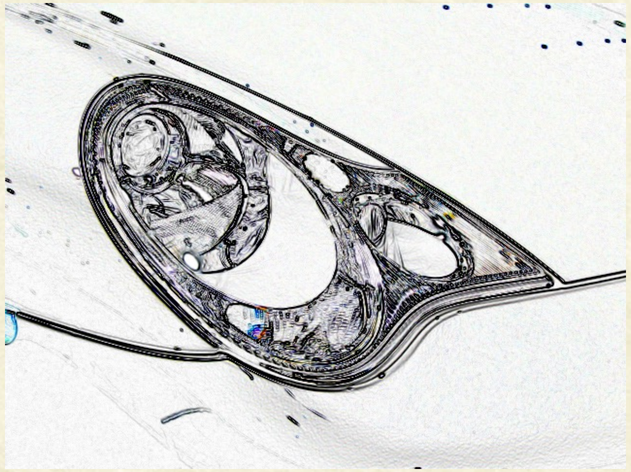
CS7.404: Digital Image Processing

Monsoon 2023: Image Segmentation - 2


Anoop M. Namboodiri

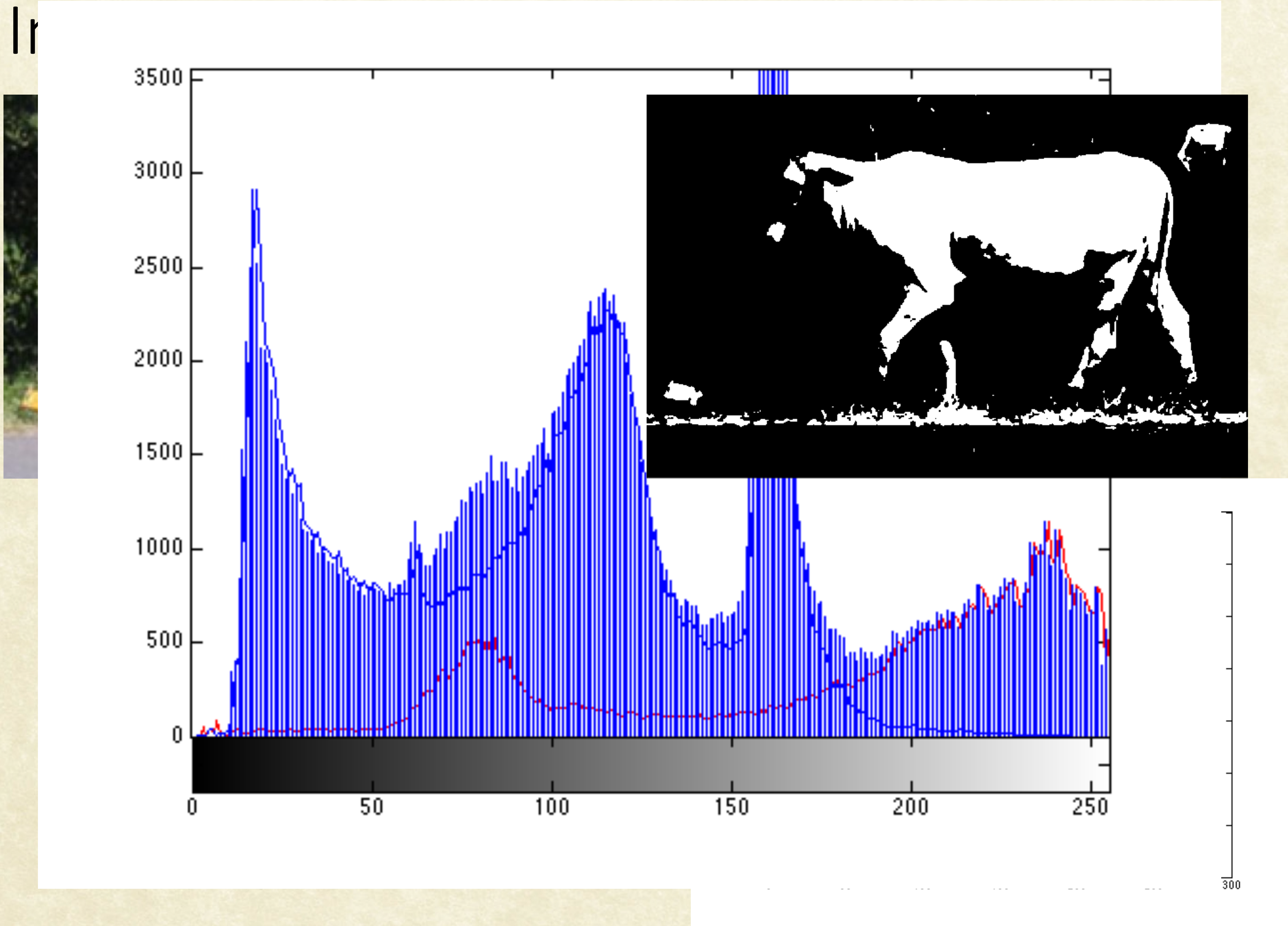
Biometrics and Secure ID Lab, CVIT,

IIIT Hyderabad

Types of Segmentation

- Classification-based
  - Label pixels based on region properties
  - Label each pixel based on object models
- Region-based
  - Region growing and splitting
- Boundary-based
  - Find edges in the image and use them as region boundary
- Motion-based
  - Group pixels that have consistent motion (e.g., move in the same direction)

Is In



Segmentation by Pixel Classification

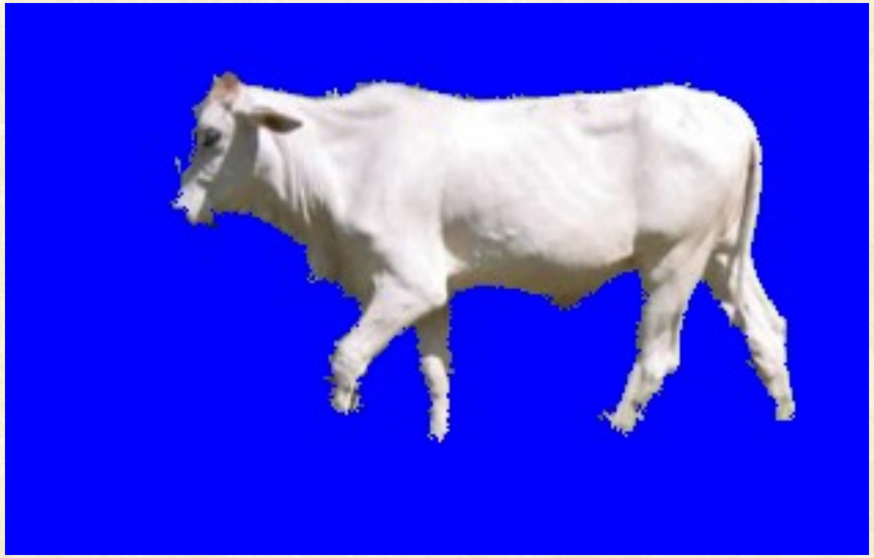
Two Primary Challenges:

1. How to use object / background properties to decide on pixel label?
   - e.g., Ducks are white and yellow, while background is green and brown
2. How to ensure that regions are continuous regions?
   - Avoid fragmentation of object regions

Segmentation as Optimal Labeling

- Model knowledge about the world
- Classify each pixel as belonging to a specific object
  - Independent classification does not work
  - Need to incorporate neighborhood information
- Consider a graph over the image
  - Each node in the graph need to be labeled
  - Edges in the graph represent neighborhood constraints
- Define a cost function, $Q(f)$, using the above
- Compute the optimal labeling wrt $Q(f)$.

Binary Image Segmentation

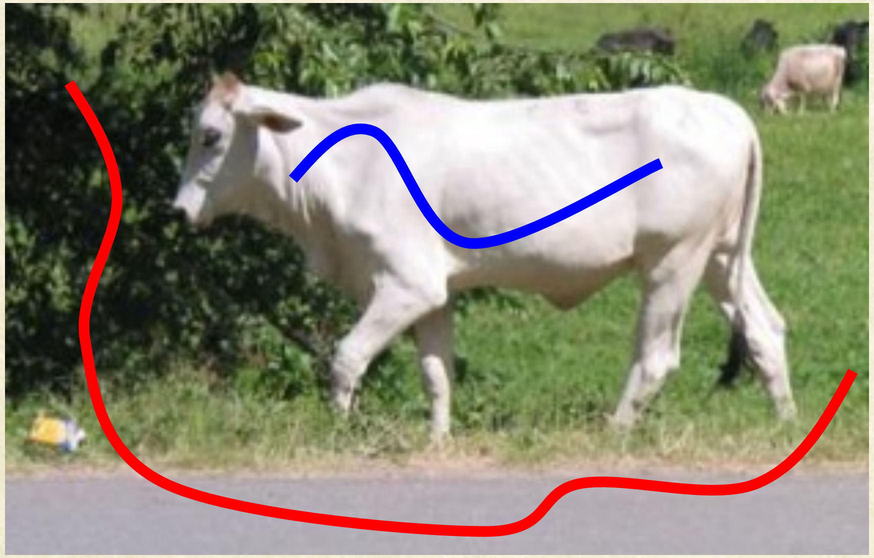


How ?

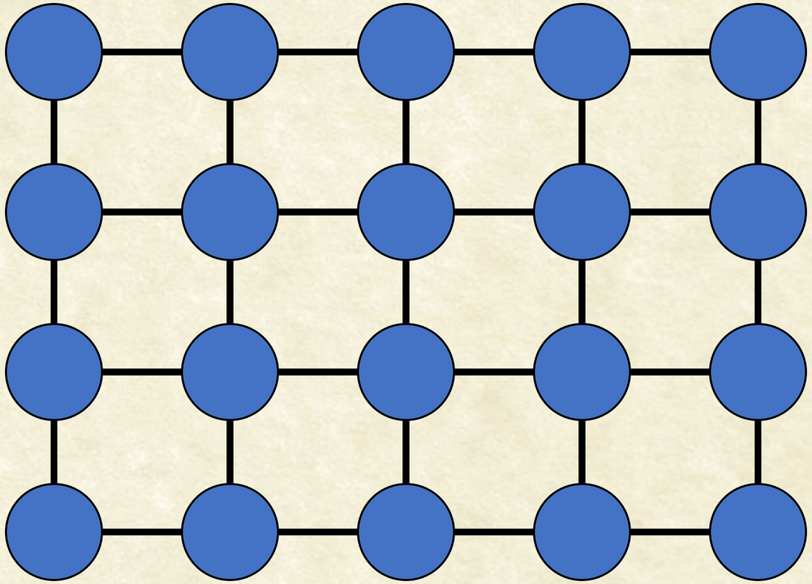
Cost function Models *our* knowledge about natural images

