CS664 Lecture #10:
Graph cuts for more than two labels

Some material taken from:

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

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

Recap of recap

- Markov chains allow you to sample from a high-dimensional distribution
  - Biased walk on a graph (Metropolis)
- Consider all possible binary images
- The distribution we want makes the low energy solutions have high probability
  - In an unbiased walk we get everywhere equally often
  - So we use Metropolis intead!
- Graph cuts do this better

Recap

- We want to minimize the energy $E(f)$

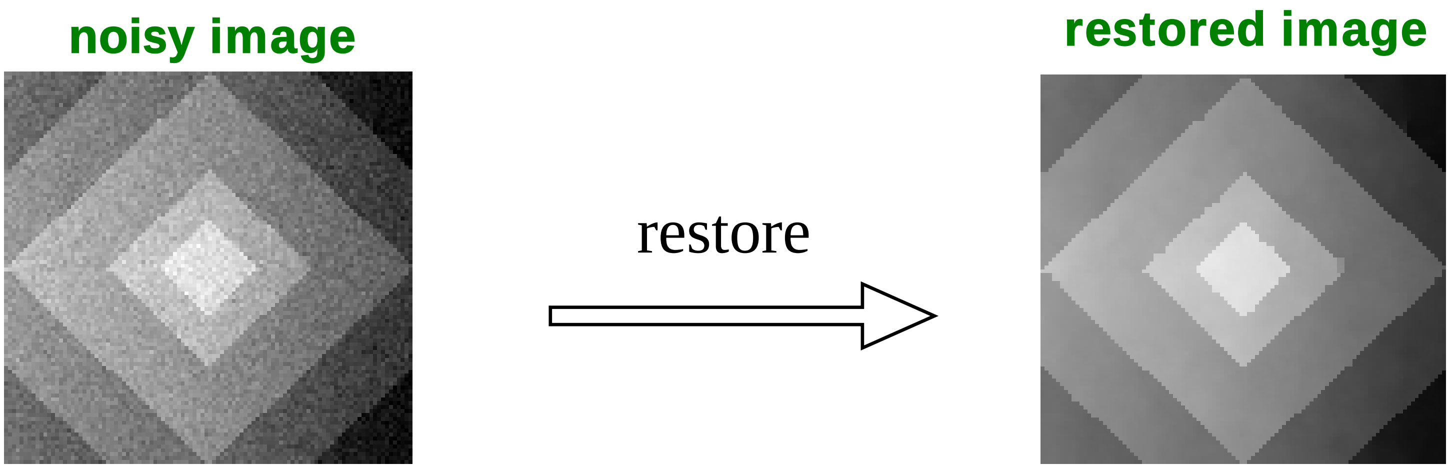
$$\arg\min_f \underbrace{\sum_{p \in \mathcal{S}} D_p(f_p)}_{\text{assignment costs}} + \underbrace{\sum_{p,q \in \mathcal{N}} V(f_p, f_q)}_{\text{separation costs}}$$

- Binary labeling problem can be solved exactly with graph cuts
  - Lots of cool applications!

