Optimal solution of binary problems

Much material taken from:

- **Olga Veksler, University of Western Ontario**

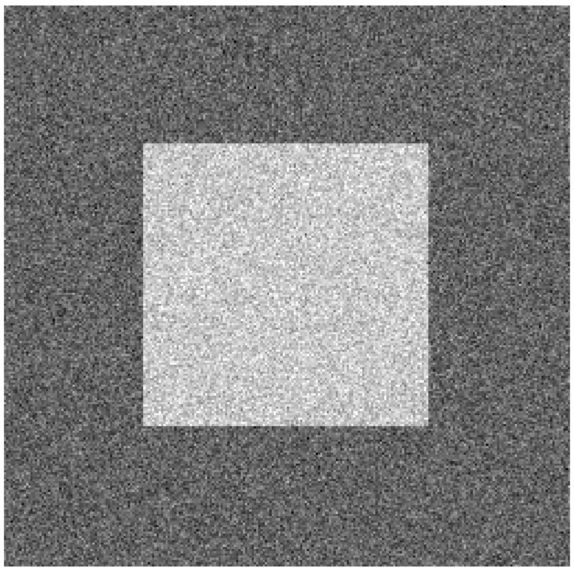
http://www.csd.uwo.ca/faculty/olga/

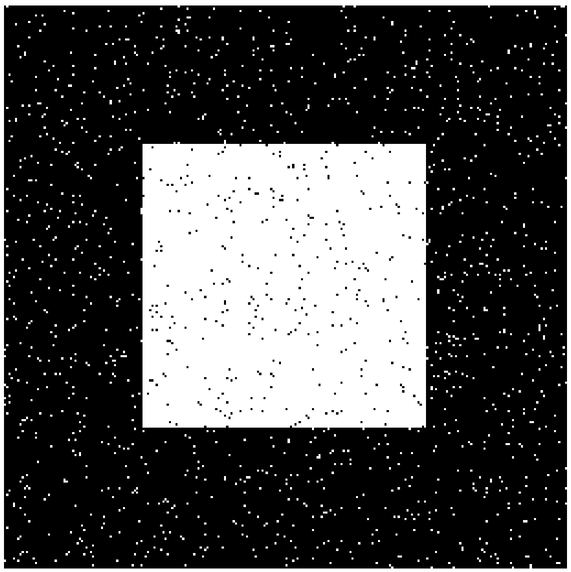
Announcements

- Project proposal due March 31
  - Email to RDZ
  - One or two paragraphs
- Project will be due the last week of classes, or perhaps a bit later
- Ashish will lecture on Friday

Motivating example

- Suppose we want to find a bright object against a dark background
  - But some of the pixel values are slightly wrong





Optimization viewpoint

- Find best (least expensive) binary image
  - Costs: C1 (labeling) and C2 (boundary)
- C1: Labeling a dark pixel as foreground
  - Or, a bright pixel as background
- If we only had labeling costs, the cheapest solution is the thresholded output
- C2: The length of the boundary between foreground and background
  - Penalizes isolated pixels or ragged boundaries

MAP-MRF energy function

- Generalization of C2 is $\sum_{p,q} V_{p,q}(x_p, x_q)$
  - Think of V as the cost for two adjacent pixels to have these particular labels
  - For binary images, the natural cost is uniform
- Bayesian energy function:

$$E(x_1, \ldots, x_n) = \sum_{p} D_p(x_p) + \sum_{p,q} V_{p,q}(x_p, x_q)$$

Generalizations

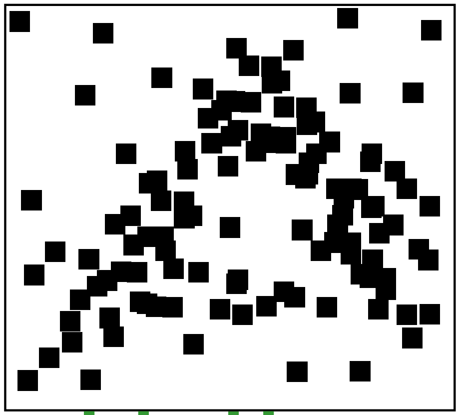
- Many vision problems have this form
- The optimization techniques are specific to these energy function, but not to images
  - See: [Kleinberg & Tardos JACM 02]
- Historically solved by gradient descent or related methods (e.g. annealing)
  - Optimization method and energy function are not independent choices!
  - Use the most specific method you can
    - And, be prepared to tweak your problem

