Combinatorial Optimization in Computer Vision (IN2245)

Frank R. Schmidt
Csaba Domokos

Winter Semester 2015/2016

Multi-label problem 3 / 43

Multi-label problem revisited

Consider an *undirected graphical model* given by $G = (\mathcal{V}, \mathcal{E})$ which takes values from an **arbitrary** (finite) label set $\mathcal{L}$.

More specially, assume that the corresponding *energy function* is given by

$$E(\mathbf{x}) = \sum_{i \in \mathcal{V}} \varphi_i(\mathbf{x}_i) + \sum_{(i,j) \in \mathcal{E}} w_{ij} \cdot d(\mathbf{x}_i, \mathbf{x}_j) \ ,$$

where φ_i stands for the *data term*, $w_{ij} \in \mathbb{R}$ are *weighting factors*, and d is a *metric* or a *semi-metric* (i.e. the triangle inequality is not necessary satisfied).

We have already seen some applications in Computer Vision corresponding to this energy function (e.g., stereo matching, image denoising, optical flow).

As we have discussed (in Lecture 13) one possible way to *approximately* solve this problem is to apply *move making algorithms* (e.g., α-expansion).

Equivalent integer linear program

We are generally interested to find a *MAP labelling* $\mathbf{x}^*$:

$$\mathbf{x}^* \in \underset{\mathbf{x}\in\mathcal{L}^{|\mathcal{V}|}}{\operatorname{argmin}} E(\mathbf{x}) = \underset{\mathbf{x}\in\mathcal{L}^{|\mathcal{V}|}}{\operatorname{argmin}} \left\{ \sum_{i\in\mathcal{V}} \varphi_i(x_i) + \sum_{(i,j)\in\mathcal{E}} w_{ij} \cdot d(x_i, x_j) \right\} .$$

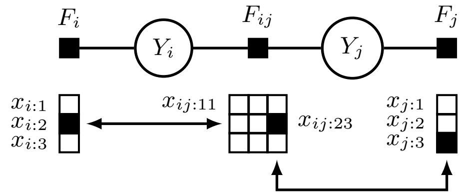
This can be equivalently written as an **integer linear program** (ILP):

$$\begin{aligned}
\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \sum_{i\in\mathcal{V}} \sum_{\alpha\in\mathcal{L}} \varphi_i(a) x_{i:\alpha} &+ \sum_{(i,j)\in\mathcal{E}} w_{ij} \sum_{\alpha,\beta\in\mathcal{L}} d(\alpha,\beta) x_{ij:\alpha\beta} \\
\text{subject to} \quad \textstyle\sum_{\alpha\in\mathcal{L}} x_{i:\alpha} &= 1 \qquad \forall i \in \mathcal{V} \\
\textstyle\sum_{\alpha\in\mathcal{L}} x_{ij:\alpha,\beta} &= x_{j:\beta} \qquad \forall \beta \in \mathcal{L}, (i,j) \in \mathcal{E} \\
\textstyle\sum_{\beta\in\mathcal{L}} x_{ij:\alpha,\beta} &= x_{i:\alpha} \qquad \forall \alpha \in \mathcal{L}, (i,j) \in \mathcal{E} \\
x_{i:\alpha}, x_{ij:\alpha,\beta} &\in \mathbb{B} \qquad \forall \alpha, \beta \in \mathcal{L}, (i,j) \in \mathcal{E}
\end{aligned}$$

$x_{i:\alpha}$ indicates whether vertex i is assigned label α, while $x_{ij:\alpha\beta}$ indicates whether (neighboring) vertices i, j are assigned labels α, β, respectively.

Interpretation of the constraints

Let us assume that $\mathcal{L} = \{1, 2, 3\}$ and consider the following example:



Uniqueness: The constraint $\sum_{\alpha \in \mathcal{L}} x_{i:\alpha} = 1$ simply express the fact that each vertex must receive exactly one label.

Consistency: The constraints $\sum_{\alpha \in \mathcal{L}} x_{ij:\alpha\beta} = x_{j:\beta}$ and $\sum_{\beta \in \mathcal{L}} x_{ij:\alpha\beta} = x_{i:\alpha}$ maintain consistency between variables, i.e. if $x_{i:\alpha} = 1$ and $x_{j:\beta} = 1$ holds true, then these constraints force $x_{ij:\alpha\beta} = 1$ to hold true as well.

FastPD algorithm vs. α-expansion

The *FastPD algorithm* is a max-flow based combinatorial method which is suitable for *approximate optimization* of a very wide class of MRFs.

It utilizes tools from the **duality theory of linear programming** in order to provide a **more general view of move making techniques**.

This algorithm solves similar problems as the α-expansion (which is included merely as a special case), but it has some advantages:

- **It is more general**: It can be applied for a much wider class of problems, e.g., MRFs with non-metric potentials.
- **It is more efficient**: It is guaranteed that the generated solution will always be within a known factor of the global optimum. In practice, these bounds prove to be very tight (i.e. very close to 1).
- **It is conceptually more elegant**.

Primal-dual LP revisited

Consider a linear program (given in **standard form**):

$$\min_{\mathbf{x}\in\mathbb{R}^n} \langle \mathbf{c}, \mathbf{x} \rangle$$
$$\text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \,,$$

for a *constraint matrix* $\mathbf{A} \in \mathbb{R}^{m\times n}$, a *constraint vector* $\mathbf{b} \in \mathbb{R}^m$ and a *cost vector* $\mathbf{c} \in \mathbb{R}^n$.

The *dual LP* is defined as

$$\max_{\mathbf{y}\in\mathbb{R}^m} \langle \mathbf{b}, \mathbf{y} \rangle$$
$$\text{subject to } \mathbf{A}^T\mathbf{y} \leqslant \mathbf{c} \,.$$

Due to weak duality $\langle \mathbf{b}, \mathbf{y} \rangle \leqslant \langle \mathbf{c}, \mathbf{x} \rangle$ is held for feasible solutions.

For more details you may refer to Lecture 7.

LP relaxation

The ILP defined before is in general NP-hard. Therefore we deal with the **LP relaxation** of our ILP. The relaxed LP can be written in *standard form* as follows:

$$\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \langle \mathbf{c}, \mathbf{x} \rangle$$
$$\text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \, .$$

Reminder: The **lexicographical order relation $\prec$ on $\mathbb{N}^k$** is defined as

$$(u_1, \dots, u_k) \prec (v_1, \dots, v_k) \quad \Leftrightarrow \quad \exists l : \; \forall i < l \; (u_i = v_i) \text{ and } (u_l < v_l) \, .$$

Reminder: Assume $\mathbf{A} \in \mathbb{R}^{k \times l}$ and $\mathbf{B} \in \mathbb{R}^{m \times n}$, then the **Kronecker product** $\mathbf{A} \otimes \mathbf{B}$ is the $km \times ln$ block matrix: $\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1l}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{k1}\mathbf{B} & \cdots & a_{kl}\mathbf{B} \end{bmatrix}$.

