TRANSFORMING AN ARBITRARY MINSUM PROBLEM INTO A BINARY ONE

DMITRIJ SCHLESINGER, BORIS FLACH

ABSTRACT. In this report we show, that an arbitrary MinSum problem (i.e. a MinSum problem with an arbitrary finite set of states) can be adequately transformed into a binary one (i.e. into a MinSum problem with only two states). Consequently all known results for binary MinSum problems can be easily extended to the general case. For instance it gives the possibility to solve exactly submodular MinSum problems with more than two states by using MinCut-MaxFlow based technics.

CONTENTS

1. INTRODUCTION

In this report we deal with labeling problems – a field attracting growing attention in pattern recognition. We regard labeling problems as a kind of theoretical frame, which covers many well known problems from different fields. Some examples are: Constraint Satisfaction problems, formal languages and grammars, energy minimization problems, Markov random fields etc.

Here we discuss in detail only certain special cases. For example, all definitions, formulations, explanations are given only for labeling problems of second order. The main result is considered in detail only for MinSum problems etc. It is however easy to see, that most of these results can be immediately generalized. We will give remarks at corresponding places.

The main result presented in this report consists in the following. We will show, that an arbitrary MinSum problem (i.e. a MinSum problem with an arbitrary finite set of states) can be adequately transformed into a binary one (i.e. into a MinSum problem with only two states). We call this transformation "K to 2". It follows, that *all* results for binary MinSum problems obtained before can be straightforward generalized to the case of more than two states. As an example we will give a scheme for constructing a MinCut problem for an

arbitrary MinSum task with many states. If the initial MinSum task (of order not greater than three) is in addition submodular, the constructed MinCut problem can be solved in polynomial time using MaxFlow algorithms.

2. Notations and definitions

2.1. **Labeling problems (of second order).** To begin with, we cite a general definition of labeling problems as given in [16].
A labeling problem is defined by means of the following components:

$V=(R,E)$	an unoriented graph (we call it "base graph of the task") with the set of nodes R and the set of edges $E \subset R \times R$. Nodes and edges are denoted by $r \in R$ and $e=(r,r') \in E$ respectively;
K	a finite set of states, we call an element "state" and denote it by k;
$(W,\oplus,\otimes)$	a semiring with a set of elements W and operations $\oplus$ and $\otimes$;
$q_r : K \to W$	a function for each node r of the base graph, that assigns a value from W to each state in that node;
$g_{rr'} : K \times K \to W$	a function for each edge $(r,r') \in E$ of the base graph, that assigns a value from W to each pair of states on that edge.

Let $f : R \to K$ be a mapping (labeling), that maps the set of nodes into the set of states, $f(r)$ denotes the chosen state in the node r. The set of all labelings is denoted by $\mathscr{F}=K^R$. The task is to calculate the quantity

$$G = \bigoplus_{f\in\mathscr{F}} \left[\bigotimes_{r\in R} q_r\big(f(r)\big) \;\otimes \bigotimes_{(rr')\in E} g_{rr'}\big(f(r),f(r')\big) \right]. \tag{1}$$

We are mainly interested in the following special cases (with respect to the semiring):
OrAnd: $(\{0,1\},\vee,\wedge)$. Such problems can be formulated verbally as follows. It is necessary to answer, whether there exists at least one labeling, which satisfies all given constraints – i.e. so called Constraint Satisfaction problems. Functions q and g are boolean constraints. The task is therefore to compute

$$G = \bigvee_{f\in\mathscr{F}} \left[\bigwedge_{r\in R} q_r\big(f(r)\big) \;\wedge \bigwedge_{(rr')\in E} g_{rr'}\big(f(r),f(r')\big) \right]. \tag{2}$$