Binary labeling problems

- Consider the case of two labels only
  - Surprisingly important
    - Lots of nice applications
    - Same basic ideas used for more labels
- There is a fast exact solution!
  - Turn a problem you don’t know how to solve into a problem you do (reduction)
    - Due to Hammer (1965) originally
    - Job scheduling problems: Stone (1977)
    - Images: Greig, Porteus & Seheult (1989)

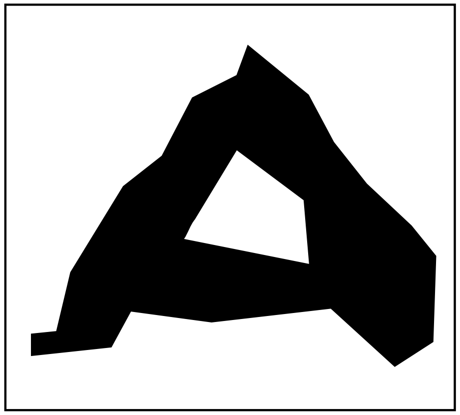
Binary Labeling: Motivation

- Suppose we receive a noisy fax:
  - Some black pixels in the original image were flipped to white pixels, and some white pixels were flipped to black



original image

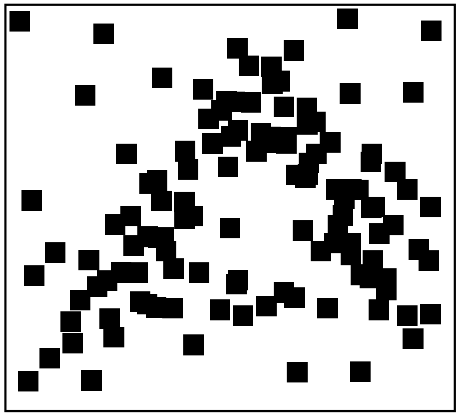
- We want to try to clean up (or **restore**) the original image:
- This problem is called **image restoration (denoising)**



restored image

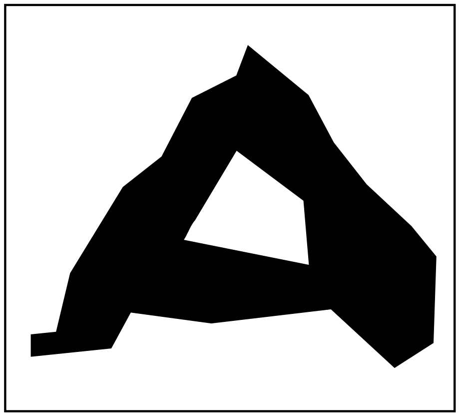
Binary Labeling Problem : Motivation

- Fax image is binary: each pixel is either
  - black (stored as 0)
  - or white (stored as 1)



original image

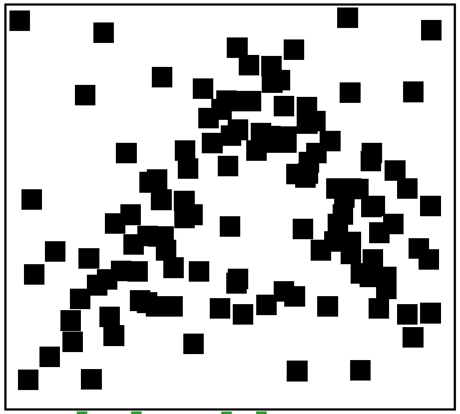
- What we know:
  1) In the restored image, each pixels should also be either black (0) or white (1)
  2) Data Constraint: if a pixel is black in the original image, it is more likely to be black in the restored image; and a white pixel in the original image is more likely to be white in the restored image
  3) Prior Constraint: in the restored image, white pixels should form coherent groups, as should black pixels



restored image

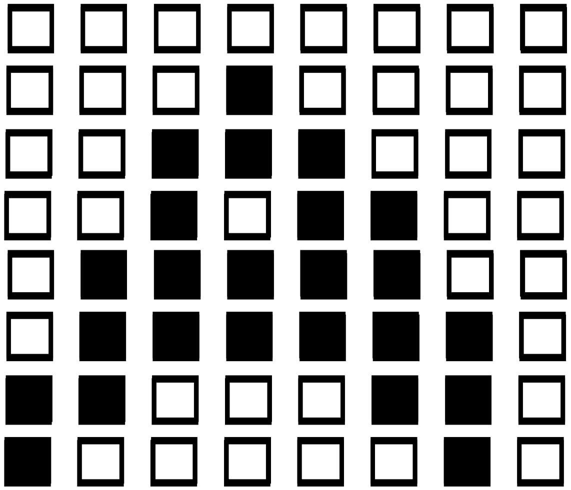
Binary Labeling Problem

- We can formulate our restoration problem as a labeling problem:
  - Labels: black (0) and white (1)
  - Set of sites: all image pixels
    - I will say set of "sites" or set of pixels interchangeably



