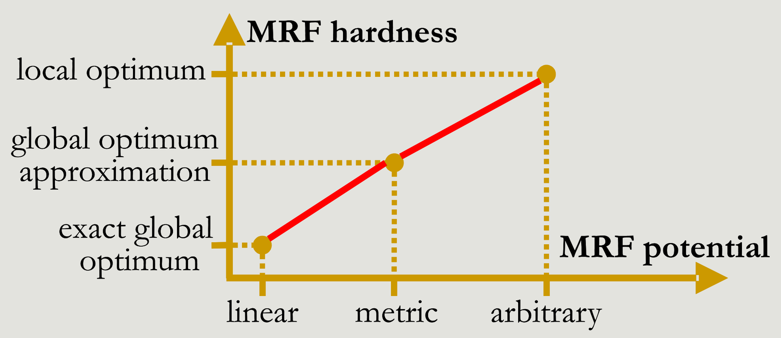
A New Framework for Approximate Labeling via Graph Cuts

Nikos Komodakis & Georgios Tziritas

Computer Science Department, University of Crete

Introduction

Markov Random Fields (MRFs) are of paramount importance to vision. They can capture a broad range of NP-complete vision problems (e.g. stereo matching, image restoration, optical flow estimation etc.).
However, the hardness of optimizing an MRF (i.e. what type of optimum an algorithm can compute) depends critically on the type of its potential:



Up to now graph-cuts could cope well only with metric MRFs. We need new approximation algorithms that go beyond that limitation.

Contributions

A new framework for designing approximation algorithms is proposed which is based on duality theory of Linear Programming:

- ❑It can handle a very wide class of MRFs with both metric & non-metric energies (an important class of problems in vision)
- ❑Even for non-metric energies its algorithms have guaranteed optimality properties (i.e. worst-case suboptimality bounds)
- ❑It includes the state-of-the-art "min-cut α-expansion" method merely as a special case (for metric energies)
- ❑Besides the worst-case bounds, its algorithms also provide much smaller per-instance suboptimality bounds which prove to be very tight in practice (i.e. very close to 1) ⇒ generated solutions are nearly optimal!
- ❑Offers new insights into existing graph-cut techniques

A comparison of our framework to existing optimization approaches

Method	Metric energy	Semimetric energy
α-β-swap	local optimum	local optimum
α-expansion	approximation	×
our framework	*approximation*	*approximation*
belief propagation	not guaranteed to converge	

In fact, our framework easily extends to apply not only to semimetric energies but also to a much more general set of non-metric energies.

Metric Labeling (ML) problem

❑Given objects V (residing in graph G) and labels L, find labeling $f: V \to L$ of minimum total cost defined as follows:

$$cost(f) = \sum_{p \in V} c_{p,f(p)} + \sum_{(p,q) \in edges(G)} w_{pq} d_{f(p)f(q)}$$

❑Total cost decomposes into "label cost" terms & "separation cost" terms:
$c_{p,a}$ = label cost for assigning label $a = f(p)$ to object p
$w_{pq} d_{ab}$ = separation cost for assigning pair of labels $a = f(p), b = f(q)$ to neighbors p,q

❑ The **edge weights** w_{pq} measure the strength of p and q relationship while the **distance function** d_{ab} measures the dissimilarity between labels a,b (here we DO NOT assume that this distance is a metric)

❑The ML problem generalizes pairwise Markov Random Fields (MRFs):
Label costs replace MRF single-node potentials
Distance function d_{ab} replaces MRF pairwise potentials

❑It therefore suffices to design our approximation algorithms for this problem

The primal-dual schema

❑Primal-dual schema is a powerful tool for deriving approximation algorithms based on duality theory.

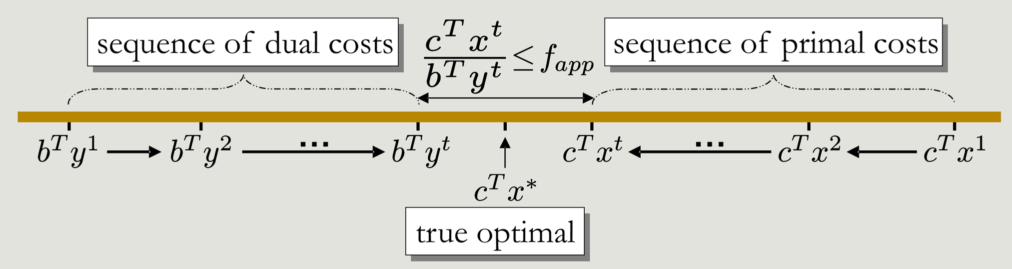
❑Given as input a particular pair of primal and dual Linear Programming (LP) problems:

$$\text{PRIMAL:} \quad \min c^T x \quad \text{s.t. } Ax = b, x \geq 0 \qquad \text{DUAL:} \quad \max b^T y \quad \text{s.t. } A^T y \leq c$$

we then seek an optimal integral solution x^* to the primal (NP-complete problem)