OrAnd problems are in a certain sense the simplest case. Studying this case can be however very helpfull for a better understanding of more complex problems such as e.g. MinSum or SumProd. Another reason is, that it is sometimes necessary to solve (or at least to consider) certain OrAnd tasks as subtasks in the context of MinSum problems.
MinMax: $(\mathbb{R},\min,\max)$[1]. MinMax problems really do not differ from OrAnd due to the fact of the "equivalence" of used operations. They can be seen as a fuzzy variant of Constraint Satisfaction problems.
MinSum: $(\mathbb{R},\min,+)$. It can be hardly overseen, that MinSum problems are very popular at the moment (they are mainly known as "Energy minimization tasks"). The task is an optimization problem of type

$$G = \min_{f\in\mathscr{F}} \left[\sum_{r\in R} q_r\big(f(r)\big) \;+ \sum_{(rr')\in E} g_{rr'}\big(f(r),f(r')\big) \right]. \tag{3}$$

[1] $\mathbb{R}$ denotes the set of real numbers enlarged by $\pm\infty$.

Many practical applications can be (in principle) formulated as optimization problems of such kind. Beside of this, solutions of corresponding MinSum tasks can be often used as a powerful preprocessing stage in more complex recognition schemes.

SumProd: $(\mathbb{R},+,\times)$. MinSum problems (3) as considered above, can be interpreted as maximum a-posteriori decisions for coresponding probabilistic models, based on Markov random fields. That is, MinSum problems are in fact a special case of more general Bayes decision tasks. For many practical applications they are obviously not the best choice, because the maximum a-posteriori decision follows from a very primitive cost function. When using more appropriate cost functions instead, it is very often necessary to calculate certain marginal a-posteriori probabilities, that is to solve a SumProd problem of type

$$G = \sum_{f \in \mathscr{F}} \left[\prod_{r \in R} q_r\big(f(r)\big) \cdot \prod_{(rr') \in E} g_{rr'}\big(f(r), f(r')\big) \right]. \tag{4}$$

In the following we will concentrate mainly on MinSum problems. We describe such a task (let us denote it by $\mathscr{A}$) by means of its components, i.e. $\mathscr{A} = (V, K, q, g)$ where

$V = (R, E)$	is the base graph,
K	is the set of states,
$q : R \times K \to \mathbb{R}$	are the state qualities,
(or $q_r : K \to \mathbb{R}$, $\forall r \in R$)	
$g : E \times K \times K \to \mathbb{R}$	are the qualities of state pairs.
(or $g_{rr'} : K \times K \to \mathbb{R}$, $\forall (r, r') \in E$)	

2.2. **Submodularity (for MinSum problems of second order).** Sub-/supermodular functions are well known in mathematics for a very long time. The story begins in 1781, when a special class of matrices was introduced by G.Monge [17]. A detailed overview of their properties can be found e.g. in [4]. At the moment there are many researchers dealing with sub-/supermodular optimization problems. In the field of image processing the first promising result was obtained in [8], where the authors realized, that some binary MinSum problems (i.e. MinSum problems with only two states) can be formulated as exact solvable MinCut problems. They used this approach for restoration of binary images. A theoretical frame for using sub-/supermodular functions in terms of labeling problems (without the restriction to be a binary one) was firstly introduced in [23]. More detailed considerations can be found in [5, 24]. We would like to note especially the paper [14], where some important results were obtained for binary MinSum problems.

In all these works the considered property is denoted differently – regular functions, monoton-interval functions, subconvex functions etc. We prefer to denote it as "submodular functions" to keep traditions. Let us consider the definition of this property.

Let the set of states K be a completely ordered set. It means, that for each pair of states k_1 and k_2 a relation "above/below" is defined, i.e. $k_2 \succeq k_1$ means "the state k_2 is located above the state k_1 (or coincides with k_1)". Let k_1 and k_2 be two states in a node r, so that $k_2 \succeq k_1$. Similar let k'_1 and k'_2 be two states in another node r', so that $k'_2 \succeq k'_1$. A function $g_{rr'}$ is called submodular if

$$g_{rr'}(k_1, k'_1) + g_{rr'}(k_2, k'_2) \leq g_{rr'}(k_1, k'_2) + g_{rr'}(k_2, k'_1) \tag{5}$$

holds for each such four-tuple $k_2 \succeq k_1$, $k'_2 \succeq k'_1$. A MinSum task is called submodular if all functions $g_{rr'}$, $\forall (r, r') \in E$ are submodular. Functions q_r can be thereby arbitrary.

