Image Processing

Discrete Energy Minimization

Denoising → Segmentation

Both the domain of definition and range are discrete

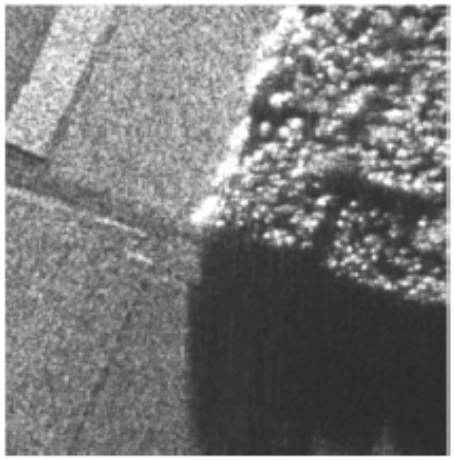
Imagine:

1. Colors are “semantically” meaningful
2. There are not many colors

The task is to partition the image in the “meaningful” regions

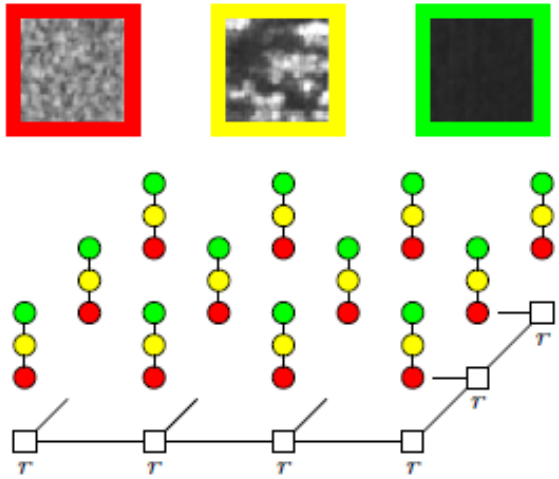
→ **Segmentation**

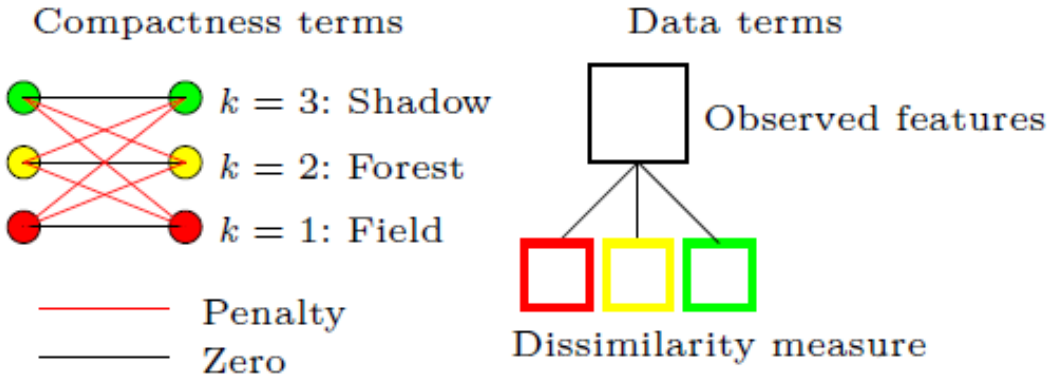
Segmentation



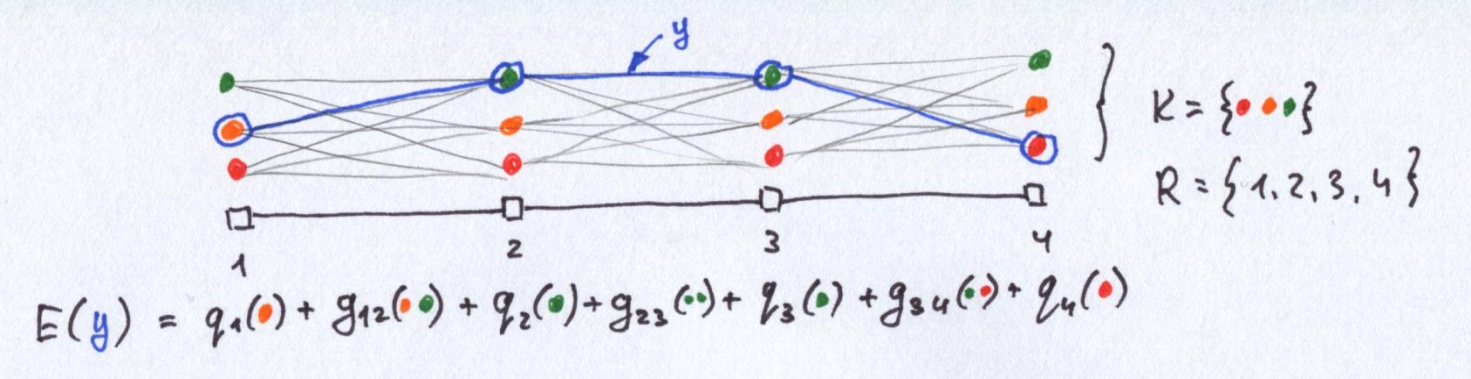
Original

A possible segmentation

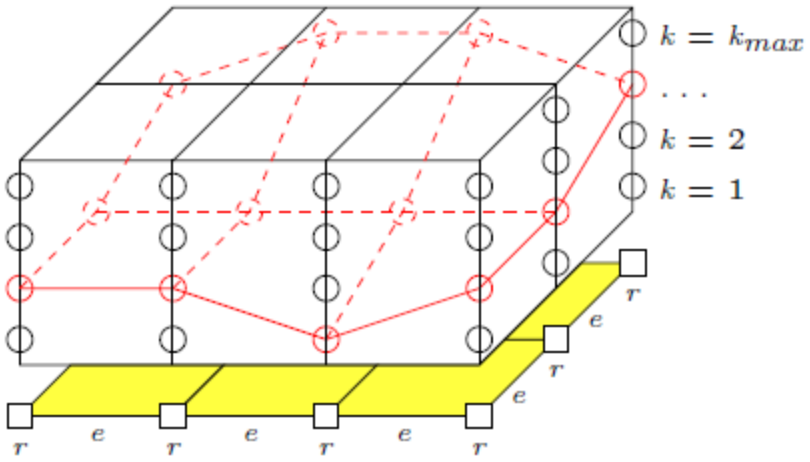