Optimize cost function to obtain the segmentation

Binary Image Segmentation



Object - white, Background - green/grey



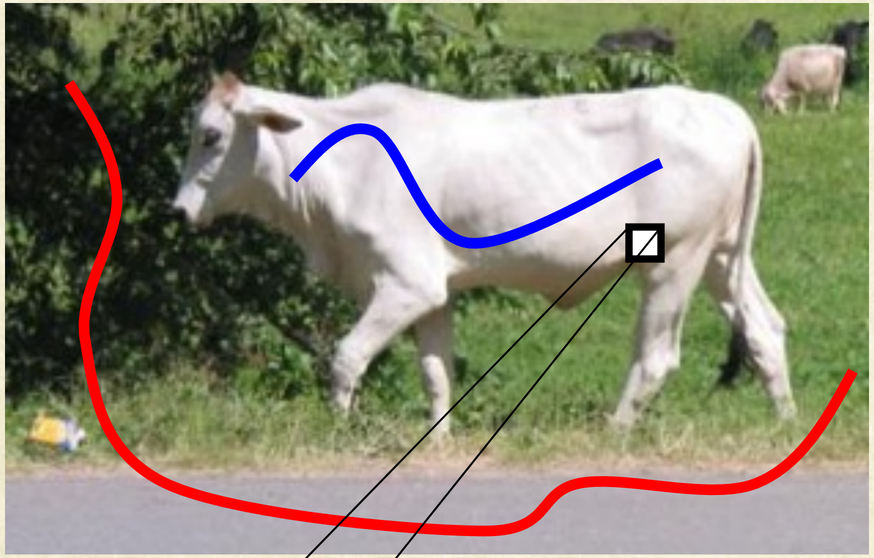
Graph G = (V,E)

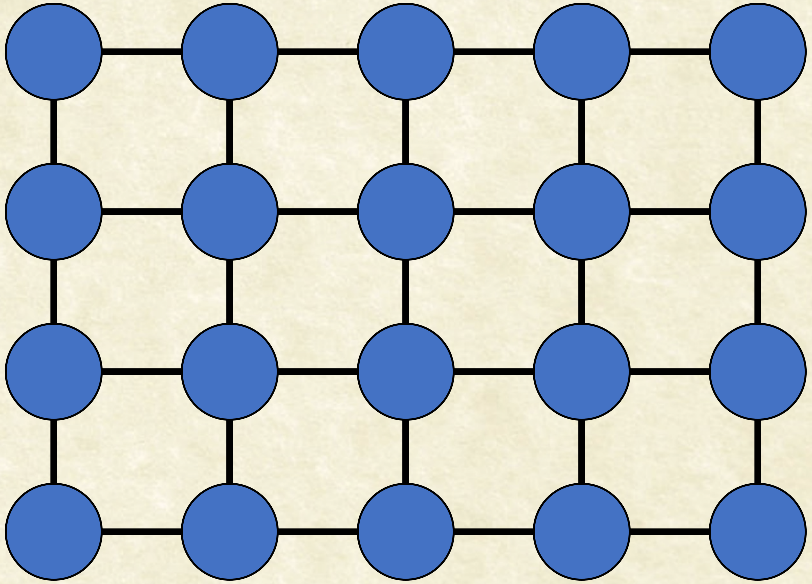
Each vertex corresponds to a pixel

Edges define a 4-neighbourhood *grid* graph

Assign a label to each vertex from L = {obj,bkg}

Binary Image Segmentation





Object - white, Background - green/grey

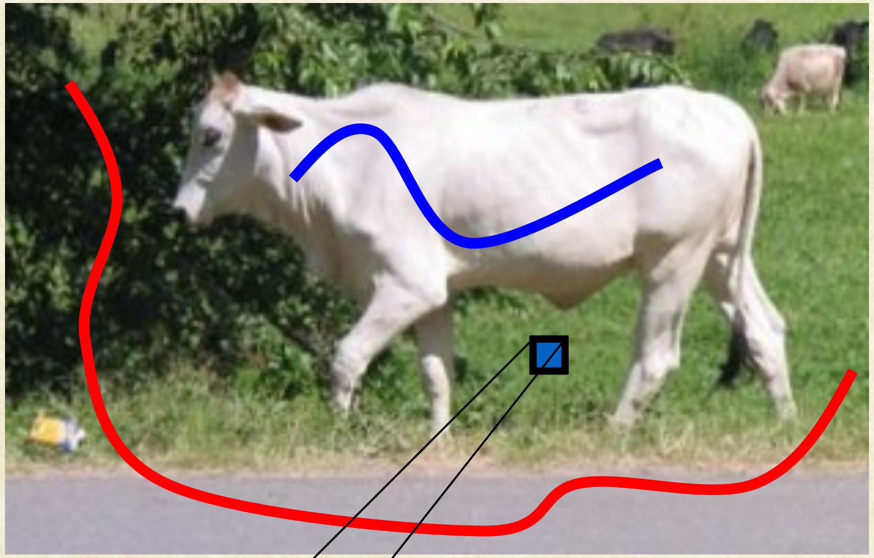
Graph G = (V,E)

Cost of a labelling f : V ➔ L

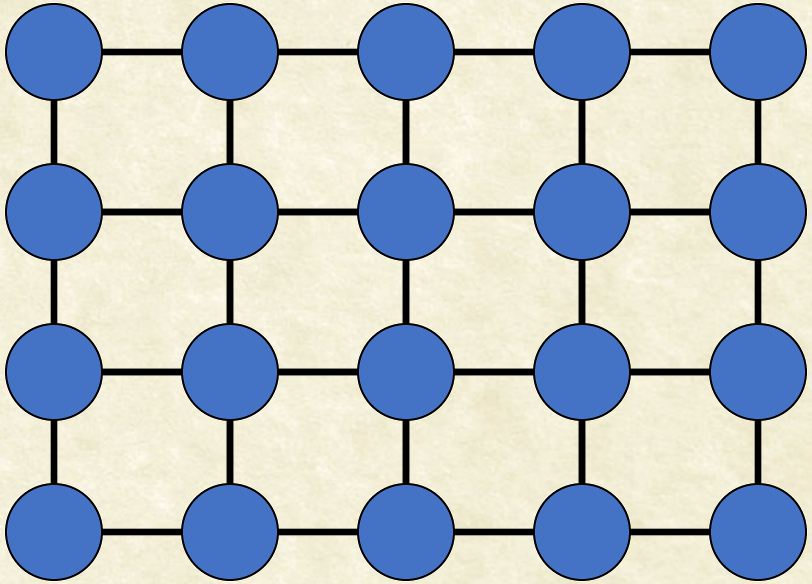
Per Vertex Cost

Cost of label ‘obj’ low Cost of label ‘bkg’ high

Binary Image Segmentation



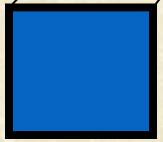
Object - white, Background - green/grey



Graph G = (V,E)

Cost of a labelling f : V ➔ L

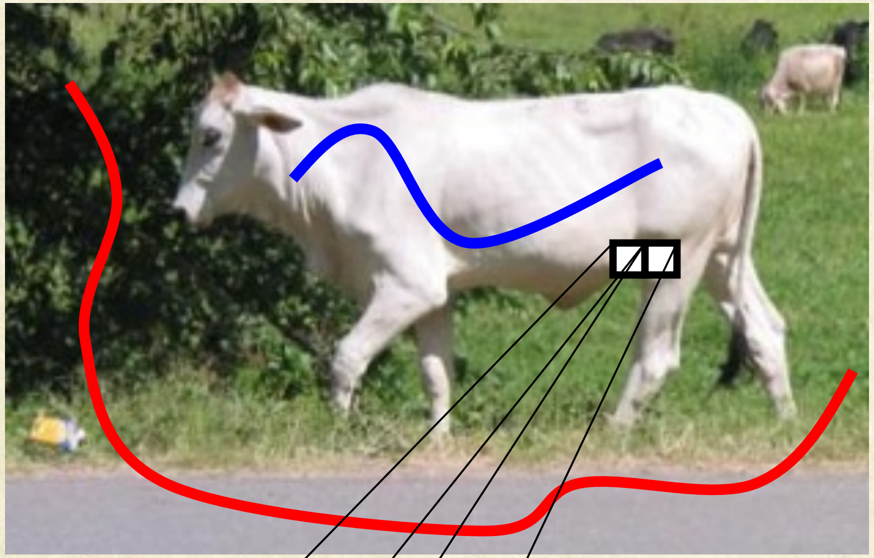
Per Vertex Cost



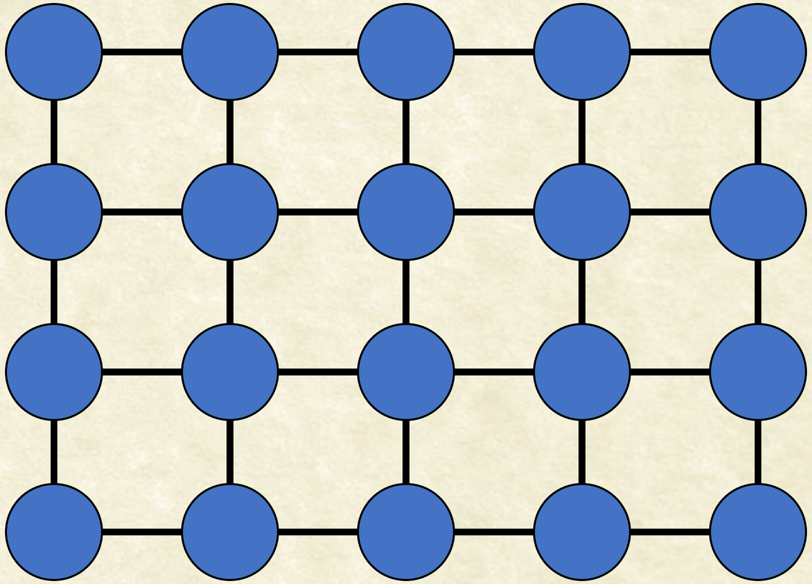
Cost of label ‘obj’ high Cost of label ‘bkg’ low

UNARY COST

Binary Image Segmentation



Object - white, Background - green/grey



Graph G = (V,E)