original image

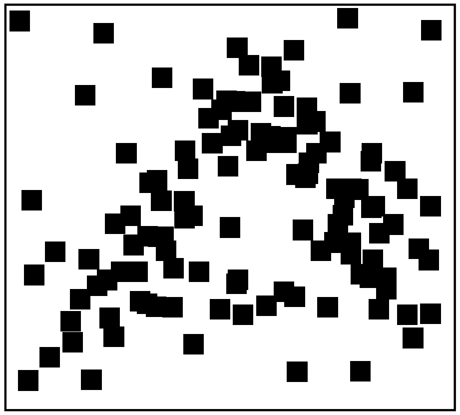
- Assign a label to each site (either the black or the white label) s.t.
  - If a pixel is black (white) in the original image, it is more likely to get the black (white) label
  - Black labeled pixels tend to group together, and white labeled pixels tend to group together



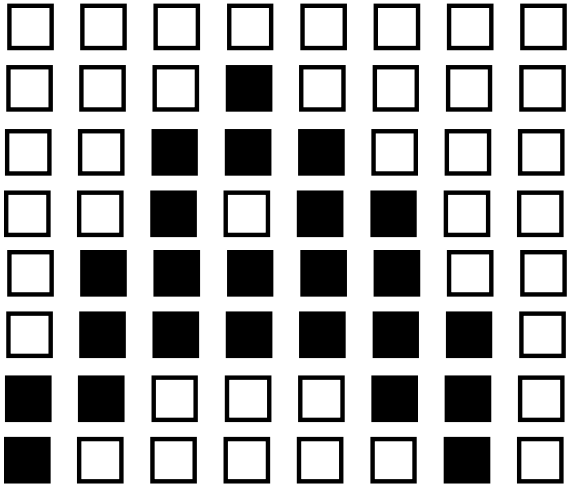
set of sites P, and one possible labeling

Binary Labeling Problem: Constraints

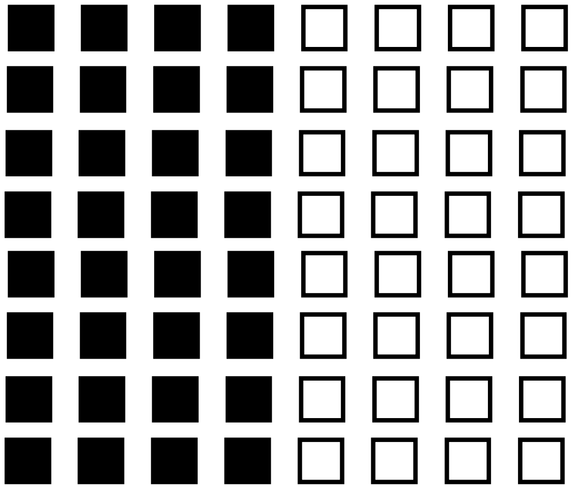
- Our Constraints:
  1. If a pixel is black (white) in the original image, it is more likely to get the black (white) label
  2. Black labeled pixels tend to group together, and white labeled pixels tend to group together



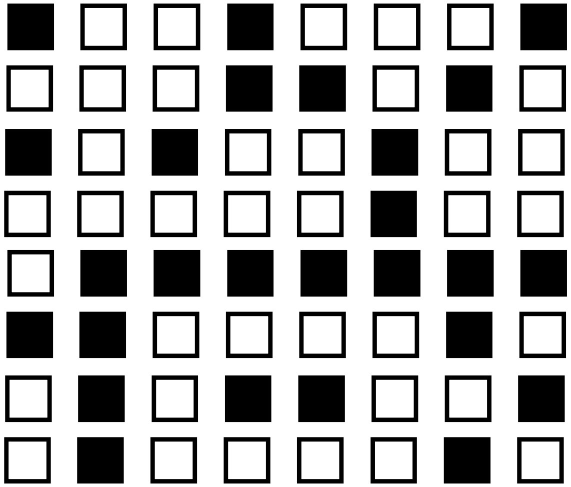
original image



good labeling



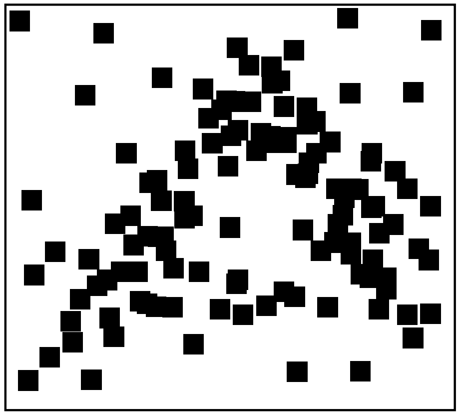
bad labeling (constraint 1)