❑The **primal-dual schema** tries to find an approximately optimal solution as follows:

PRIMAL DUAL SCHEMA: Keep generating pairs $\{(x^k, y^k)\}_{k=1}^t$ of feasible integral-primal and dual solutions until the primal & dual costs of the last pair are "close enough": $c^T x^t \leq f_{app} \cdot b^T y^t$
Then x is guaranteed to be an f_{app}-approximation to the true optimal solution x^*.



❑In practice an easy way to check that the costs of a primal-dual pair x, y are "close enough" is through the so-called **complementary slackness** conditions:

$$\forall x_j > 0 \Rightarrow \sum_{i=1}^m a_{ij} y_i \geq c_j / f_j \quad \text{with} \quad \max f_j = f_{app}$$

(It is very easy to show that these conditions do imply the inequality $c^T x \leq f_{app} \cdot b^T y$) So our objective will be to find x,y satisfying one such set of slackness conditions.

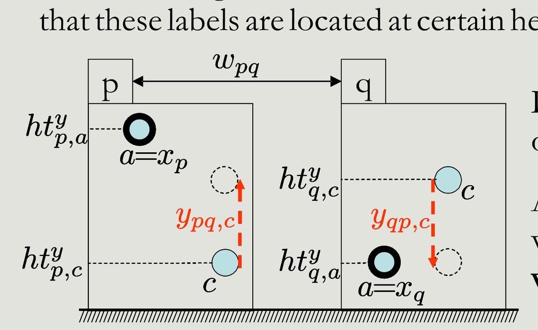
❑Simply by choosing different sets of slackness conditions (i.e. different f_j), then different approximation algorithms can be derived!

Primal & dual of Metric Labeling

❑In order to apply the primal-dual schema, the ML problem has been expressed as a linear program (see paper for derivation)

❑The primal variables $\{x_p\}_{\in V}$ represent the **active labels** (i.e. the labels assigned to objects). In particular, x_p denotes the label assigned to object p.

❑For visualizing the dual variables we can think that each object holds a copy of all labels and that these labels are located at certain heights.



Dual variable $ht^y_{p,c}$ denotes the height of label c at object p (these are the **height variables**).

Also, labels can move up or down through the dual variables $y_{pq,c}, y_{qp,c}$ (these are called the **balance variables**).

However each time a label c at p goes up by $y_{pq,c}$ then label c at neighbor q must go down by $y_{qp,c} = -y_{pq,c}$ (For this reason variables $y_{pq,c}, y_{qp,c}$ are called **conjugate** to each other)
Initially the height of a label is set equal to its label cost i.e. $ht^y_{p,a} = c_{p,a}$

The PD1 algorithm

I. Complementary slackness conditions

PD1 uses a particular set of complementary slackness conditions in which $f_{app} = 2 \frac{max_{a \neq b} d_{ab}}{min_{a \neq b} d_{ab}}$
In particular it tries to find a primal-dual pair (x,y) that satisfies the following set of slackness conditions (see paper for derivation):

(a) $ht^y_{p,x_p} = \min_a ht^y_{p,a}$ → **active label x_p must be lower than all labels at p**

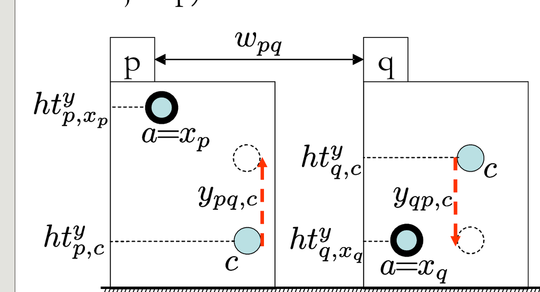
(b) $y_{pq,a} \leq w_{pq} d_{min}/2, \forall a$ → balance variables are upper bounded (so we are not allowed to raise labels as much as we like)

(c) $y_{pq,x_p} + y_{qp,x_q} \geq w_{pq} d_{x_p x_q} / f_{app}$ → the sum of raises of active labels x_p, x_q at neighbors p,q should not be too small