Non-binary Labeling Problem

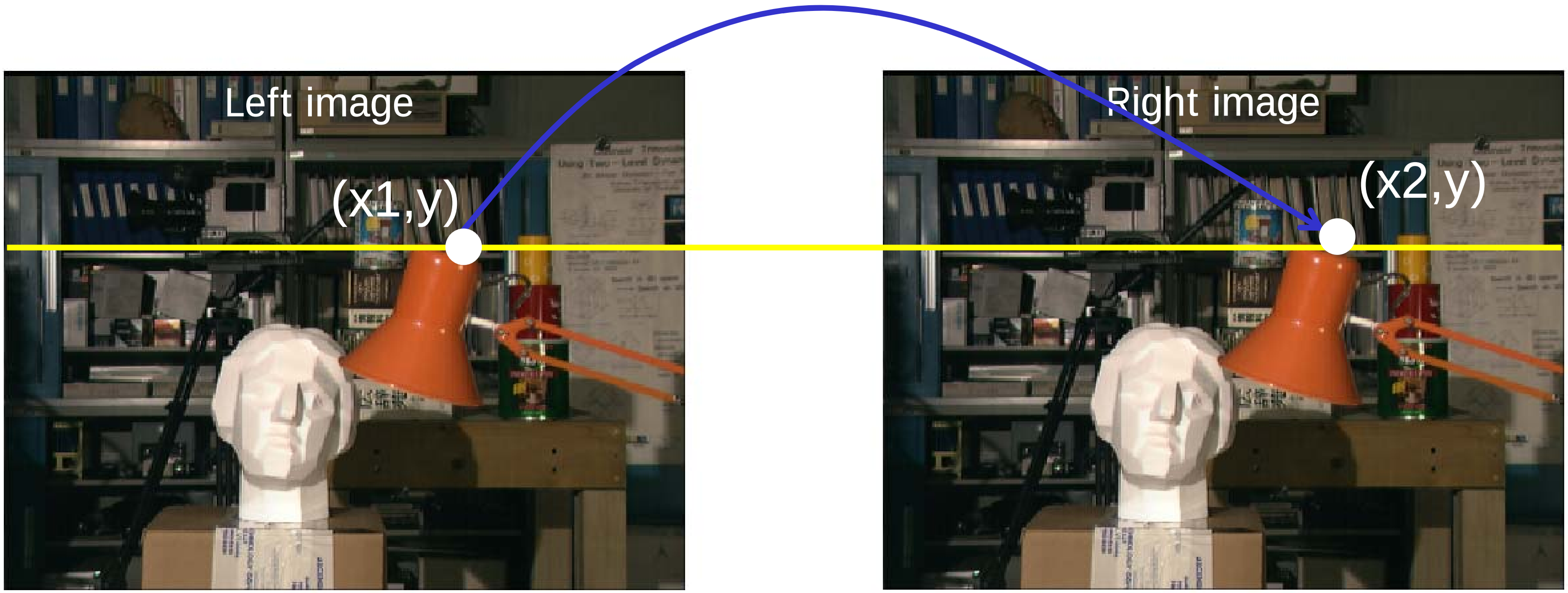
- In many problems, the number of labels is larger than 2
  - General image restoration, stereo correspondence, motion, etc.
- Everything else stays the same
  - Each pixel (site) has preferences for some labels over the others
  - We seek a solution where nearby sites (pixels) are assigned the same label (as much as possible)

Example: Image Denoising



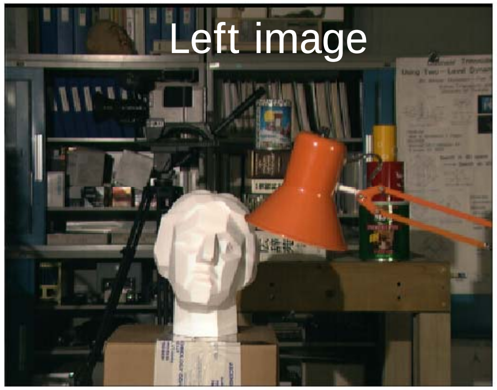
- This is basically same problem as fax image restoration, only for non-binary image
- The number of labels is 256, all grayscale values
  - Labels are integers in the range {0,1,…,255}
  - In restored image, pixels should be close to original intensity
    - $D_p(label)=\text{abs}[label - I(p)]$
  - Pixels with similar intensity should "group together"

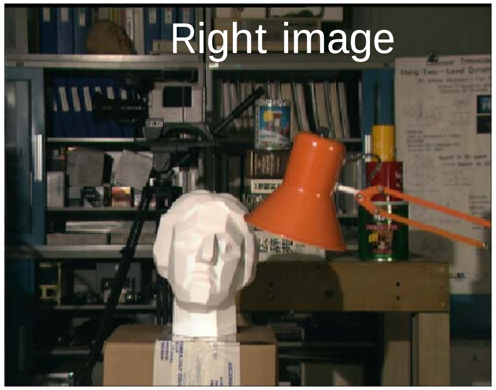
Example: Stereo Matching

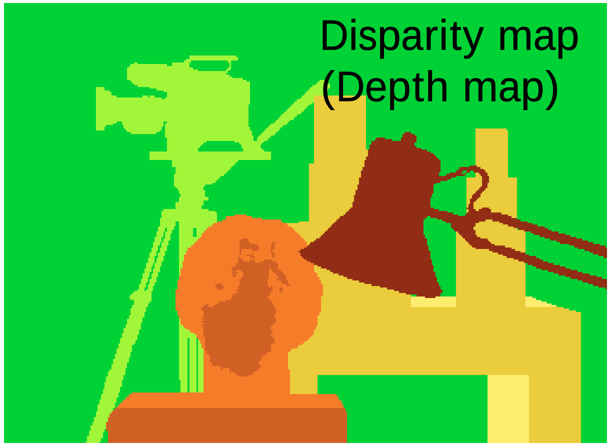


- Disparity is x1-x2
  - Can make this non-negative, in a limited range
- Labels = All possible disparities = {0,1,…,max}