This definition can be written in a little bit different form. Let $k_2 = k_1 + 1$, i.e. "the state k_2 is situated above the state k_1 and it is the lowest one among all such states (besides of

k_1 itself)". The previous definition is then equivalent to the following one. A function $g_{rr'}$ is submodular if

$$g_{rr'}(k,k') + g_{rr'}(k{+}1,k'{+}1) \leq g_{rr'}(k,k'{+}1) + g_{rr'}(k{+}1,k') \tag{6}$$

holds for each $k,k' \in K$ (despite the "highest" states, which have no successor $k{+}1$). Obviously (6) follows from (5) directly. It is easy to see, that the reverse statement holds as well, i.e. (5) follows from (6). Let us consider the numbers

$$\alpha(k_1,k_2,k'_1,k'_2) = g_{rr'}(k_1,k'_1) + g_{rr'}(k_2,k'_2) - g_{rr'}(k_1,k'_2) - g_{rr'}(k_2,k'_1)$$

for each four-tuple of states $k_2 \succeq k_1$, $k'_2 \succeq k'_1$. These numbers fulfill

$$\alpha(k_1,k_2,k'_1,k'_2) = \sum_{k_3=k_1}^{k_2-1} \sum_{k'_3=k'_1}^{k'_2-1} \alpha(k_3,k_3{+}1,k'_3,k'_3{+}1) \tag{7}$$

for each such four-tuple. If a function $g_{rr'}$ is submodular according to the second definition (6), all addends in above formula are not positive, that is the function $g_{rr'}$ is submodular according to the first definition (5) as well.

Obviously the notation of submodularity relates to the order, introduced for the set of states. A task, which is submodular with respect to a particular order, is in general not submodular with respect to another order. Consequently it makes no sense to speak about submodularity itself. On the other hand, any finite set can be always ordered (its elements can be arbitrary enumerated). Therefore in the following we will allways assume, that the set of states is ordered, i.e. it can be represented by a subset of integer numbers – $K = \{1,2\ldots|K|\}$. Considering a task we will say "the task is submodular" if it is submodular with respect to the introduced order.

Remark 1. The definitions (5) and (6) are equivalent only if there are no infinite qualities g in the task. Otherwise (6) is a necessary condition only.

Remark 2. Submodularity can be defined in a similar way for other semirings by simply substituting operations $\oplus$ and $\otimes$. Obviously, it is possible only if the relation "grater/less" is defined on the set of elements W of the considered semiring. Note, that submodular OrAnd tasks are exactly solvable by Relaxation Labeling algorithm [19]. For details we refer to [5, 23].

Remark 3. It is easy to see, that the class of functions g given in [9] is a special case of submodular functions.

2.3. **Equivalent transformations (for MinSum problems of second order).** Let us consider two tasks $\mathscr{A} = (V,K,q,g)$ and $\mathscr{A}' = (V,K,q',g')$ which have the same base graph and the same set of states, but different functions q and g. Let f be a labeling. Its qualities in the tasks $\mathscr{A}$ and $\mathscr{A}'$ are

$$G(f) = \sum_{r\in R} q_r\big(f(r)\big) \;+\; \sum_{(r,r')\in E} g_{rr'}\big(f(r),f(r')\big)$$

and

$$G'(f) = \sum_{r\in R} q'_r\big(f(r)\big) \;+\; \sum_{(r,r')\in E} g'_{rr'}\big(f(r),f(r')\big)$$

respectively. Such two tasks are called equivalent, if $G(f) = G'(f)$ holds for each labeling $f \in \mathscr{F}$ [22]. All equivalent tasks form an equivalence class (let us denote it by $\mathscr{E}(\mathscr{A})$ – the set of all tasks which are equivalent to $\mathscr{A}$).