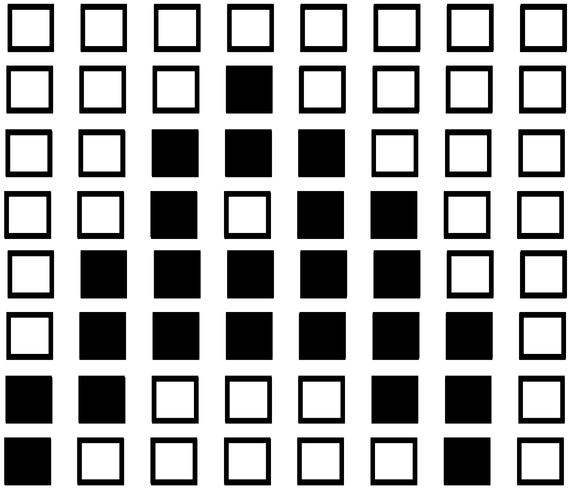
bad labeling (constraint 2)

Binary Labeling Problem: Data Constraints

- How can we find a a good labeling (i.e., satisfying our constraints)?
  - Formulate and minimize an energy function on labelings L:
    - Let L_p be the label assigned to pixel p
      - $L_p = 0$ or $L_p = 1$
    - Let $I(p)$ be the intensity of pixel p in the original image
    - Data constraint is modeled by the Data Penalty $D_P(L_P)$
      - $D_P(0) < D_P(1)$ if $I(p) = 0$
      - $D_P(0) > D_P(1)$ if $I(p) = 1$



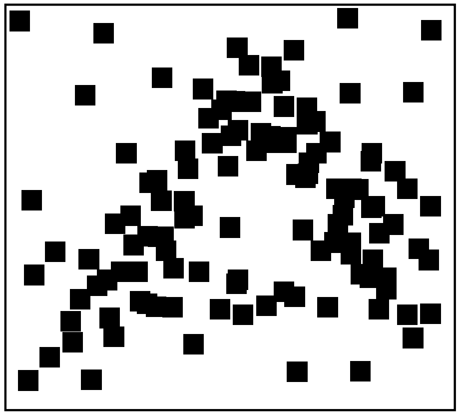
original image



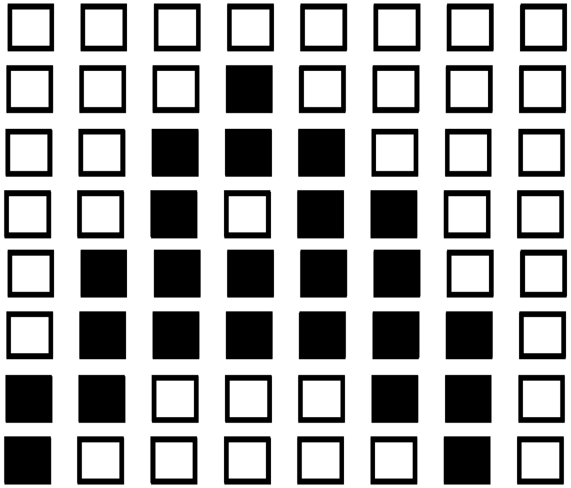
a good labeling L

Binary Labeling Problem: Data Constraints

- In our example, we can make
  - if $I(p)$ = ***black*** then
    $D_p(black) = 0$
    $D_p(white) = 4$
  - if $I(p)$ = ***white*** then
    $D_p(black) = 4$
    $D_p(white) = 0$