Discrete Energy Minimization



$$y^* = \arg\min_y \left[\sum_r q_r(y_r) + \sum_{rr'} g_{rr'}(y_r, y_{r'}) \right]$$



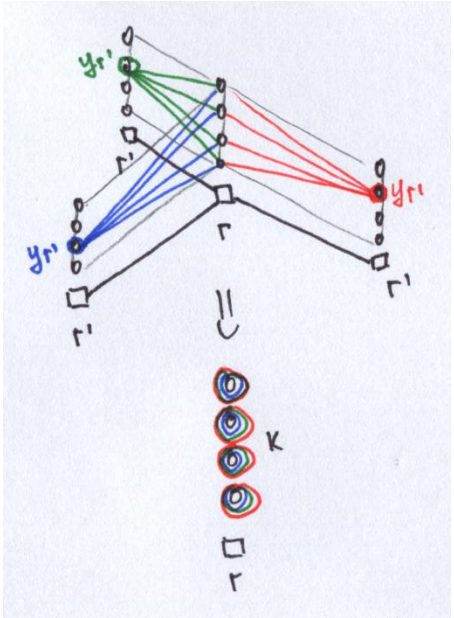
Iterated Conditional Modes

$$y^* = \arg\min_y \left[\sum_r q_r(y_r) + \sum_{rr'} g_{rr'}(y_r, y_{r'}) \right]$$

Idea: choose (locally) the best label for the fixed rest [Besag, 1986]

Repeat:

$$y_r = \arg\min_k \left[q_r(k) + \sum_{r':rr' \in E} g_{rr'}(k, y_{r'}) \right]$$



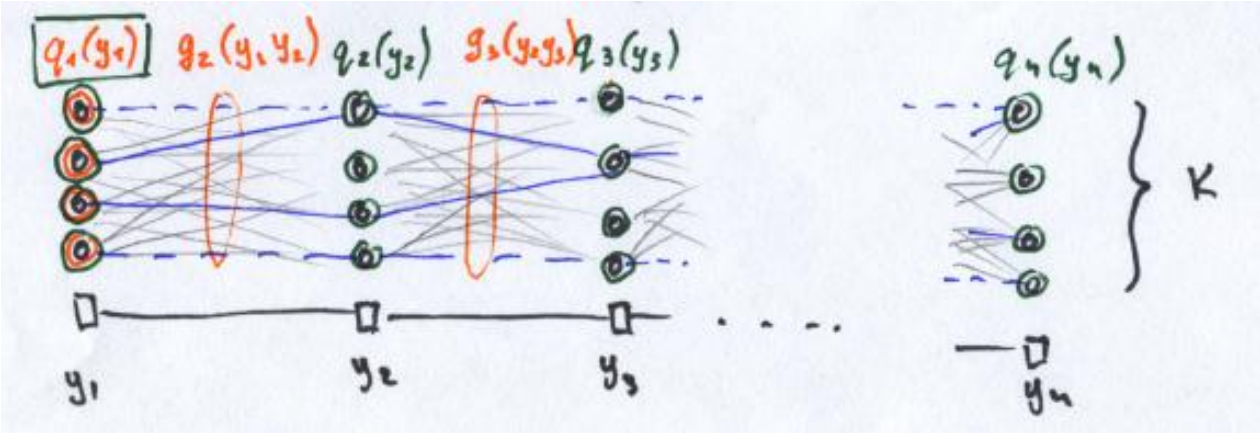
(PR: remember on the synchronous dynamics for Hopfield-networks)