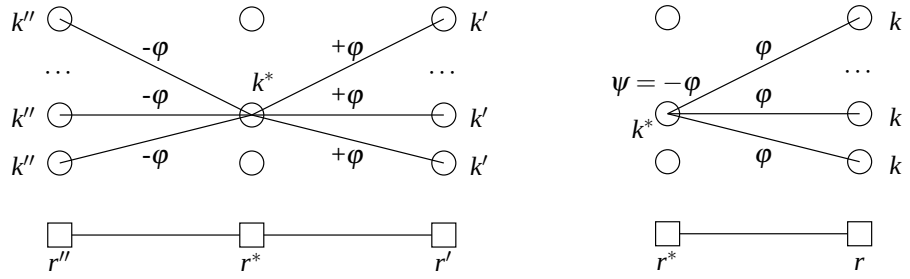

FIGURE 1. Equivalent transformations.

Starting from a given task $\mathscr{A}$ an equivalent task $\mathscr{A}'$ can be built using e.g. the transformation shown in fig.1.a. In the figure nodes are marked by squares and states by circles. Let us consider a node and a state in that node – a pair (r^*,k^*). Further let us consider two edges of the base graph, which are incident to r^*, e.g. (r^*,r') and (r^*,r''). If all function values $g_{r^*r'}(k^*,k')$, $k' \in K$ are increased by some quantity φ and at the same time all function values $g_{r^*r''}(k^*,k'')$, $k'' \in K$ are decreased by the same quantity, no labeling will change its quality. The quantities $\varphi_{rr'}(k)$ are called "potentials" and this transformation is called "equivalent transformation" of a task. Obviously potentials should satisfy certain constraints to describe an equivalent transformation. For instance in the example considered above $\varphi_{r^*r'}(k^*) = -\varphi_{r^*r''}(k^*)$ should be satisfied.

Equivalent transformations for state qualities $q_r(k)$ can be defined in a similar way. Let us consider a pair (r^*,k^*) as well as an edge (r^*,r), which is incident to this node. If the quality of the state k^* is increased by the same value (let us denote it by $\psi_{r^*}(k^*)$) as the qualities of corresponding pairs of states $g_{r^*r}(k^*,k)$, $k \in K$ are decreased, no labeling will change its quality (see fig.1.b). As in the previous example $\psi_{r^*}(k^*) = -\varphi_{r^*r}(k^*)$ should hold for this one.

In general an equivalent transformation is defined as a tuple of potentials $\Phi = \big(\varphi_{rr'}(k), \psi_r(k)\big)$, so that

$$\psi_r(k) + \sum_{r':(r,r')\in E} \varphi_{rr'}(k) = 0, \quad \forall r \in R,\ k \in K \tag{8}$$

holds. By applying such a transformation to a task $\mathscr{A} = (V,K,q,g)$, a new task $\mathscr{A}' = (V,K,q',g')$ is constructed, which is equivalent to $\mathscr{A}$. The new values of functions q' and g' are given by

$$\begin{aligned} q'_r(k) &= q_r(k) + \psi_r(k) \\ g'_{rr'}(k,k') &= g_{rr'}(k,k') + \varphi_{rr'}(k) + \varphi_{r'r}(k'). \end{aligned} \tag{9}$$

It is easy to see, that conditions (8) guarantee the equivalence of both tasks.