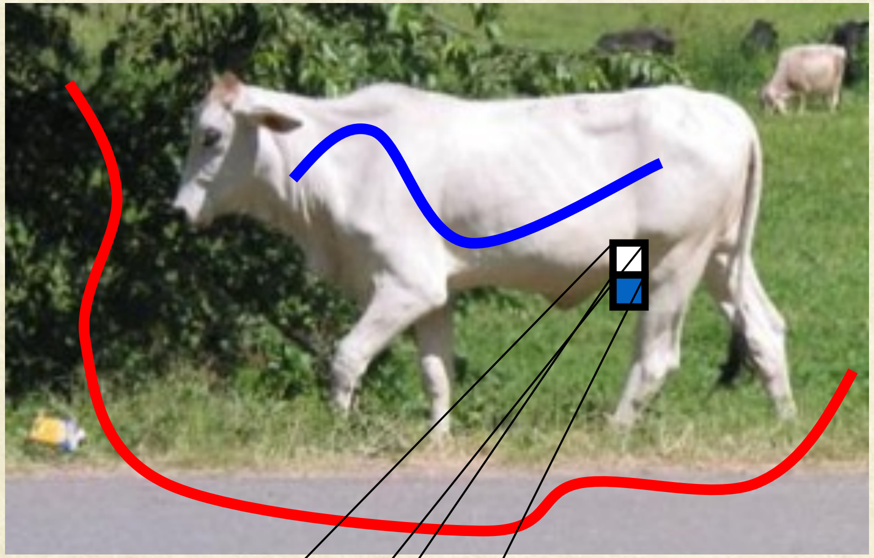
Cost of a labelling f : V ➔ L

Per Edge Cost

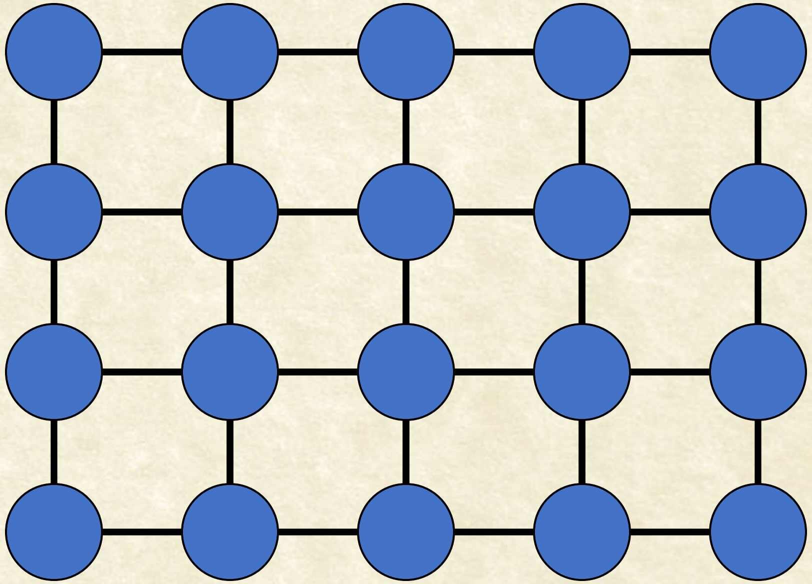
Cost of same label low

Cost of different labels high

Binary Image Segmentation



Object - white, Background - green/grey



Graph G = (V,E)

Cost of a labelling f : V ➔ L

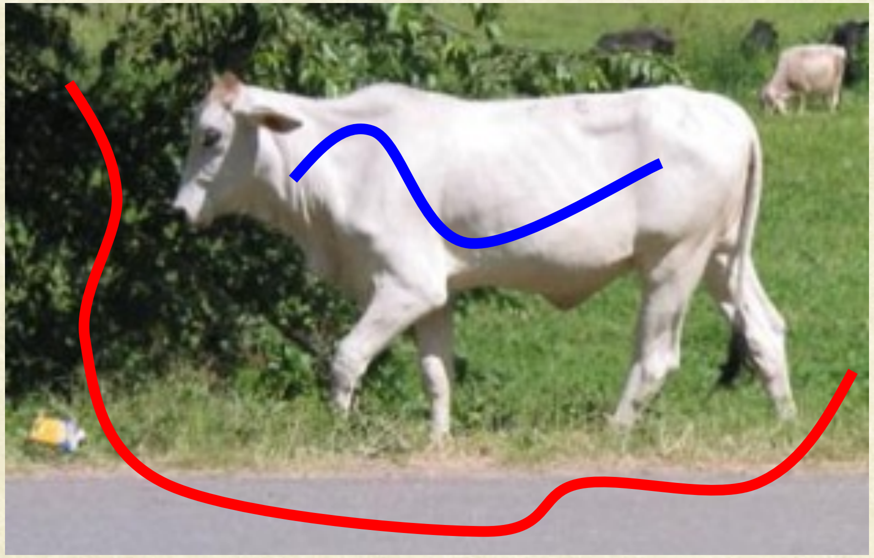
Per Edge Cost

Cost of same label high

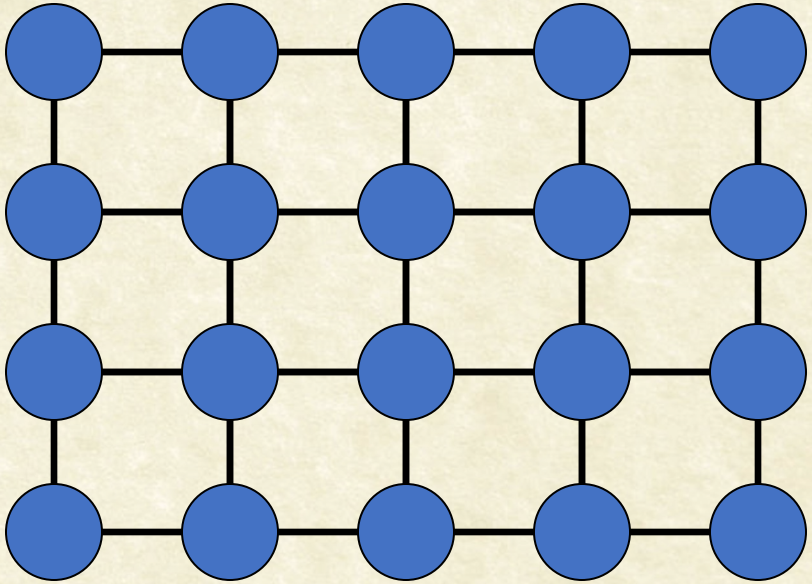
Cost of different labels low

PAIRWISE COST

Binary Image Segmentation



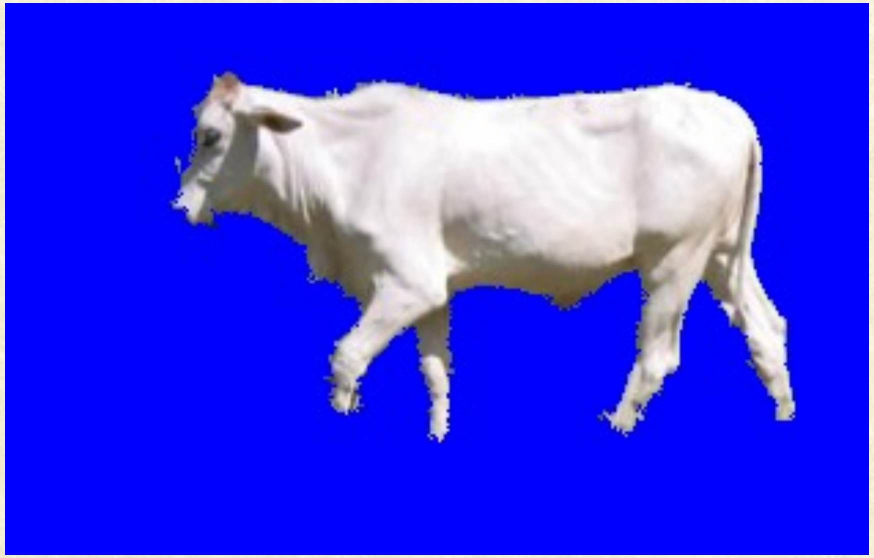
Object - white, Background - green/grey

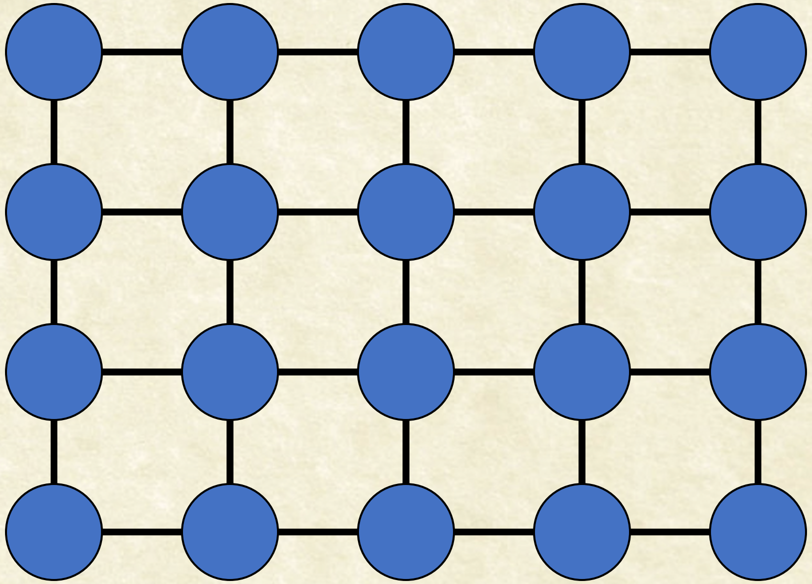


Graph G = (V,E)

Problem: Find the labeling with minimum cost f*

Binary Image Segmentation





Graph G = (V,E)

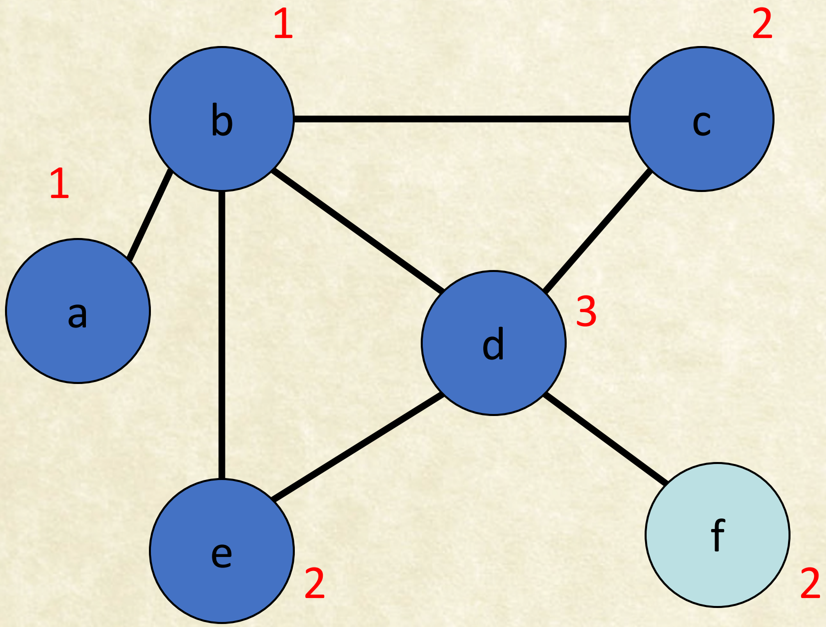
L = {fg, bg}

Vertex corresponds to a pixel

Edges define grid graph

Problem: Find the labeling with minimum cost f*

The General Problem



Graph G = (V, E)