II. Updating the primal & dual variables

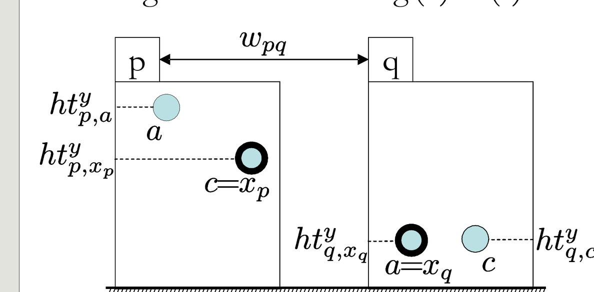
PD1 ensures that slackness conditions (b) and (c) always hold true (which is easy) and drives (x,y) towards satisfying (a) by alternating between updates of dual & primal variables:

DUAL VARIABLES UPDATE: Given the current active labels (i.e. the current primal), try to raise all non-active labels at p above the active label at p (for each object p).



However, this may not be possible for all p. E.g. in the above example we cannot raise label c at p any further than the red arrow or else label c at q will go below the active label x_q

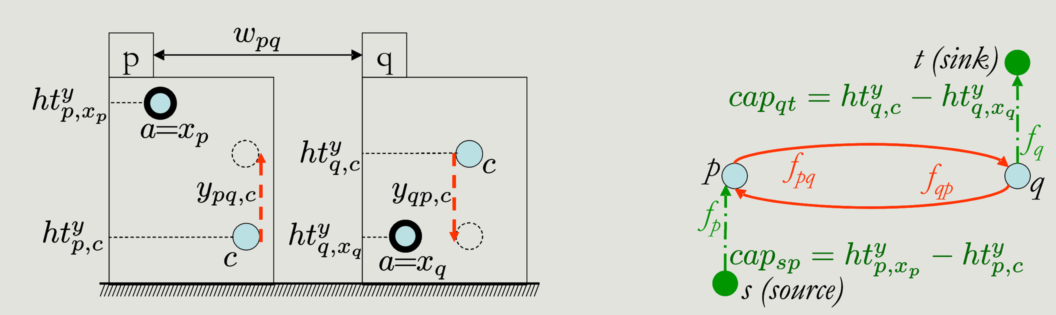
PRIMAL VARIABLES UPDATE: Given the new heights (i.e. the new current dual), if the active label of p is not the lowest one at p then assign to p a new active label of lower height without violating (b) or (c).



Try to do that for as many p as possible. E.g. in the above example the new active label assigned to p is label c because that label is still below the previous active label of p i.e. label *a*

III. Simulation via max-flow

❑The above update of the dual variables can be simulated by pushing flow through a graph as follows (here y^{k+1} denotes the dual variable resulting after the update while $y^k = y$ denotes the dual variable before the update):



If at an object p label c needs to go up (i.e. it is below the active label x_p) then we connect p to the source node s and flow f_p through sp will represent the total raise of label c at p.	$ht^{y^{k+1}}_{p,c} = ht^{y^k}_{p,c} + f_p$
If at an object q label c may go down (i.e. it is above the active label x_q) then we connect q to the sink node t and flow f_q through qt will represent the total decrease in the height of label c at q.	$ht^{y^{k+1}}_{q,c} = ht^{y^k}_{q,c} - f_q$
Flows f_{pq}, f_{qp} represent the increase/decrease of $y_{pq,c}$ balance var	$y^{k+1}_{pq,c} = y^k_{pq,c} + f_{pq} - f_{qp}$

❑Optimal update occurs at the maximum flow. One can then update the primal variables (i.e. check which labels did not manage to go above the active labels) using the following simple rule:

REASSIGN RULE: Label c will be the new label of node p (i.e. $x^{k+1}_p = c$) ⇔ There is an unsaturated path between the source node s and p.

The PD2 & PD3 algorithms

❑PD1 made use of just one particular set of complementary slackness conditions.

❑Simply by choosing a different set of slackness conditions and restricting distance d_{ab} to be a metric **an algorithm equivalent to the a-expansion graph-cut method is derived!** This is the PD2 algorithm.

❑Moreover, thanks to the power of our framework, PD2 can be extended (in many ways) so that it can handle non-metric distances d_{ab} as well, leading to algorithms $PD3_a$, $PD3_b$ and $PD3_c$ which still have guaranteed optimality properties.