☺ extremely simple, parallelizable

☹ coordinate-wise optimization, does not converge to the global optimum even for very simple energies

Dynamic Programming

Suppose that the image is one pixel high → a **chain**

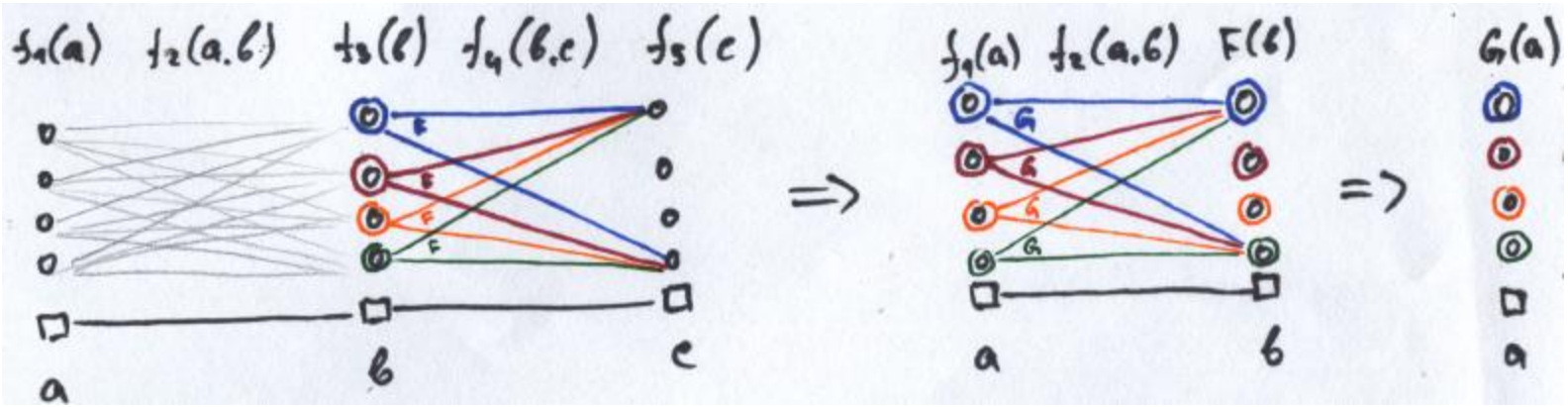


The goal is to compute

$$\min_y \left[\sum_{i=1}^{n} q_i(y_i) + \sum_{i=2}^{n} g_i(y_{i-1}, y_i) \right]$$

Dynamic Programming – example

$$f(a,b,c) = f_1(a) + f_2(a,b) + f_3(b) + f_4(b,c) + f_5(c)$$



$$\min_{abc} f(a,b,c) = \min_a \min_b \min_c \left[f_1(a) + f_2(a,b) + f_3(b) + f_4(b,c) + f_5(c)\right] =$$

$$= \min_a \left[f_1(a) + \min_b \left[f_2(a,b) + f_3(b) + \min_c \left[f_4(b,c) + f_5(c)\right]\right]\right] =$$

$$\ldots\text{denote} \quad F(b) = f_3(b) + \min_c \left[f_4(b,c) + f_5(c)\right] \ldots$$

$$= \min_a \left[f_1(a) + \min_b \left[f_2(a,b) + F(b)\right]\right] =$$

$$\ldots\text{denote} \quad G(a) = f_1(a) + \min_b \left[f_2(a,b) + F(b)\right] \ldots$$

$$= \min_a G(a)$$

Dynamic Programming (derivation)

$$\min_{y} \left[\sum_{i=1}^{n} q_i(y_i) + \sum_{i=2}^{n} g_i(y_{i-1}, y_i) \right] =$$

$$\min_{y_1} \min_{y_2} \dots \min_{y_n} \left[\sum_{i=1}^{n} q_i(y_i) + \sum_{i=2}^{n} g_i(y_{i-1}, y_i) \right] =$$

$$\min_{y_2} \dots \min_{y_n} \left[\sum_{i=2}^{n} q_i(y_i) + \sum_{i=3}^{n} g_i(y_{i-1}, y_i) + \min_{y_1} \Big(q_1(y_1) + g_2(y_1, y_2) \Big) \right] =$$

$$\min_{y_2} \dots \min_{y_n} \left[\sum_{i=2}^{n} q_i(y_i) + \sum_{i=3}^{n} g_i(y_{i-1}, y_i) + F(y_2) \right] =$$