Without going deep into details we would like to present some important properties of equivalent transformations. Let Φ_1 and Φ_2 be two equivalent transformations. Applying them consecutively is equivalent to applying one equivalent transformation Φ_3, where $\varphi_{3rr'}(k) = \varphi_{1rr'}(k) + \varphi_{2rr'}(k)$ and $\psi_{3r}(k) = \psi_{1r}(k) + \psi_{2r}(k)$. We call transformation Φ_3 a "superposition" of two transformations and denote it as $\Phi_3 = \Phi_1 \circ \Phi_2$. It follows immediately, that the set of all equivalent transformations forms a cummutative group with respect to the operation $\circ$. A further important property of equivalent transformations is the following one. It can be proved, that the set of all equivalent transformations "completely

describes" an equivalence class in the following sense:

$$\mathscr{A}' \in \mathscr{E}(\mathscr{A}) \;\Leftrightarrow\; \exists \Phi : \mathscr{A}' = \Phi(\mathscr{A}). \tag{10}$$

Another characteristic of equivalent transformations is the fact, that these transformations preserve the submodularity of a task. To show this, let us compair the left and the right sides of submodularity conditions (5) after applying an equivalent transformation:

$$\begin{aligned}
&g'_{rr'}(k_1,k'_1)+g'_{rr'}(k_2,k'_2) \lesseqgtr g'_{rr'}(k_1,k'_2)+g'_{rr'}(k_2,k'_1),\\
&\big[g_{rr'}(k_1,k'_1)+\varphi_{rr'}(k_1)+\varphi_{r'r}(k'_1)\big]+\big[g_{rr'}(k_2,k'_2)+\varphi_{rr'}(k_2)+\varphi_{r'r}(k'_2)\big] \lesseqgtr\\
&\big[g'_{rr'}(k_1,k'_2)+\varphi_{rr'}(k_1)+\varphi_{r'r}(k'_2)\big]+\big[g'_{rr'}(k_2,k'_1)+\varphi_{rr'}(k_2)+\varphi_{r'r}(k'_1)\big],\\
&g_{rr'}(k_1,k'_1)+g_{rr'}(k_2,k'_2) \lesseqgtr g_{rr'}(k_1,k'_2)+g_{rr'}(k_2,k'_1),
\end{aligned}$$

i.e. the signs of the conditions remain unchanged. This means, that either both tasks $\mathscr{A}$ and $\mathscr{A}'$ are submodular (all inequalities are satisfied with $\leq$) or both tasks are not submodular (there exists at least one inequality, which is satisfied with $>$). Due to the fact, that the set of all equivalent transformations completely describes an equivalence class, the following holds. An arbitrary equivalence class consists either only of submodular tasks or only of not submodular tasks.

We also introduce another transformation, which we will need later. If all values of a function $g_{rr'}$ are changed simultaneously by some quantity, i.e. $g'_{rr'}(k,k') = g_{rr'}(k,k') + c_{rr'}$, $\forall k,k' \in K$, then the qualities of all labelings are changed by the same quantity. Therefore the location of the best labeling (arg min) remains unchanged. The same holds also for changing of q-s, i.e. $q'_r(k) = q_r(k) + c_r$, $\forall k \in K$. We call such transformations "uniform transformations" of a task and denote them as $U = (c_r, c_{rr'})$. Applying an uniform transformation changes the qualities of all labelings by the value $\sum_r c_r + \sum_{rr'} c_{rr'}$. The set of all uniform transformations also forms a group (the operation $\circ$ is in that case the componentwise summation). It is easy to see, that uniform transformations preserve submodularity of a task.

Remark 4. All these considerations are correct only if the values of the potentials φ, ψ and c are finite. Otherwise, the conditions (8) are not satisfied.

Remark 5. Strictly speaking, completness of equivalent transformations (10) holds only if the base graph of the task is connected. The set of all equivalent transformations together with the set of all uniform transformations completely describes an equivalence class even in the case, if the base graph of the task is not connected.

Remark 6. Equivalent transformations can be formulated for certain other semirings in a similar manner by substituting the operation $\otimes$.