LP relaxation: cost function

$$\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \langle \mathbf{c}, \mathbf{x} \rangle \qquad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \; .$$

We may write $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1^T & \mathbf{x}_2^T \end{bmatrix}^T$, where

$$\mathbf{x}_1 = \begin{bmatrix} x_{1:1} & \cdots & x_{1:m} & x_{2:1} & \cdots & x_{2:m} & x_{n:1} & \cdots & x_{n:m} \end{bmatrix}^T \in \mathbb{R}^{mn} \; ,$$

where $n = |\mathcal{V}|$ and $m = |\mathcal{L}|$, and $\mathbf{x}_2 \in \mathbb{R}^{|\mathcal{E}|m^2}$ is the vector consisting of all the variables $x_{ij:\alpha\beta}$ in *lexicographic order* based on the corresponding 4-tuples (i, j, α, β).

Similarly, we can write $\mathbf{c} = \begin{bmatrix} \mathbf{c}_1^T & \mathbf{c}_2^T \end{bmatrix}^T$, where

$$\mathbf{c}_1 = \begin{bmatrix} \varphi_1(1) & \cdots & \varphi_1(m) & \cdots & \varphi_n(1) & \cdots & \varphi_n(m) \end{bmatrix}^T \in \mathbb{R}^{mn} \; ,$$

and $\mathbf{c}_2 \in \mathbb{R}^{|\mathcal{E}|m^2}$ is the vector consisting of the values $w_{ij} d(\alpha, \beta)$ in *lexicographic order* based on the corresponding 4-tuples (i, j, α, β).

Therefore, $\langle \mathbf{c}, \mathbf{x} \rangle = \langle \mathbf{c}_1, \mathbf{x}_1 \rangle + \langle \mathbf{c}_2, \mathbf{x}_2 \rangle$.

LP relaxation: constraints

$$\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \langle \mathbf{c}, \mathbf{x} \rangle \qquad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \; .$$

We can write the (uniqueness) constraints $\sum_{\alpha \in \mathcal{L}} x_{i:\alpha} = 1$ for all $p \in \mathcal{V}$ as

$$\left[\mathbf{I}_{n \times n} \otimes \mathbf{1}_m^T \right] \mathbf{x}_1 = \mathbf{1}_n =: \mathbf{b}_1 \; ,$$

where $\mathbf{1}_n \in \mathbb{R}^n$ is the vector of all-ones.

We introduce the notation $\pi_{\mathcal{E}}(i, j)$ for the **index of an element** $(i, j) \in \mathcal{E}$ according to the lexicographic order $\prec$ on $\mathcal{E}$, that is

$$\pi_{\mathcal{E}}(i, j) \triangleq |\{(k, l) \in \mathcal{E} \mid (k, l) \prec (i, j)\}| \; .$$

LP relaxation: constraints

$$\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \langle \mathbf{c}, \mathbf{x} \rangle \qquad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \ .$$

The (consistency) constraint $\sum_{\alpha \in \mathcal{L}} x_{ij:\alpha\beta} = x_{j:\beta} \;\Leftrightarrow\; -x_{j:\beta} + \sum_{\alpha \in \mathcal{L}} x_{ij:\alpha\beta} = 0$ can be expressed as

$$\begin{bmatrix} -\mathbf{u}^T_{(j-1)m+\beta} & \sum_{\alpha \in \mathcal{L}} \mathbf{v}^T_{m^2 \pi_\varepsilon(i,j)+(\alpha-1)m+\beta} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = 0 \ ,$$

where $\mathbf{u}_k \in \mathbb{R}^{mn}$ and $\mathbf{v}_k \in \mathbb{R}^{|\mathcal{E}|m^2}$ are k^{th} **standard unit vectors** whose k^{th} component is equal to one and all the other elements are equal to zero.

One can collect **all** the *consisteny constraints* as follows

$$\begin{bmatrix} -\mathbf{U} \mid \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \mathbf{0}_{2|\mathcal{E}|m} =: \mathbf{b}_2 \ ,$$

where $\mathbf{U} \in \mathbb{R}^{2|\mathcal{E}|m \times mn}$ and $\mathbf{V} \in \mathbb{R}^{2|\mathcal{E}|m \times |\mathcal{E}|m^2}$.

LP relaxation: constraints

$$\min_{x_{i:\alpha}, x_{ij:\alpha\beta}} \langle \mathbf{c}, \mathbf{x} \rangle \qquad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0} \ .$$

We can write all the constraints in a matrix-vector notation as follows.

$$\mathbf{A}\mathbf{x} = \left[\begin{array}{c|c} \mathbf{I}_{n \times n} \otimes \mathbf{1}^T_m & \mathbf{0}_{n \times |\mathcal{E}|m^2} \\ \hline -\mathbf{U} & \mathbf{V} \end{array}\right] \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_{2|\mathcal{E}|m} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix} = \mathbf{b} \ .$$

Hence, $\mathbf{A} \in \mathbb{R}^{n+2|\mathcal{E}|m \times mn+|\mathcal{E}|m^2}$ is a **sparse matrix** with elements -1,0 and 1, furthermore $\mathbf{b} \in \mathbb{R}^{n+2|\mathcal{E}|m}$, where the first mn elements are equal to one and the others are equal to zero.

Column consistency: We assume that the first $|\mathcal{E}|m$ rows of $\mathbf{U}$ and $\mathbf{V}$ correspond to the constraints $\sum_{\alpha \in \mathcal{L}} x_{ij:\alpha\beta} = x_{j:\beta}$ enumerated in *lexicographic order* based on (i, j, β).
Row consistency: the second half of the rows in $\mathbf{U}$ and $\mathbf{V}$ correspond to the constraints $\sum_{\beta \in \mathcal{L}} x_{ij:\alpha\beta} = x_{i:\alpha}$ enumerated in lexicographic based on (i, j, α).

Dual LP

$$\max_{y_i, y_{ij:\alpha}, y_{ji:\beta}} \langle \mathbf{b}, \mathbf{y} \rangle \qquad \text{subject to } \mathbf{A}^T\mathbf{y} \leqslant \mathbf{c} \; .$$

Note that the dual variables y_i for all $i \in \mathcal{V}$ and $y_{ij:\alpha}$, $y_{ji:\beta}$ for all $(i,j) \in \mathcal{E}$, $\alpha, \beta \in \mathcal{L}$ correspond to the constraints of the primal LP.

We can write $\mathbf{y} = \begin{bmatrix} \mathbf{y}_1^T & \mathbf{y}_2^T & \mathbf{y}_3^T \end{bmatrix}^T$, where $\mathbf{y}_1 = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix}^T \in \mathbb{R}^n$, and $\mathbf{y}_2 \in \mathbb{R}^{|\mathcal{E}|m}$ and $\mathbf{y}_3 \in \mathbb{R}^{|\mathcal{E}|m}$ are the vectors consisting of the variables $y_{ji:\beta}$ and $y_{ij:\alpha}$ in the same order as it is defined in the case of the primal LP.

The cost function results in

$$\langle \mathbf{b}, \mathbf{y} \rangle = \langle \mathbf{b}_1, \mathbf{y}_1 \rangle + \langle \mathbf{b}_2, \begin{bmatrix} \mathbf{y}_2^T & \mathbf{y}_3^T \end{bmatrix}^T \rangle = \langle \mathbf{1}_n, \mathbf{y}_1 \rangle = \sum_{i=1}^{n} y_i \; .$$

The constraints $\mathbf{A}^T\mathbf{y} \leqslant \mathbf{c}$ are given by

$$\mathbf{A}^T\mathbf{y} = \left[\begin{array}{c|c} \mathbf{I}_{n\times n} \otimes \mathbf{1}_m & -\mathbf{U}^T \\ \hline \mathbf{0}_{|\mathcal{E}|m^2\times n} & \mathbf{V}^T \end{array}\right] \mathbf{y} \leqslant \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = \mathbf{c} \; .$$

Dual LP

$$\max_{y_i, y_{ij:\alpha}, y_{ji:\beta}} \langle \mathbf{1}_n, \mathbf{y}_1 \rangle$$
$$\text{subject to } \left[\begin{array}{c|c} \mathbf{I}_{n\times n} \otimes \mathbf{1}_m & -\mathbf{U}^T \\ \hline \mathbf{0}_{|\mathcal{E}|m^2\times n} & \mathbf{V}^T \end{array}\right] \mathbf{y} \leqslant \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} \; .$$

Or equivalently, we can formulate the dual LP as

$$\begin{aligned} &\max_{y_i, y_{ij:\alpha}, y_{ji:\beta}} \sum_{i\in\mathcal{V}} y_i \\ &\text{subject to} \quad y_i - \sum_{j\in\mathcal{V}:(i,j)\in\mathcal{E}} y_{ij:\alpha} \leqslant \varphi_i(\alpha) \qquad \forall i \in \mathcal{V}, \alpha \in \mathcal{L} \\ &\qquad\qquad\quad y_{ij:\alpha} + y_{ji:\beta} \leqslant w_{ij} d(\alpha, \beta) \quad \forall (i,j) \in \mathcal{E}, \alpha, \beta \in \mathcal{L} \end{aligned}$$

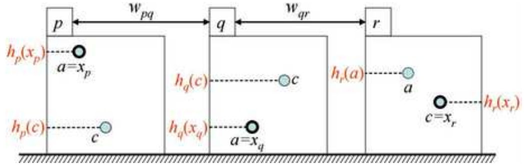
An intuitive view of the dual variables

We will use the notation $x_i \in \mathcal{L}$ for the **active label** given the vertex $i \in \mathcal{V}$.

For each vertex we have a different copy of all labels in $\mathcal{L}$. It is assumed that all these labels represent **balls** floating at certain heights relative to a *reference plane*.

For this sake we introduce **height variables** defined as

$$h_i(\alpha) = \varphi_i(\alpha) + \sum_{j \in \mathcal{V}, (i,j) \in \mathcal{E}} y_{ij:\alpha} \ .$$



The constraints $y_i - \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:\alpha} \leqslant \varphi_i(\alpha)$ can be equivalently written as

$$y_i \leqslant \varphi_i(\alpha) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:\alpha} = h_i(\alpha) \qquad \forall i \in \mathcal{V}, \alpha \in \mathcal{L} \ .$$

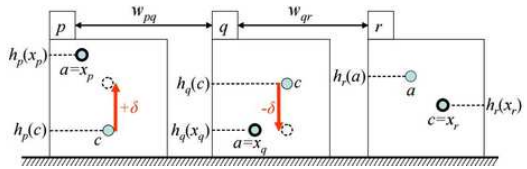
Since our objective is to maximize $\sum_{i \in \mathcal{V}} y_i$, the following relation holds

$$y_i = \min_{\alpha \in \mathcal{L}} h_i(\alpha) \qquad \forall i \in \mathcal{V} \ .$$

Balance variables and load

We will refer to the variables $y_{ij:\alpha}$, $y_{ji:\beta}$ as **balance variables**. Specially, the pair of $y_{ij:\alpha}$, $y_{ji:\alpha}$ is called **conjugate balance variables**.

The *balls* are not static, but may move in pairs through updating pairs of *conjugate balance variables* as $h_i(\alpha) = \varphi_i(\alpha) + \sum_{j \in \mathcal{V}, (i,j) \in \mathcal{E}} y_{ij:\alpha}$. Therefore, the role of *balance variables* is to raise or lower labels.



It is due to $y_{ij:\alpha} + y_{ji:\alpha} \leqslant w_{ij} d(\alpha, \alpha) = 0 \quad \Rightarrow \quad y_{ij:\alpha} \leqslant -y_{ji:\alpha}$.

We will call the variables $y_{ij:x_i}$ as **active balance variable** and use the following notation for the **"load"** between neighbors i, j, defined as