original image



a good labeling L

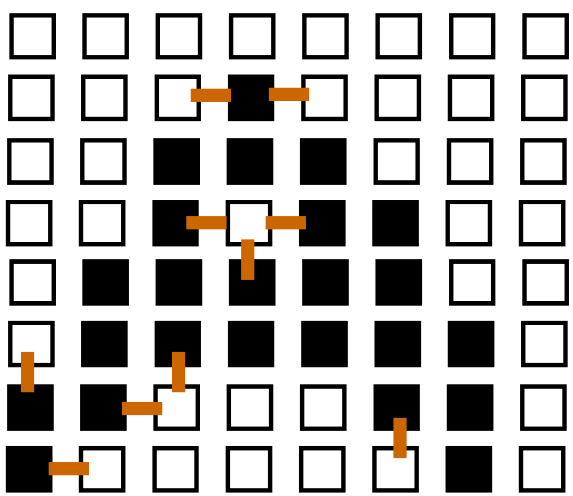
- To figure out how well image as a whole satisfies the data constraint, sum up data terms over all image pixels:

$$\sum_p D_P(L_p)$$

Binary Labeling Problem: Prior Constraints

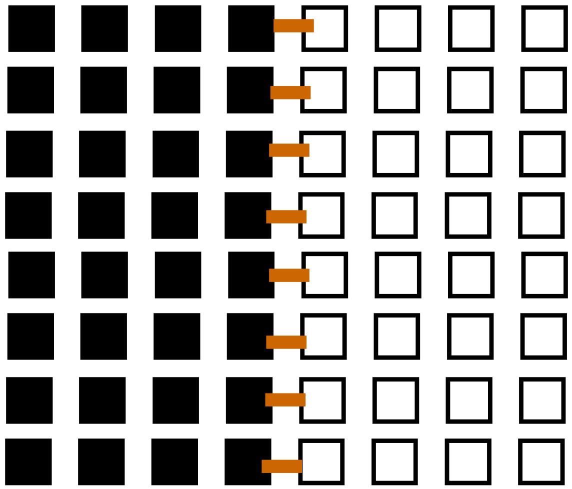
- To model our prior constraints, simply count the number "discontinuities" in a labeling ***L***:
  - There is a discontinuity between pixels ***p*** and ***q*** if labels ***p*** and ***q*** have different labels
  - The more pixels with equal labels group together in coherent groups, the less discontinuities there are

some discontinuities are marked



31 discontinuities

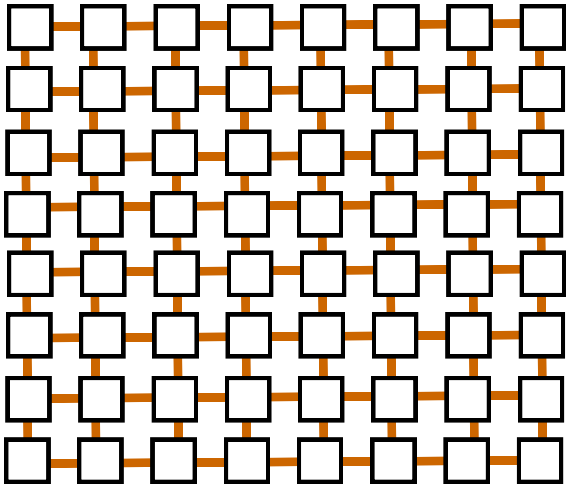
all discontinuities are marked



8 discontinuities

Binary Labeling: Data Constraints

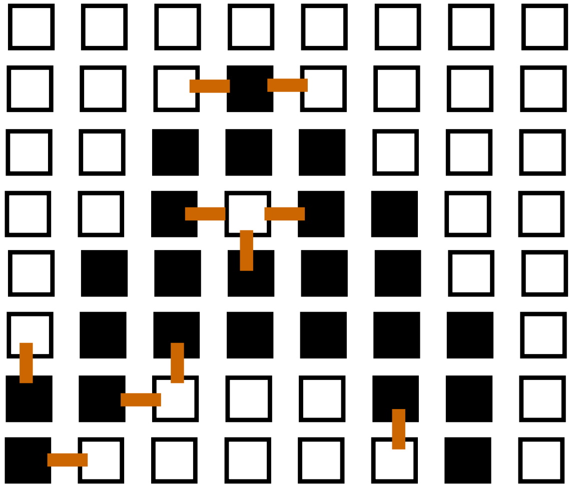
- How to count the number of discontinuities in a labeling ***L***?
  - Let ***N*** be the set of all adjacent pixels
  - Thus our ***N*** consists of edges between immediately adjacent pixels



N consists of all edges

- Let $\mathcal{I}(b)$ be the identity function
  - $\mathcal{I}(b)$ =1 when its argument ***b*** is true
  - $\mathcal{I}(b)$ =0 when its argument ***b*** is false
- To count all discontinuities in a labeling ***L***,

$$\sum_{pq \in N} \mathcal{I}\left(L_p \neq L_q\right)$$



31 discontinuities

Binary Labeling Problem: Energy Formulation

- Our final energy, which takes into consideration the data and the prior constraints is:

data term   prior term

$$E(L) = \sum_{p} D_P(L_p) \quad + \quad \lambda \sum_{pq \in N} \mathcal{I}(L_p \neq L_q)$$

Data term says that in a good labeling L pixels should be labeled as close as possible to their colors in the original (noisy) image

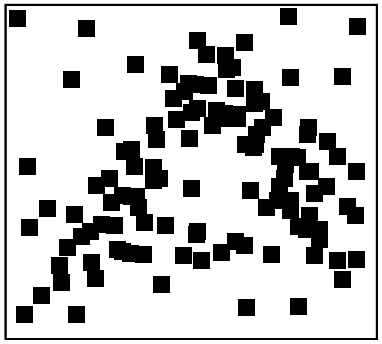
Prior term says that in a good labeling L pixels should be labeled as to form spatially coherent blocks (as few discontinuities as possible)

Binary Labeling Problem: Energy Formulation

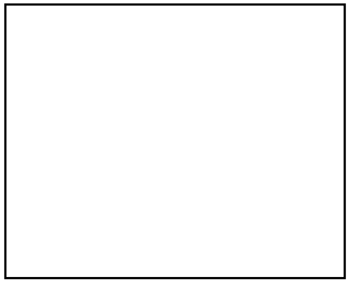
data term   prior term

$$E(L)=\sum_{p} D_p(L_p) + \lambda \sum_{pq\in N} I(L_p \neq L_q)$$

- Note that data term and prior term want different things:
  - Best labeling as far as the data term is concerned is the original image:



  - Best labeling considering only the prior term is completely black or completely white:



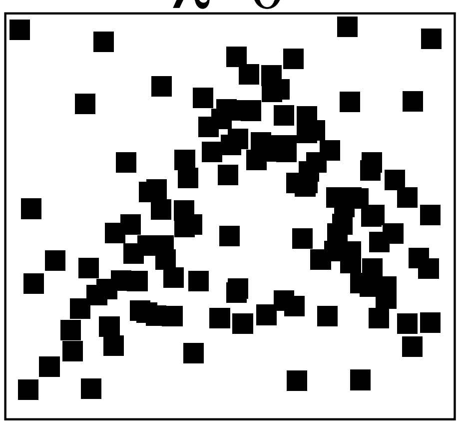
Binary Labeling Problem: Energy Formulation

data term **prior term**

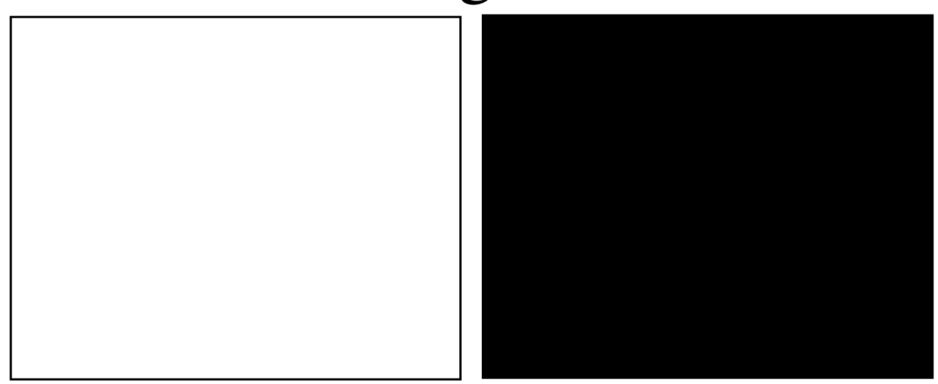
$$E(L) = \sum_{p} D_p(L_p) \quad + \quad \lambda \sum_{pq \in N} I(L_p \neq L_q)$$

- λ serves as a balancing parameter between the data term and the prior term:
  - The larger the λ, the less discontinuities in the optimal labeling ***L***
  - The smaller the λ, the more the optimal labeling ***L*** looks like the original image