Example: Stereo Matching







- Here Disparity map is colored for visibility
  - Dark red = 14, medium red = 10,…, medium green = 6, dark green = 5
- A possible data term:
  - Corresponding pixels are expected to have similar colors
  - For disparity ***d***, $D_P(d)$=abs(LeftImage(***p***)-RightImage(***p-d***))
    - If $p = (x,y)$, $p - d = (x-d\,,y)$

Multi-label Energy

- Many different energy functions in multi-label case
- Our old energy function still makes sense:

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

  - It still says the same thing, except that the number of labels that can be assigned to a pixel is more than 2
    - Find a labeling of pixels s.t. each pixel is assigned a label it likes as much as possible, and nearby pixels are assigned the same label as much as possible
- Unfortunately, minimizing this energy in case of more than 2 labels is NP-hard

Multi-label Energy: Linear Interactions

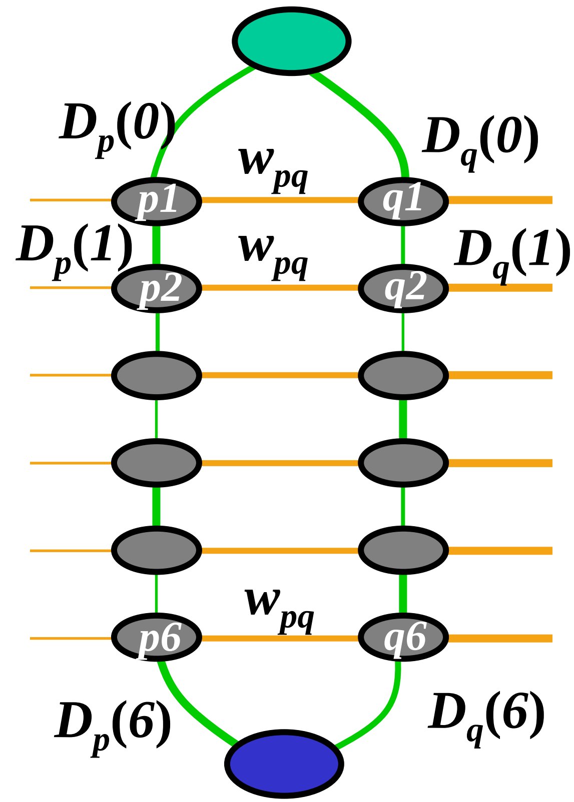
- The following energy **can** be minimized exactly with an *s-t* graph cut:

$$E(L) = \sum_p D_p(L_p) \; + \; \sum_{pq \in N} w_{pq} \left| L_p - L_q \right|$$

- Data term is as before
- Penalty for discontinuity is now proportional to the discontinuity “size”
  - If neighboring pixels $\boldsymbol{p}$ and $\boldsymbol{q}$ do not have the same label, penalty is the absolute difference between the labels (multiplied by $\boldsymbol{w_{pq}}$)
  - Thus the larger is the difference between their labels ($\boldsymbol{L_p}$ and $\boldsymbol{L_q}$) the more costly is it assigning label $\boldsymbol{L_p}$ to pixel $\boldsymbol{p}$ and label $\boldsymbol{L_q}$ to pixel $\boldsymbol{q}$
- Energy function with such prior term is said to have ***linear interactions***