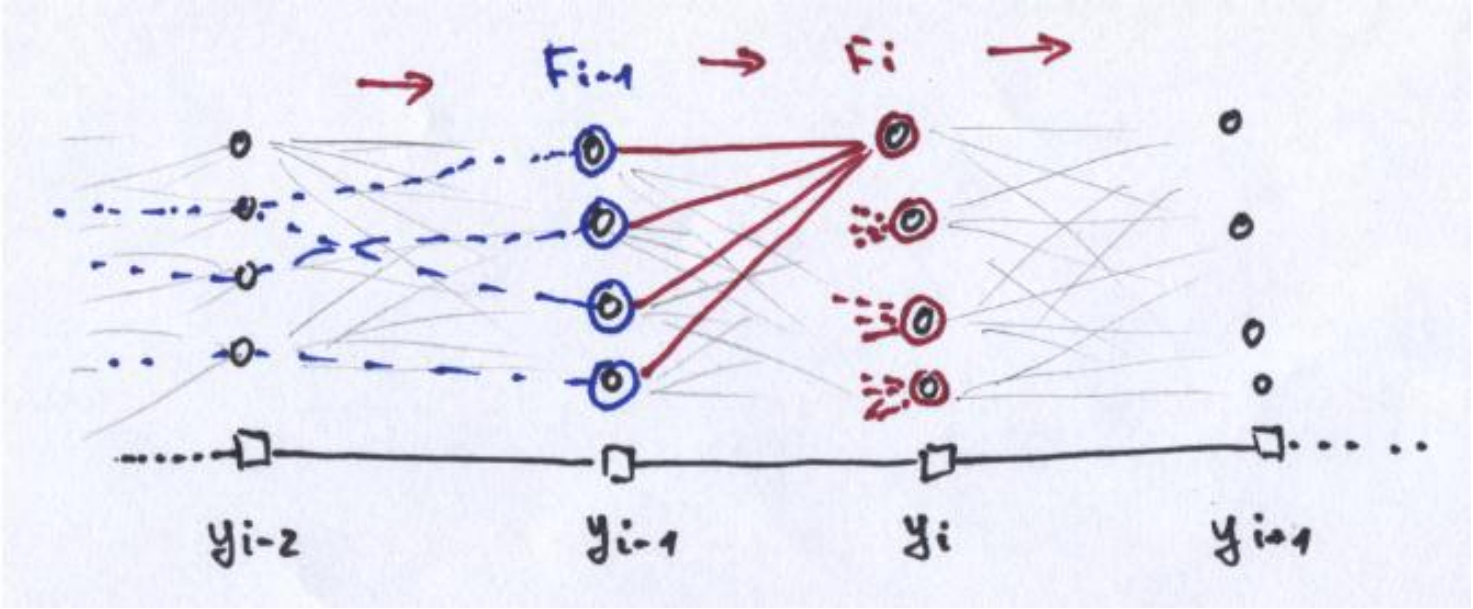
$$\min_{y_2} \dots \min_{y_n} \left[\sum_{i=2}^{n} \tilde{q}_i(y_i) + \sum_{i=3}^{n} g_i(y_{i-1}, y_i) \right]$$

with $\tilde{q}_2(k) = q_2(k) + F(k)$, $\tilde{q}_i(k) = q_i(k)$ for $i = 3 \dots n$

Dynamic Programming

General idea – propagate **Bellman functions** F_i by

$$F_i(k) = q_i(k) + \min_{k'} \left[F_{i-1}(k') + g_i(k', k) \right]$$



The Bellman functions represent the quality of the best expansion onto the processed part.

Dynamic Programming (algorithm)