λ=0



λ=huge number



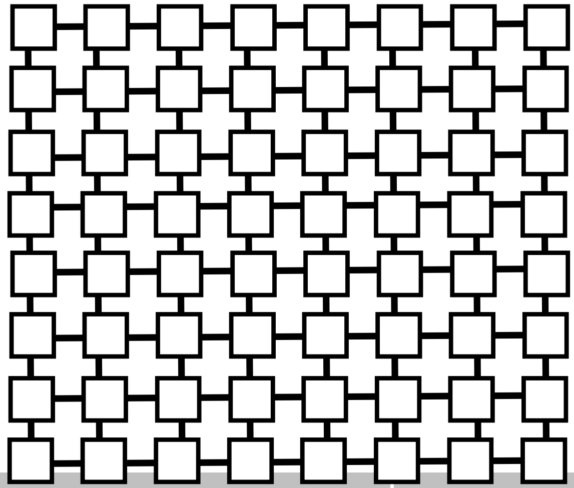
s-t Graph Cuts for Binary Energy Minimization

data term / prior term

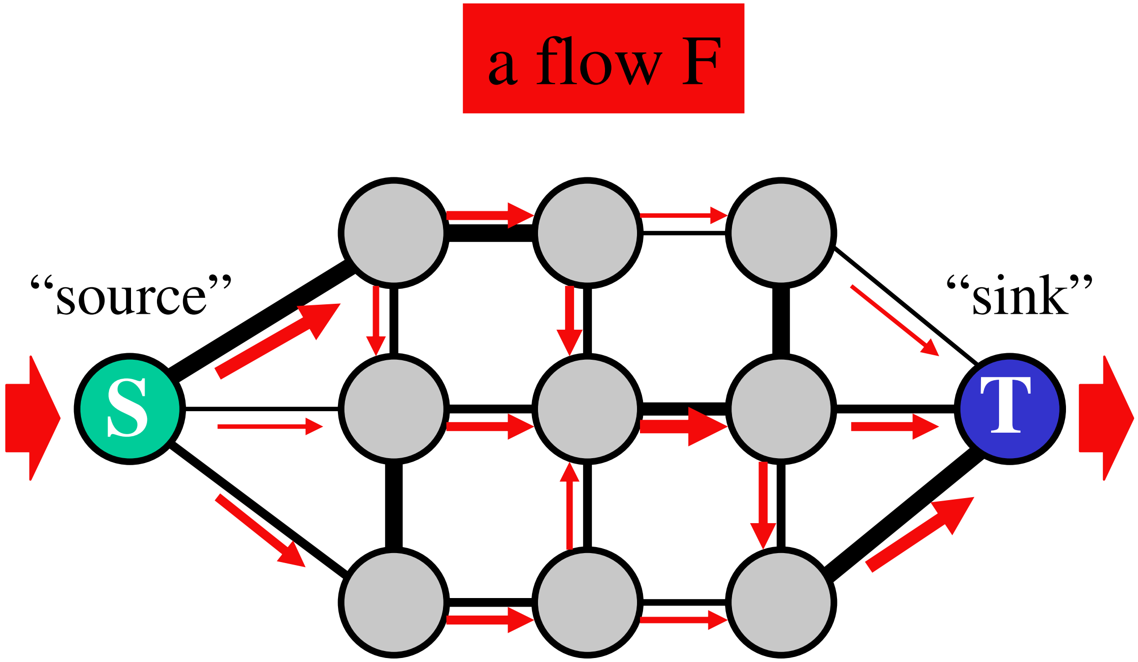
$$E(L) = \sum_{p} D_p(L_p) + \lambda \sum_{pq \in N} I(L_p \neq L_q)$$

- Now that we have an energy function, the big question is how do we minimize it?

- Exhaustive search is exponential: if n is the number of pixels, there are 2^n possible labelings L



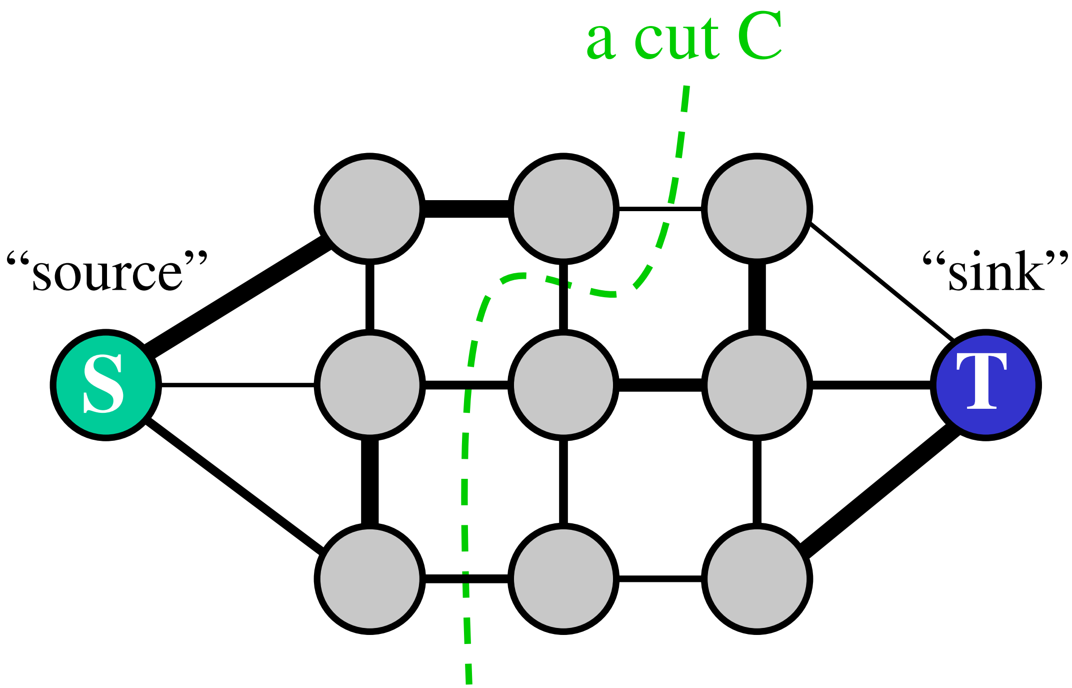
Maximum flow problem



A graph with two terminals

- Max flow problem:
  - Each edge is a "pipe"
  - Find the largest flow F of "water" that can be sent from the "source" to the "sink" along the pipes
  - Source output = sink input = flow value
  - Edge weights give the pipe's capacity

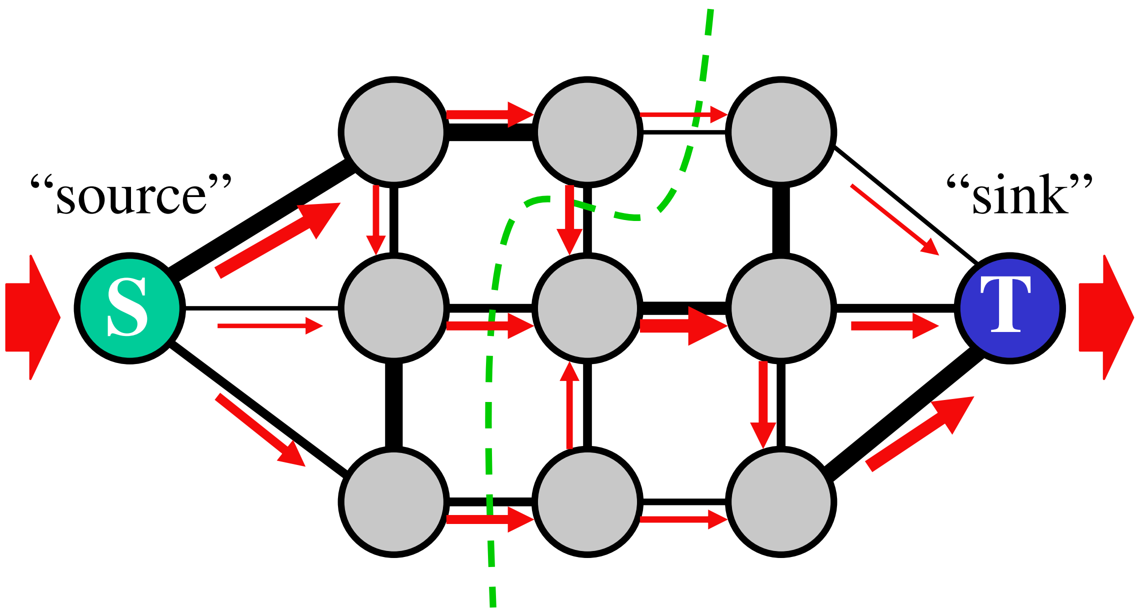
Minimum cut problem



A graph with two terminals

- Min cut problem:
  – Find the cheapest way to cut the edges so that the “source” is separated from the “sink”
  – Cut edges going from source side to sink side
  – Edge weights now represent cutting “costs”

Max flow/Min cut theorem



A graph with two terminals

- Max Flow = Min Cut:
  - Proof sketch: value of a flow is value over any cut
  - Maximum flow saturates the edges along the minimum cut
    - Ford and Fulkerson, 1962
    - Problem reduction!
- Ford and Fulkerson gave first polynomial time algorithm for globally optimal solution