❑Intuitively these algorithms obtain a nearly optimal solution by dynamically approximating the non-metric distance d_{ab} with a "metric" $\bar{d}_{ab}$. The "metric" $\bar{d}_{ab}$ adapts itself at every primal-dual update so that the complementary slackness conditions finally hold true.

❑Thanks to our framework many theorems can be proved about these algorithms, thus offering new insights into graph-cut methods.

Experimental results

I. Stereo Matching

Tsukuba left image — Estimated disparity — SRI tree left image — Disparity

II. Stereo Matching with occlusions detection

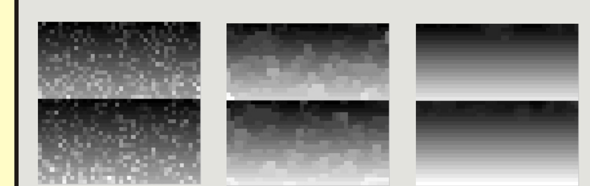
'map' left image — disparity & occlusions

For detecting occlusions a non-metric distance between labels (i.e. disparities) has been used which penalizes disparity pairs that violate the uniqueness constraint.

III. Image restoration & completion

Noisy input image — Restored image — Noisy input image with missing pixels — Restored & completed image

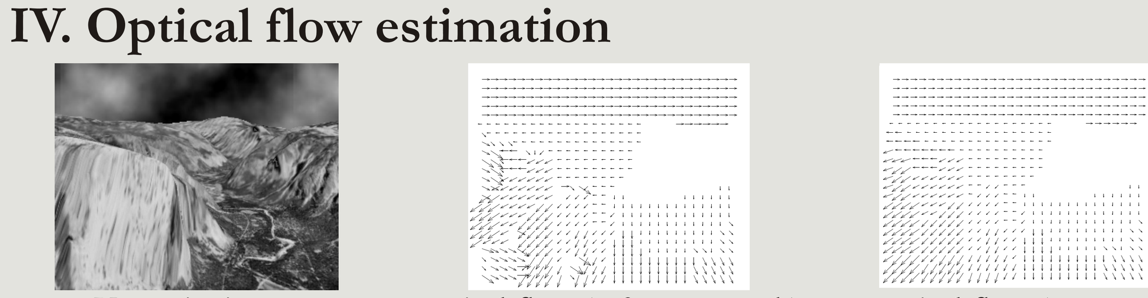
The truncated quadratic semimetric has been used as distance between labels in this case. Moreover, in the case of missing pixels the label costs of those pixels have been set to zero.



Noisy input image — Metric restoration — Semimetric restoration

For this example, a semimetric distance that assigns a small penalty to smoothly varying intensity patterns & a higher penalty to large intensity discontinuities has been applied.
semimetric case: 8.2% pixels with errors
metric case: 41.4% pixels with errors

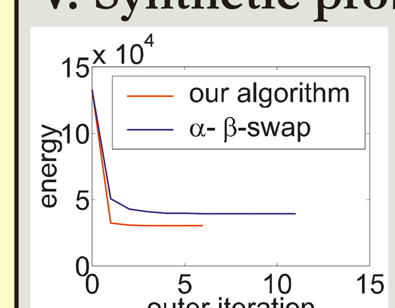
IV. Optical flow estimation



Yosemite image — optical flow (α-β-swap result) avg. angular error: 14.73° — optical flow (our result) avg. angular error: 6.97°

A non-metric distance that penalizes abrupt changes in the direction of neighboring optical flow vectors has been used. Both our algorithm & α-β-swap minimized exactly the same objective function, yet α-β-swap solution had 19.2% higher energy.

V. Synthetic problems



Both our algorithm and α-β-swap have been applied to synthetic problems with random label distances and random label costs (180 labels were used and the objects were on a 30 × 30 grid).
On average the α-β-swap energy was higher by 28% while also α-β-swap needed more iterations to converge (as can be observed in the plot on the left).

VI. Per-instance suboptimality bounds

❑**An advantage of any primal-dual algorithm is that it always tells you (for free) how well it performed!**

❑This is thanks to its ability to also provide (for free) per-instance suboptimality bounds using the primal-dual pairs $\{(x^k, y^k)\}_{k=1}^t$ that has already generated during its execution.
In particular, any ratio $r_k = c^T x^k / b^T y^k$ of primal and dual costs makes up such a bound.

❑In practice these bounds prove to be very tight (i.e. very close to 1) for our algorithms, **meaning that their solutions are almost optimal!**
E.g. the ratios for the tsukuba stereo pair (and various d_{ab} distances) are displayed below: