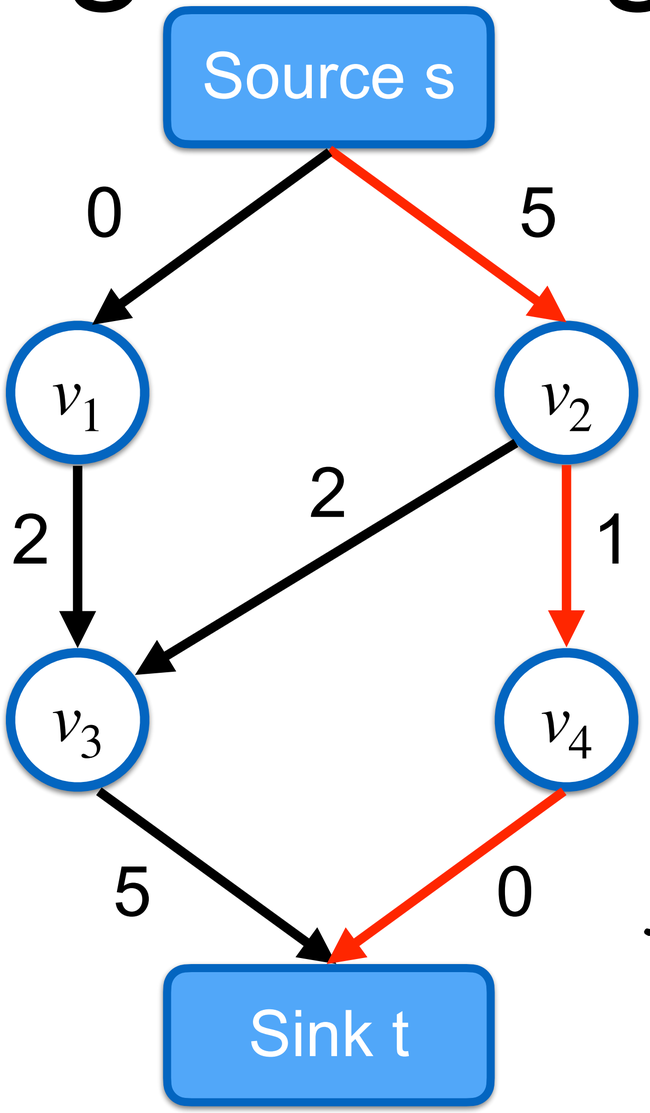


1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

$flow = 4$

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

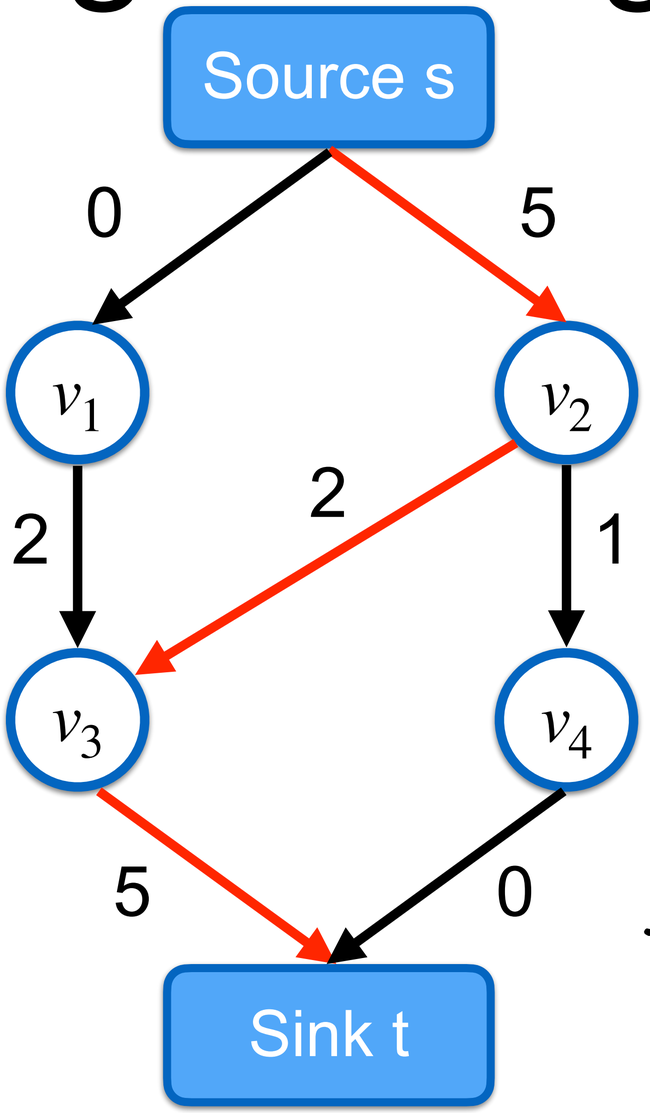


flow = 6

1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

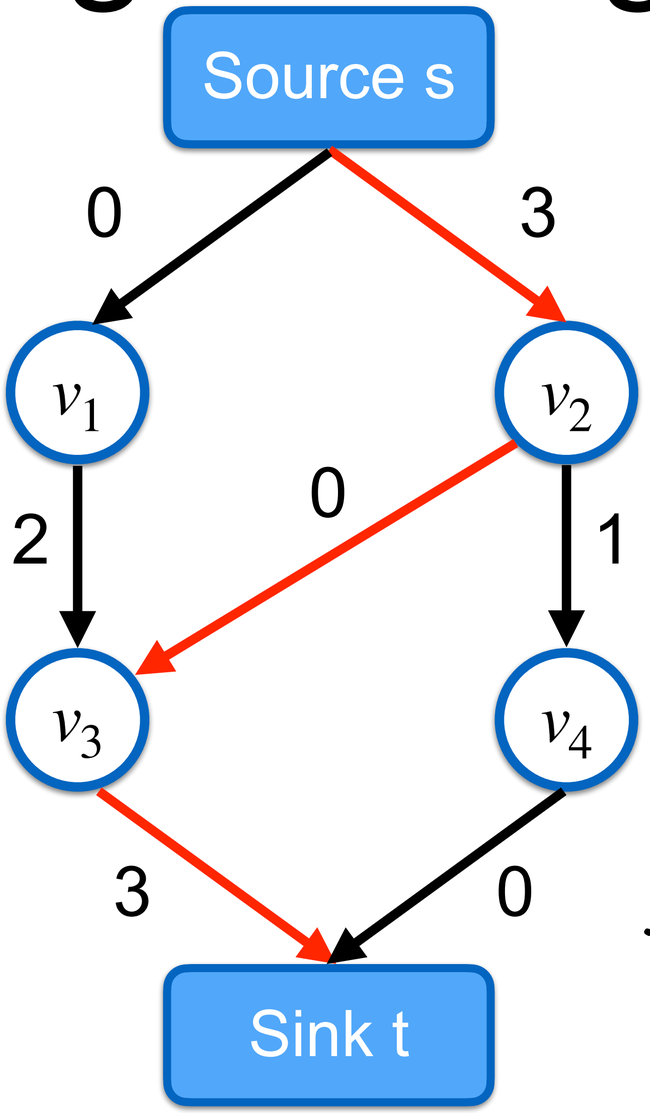


flow = 6

1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

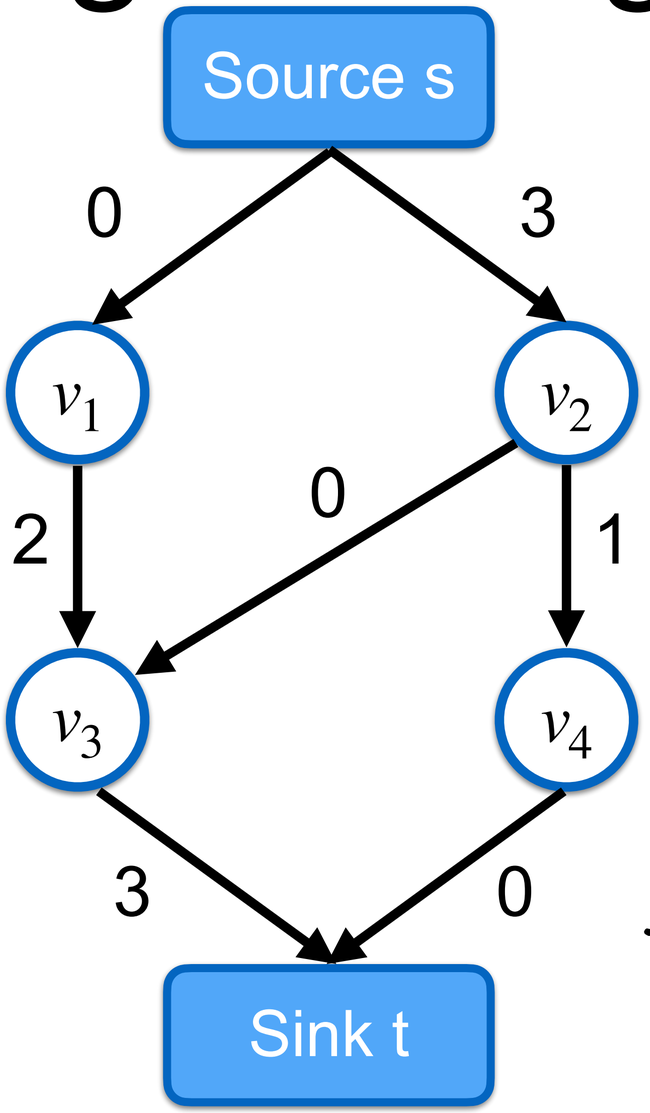


flow = 8

1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms

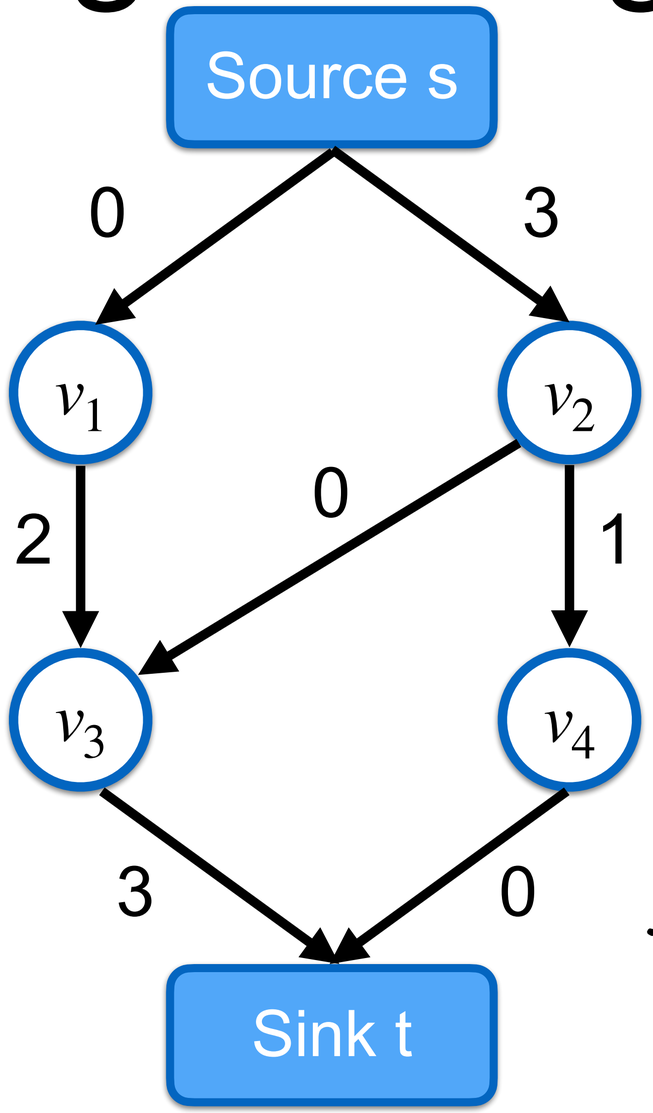


flow = 8

1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

slide credit: Václav Hlaváč, Bastian Leibe

Augmenting Path-Based Algorithms



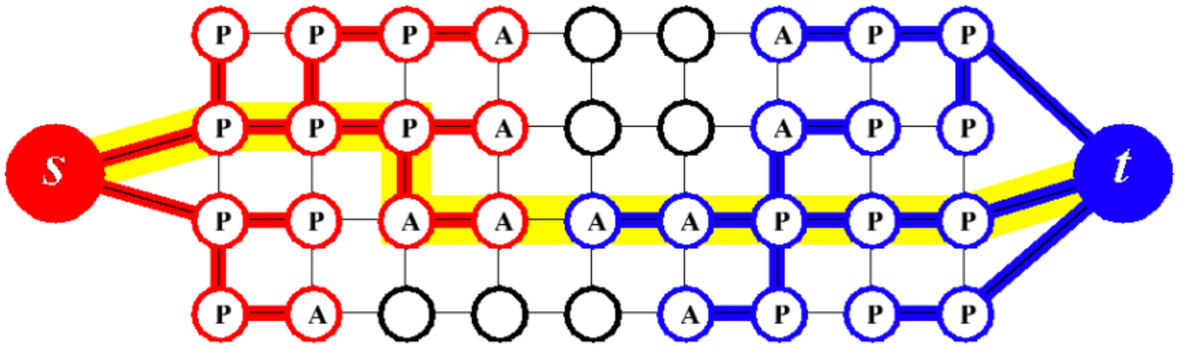