Multi-label Energy

$$E(L)=\sum_p D_p(L_p) + \sum_{pq\in N} w_{pq}|L_p - L_q|$$

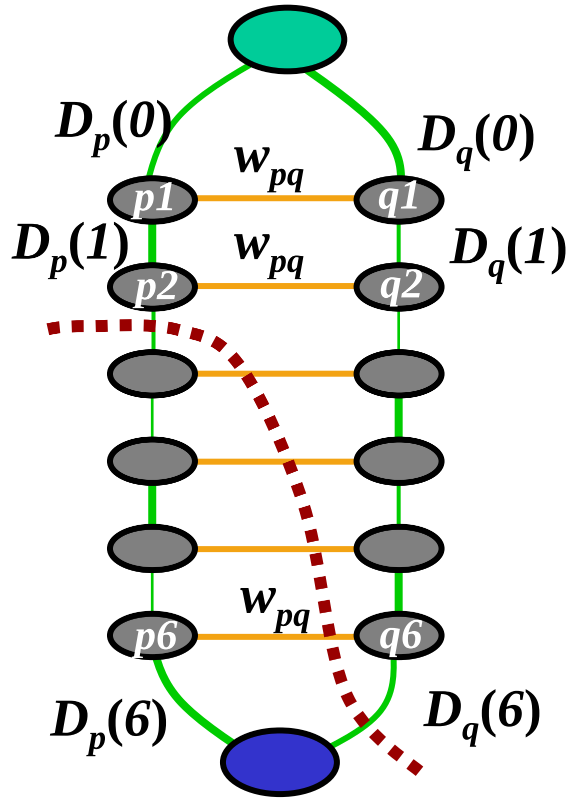


- Suppose there are ***d*** +1 labels, {0,1,…,***d***}
- For each pixel ***p***, create ***d*** vertices, named $p_1, p_2, \ldots, p_d$
- Create ***d***+1 ***t***-links for each pixel ***p***
  - ***t*** –link #1 goes between source ***s*** and p_1 and has weight $D_p(0)$ +***K***
  - ***t*** –link #2 goes between vertices p_1 and p_2 and has weight $D_p(1)$ +***K***
  - …..
  - ***t*** –link #***d***+1 goes between source p_d and p_2 and has weight $D_p(1)$ +***K***
- For neighboring pixels ***p*** and ***q***, for all ***i***, p_i is connected to q_i by an ***n***-link of weight w_{pq}
- If ***t*** –link #***i*** is cut, then pixel ***p*** is assigned disparity ***i***
  - ***K*** is large enough to ensure that only one ***t***-link per pixel is in a minimum cut, for example ***K*** = sum of all ***n***-links

Multi-label Energy

$$E(L) = \sum_p D_p(L_p) + \sum_{pq \in N} w_{pq} |L_p - L_q|$$

- The mincut severs only one ***t***-link per pixel
- If a cut severs t-link ***i*** for pixel ***p*** and ***t***-link ***j*** for pixel ***q***, it must sever all ***n***-links between p_i and q_{j-1}
- If a cut severs t-link ***i*** for pixel ***p*** and ***t***-link ***j*** for pixel ***q***, it must sever all ***n***-links between p_i and q_{j-1}
- Thus the cost of the severed ***n***-links is exactly as required by the energy function

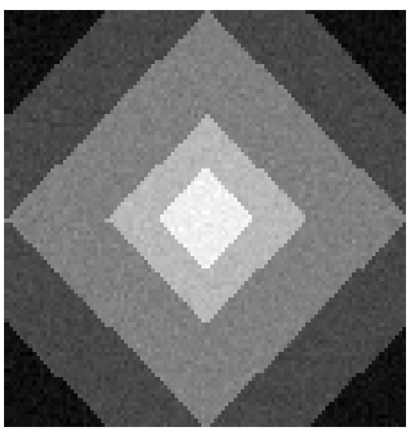


- Notice that
  - the cost of severed ***t***-links contributes to the data term
  - the cost of severed ***n***-links contributes to the prior term

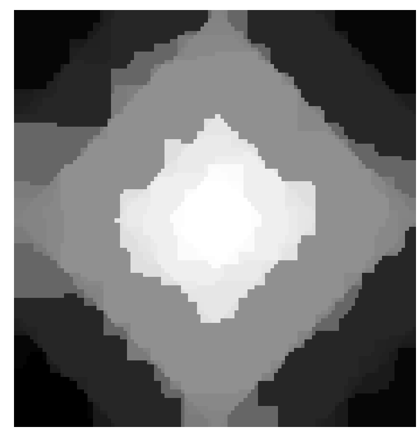
Over-smoothing

$$E(L)=\sum_{p} D_p(L_p) \; + \sum_{pq\in N} w_{pq}\left|L_p - L_q\right|$$

- Above energy can be minimized exactly, but there are serious problems with it
- The problem is that it is not discontinuity preserving
  - Sharp (large $|L_p\text{-}L_q|$) discontinuities are penalized too much
  - Instead of one large discontinuity, it is cheaper to create a few smaller discontinuities