// Forward pass

for $i = 2$ **bis** n

  for $k = 1$ **bis** K

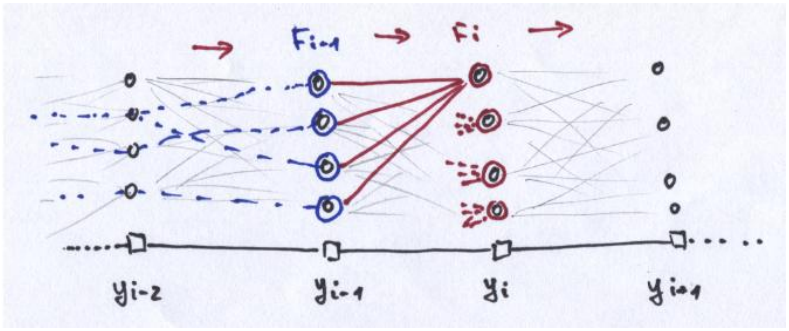
    $best = \infty$

    for $k' = 1$ **bis** K

      if $q_{i-1}(k') + g_i(k', k) < best$

        $best = q_{i-1}(k') + g_i(k', k)$, $pointer_i(k) = k'$

    $q_i(k) = q_i(k) + best$



// Backward pass

$best = \infty$

for $k = 1$ **bis** K

  if $q_n(k) < best$

    $best = q_n(k)$, $x_n = k$

for $i = n - 1$ **bis** 1

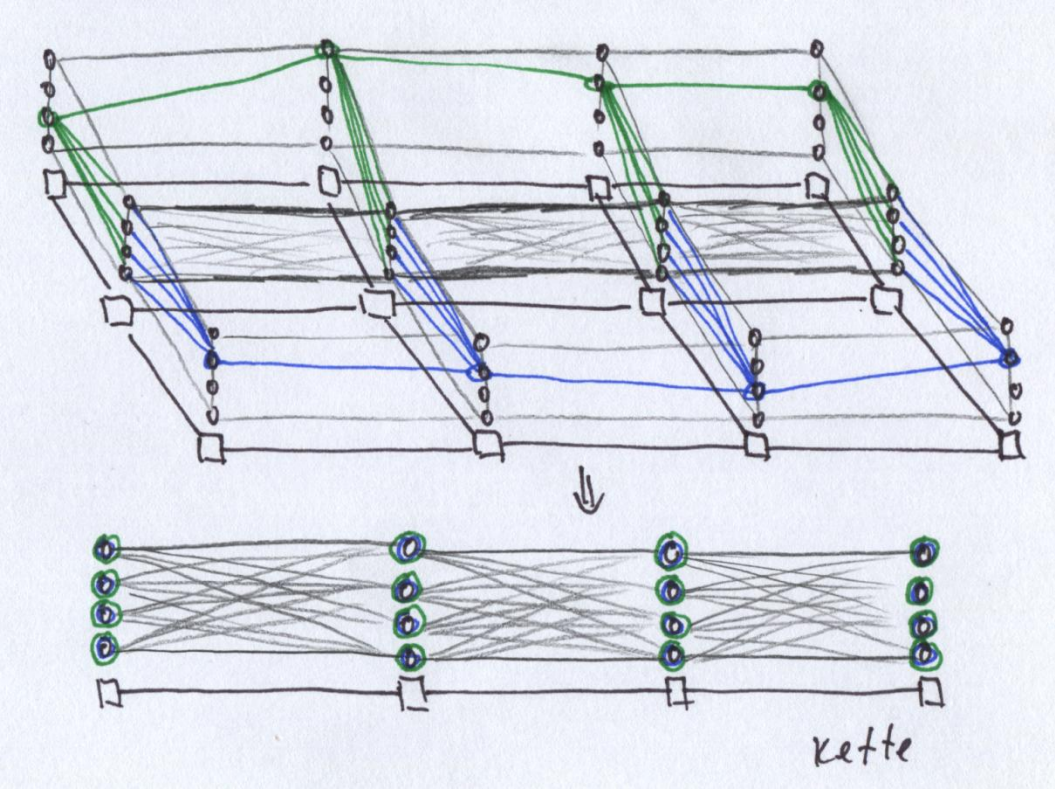
  $x_i = pointer_{i+1}(x_{i+1})$

Time complexity – $O(nK^2)$

$pointer_i(k)$ is the best predecessor for k-th label in the i-th node

Iterated Conditional Mode (2D again)

Fix labels in all nodes but for a chain (e.g. an image row)



The “auxiliary” task is solvable exactly and efficiently by DP

The overall schema – iterate over rows and columns until convergence

Equivalent Transformation (re-parametrization)