$flow = 8$

1. Find path with non-zero capacity
2. Maximize flow on path
3. Repeat until no such path exists

Assumes non-negative capacities

slide credit: Václav Hlaváč, Bastian Leibe

Maxflow on Images



- Popularized by Boykov & Kolmogorov
- Dual search tree augmenting path algorithm [Boykov & Kolmogorov, PAMI 2004]
  - Efficient approach for finding approximate shortest augmenting paths
  - High worst-case complexity
  - But works better than other methods on the grid graphs defined by images

slide credit: Pushmeet Kohli, Bastian Leibe

A Few More Words on Graph Cuts and MRFs

- Graph cuts globally optimal for **binary energies** that are **submodular**
- Multi-label problems with 3 or more labels are NP-hard
- Approximation algorithms extending graph cuts to multi-label case:
  - $\alpha\beta$-swap
  - α-expansion: iteratively solve binary graph cuts (one label vs. all), factor 2 approximation to optimal solution

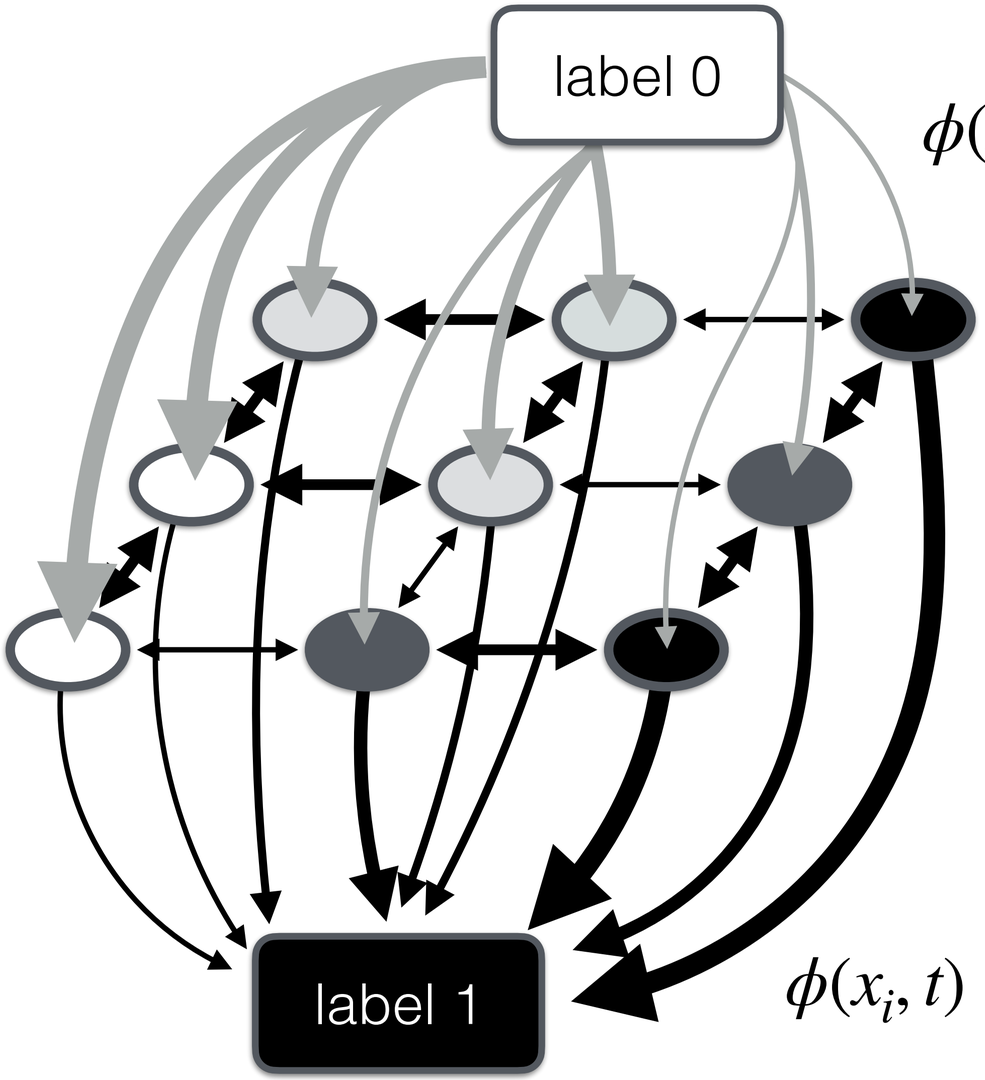
slide credit: Bastian Leibe

A Few More Words on Graph Cuts and MRFs

- Minimum cuts can penalize large segments
- Avoid this problem through normalization by component size
- Normalized cut: $\frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$ ($\text{assoc}(A,V)$ = sum of weights of all edges in V that touch A)
- NP-hard, approximation via generalized Eigenvalue problem [Shi & Malik, Normalized Cuts and Image Segmentation, PAMI 2000]

slide credit: Václav Hlaváč

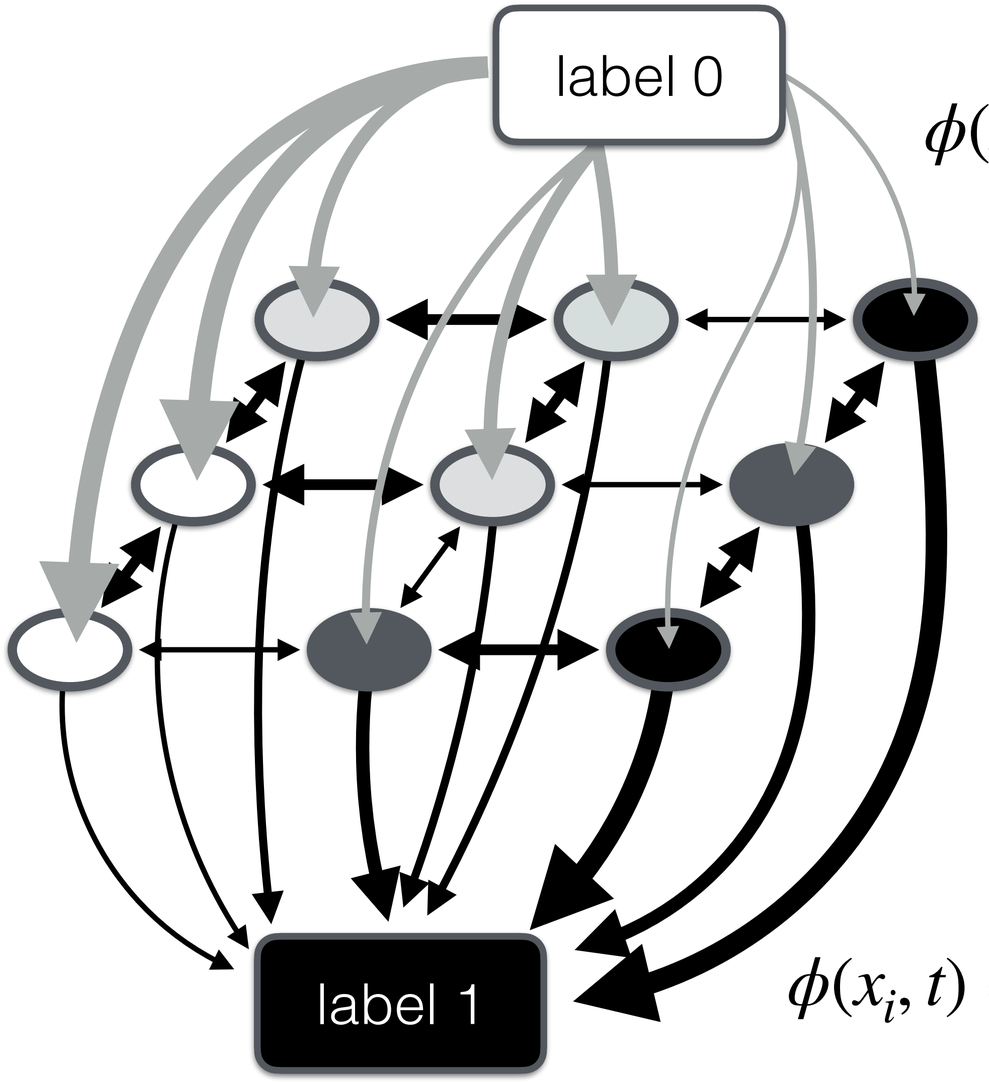
Application: Interactive Segmentation via GrabCut



$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

$$\phi(x_i, t) \propto exp(-\|I_i - I_t\|^2/2\sigma^2)$$

Application: Interactive Segmentation via GrabCut



$$\phi(x_i, s) \propto exp(-\|I_i - I_s\|^2/2\sigma^2)$$

How to obtain models for foreground and background?

$$\phi(x_i, t) \propto exp(-\|I_i - I_t\|^2/2\sigma^2)$$

Application: Interactive Segmentation via GrabCut

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Application: Interactive Segmentation via GrabCut

train GMM for foreground and background

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Application: Interactive Segmentation via GrabCut

train GMM for foreground and background

Automatic

Segmentation

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Application: Interactive Segmentation via GrabCut

train GMM for foreground and background

graph cut

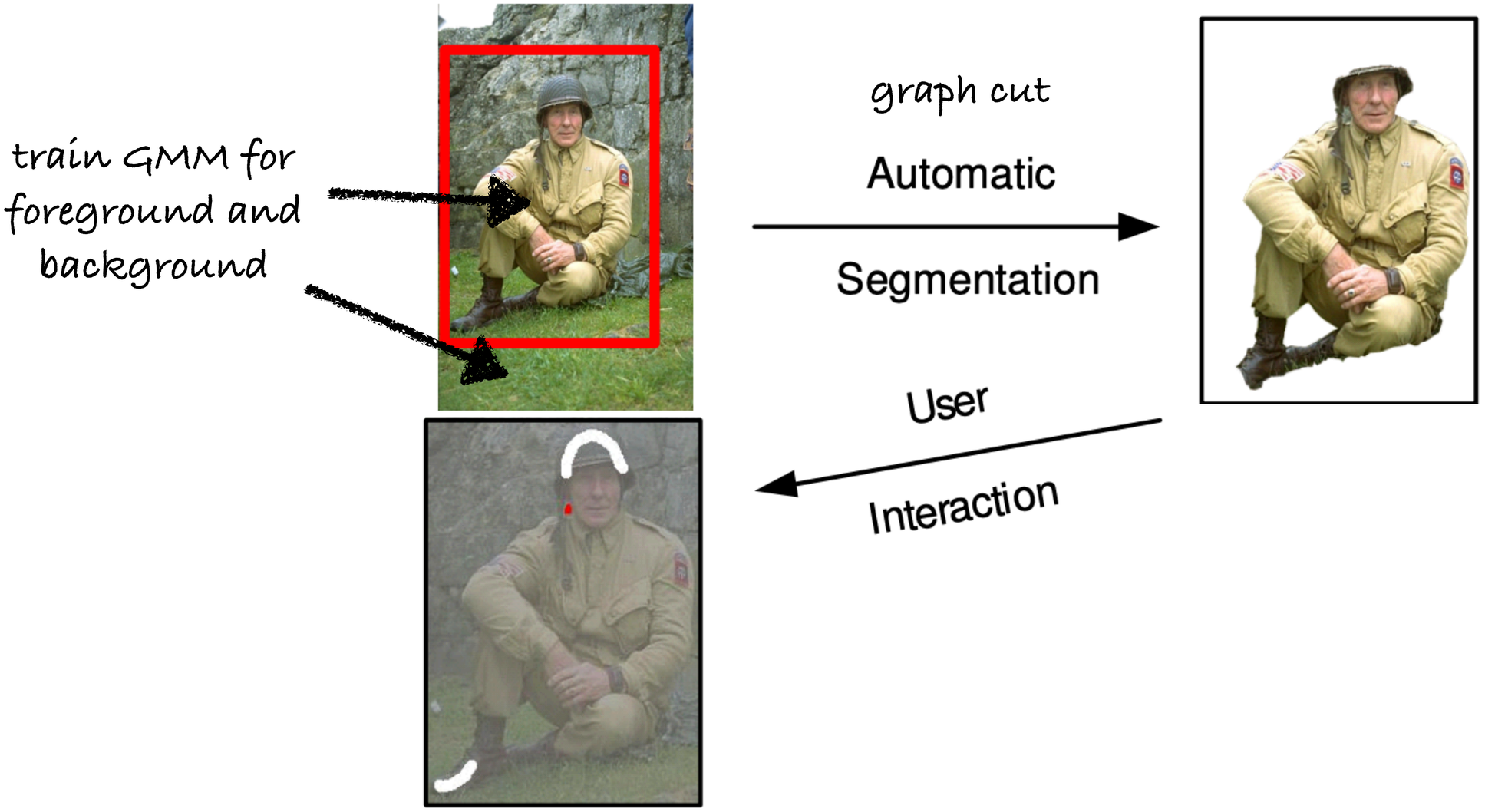
Automatic

Segmentation

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

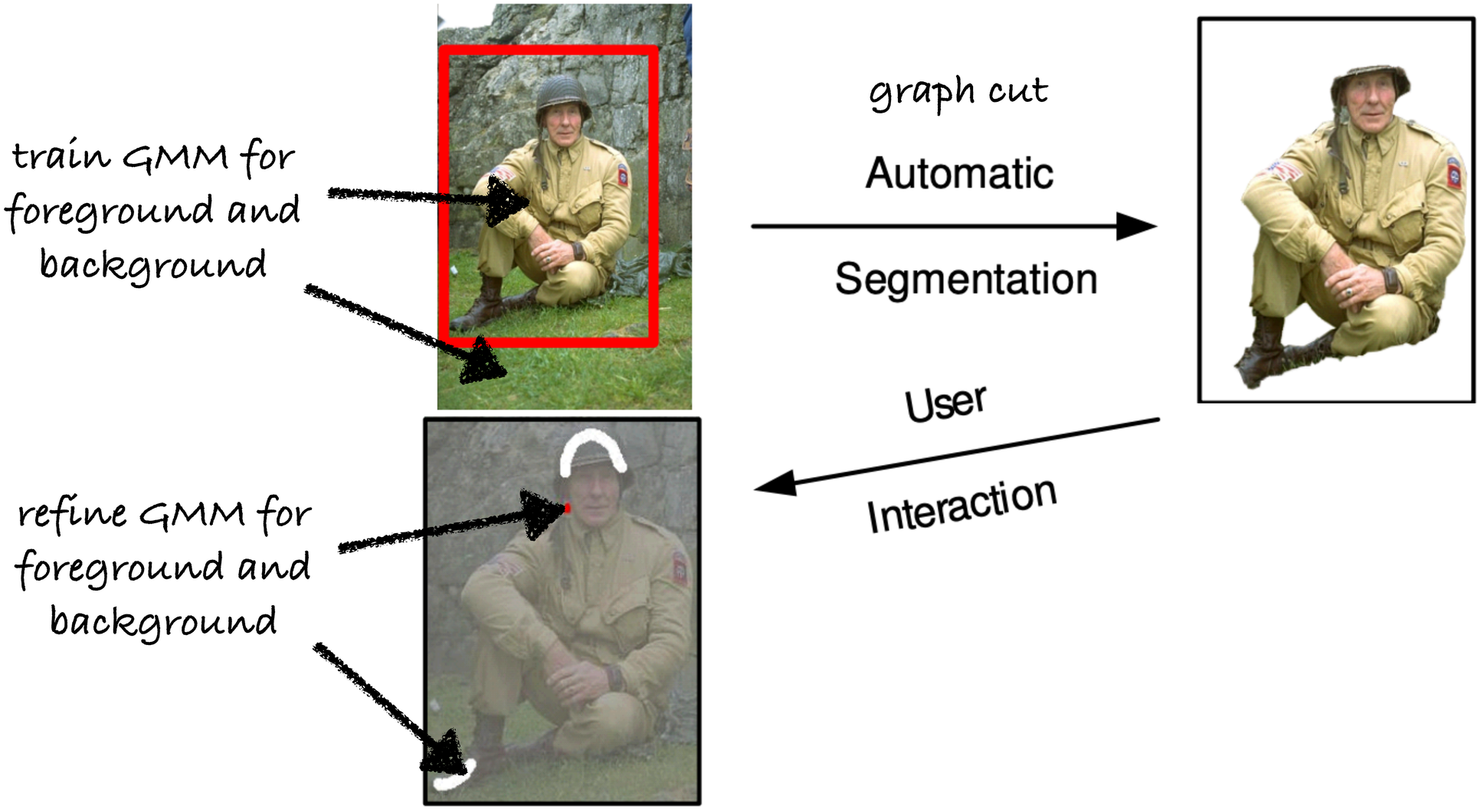
Application: Interactive Segmentation via GrabCut



slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

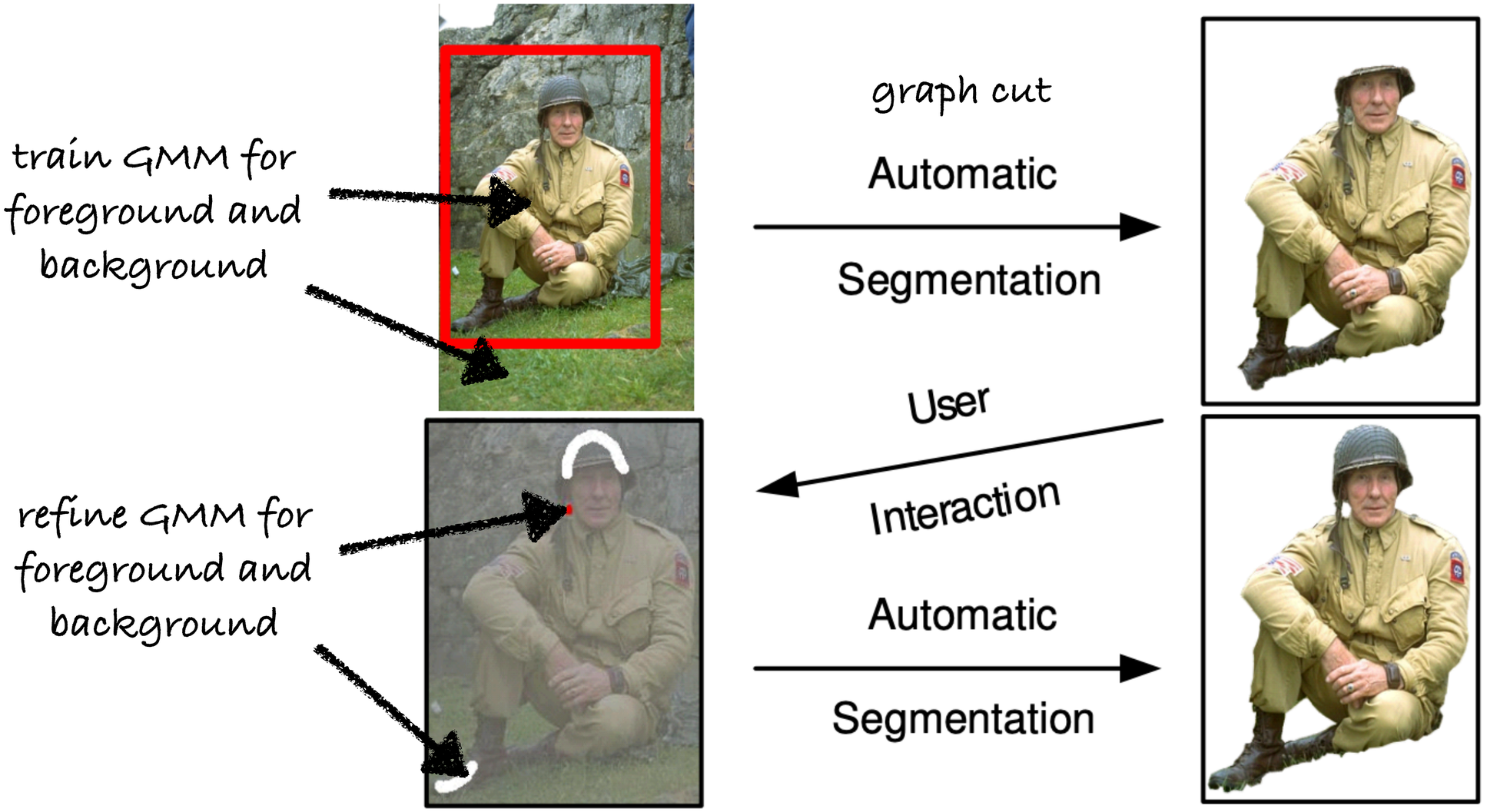
Application: Interactive Segmentation via GrabCut



slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Application: Interactive Segmentation via GrabCut



slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Iterated Graph Cuts

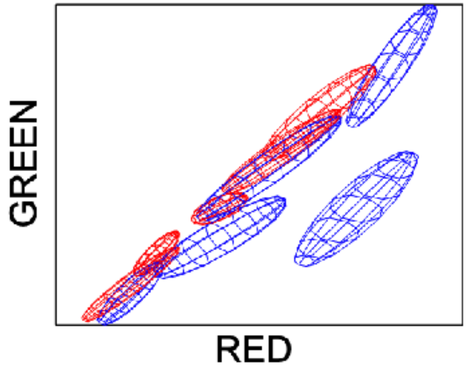
Input

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Iterated Graph Cuts

initial Gaussian Mixture Model (GMM)



Input

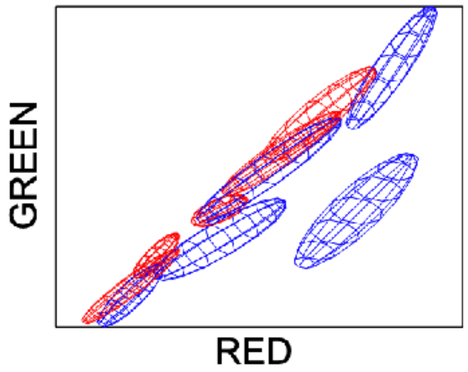
slide credit: Carsten Rother

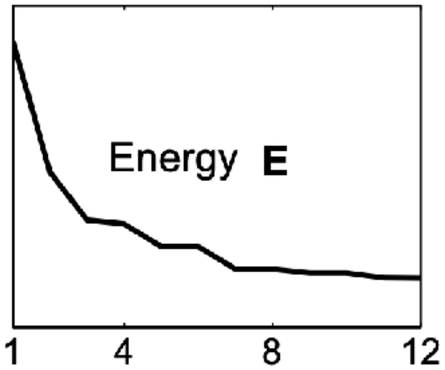
[Rother, Kolmogorov, Blake, “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Iterated Graph Cuts

initial Gaussian Mixture Model (GMM)

Input





EM optimization:

- Apply graph cut with current GMM
- Re-estimate GMM based on segmentation

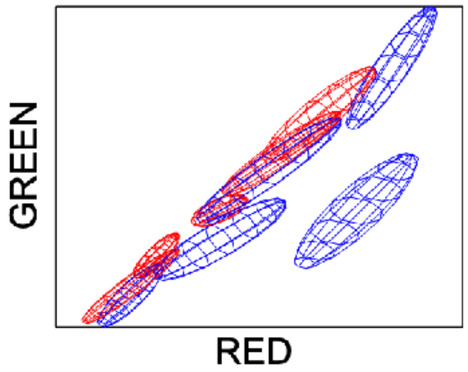
slide credit: Carsten Rother

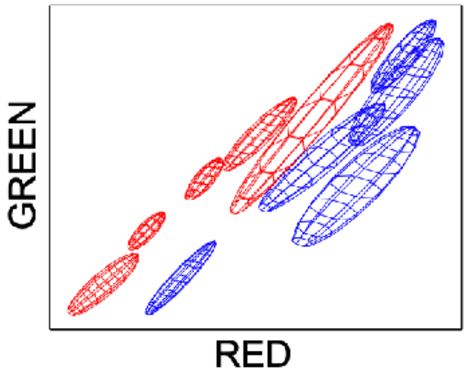
[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Iterated Graph Cuts

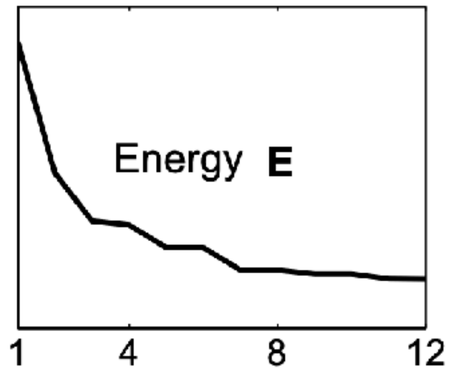
initial Gaussian Mixture Model (GMM)

Input





final GMM



EM optimization:

- Apply graph cut with current GMM
- Re-estimate GMM based on segmentation

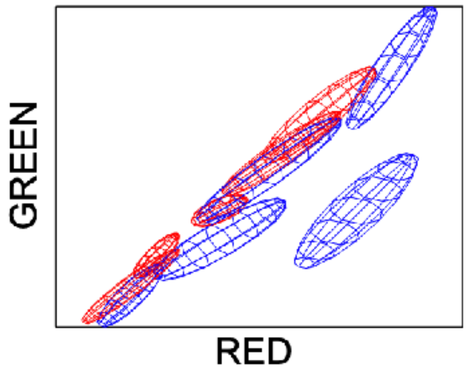
slide credit: Carsten Rother

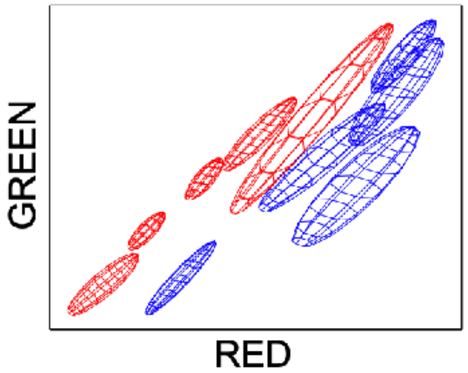
[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Iterated Graph Cuts

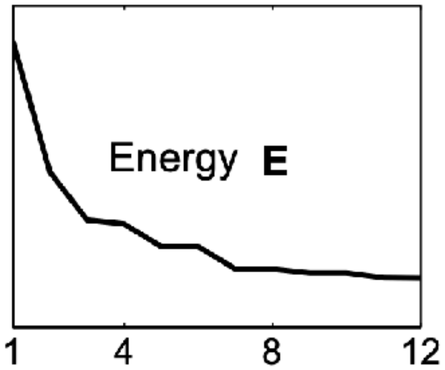
initial Gaussian Mixture Model (GMM)

Input





final GMM



EM optimization:

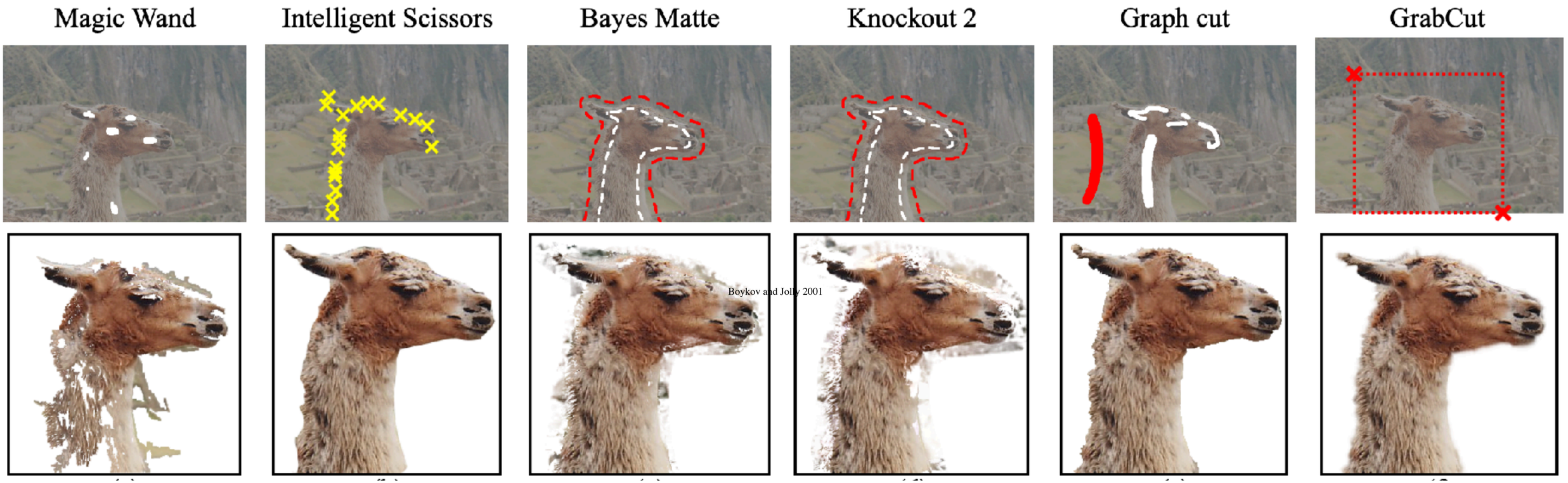
- Apply graph cut with current GMM
- Re-estimate GMM based on segmentation

Segmentation

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

Application: Interactive Segmentation via GrabCut



[Mortensen and Barrett 1995]

[Chuang et al. 2001]

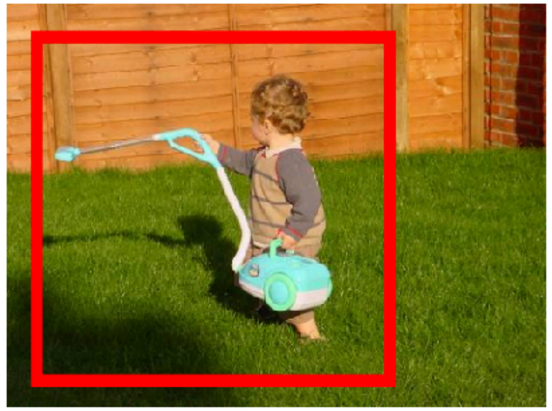
[Corel Corp. 2002]

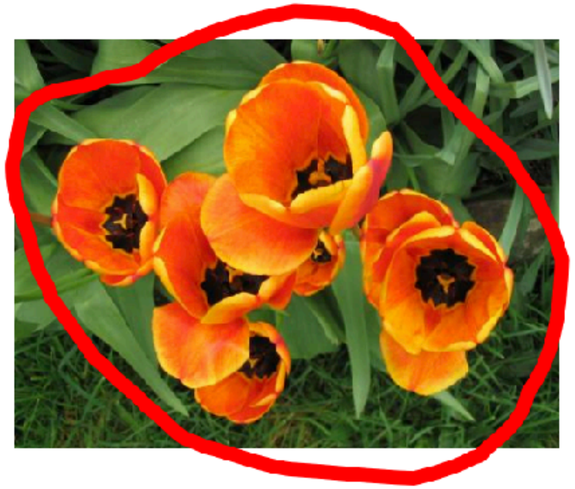
[Boykov and Jolly 2001]

slide credit: Carsten Rother

[Rother, Kolmogorov, Blake, "GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts, SIGGRAPH 2004]

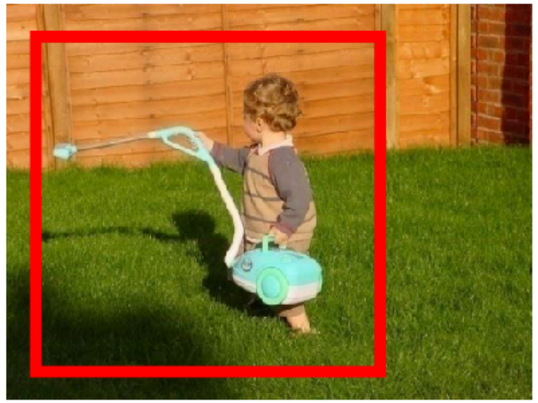
Quiz: Which Examples Are Easy, Which Are Hard?





slide credit: Václav Hlaváč, Carsten Rother

Easier Examples






slide credit: Václav Hlaváč, Carsten Rother

Harder Examples

slide credit: Václav Hlaváč, Carsten Rother

Other Examples for Graph Cuts

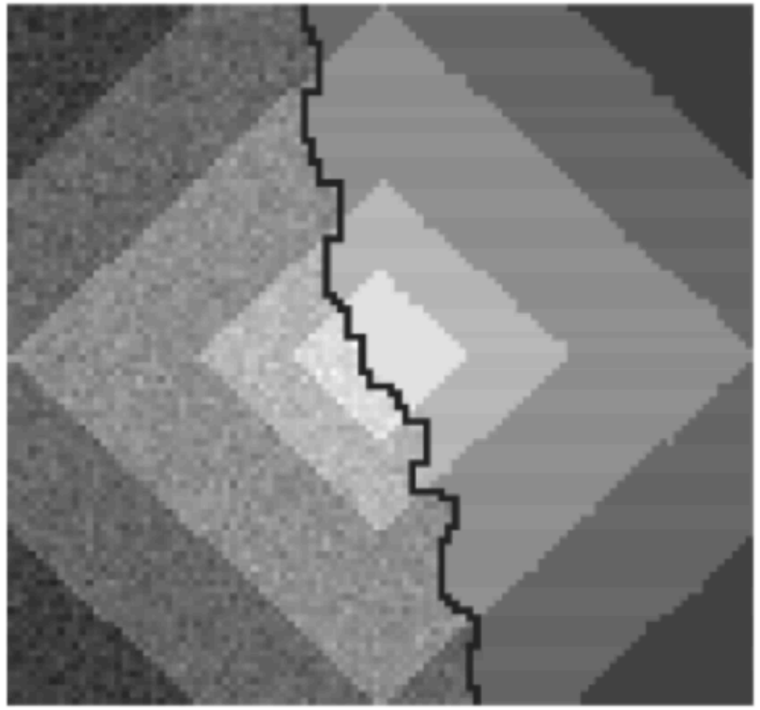


Image Restoration

Stereo Disparity Estimation

slide credit: Václav Hlaváč

Graph Cuts - Summary

- **Pros**:
  - Powerful approach, applicable to many labelling problems
  - Based on probabilistic model (MRF)
  - Efficient algorithms available for many computer vision / image analysis problems

slide credit: Bastian Leibe

Graph Cuts - Summary

- **Pros**:
  - Powerful approach, applicable to many labelling problems
  - Based on probabilistic model (MRF)
  - Efficient algorithms available for many computer vision / image analysis problems
- **Cons**:
  - Graph cuts can only (optimally) solve limited range of problems (binary submodular energy functions), not full range of what can be modeled by MRFs)
  - Only approximations for multi-label case

slide credit: Bastian Leibe