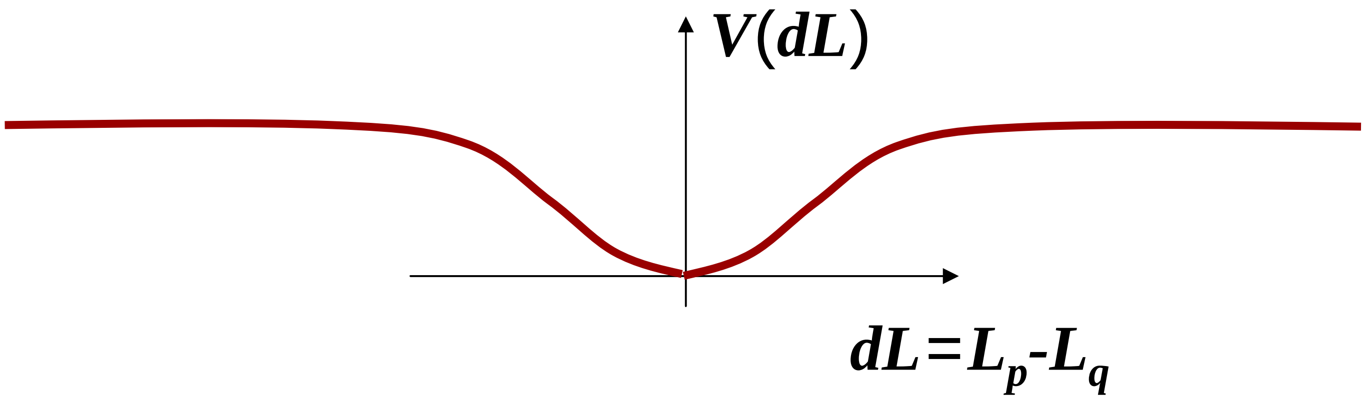
restoration with linear interactions



Discontinuity Preserving Interactions V

$$E(L)=\sum_{p} D_P(L_p) \;+\; \sum_{pq\in N} w_{pq}|L_p - L_q|$$

- In order to be discontinuity preserving, pixel interaction function $V(L_p, L_q)$ should not grow to large
  - At some point, we decide that there is a discontinuity between neighboring pixels, and assign a fixed penalty for this discontinuity, independent of the size of this discontinuity



Generalization

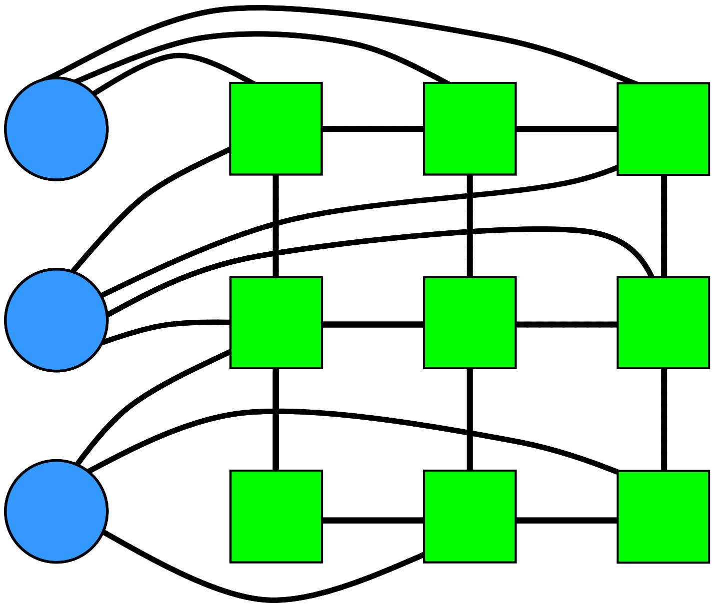
- Ishikawa '03 showed how to handle *any* convex function V
  - Add "diagonal" n-links between pixel nodes
    - With the right choice of edge weights
  - Labels must be 0,1,2,…
  - Graph has |P|·|L| nodes
    - Space is more expensive than time
- Linear model is the "most robust non-robust" solution
  - Clear line between easy and NP-hard

Handling robust V

- How do we handle the problem we really want to solve?
  - Multiple labels
  - Discontinuity-preserving (robust) V
- Can we generalize the binary construction?
- Initially we will focus on the Potts model
  - Raises all the major issues
  - Surprisingly important and flexible