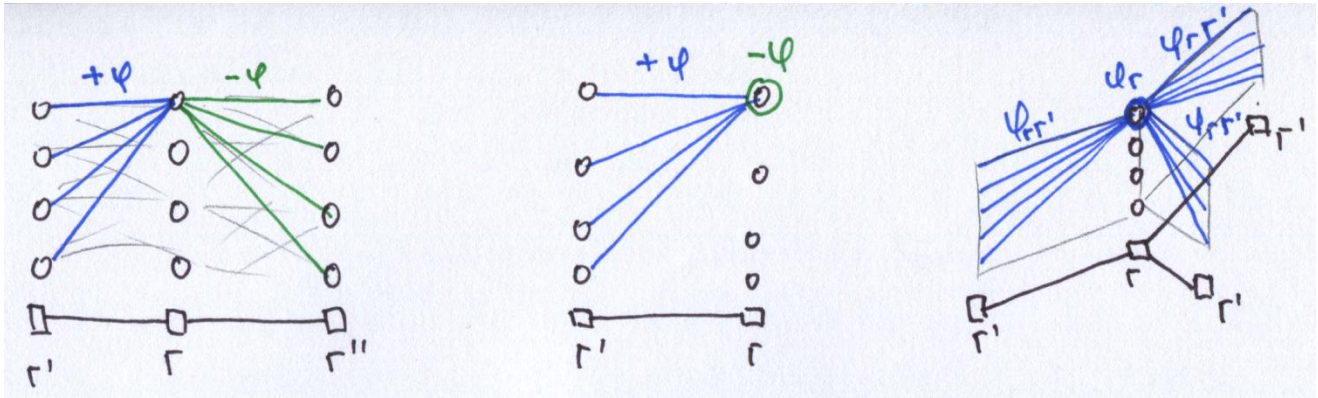
Two tasks $A = (q, g)$ and $A' = (q', g')$ are **equivalent** if

$$\left[\sum_r q_r(y_r) + \sum_{rr'} g_{rr'}(y_r, y_{r'})\right] = \left[\sum_r q'_r(y_r) + \sum_{rr'} g'_{rr'}(y_r, y_{r'})\right]$$

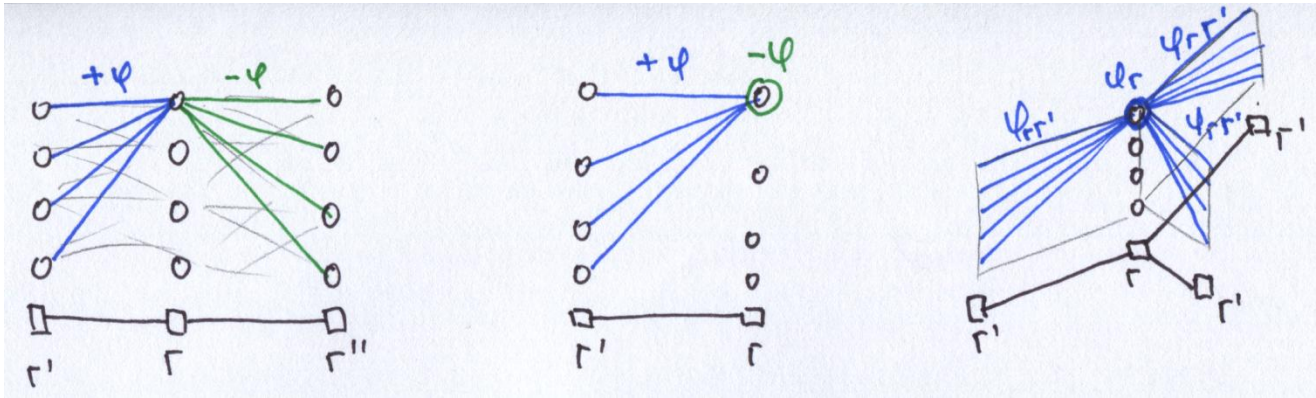
holds for all labelings y.

$\mathcal{A}(A)$ – equivalence class (all tasks equivalent to A).

Equivalent transformation:



Equivalent Transformation



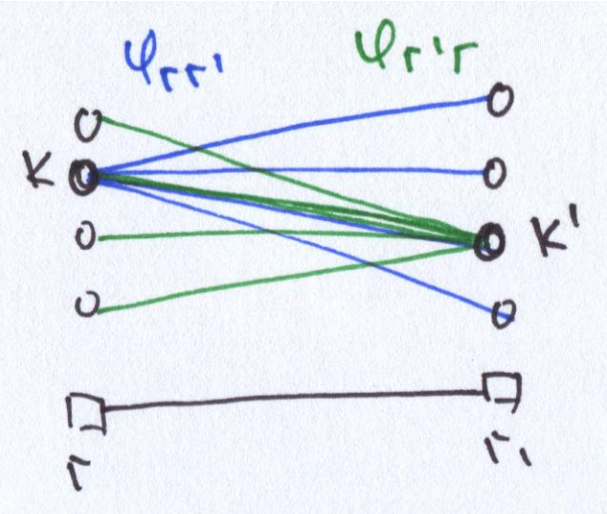
Equivalent transformation can be seen as “vectors”, that satisfy certain conditions:

$$\Phi = \left(\varphi_r(k)\ \forall r, k, \quad \varphi_{rr'}(k),\ \forall rr', k\right)$$

$$\varphi_r(k) + \sum_{r' : rr' \in E} \varphi_{rr'}(k) = 0 \quad \forall r, k$$

Equivalent Transformation

Let $A = (q, g)$ be a task and $A' = (q', g') = \Phi(A)$ be another one after applying an ET Φ (ET can be seen also as operators), i.e.



$$q'_r(k) = q_r(k) + \varphi_r(k)$$
$$g'_{rr'}(k, k') = g_{rr'}(k, k') + \varphi_{rr'}(k) + \varphi_{r'r}(k')$$

→ the tasks A and A' are equivalent.

Less trivial: if two tasks A and A' are equivalent, there exists such transformation Φ, that $A' = \Phi(A)$ holds.

Equivalent Transformation

Further useful properties:

Superposition – consecutive application of two ET:

$$\Phi\Big(\Phi'(A)\Big) = \Phi'\Big(\Phi(A)\Big) = (\Phi \oplus \Phi')(A)$$

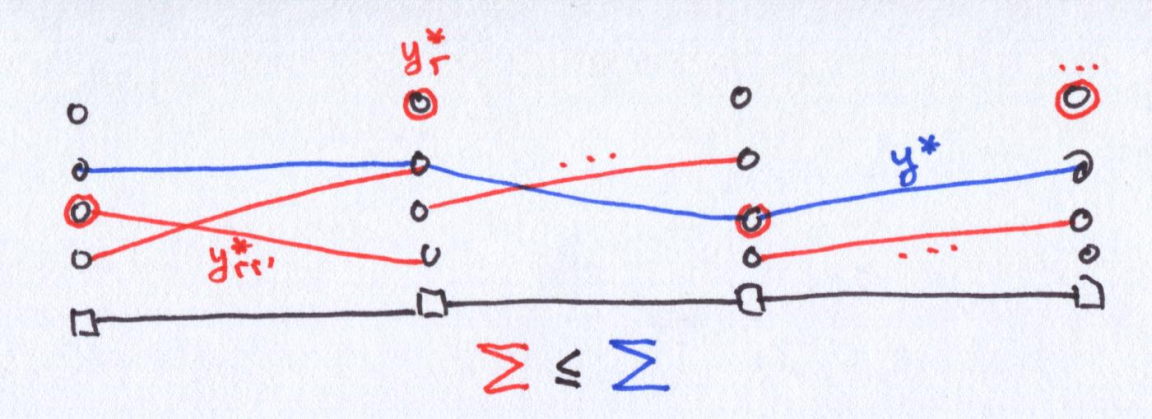
$\oplus$ is the "componentwise" summation.

There exists an "inverse" for each ET:

$$\Phi^{-1}\Big(\Phi(A)\Big) = A, \text{ d.h. } \Phi \oplus \Phi^{-1} = \Phi^0$$

→ the set of all ET Φ composes a **group**

Seeming Quality



Consider the following “stupid” algorithm:

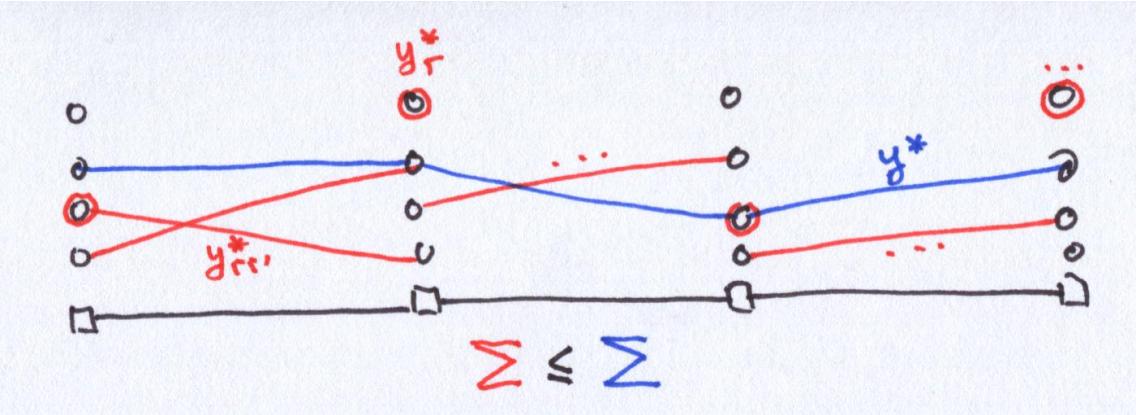
1. Choose independently the best (according to q) label in each node and the best label pair (according to g) in each edge;

2. Sum up all their qualities (call it “**seeming quality**”);

$$SQ(A) = \sum_r \min_k q_r(k) + \sum_{rr'} \min_{kk'} g_{rr'}(k, k')$$

3. Hope that the result is equal to the quality of the best labeling.

Seeming Quality



Compare the best energy

$$E(A) = \min_y \left[\sum_r q_r(y_r) + \sum_{rr'} g_{rr'}(y_r, y_{r'}) \right]$$

and the seeming quality

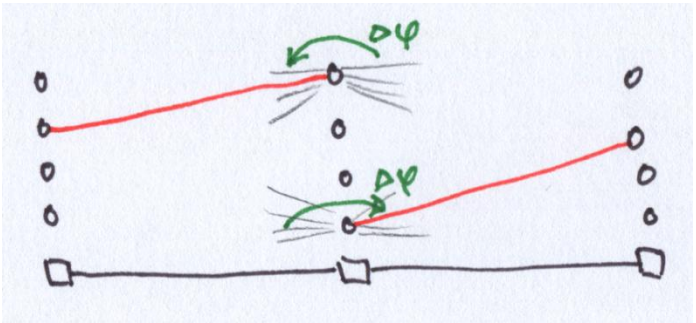
$$SQ(A) = \sum_r \min_k q_r(k) + \sum_{rr'} \min_{kk'} g_{rr'}(k, k')$$

Obviously: $SQ(A) \leq E(A)$ (lower bound)

The task is **trivial**, if $E(A) = SQ(A)$ holds.

Maximize the Seeming Quality

Observation: equivalent transformation do not change the Energy $E(A)$.
However, they change the seeming quality $SQ(A)$!!!



Idea: Try to **maximize** the seeming quality – search for a trivial task in the equivalence class $\mathcal{A}(A)$.

$$\sum_{r} \min_{k} \Big(q_r(k) + \varphi_r(k) \Big) + \sum_{rr'} \min_{kk'} \Big(g_{rr'}(k,k') + \varphi_{rr'}(k) + \varphi_{r'r}(k') \Big) \to \max_{\Phi}$$

$$\text{s.t. } \varphi_r(k) + \sum_{r' : rr' \in E} \varphi_{rr'}(k) = 0 \quad \forall r, k$$

Maximize the Seeming Quality

$$\sum_r \min_k \Big(q_r(k) + \varphi_r(k) \Big) + \sum_{rr'} \min_{kk'} \Big(g_{rr'}(k,k') + \varphi_{rr'}(k) + \varphi_{r'r}(k') \Big) \to \max_{\Phi}$$

$$\text{s.t. } \varphi_r(k) + \sum_{r' : rr' \in E} \varphi_{rr'}(k) = 0 \quad \forall r, k$$

The subject is concave but not everywhere differentiable, the conditions are linear.

Problems:

1. How to optimize SQ **efficiently**?
2. Checking the triviality is NP-complete ☹
3. Not for every A there exists a trivial equivalent

Diffusion Algorithm

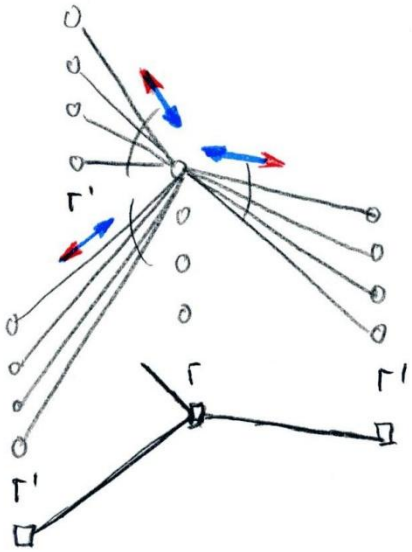
Repeat for all r, k :

1. Accumulate: put as much as possible to the node:

$$\triangle_{rr'}(k) = \min_{k'} g_{rr'}(k, k')$$

$$q_r(k) = q_r(k) + \sum_{r' : rr' \in E} \triangle_{rr'}(k, k')$$

$$g_{rr'}(k, k') = g_{rr'}(k, k') - \triangle_{rr'}(k, k')$$



2. Distribute equally to the incident edges $g_{rr'}(k, k')$:

$$\triangle_r(k) = q_r(k)/4 \quad \text{(4-neighborhod)}$$

$$g_{rr'}(k, k') = g_{rr'}(k, k') + \triangle_r(k)$$

$$q_r(k) = 0$$

Diffusion Algorithm

It Is not clear, what for a task the Diffusion Algorithm does solve (the algorithm is not derived from the original optimization task).

In general the seeming quality is not globally optimized.

In practice works often satisfactory

Other algorithms: Message Passing (specific equivalent transformation), Sub-gradient methods …

Solvable classes of discrete energy minimization:

- The graph of the task is simple (chain, tree, partial k-trees …)
- The pairwise functions are **submodular** (MinCut methods)