$$\text{load}_{ij} = y_{ij:x_i} + y_{ji:x_j} \ .$$

Primal-dual LP for multi-label problem

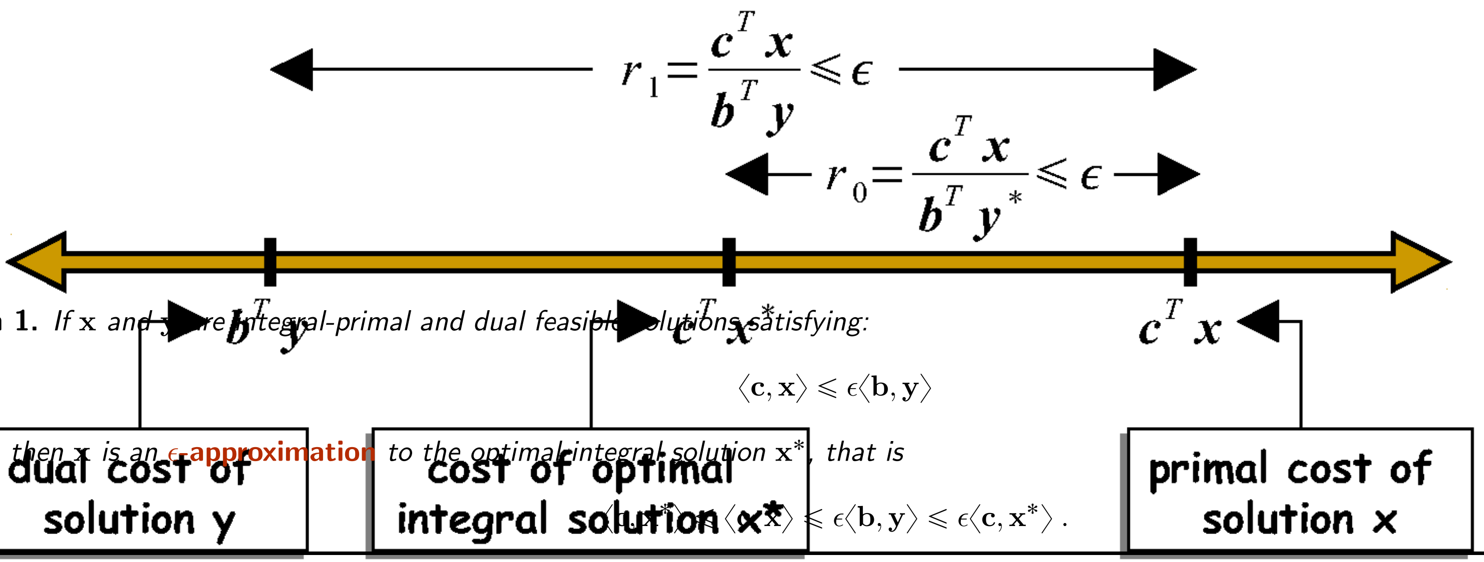
The (relaxed) primal LP:

$$\begin{aligned} \min_{x_{i:\alpha}, x_{ij:\alpha\beta} \geqslant 0} & \sum_{i \in \mathcal{V}} \sum_{\alpha \in \mathcal{L}} \varphi_i(a) x_{i:\alpha} + \sum_{(i,j) \in \mathcal{E}} w_{ij} \sum_{\alpha,\beta \in \mathcal{L}} d(\alpha,\beta) x_{ij:\alpha\beta} \\ \text{subject to} \quad & \textstyle\sum_{\alpha \in \mathcal{L}} x_{i:\alpha} = 1 \qquad \forall i \in \mathcal{V} \\ & \textstyle\sum_{\alpha \in \mathcal{L}} x_{ij:\alpha\beta} = x_{j:\beta} \quad \forall \beta \in \mathcal{L}, (i,j) \in \mathcal{E} \\ & \textstyle\sum_{\beta \in \mathcal{L}} x_{ij:\alpha\beta} = x_{i:\alpha} \quad \forall \alpha \in \mathcal{L}, (i,j) \in \mathcal{E} \end{aligned}$$

The dual LP:

$$\begin{aligned} \max_{y_i, y_{ij:\alpha}, y_{ji:\beta}} & \sum_{i \in \mathcal{V}} y_i \\ \text{subject to} \quad & y_i - \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:\alpha} \leqslant \varphi_i(\alpha) \qquad \forall i \in \mathcal{V}, \alpha \in \mathcal{L} \\ & y_{ij:\alpha} + y_{ji:\beta} \leqslant w_{ij} d(\alpha,\beta) \quad \forall (i,j) \in \mathcal{E}, \alpha, \beta \in \mathcal{L} \end{aligned}$$

Primal-dual principle



Theorem 1. *If* $\mathbf{x}$ *and* $\mathbf{y}$ *are integral-primal and dual feasible solutions satisfying:*

$$\langle \mathbf{c}, \mathbf{x} \rangle \leqslant \epsilon \langle \mathbf{b}, \mathbf{y} \rangle$$

for $\epsilon \geqslant 1$, *then* $\mathbf{x}$ *is an* **ϵ-approximation** *to the optimal integral solution* $\mathbf{x}^*$, *that is*

$$\langle \mathbf{c}, \mathbf{x}^* \rangle \leqslant \langle \mathbf{c}, \mathbf{x} \rangle \leqslant \epsilon \langle \mathbf{b}, \mathbf{y} \rangle \leqslant \epsilon \langle \mathbf{c}, \mathbf{x}^* \rangle \,.$$

The relaxed complementary slackness

One way to estimate a pair $(\mathbf{x}, \mathbf{y})$ satisfying the fundamental inequality $\langle \mathbf{c}, \mathbf{x} \rangle \leqslant \epsilon \langle \mathbf{b}, \mathbf{y} \rangle$ relies the **complementary slackness principle**.

Theorem 2. *If the pair $(\mathbf{x}, \mathbf{y})$ of integral-primal and dual feasible solutions satisfies the so-called* **relaxed primal complementary slackness conditions**:

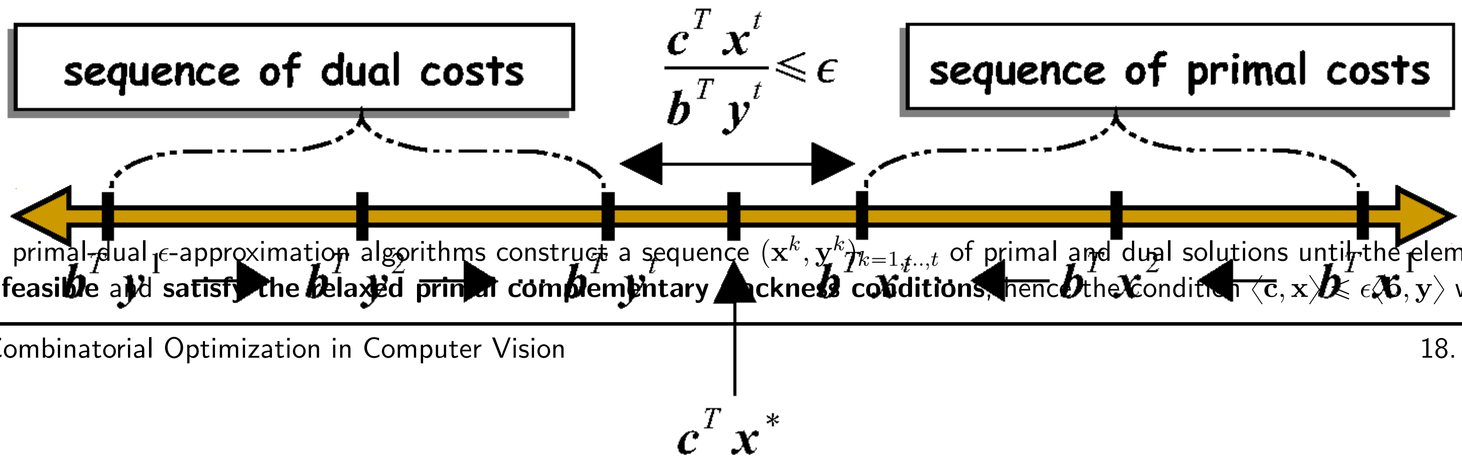
$$\forall j: \ (x_j > 0) \quad \Rightarrow \quad \left(\sum_i a_{ij} y_i \geqslant \frac{c_j}{\epsilon_j} \right) ,$$

then $(\mathbf{x}, \mathbf{y})$ also satisfies $\langle \mathbf{c}, \mathbf{x} \rangle \leqslant \epsilon \langle \mathbf{b}, \mathbf{y} \rangle$ with $\epsilon = \max_j \epsilon_j$ and therefore $\mathbf{x}$ is an ϵ-approximation to the optimal integral solution $\mathbf{x}^*$.

Proof. Exercise. □

Primal-dual schema

Primal-dual schema



Typically, primal-dual ϵ-approximation algorithms construct a sequence $(\mathbf{x}^k, \mathbf{y}^k)_{k=1,\ldots,t}$ of primal and dual solutions until the elements $\mathbf{x}^t$, $\mathbf{y}^t$ of the last pair are both **feasible** and **satisfy the relaxed primal complementary slackness conditions**, hence the condition $\langle \mathbf{c}, \mathbf{x} \rangle \leqslant \epsilon \langle \mathbf{b}, \mathbf{y} \rangle$ will be also fulfilled.

Pseudo-code of the FastPD algorithm

```
1:  [x, y] ← Init_Primals_Duals()
2:  labelChange ← false
3:  for all α ∈ ℒ do                                        ▷ α-iteration
4:      y ← PreEdit_Duals(α, x, y)
5:      [x′, y′] ← Update_Duals_Primals(α, x, y)
6:      y′ ← PostEdit_Duals(α, x′, y′)
7:      if x′ ≠ x then
8:          labelChange ← true
9:      end if
10:     x ← x′; y ← y′
11: end for
12: if labelChange then
13:     goto 2
14: end if
15: y^fit ← Dual_Fit(y)
```

Complementary slackness conditions

From now on, in case of Algorithm `PD1`, we only assume that $d(\alpha, \beta) = 0 \Leftrightarrow \alpha = \beta$, and $d(\alpha, \beta) \geqslant 0$.

The *complementary slackness conditions* reduces to

$$y_i - \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:x_i} \geqslant \frac{\varphi_i(x_i)}{\epsilon_1} \quad \Rightarrow \quad y_i \geqslant \frac{\varphi_i(x_i)}{\epsilon_1} + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:x_i}$$

$$y_{ij:x_i} + y_{ji:x_j} \geqslant \frac{w_{ij} d(x_i, x_j)}{\epsilon_2}$$

for specific values of $\epsilon_1, \epsilon_2 \geqslant 1$.

If $x_i = x_j = \alpha$ for neighboring i, j, then

$$0 = w_{ij:\alpha} d(\alpha, \alpha) \geqslant y_{ij:i\alpha} + y_{ij:j\alpha} \geqslant \frac{w_{ij} d(\alpha, \alpha)}{\epsilon_2} = 0 ,$$

therefore we get that $y_{ij:\alpha} = -y_{ij:\alpha}$.

Complementary slackness conditions

We have known that $y_i = \min_{\alpha \in \mathcal{L}} h_i(\alpha)$. If $\epsilon_1 = 1$, then we get

$$y_i \geqslant \varphi_i(x_i) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:x_i} = h_i(x_i) \ .$$

Therefore

$$h_i(x_i) = \min_{\alpha \in \mathcal{L}} h_i(\alpha) \ , \tag{1}$$

which means that, at each vertex, **the active label should have the lowest height**.

If $\epsilon_2 = \epsilon_{\mathsf{app}} := \frac{2d_{\max}}{d_{\min}}$, then the *complementary condition* simply reduces to:

$$y_{ij:x_i} + y_{ij:x_j} \geqslant \frac{w_{ij} d(x_i, x_j)}{\epsilon_{\mathsf{app}}} \ . \tag{2}$$

It requires that any **two active labels should be raised proportionally to their "load"**.

Feasibility constraints

To ensure feasibility of $\mathbf{y}$, `PD1` enforces for any $\alpha \in \mathcal{L}$:

$$y_{ij:\alpha} \leqslant w_{ij} d_{\min}/2 \quad \text{where} \quad d_{\min} = \min_{\alpha \neq \beta} d(\alpha, \beta) \tag{3}$$

says that **there is an upper bound on how much we can raise a label**.

Hence, we get the feasibility condition

$$y_{ij:\alpha} + y_{ji:\beta} \leqslant 2 w_{ij} d_{\min}/2 = w_{ij} d_{\min} \leqslant w_{ij} d(\alpha, \beta) \ .$$

Moreover the algorithm **keeps the active balance variables non-negative**, that is $y_{ij:x_i} \geqslant 0$ for all $i \in \mathcal{V}$.

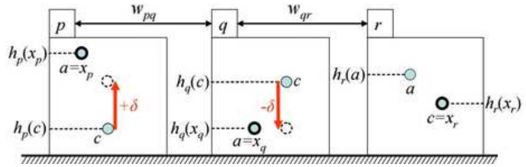
The *proportionality condition* (2) will be also fulfilled as $y_{ij:x_i}, y_{ij:x_j} \geqslant 0$ and

$$y_{ij:x_i} \geqslant \frac{w_{ij} d_{\min}}{2} \geqslant \frac{w_{ij} d_{\min}}{2} \frac{d(x_i, x_j)}{d_{\max}} = \frac{w_{ij} d(x_i, x_j)}{\frac{2 d_{\max}}{d_{\min}}} = \frac{w_{ij} d(x_i, x_j)}{\epsilon_{\mathsf{app}}} \ .$$

Subroutine Init_Primals_Duals()

function $\textsc{Init\_Primals\_Duals}$
  $\mathbf{x}$ is simply initialized by a random label assignment ▷ Init primals
  for all $(i, j) \in \mathcal{E}$ with $x_i \neq x_j$ **do** ▷ Init duals
    $y_{ij:x_i} \leftarrow w_{ij} d(x_i, x_j)/2$
    $y_{ji:x_i} \leftarrow -w_{ij} d(x_i, x_j)/2$
    $y_{ji:x_j} \leftarrow w_{ij} d(x_i, x_j)/2$
    $y_{ij:x_j} \leftarrow -w_{ij} d(x_i, x_j)/2$
  end for
  for all $i \in \mathcal{V}$ **do**
    $y_i \leftarrow \min_{\alpha \in \mathcal{L}} h_i(\alpha)$
  end for
  return $[\mathbf{x}, \mathbf{y}]$
end function

Update primal and dual variables



Dual variables update: Given the current active labels, any non-active label is raised, until it either reaches the active label, or attains the maximum raise allowed by the upper bound (3).

Primal variables update: Given the new heights, there might still be vertices whose active labels are not at the lowest height. For each such vertex i, we select a non-active label, which is below x_i, but has already reached the maximum raise allowed by the upper bound (3).

The optimal update of the α-heights can be simulated by pushing the **maximum amount of flow** through a directed graph $G = (\mathcal{V} \cup \{s, t\}, \mathcal{E}', \mathcal{C})$.

Flow construction: n-edges

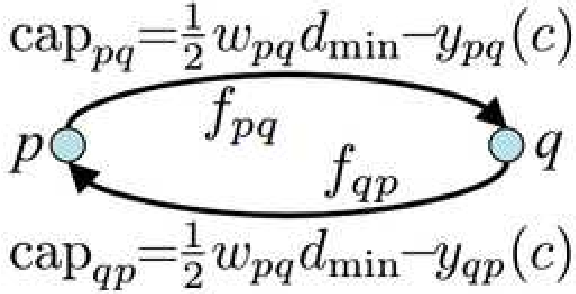
For each $(i, j) \in \mathcal{E}$, we insert two directed edges ij and ji into $\mathcal{E}'$.

The flow value f_{ij}, f_{ij} represent respectively the **increase, decrease of balance variable** $y_{pq:\alpha}$:

$$y'_{ij:\alpha} = y_{ij:\alpha} + f_{ij} - f_{ji} \quad \text{and} \quad y'_{ji:\alpha} = -y'_{ij:\alpha} \, .$$

According to (3), the capacities cap_{ij} and cap_{ji} are set based on

$$\mathsf{cap}_{ij} + y_{ij:\alpha} = \frac{1}{2} w_{ij} d_{\min} = \mathsf{cap}_{ji} + y_{ji:\alpha} \, .$$

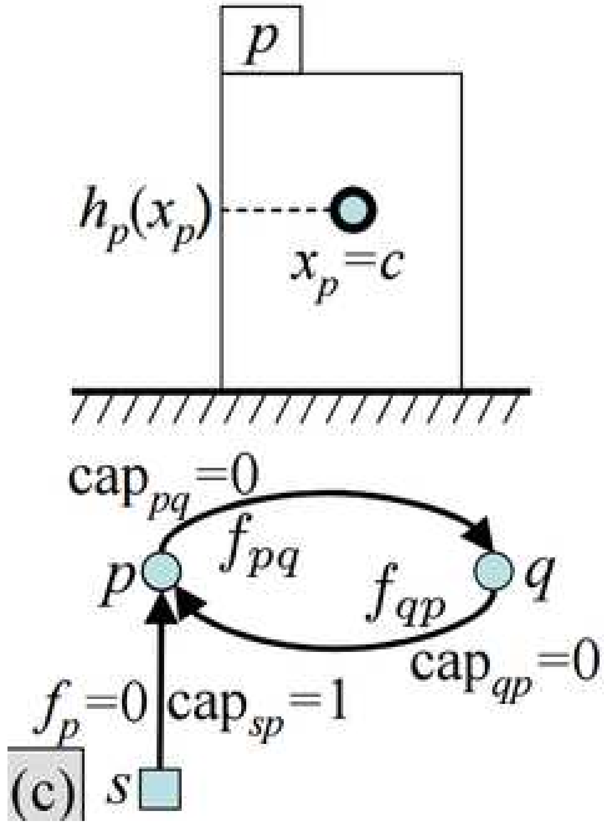


Flow construction: n-edges

If α is already the active label of i (or j), then label α at i (or j) need not move.

Therefore, $y'_{ij:\alpha} = y_{ij:\alpha}$ and $y'_{ji:\alpha} = y_{ji:\alpha}$, that is

$$x_i = \alpha \text{ or } x_j = \alpha \quad \Rightarrow \quad \text{cap}_{ij} = \text{cap}_{ji} = 0 \; .$$



Flow construction: t-edges

Each node $i \in \mathcal{V}' - \{s, t\}$ connects to either the source node s or the sink node t (but not to both of them).
There are three possible cases to consider:

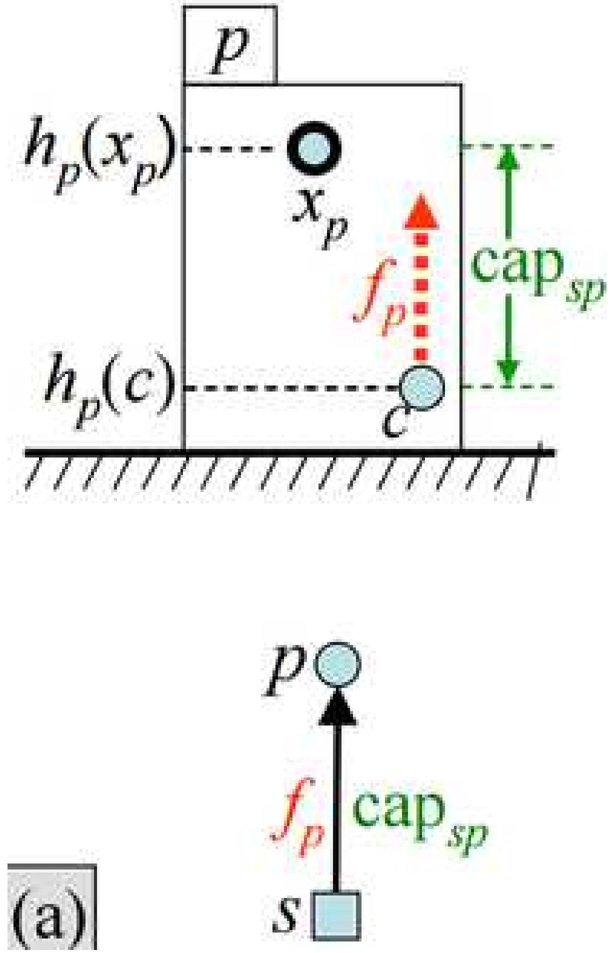
Case 1 ($h_i(\alpha) < h_i(x_i)$): we want to raise label α as much as it reaches label x_i. We connect source node s to node i.
Due to the flow conservation property, $f_i = \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} (f_{ij} - f_{ji})$.
The flow f_i through that edge will then represent the total relative raise of label α:

$$
\begin{aligned}
h_i(\alpha) + f_i &= \Big(\varphi_i(\alpha) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:\alpha}\Big) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} (f_{ij} - f_{ji}) \\
&= \Big(\varphi_i(\alpha) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y_{ij:\alpha}\Big) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} \big(y'_{ij:\alpha} - y_{ji:\alpha}\big) \\
&= \varphi_p(\alpha) + \sum_{j \in \mathcal{V}:(i,j) \in \mathcal{E}} y'_{ij:\alpha} = h'_i(\alpha)\ .
\end{aligned}
$$

Flow construction: t-edges

We need to raise α only as high as the current active label of i, but not higher than that, we therefore set:

$$\mathsf{cap}_{ij} = h_i(x_i) - h_i(\alpha) \; .$$



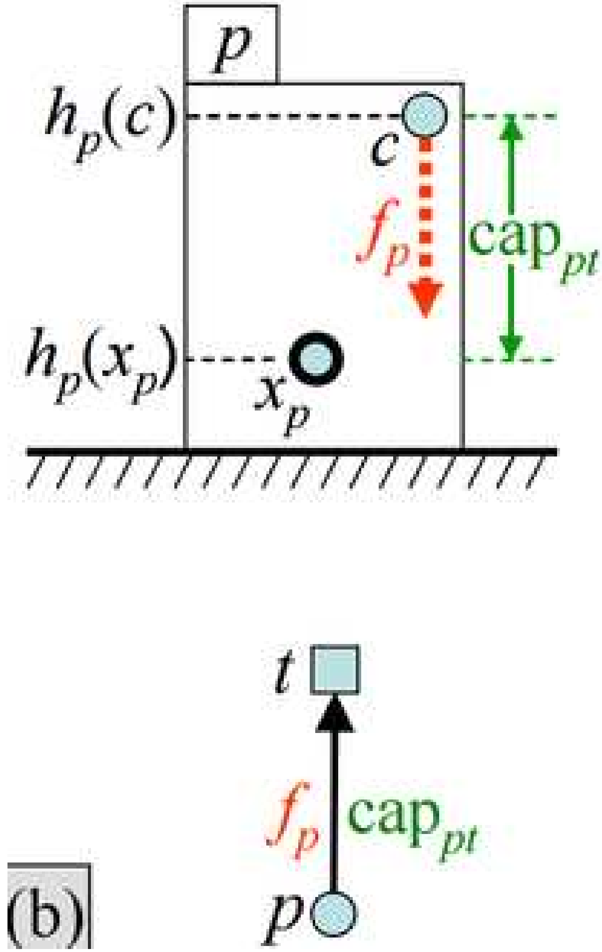
Flow construction: t-edges

Case 2 ($h_i(\alpha) \geqslant h_i(x_i)$ and $c \neq x_i$): we can then afford a decrease in the height of α at i, as long as α remains above x_p.

We connect i to the sink node t through directed edge it.

The flow f_i through edge it will equal the total relative decrease in the height of α:

$$
\begin{aligned}
h'_i(\alpha) &= h_i(\alpha) - f_i \\
\mathsf{cap}_{it} &= h_p(\alpha) - h_i(x_i) \;.
\end{aligned}
$$

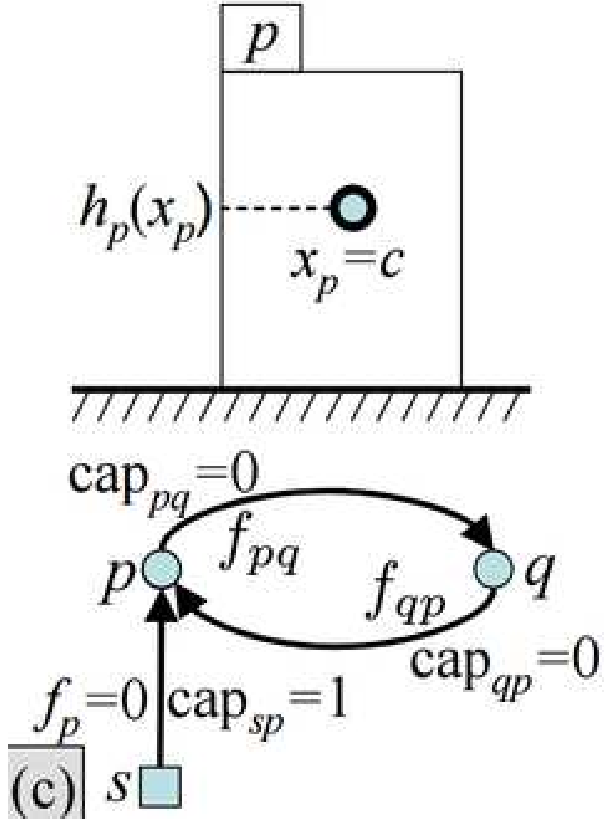


Flow construction: t-edges

Case 3 ($\alpha = x_i$): we want to keep the height of α fixed at the current iteration.

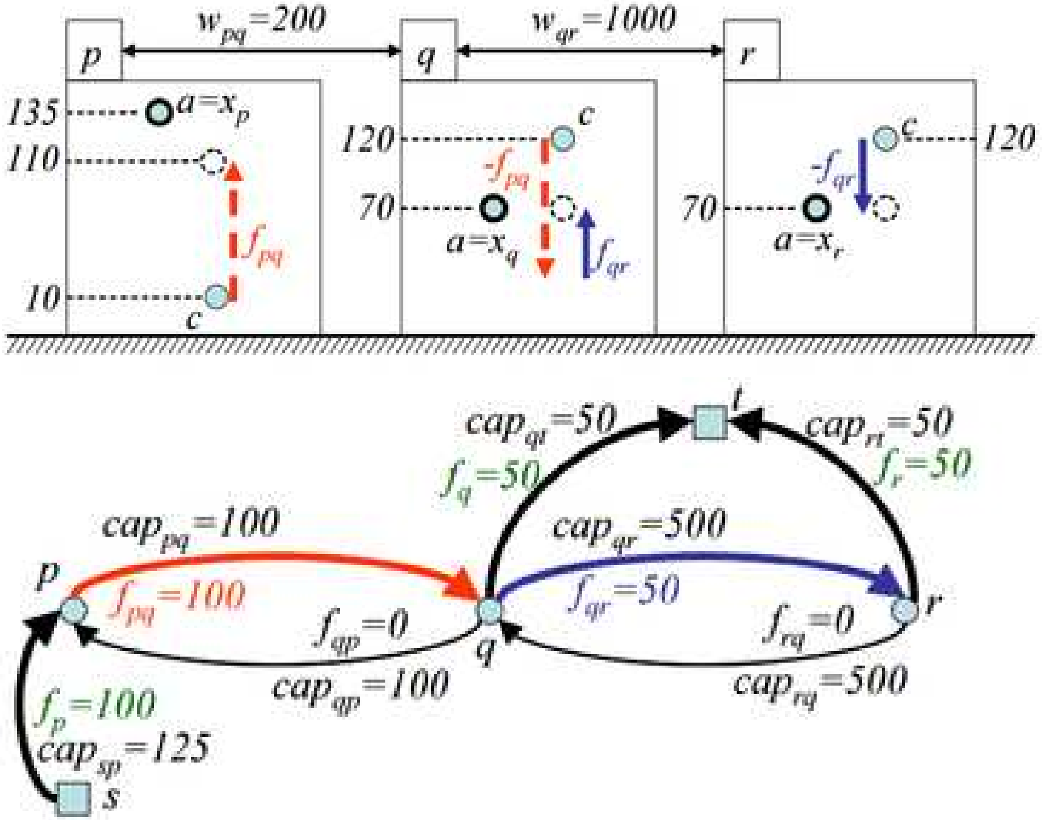
Note that the capacities of the *n-edges* for p are set to 0, since i has the *active label*. Therefore, $f_i = 0$ and $h'_{ij:\alpha} = h_{ij:\alpha}$.

By convention $\text{cap}_{ij} := 1$.



Reassign rule

Label α will be the new label of i (i.e. $x'_i = \alpha$) iff there exists unsaturated path between the source node s and node i. In all other cases, i keeps its current label (i.e. $x'_i = x_i$).



Some properties

Based on the reassign rule the following three properties hold:

- $h'_i(x'_i)) = \min\{h'_i(x_i), h'(\alpha)\}$
- $x'_i = \alpha \neq x'_j \;\Rightarrow\; y'_{ij:x'_i} = \mathsf{cap}_{ij} + y_{ij:\alpha}$
- $\mathsf{APF}^{\mathbf{x}',\mathbf{y}'} < \mathsf{APF}^{\mathbf{x},\mathbf{y}}$, where $\mathsf{APF}^{\mathbf{x},\mathbf{y}}$ is defined as

$$\begin{aligned}
\mathsf{APF}^{\mathbf{x},\mathbf{y}} &\triangleq \sum_{i\in\mathcal{V}} h_i(x_i) = \sum_{i\in\mathcal{V}} \Big(\varphi_i(x_i) + \sum_{j\in\mathcal{V},(i,j)\in\mathcal{E}} y_{ij:x_i}\Big) \\
&= \sum_{i\in\mathcal{V}} \Big(\varphi_i(x_i) + \sum_{(i,j)\in\mathcal{E}} (y_{ij:x_i} + y_{ji:x_j})\Big) \\
&\leqslant \sum_{i\in\mathcal{V}} \varphi_i(x_)) + \sum_{(i,j)\in\mathcal{E}} w_{ij} d(x_i, x_j) = E(\mathbf{x}) \; .
\end{aligned}$$

The last condition shows that the algorithm terminates (assuming integer capacities), due to the reassign rule, which ensures that a new active label has always lower height than the previous active label, i.e. $h'_i(x'_i) \leqslant h_i(x_i)$.

Subroutine Update_Duals_Primals(α,$\mathbf{x}$,$\mathbf{y}$)

function UPDATE_DUALS_PRIMALS(α,$\mathbf{x}$,$\mathbf{y}$)
  $\mathbf{x}' \leftarrow \mathbf{x}$, $\mathbf{y}' \leftarrow \mathbf{y}$
  Apply max-flow to G' and compute flows f_i, f_{ij}
  for all $(i, j) \in \mathcal{E}$ **do**
    $y'_{ij:\alpha} \leftarrow y_{ij:\alpha} + f_{ij} - f_{ji}$
  end for
  for all $i \in \mathcal{V}$ **do**
    $x_i \leftarrow \alpha \;\Leftrightarrow \exists$ unsaturated path $s \rightsquigarrow i$ in G'
  end for
  return $[\mathbf{x}', \mathbf{y}']$
end function

Subroutine PostEdit_Duals(α,$\mathbf{x}'$,$\mathbf{y}'$)

The goal is to restore all *active balance variables* $y_{ij:x_i}$ to be non-negative.

1. $x'_i = \alpha \neq x'_j$: we have $\text{cap}_{ij}, y_{ij:\alpha} \geqslant 0$, therefore $y'_{ij:\alpha} = \text{cap}_{ij} + y_{ij:\alpha} \geqslant 0$.
2. $x'_i = x'_j = \alpha$: we have $y'_{ij:\alpha} = -y'_{ji:\alpha}$, therefore $\text{load}'_{ij} = y'_{ij:\alpha} + y'_{ji:\alpha} = 0$. By setting $y'_{ij}(\alpha) = y'_{ji:\alpha} = 0$ we get $\text{load}'_{ij} = 0$ as well.

Since none of the "load" were altered, the $\text{APF}^{\mathbf{x},\mathbf{y}}$ remains unchanged.

function POSTEDIT_DUALS(α,$\mathbf{x}'$,$\mathbf{y}'$)
  for all $(i, j) \in \mathcal{E}$ with $(x'_i = x'_j = \alpha)$ and $(y'_{ij:\alpha} < 0$ or $y'_{ji:\alpha} < 0)$ **do**
    $y'_{ij:\alpha} \leftarrow 0$, $y'_{ji:\alpha} \leftarrow 0$
  end for
  for all $i \in \mathcal{V}$ **do**
    $y'_i \leftarrow \min_{\alpha \in \mathcal{L}} h'_i(\alpha)$
  end for
  return $\mathbf{y}'$
end function

ϵ_{app}-approximate solution

In summary, one can see that PD1 always leads to an ϵ-approximate solution:

Theorem 3. *The final primal-dual solutions generated by* PD1 *satisfy*

1. $h_i(x_i) = \min_{\alpha \in \mathcal{L}} h_i(\alpha)$ *for all* $i \in \mathcal{V}$,
2. $x_i \neq x_j \Rightarrow \textit{load}_{ij} \geqslant \frac{w_{ij} d(x_p, x_q)}{\epsilon_{app}}$ *for all* $(i, j \in \mathcal{E})$,
3. $y_{ij:\alpha} \leqslant \frac{w_{ij} d_{\min}}{2}$ *for all* $(i, j \in \mathcal{E})$ *and* $\alpha \in \mathcal{L}$,

and thus they satisfy the relaxed complementary slackness conditions with $\epsilon_1 = 1$, $\epsilon_2 = \epsilon_{app} = \frac{2d_{\max}}{d_{\min}}$.

Literature

- Nikos Komodakis and Georgios Tziritas. **Approximate Labeling via Graph-Cuts Based on Linear Programming**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29(8), pp. 1436–1453, August, 2007.
- Nikos Komodakis and Georgios Tziritas. **Approximate Labeling via the Primal-Dual Schema**. Technical Report, University of Crete, February 2005.