Min cost multiway cut problem

- Graphs with a set of terminal nodes
- Multiway cut is a set of edges that separates all terminals
  - Cost of a cut is the sum of edge weights



Potts model result

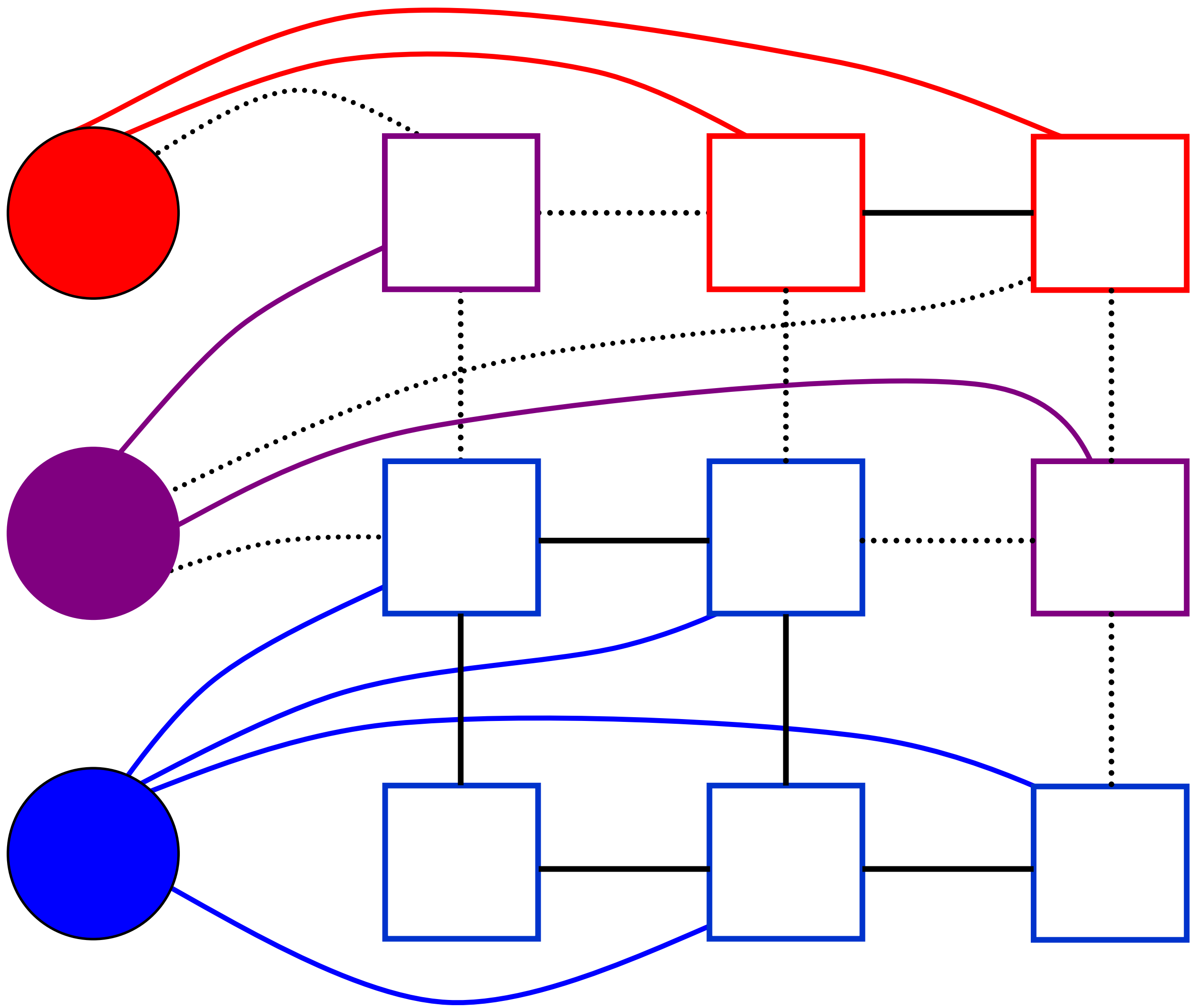
Theorem [BVZ98]: Minimizing

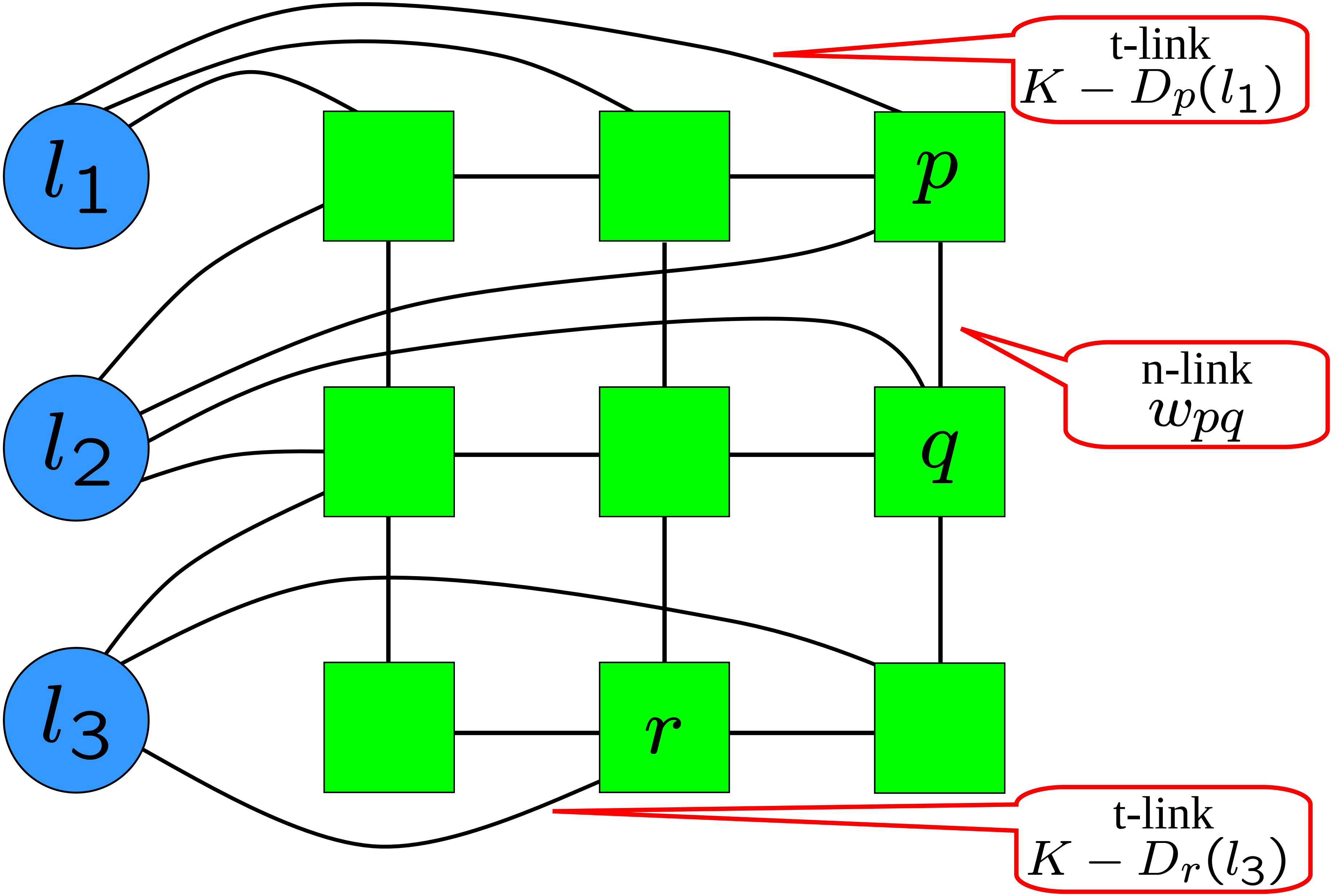
$$E(f) = \left(\sum_{p} D_p(f_p) + \sum_{pq \in N} w_{pq} \cdot \mathrm{T}(f_p \neq f_q) \right)$$

is equivalent to finding a minimum cost multiway cut on a certain graph

Note: we will construct graphs where all terminals are connected to all non-terminals. Some of these links will not be shown for legibility.

Multiway cuts are labelings




t-link
$K - D_p(l_1)$
l_1
p
n-link
w_{pq}
l_2
q
l_3
r
t-link
$K - D_r(l_3)$

Computing a multiway cut

- With 2 terminals the multiway cut problem reduces to the minimum cut problem
  – More than 2 terminals: NP-hard
    [Dahlhaus *et al.*, STOC ‘92]
- Efficient approximation algorithms exist
  – Within a factor of 2 of optimal
  – Need to build your own for vision (why?)