Discrete label set L = {1,2,…,h}

Assign a label to each vertex
f: V ➜ L

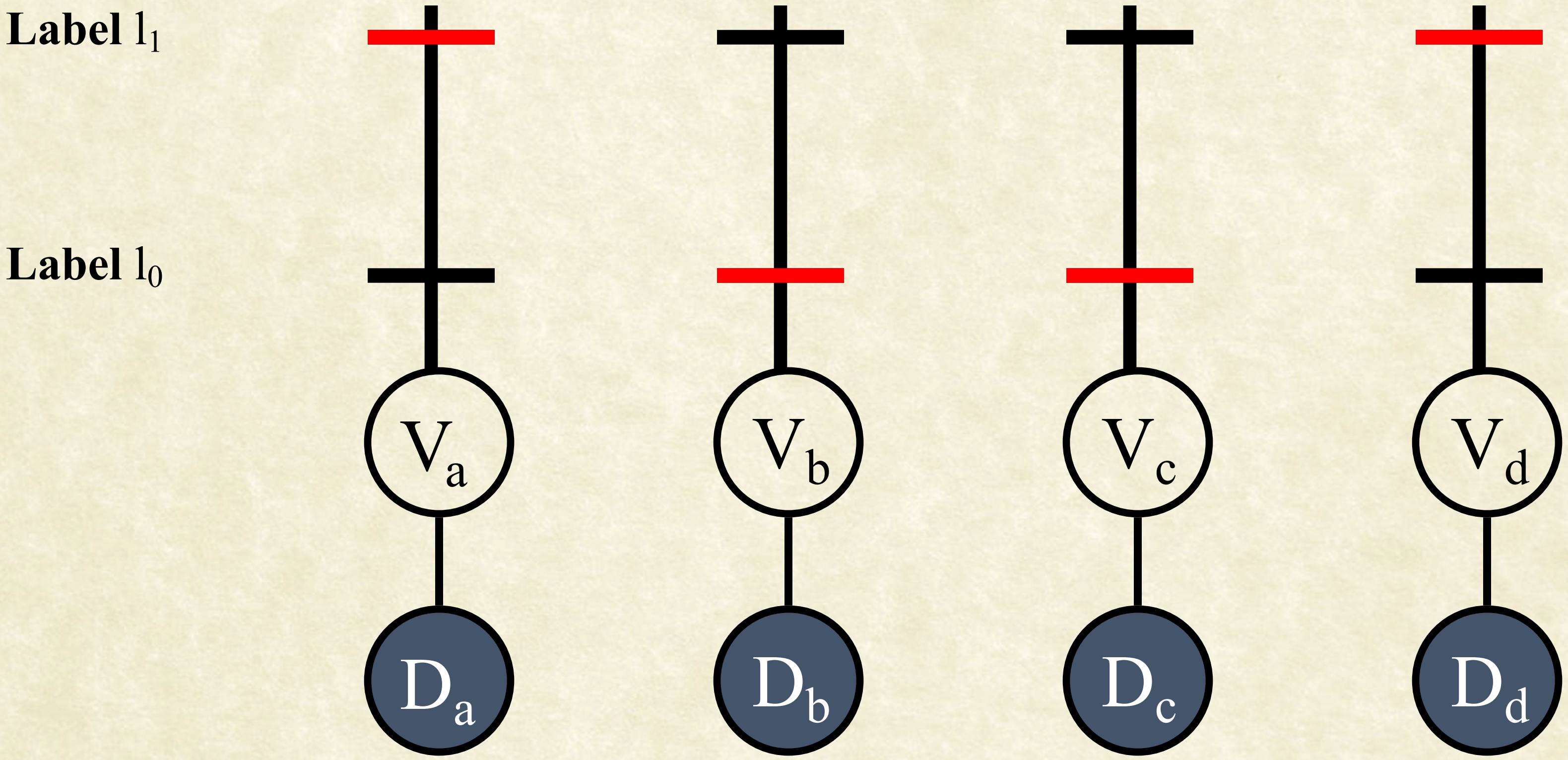
Cost of a labelling Q(f)

Unary Cost

Pairwise Cost

Find f* = arg min Q(f)

Formulation: Energy Function

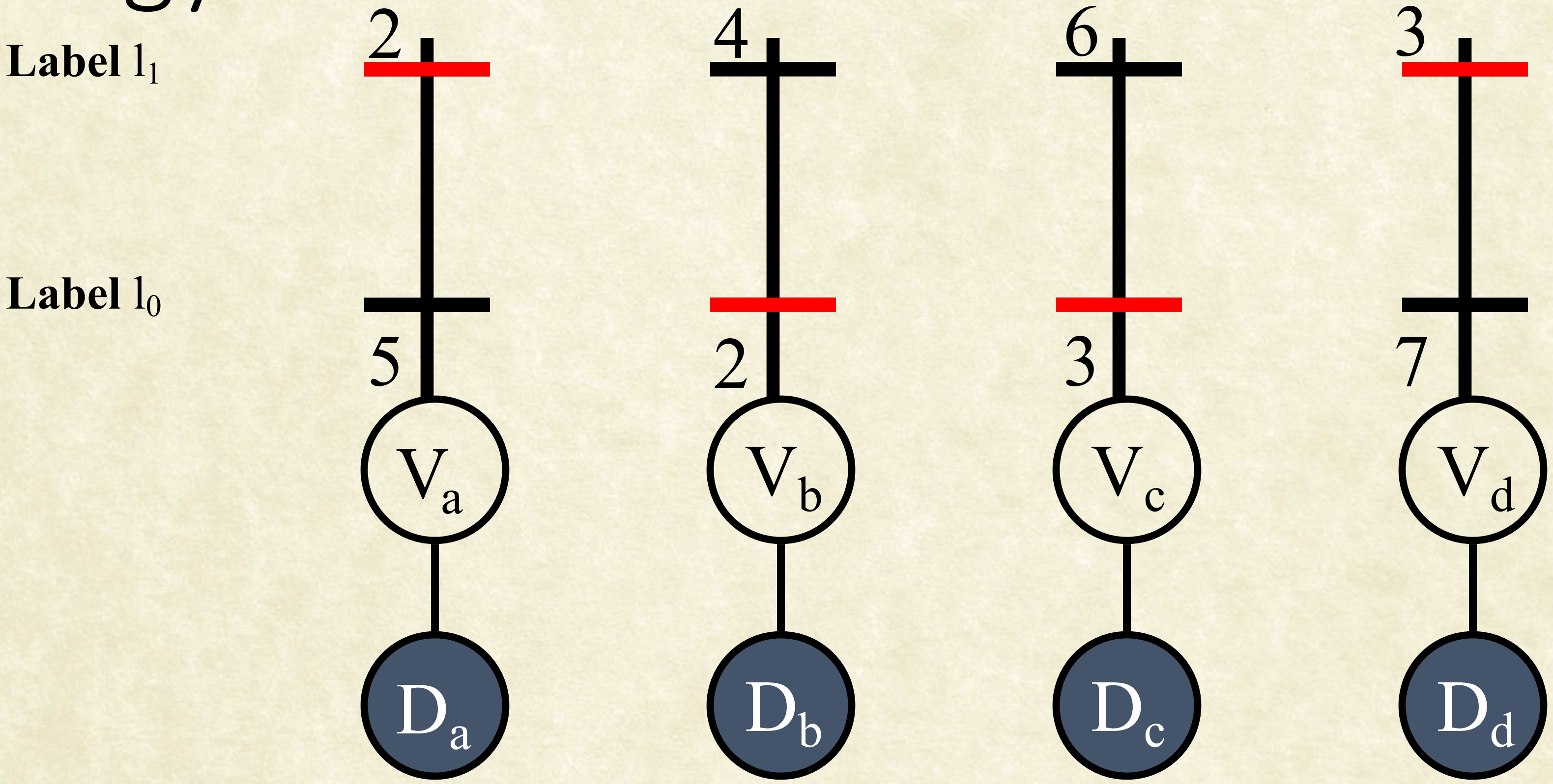


Random Variables $V = \{V_a, V_b, ...\}$

Labels $L = \{l_0, l_1, ...\}$ Data D

Labelling $f: \{a, b, ... \} \rightarrow \{0,1, ...\}$

Energy Function



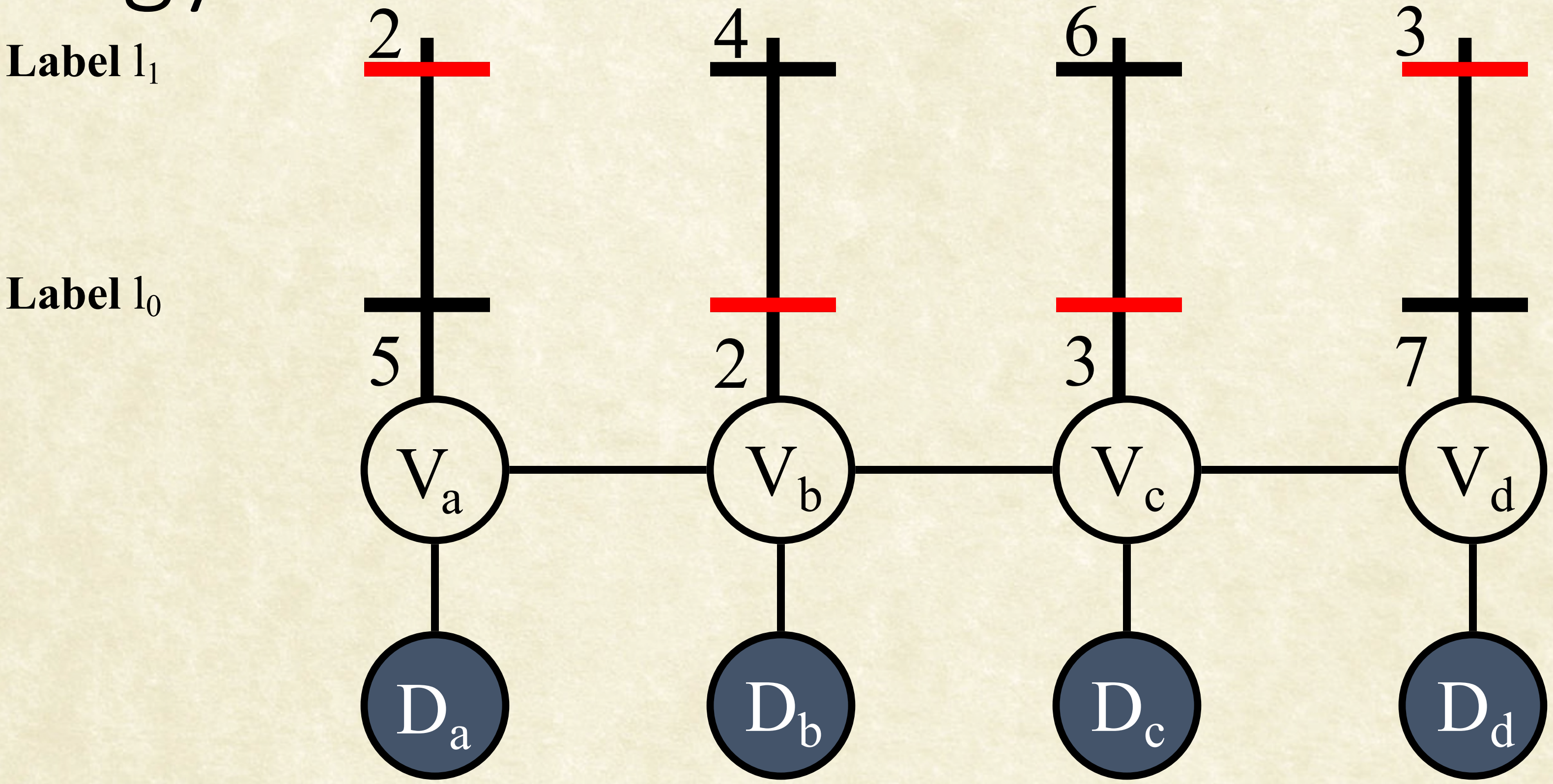
$$Q(f) \quad = \sum_a \theta_{a;f(a)}$$

Unary Potential

Easy to minimize

Neighbourhood

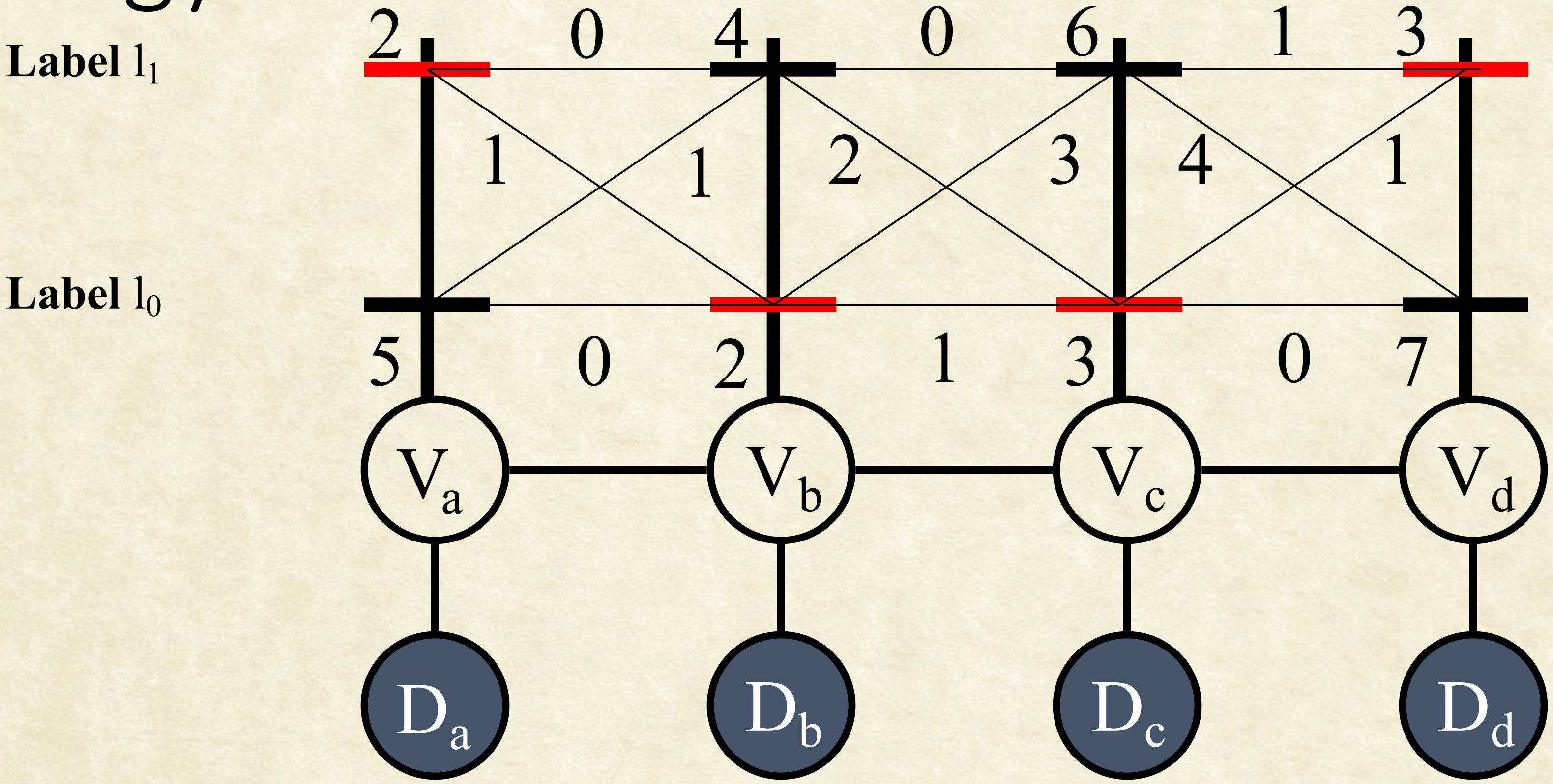
Energy Function



E : (a,b) $\in$ E iff V_a and V_b are neighbours

E = { (a,b) , (b,c) , (c,d) }

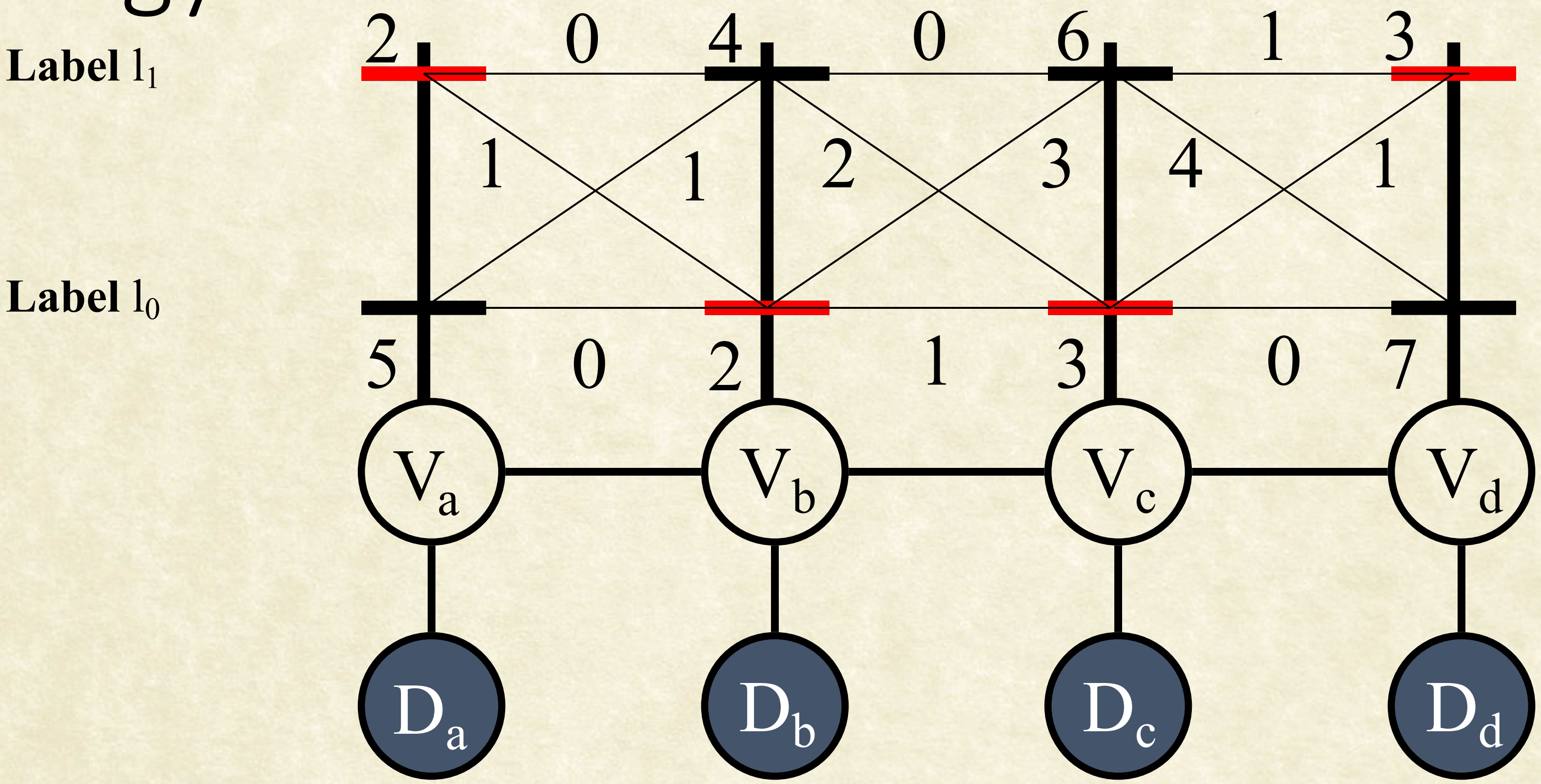
Energy Function



Pairwise Potential

$$Q(f) \quad = \sum_a \theta_{a;f(a)} \qquad + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

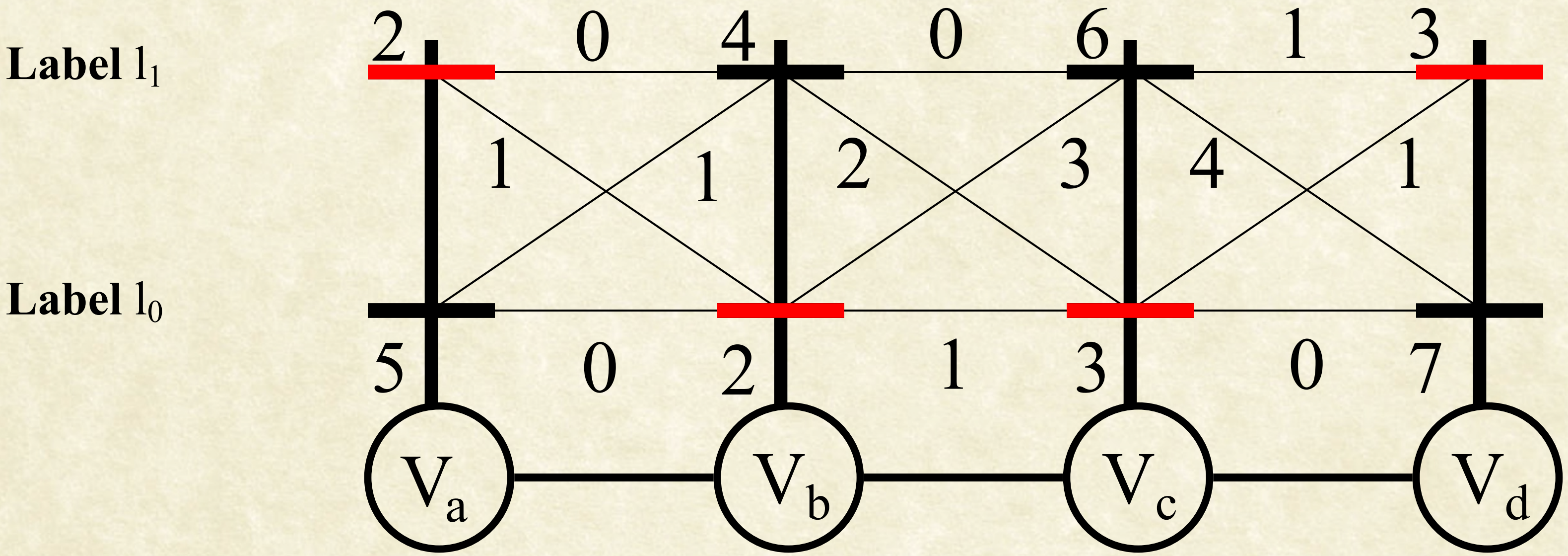
Energy Function



$$Q(f;\theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

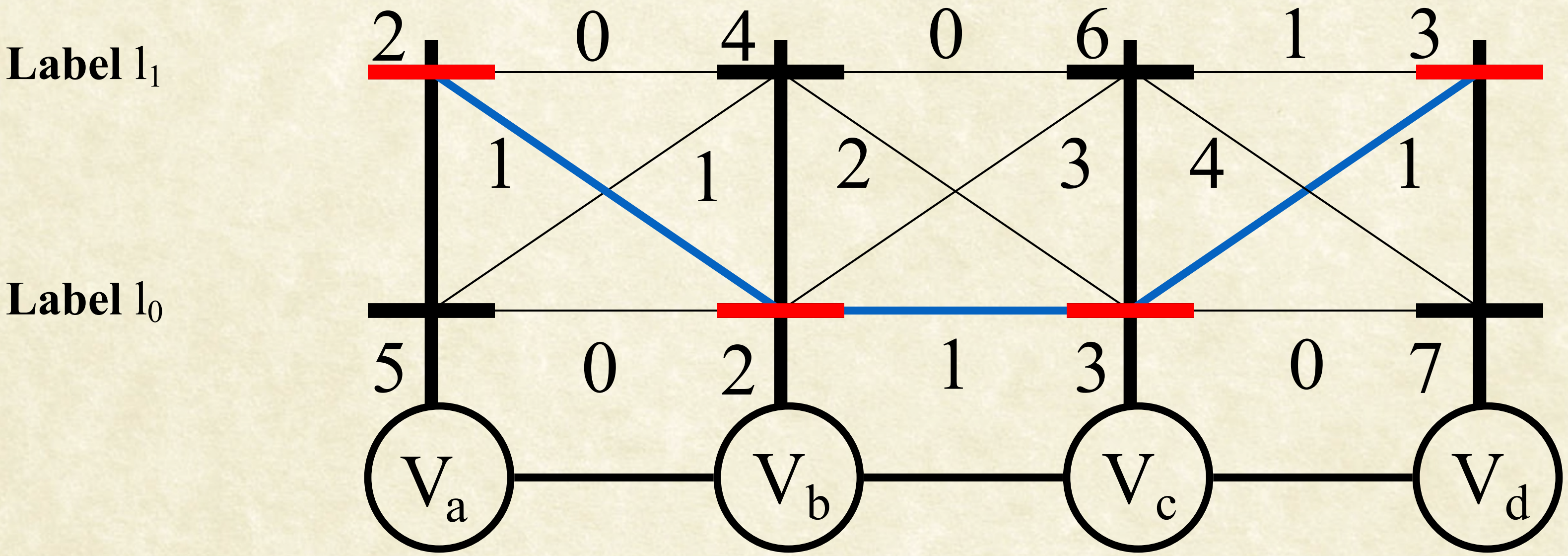
Parameter

MAP Estimation



$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

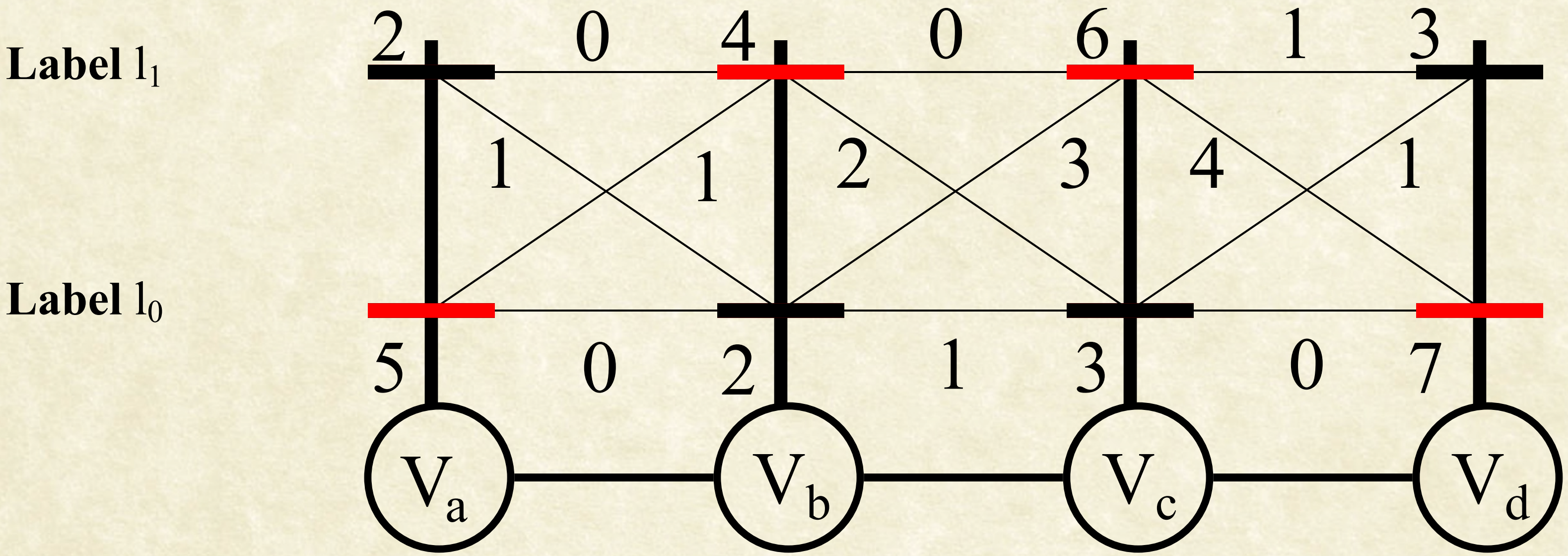
MAP Estimation



$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

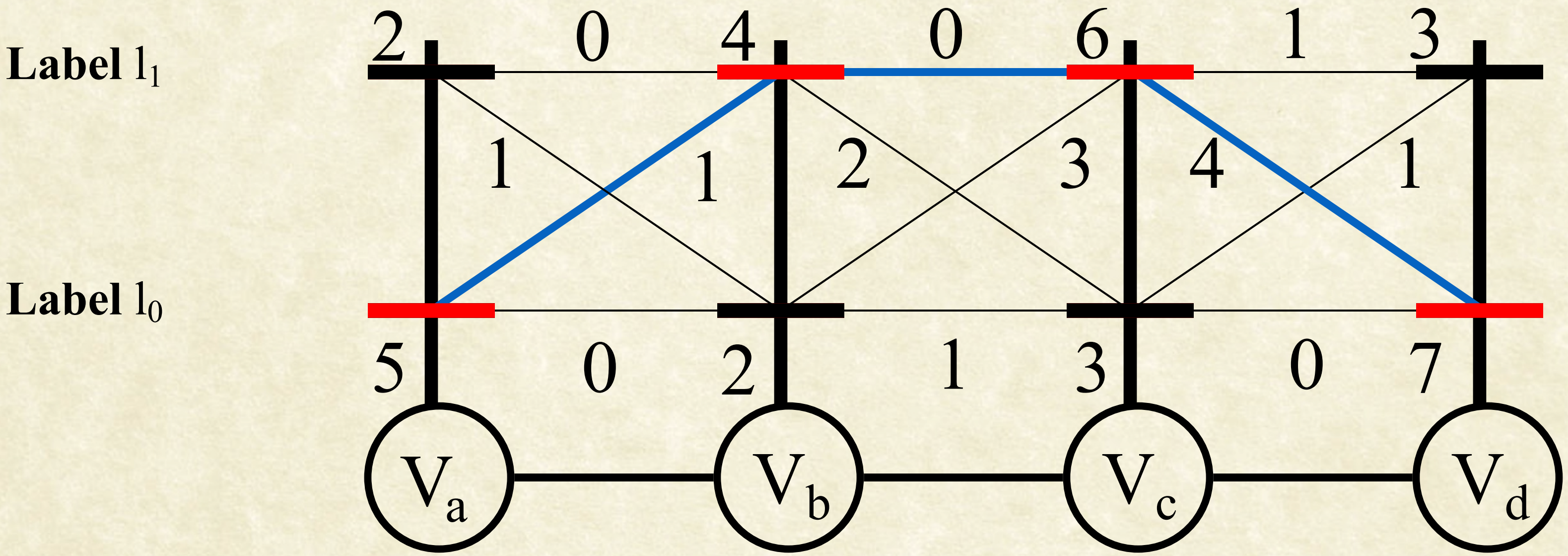
2 + 1 + 2 + 1 + 3 + 1 + 3 = 13

MAP Estimation



$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

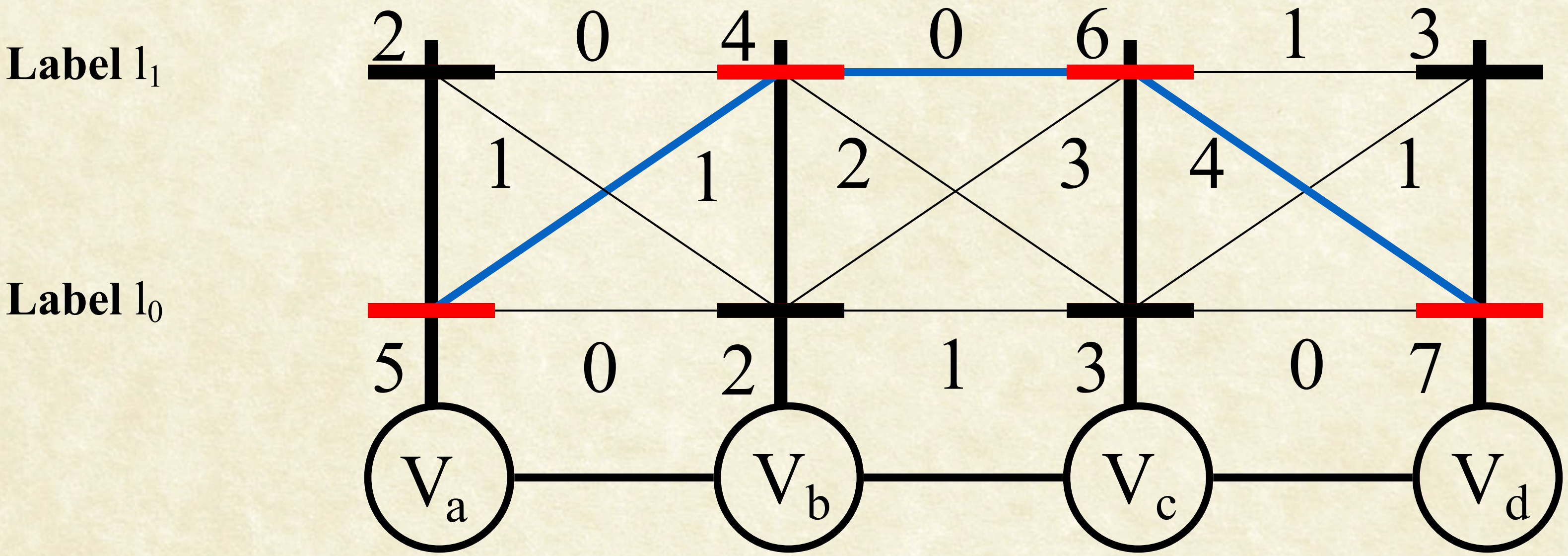
MAP Estimation



$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$

5 + 1 + 4 + 0 + 6 + 4 + 7 = 27

MAP Estimation



$q^* = \min Q(f; \theta) = Q(f^*; \theta)$

$$Q(f; \theta) = \sum_a \theta_{a;f(a)} + \sum_{(a,b)} \theta_{ab;f(a)f(b)}$$

$f^* = \arg\min Q(f; \theta)$

MAP Estimation

16 possible labellings

$f^* = \{1, 0, 0, 1\}$

$q^* = 13$

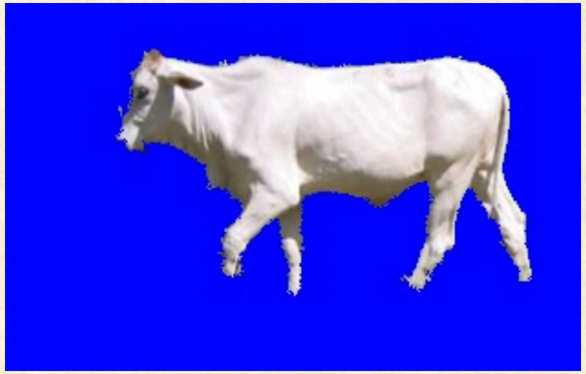
f(a)	f(b)	f(c)	f(d)	Q(f; θ)
0	0	0	0	18
0	0	0	1	15
0	0	1	0	27
0	0	1	1	20
0	1	0	0	22
0	1	0	1	19
0	1	1	0	27
0	1	1	1	20

f(a)	f(b)	f(c)	f(d)	Q(f; θ)
1	0	0	0	16
1	0	0	1	13
1	0	1	0	25
1	0	1	1	18
1	1	0	0	18
1	1	0	1	15
1	1	1	0	23
1	1	1	1	16

Computational Complexity

Segmentation

$2^{|V|}$



$|V|$ = number of pixels ≈ 320 * 480 = 153600

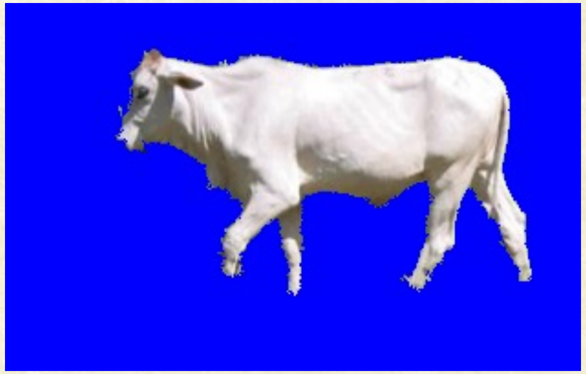
Can we do better than brute-force?

MAP Estimation is NP-hard !!

Computational Complexity

Segmentation

$2^{|V|}$



$|V|$ = number of pixels ≈ 320 * 480 = 153600

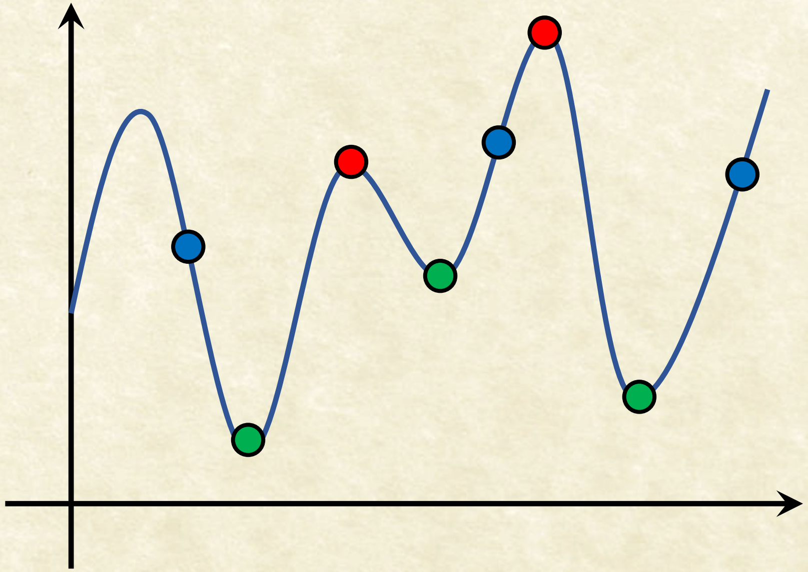
Exact algorithms do exist for special cases

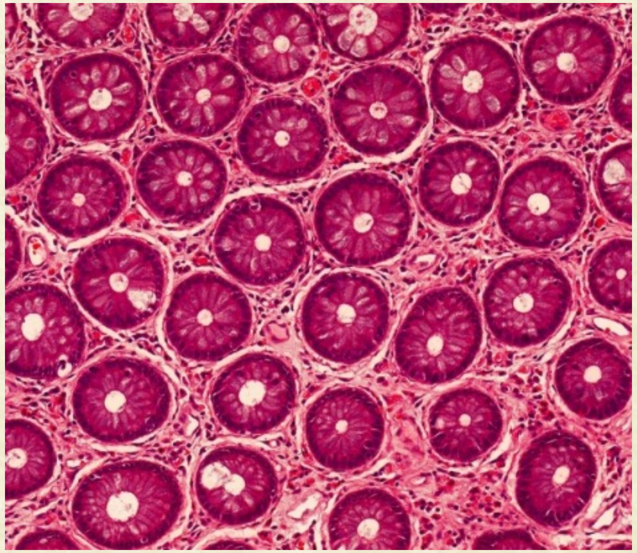
Good approximate algorithms for general case

Questions?

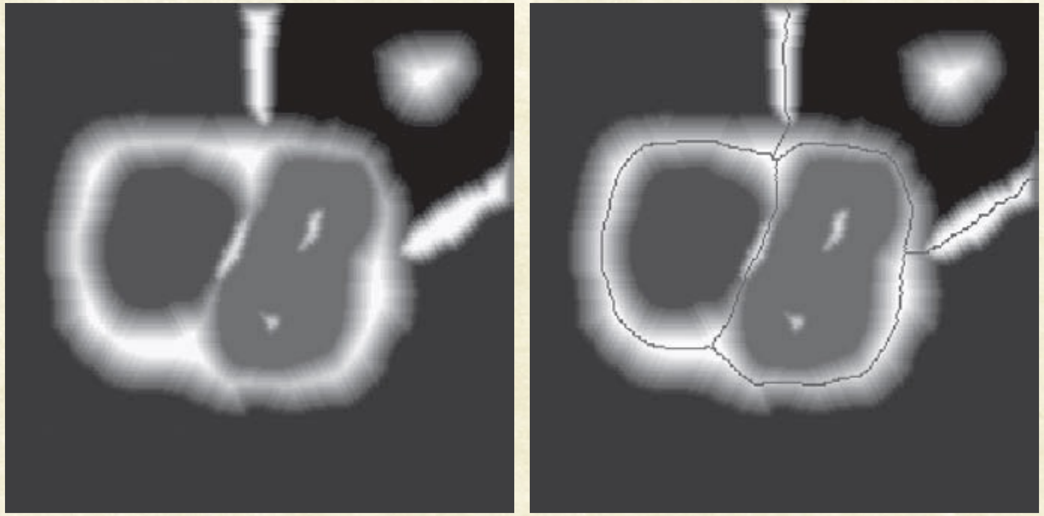
Watershed Segmentation

- Consider the intensity profile of an image as a topographical surface
- We can identify three types of points:
  1. Local minima, where water collects
  2. Points where water flows to a single minima (catchment basin)
  3. Points where water can flow to any one of multiple minima (watershed lines)
- Principal Goal: Find the watershed lines (object boundaries)

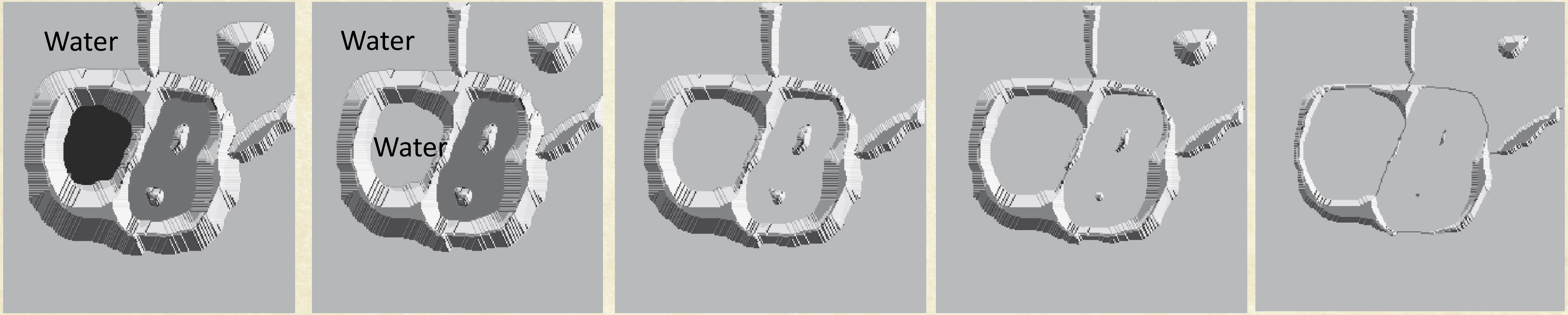




Watershed Segmentation



- Original Image, Topographical view
- Various stages of flooding.

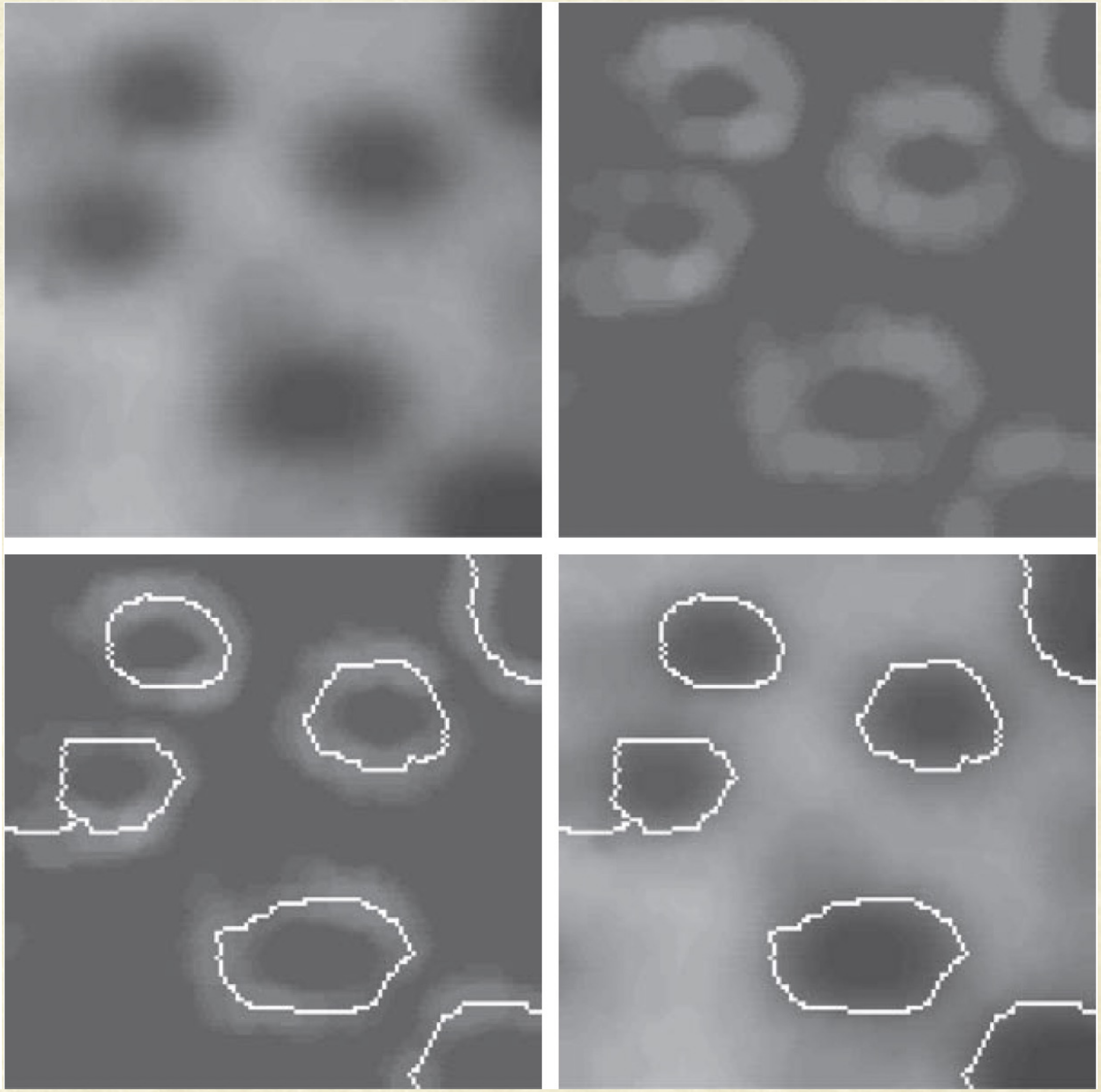


The Watershed Algorithm

- Let $M_1, M_2, .., M_R$ be the regional minima.
- Let $C(M_i)$ be points in the catchment area of M_i.
- Let $T[n]$ represent the set of co-ordinates for which $g(s,t) < n$.
- Flood the topography in integer increments
  - From $n = \min + 1$ to $n = \max + 1$.
- At each stage, we compute $C_n(M_i)$ from $C_{n-1}(M_i)$ and $T[n]$ using morphological operators
  - Essentially, add pixels of value n to the corresponding $C(M_i)$

Watershed Example

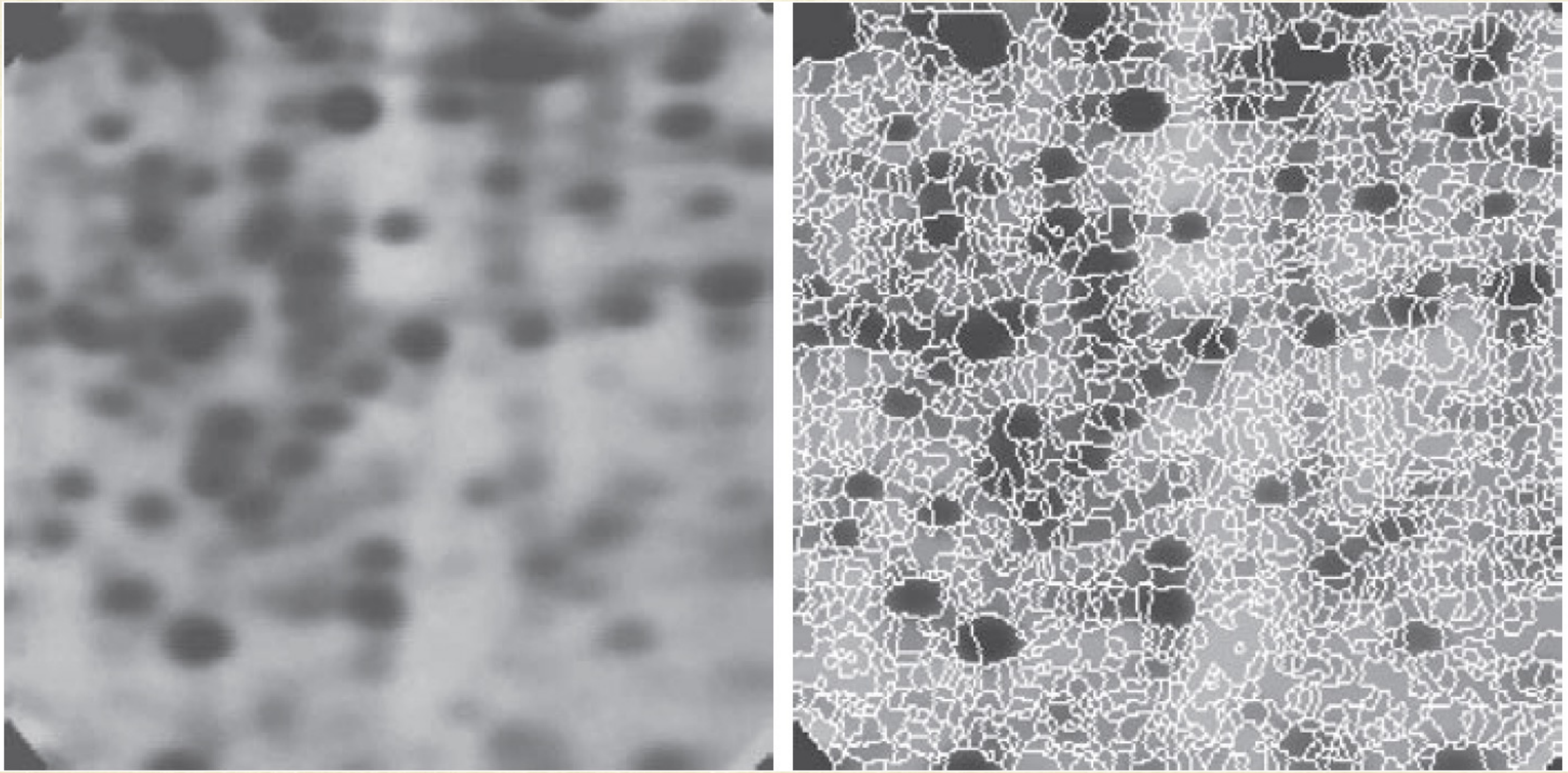
- Watershed is often performed on the gradient image (top right).



a b
c d

FIGURE 10.59
(a) Image of blobs. (b) Image gradient. (c) Watershed lines, superimposed on the gradient image. (d) Watershed lines superimposed on the original image.

Over-segmentation from Watershed



a b

FIGURE 10.60
(a) Electrophoresis image.
(b) Result of applying the watershed segmentation algorithm to the gradient image. Over-segmentation is evident.

Over-segmentation from Watershed

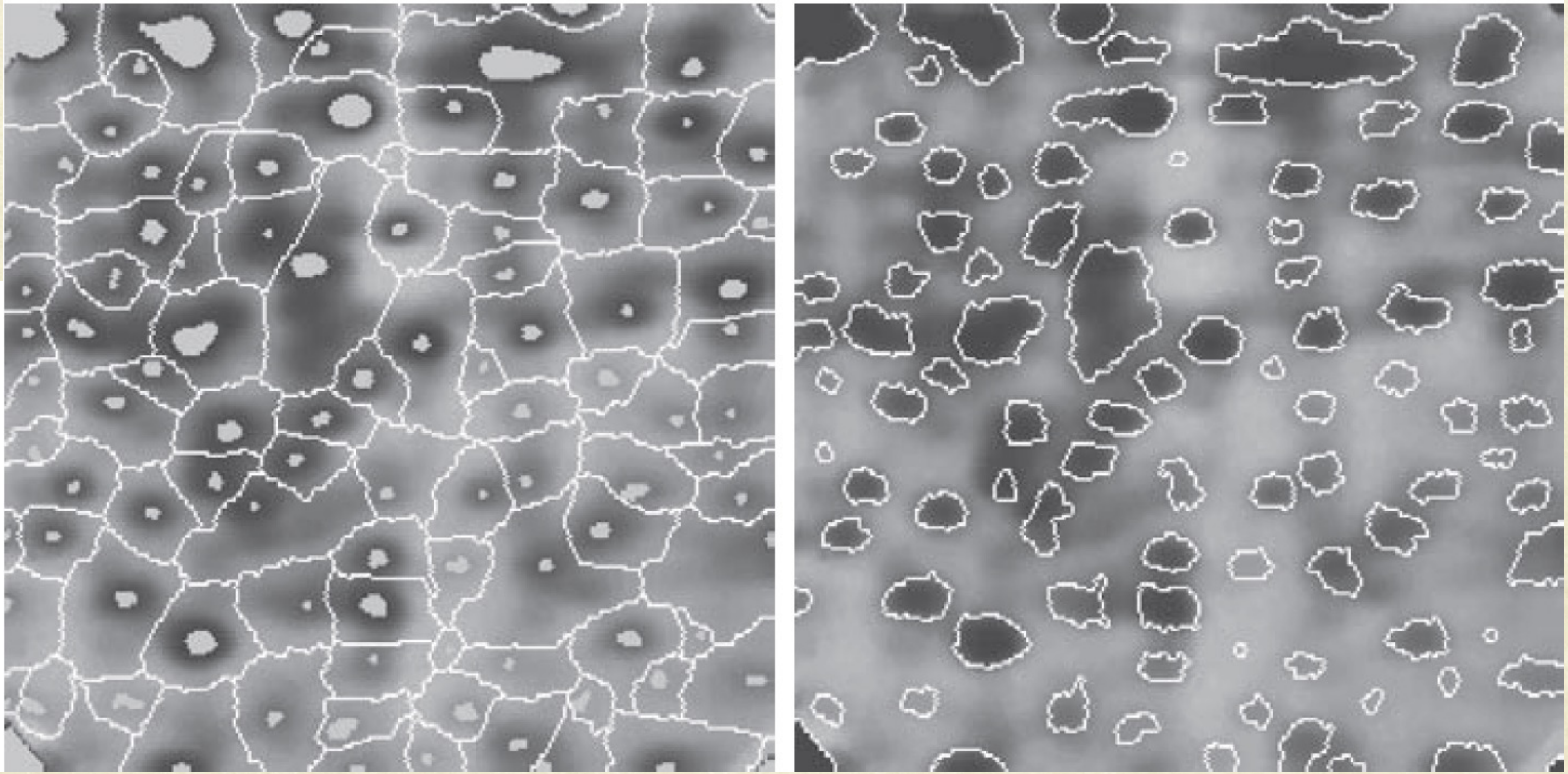


FIGURE 10.61
(a) Image showing internal markers (light gray regions) and external markers (watershed lines).
(b) Result of segmentation. Note the improvement over Fig. 10.60(b).

Questions?