3. Transformation "K to 2"

Let $\mathscr{A}(V,K,q,g)$ be a task with a finite ordered set of states $K = \{1,2\ldots|K|\}$ and let f be a labeling $f : R \to K$. Its quality is

$$G(f) = \sum_{r\in R} q_r\big(f(r)\big) \;+ \sum_{(r,r')\in E} g_{rr'}\big(f(r),f(r')\big).$$

Our goal is to construct another task (namely a task with only two states) in such a way, that each labeling in the first task has a corresponding labeling in the second one and the qualities of these two labelings are equal (may be up to a constant). The number of labelings in the new task can be greater than the number of labelings in the initial one. We will show however, that it is possible to build the new task so, that all these "additional"

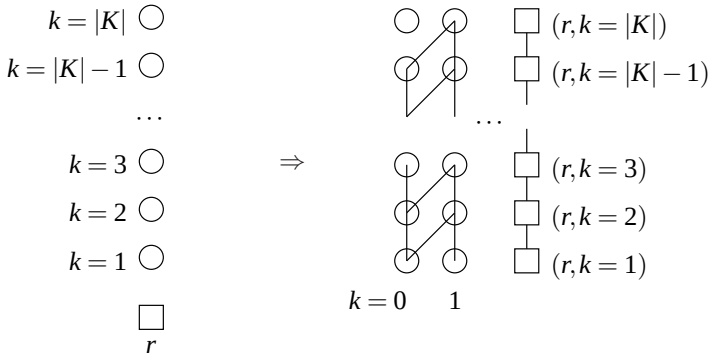

FIGURE 2. Construction of the set of nodes.

labelings have infinite quality. Let us denote the components of the second task by $\hat{V}(\hat{R},\hat{E})$, $\hat{K} = \{0,1\}$, $\hat{q}_{\hat{r}}$ and $\hat{g}_{\hat{r}\hat{r}'}$. Summarizing, we have to build a task $\hat{\mathscr{A}}(\hat{V},\{0,1\},\hat{q},\hat{g})$ so, that for each labeling f there exists an unique labeling $\hat{f} : \hat{R} \to \{0,1\}$ and $G(f) = \hat{G}(\hat{f}) + C$ holds. We build this new task as follows:

(1) First, we construct the set of nodes $\hat{R}$ of the new task. We connect these nodes by certain "special" edges. For each such edge we define a function $\hat{g}$, so that exactly $|K|^{|R|}$ labelings in the new task have a finite quality. At the same time we build a correspondence betwen labelings f in the initial task and labelings $\hat{f}$ in the constructed one.

(2) We define state qualities $\hat{q}$ in the new task so that

$$\sum_{r \in R} q_r\big(f(r)\big) = \sum_{\hat{r} \in \hat{R}} \hat{q}_{\hat{r}}\big(\hat{f}(\hat{r})\big)$$

holds for each pair of corresponding labelings f and $\hat{f}$.

(3) Furthermore we define the set of edges $\hat{E}$ and the functions $\hat{g}$ so that

$$\sum_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) = \sum_{(\hat{r},\hat{r}') \in \hat{E}} \hat{g}_{\hat{r}\hat{r}'}\big(\hat{f}(\hat{r}), \hat{f}(\hat{r}')\big)$$

holds for each pair of corresponding labelings f and $\hat{f}$.

(4) Finally we simplify the constructed task by equivalent and uniform transformations.

Let us consider these steps in more detail.

The set of nodes $\hat{R}$ is built as follows. For each state of the initial task (for each pair (r,k)) one node in the new task is introduced, i.e. $\hat{R} = R \times K$. We denote the nodes of the second task as $\hat{r} = (r,k)$. Let us consider a node r of the initial task and the subset of corresponding nodes $\hat{r}$ in the new task. We denote this subset as $\hat{R}_r = \{(r,k) \mid k = 1 \dots |K|\}$. Now, it is necessary to "link" these nodes with edges and to define functions $\hat{g}$ on these edges so, that exactly $|K|$ "sublabelings" in the new task have a finite quality. We understand the notion "sublabeling" as a mapping $\hat{f}_r : \hat{R}_r \to \{0,1\}$ – i.e. a restriction of a labeling $\hat{f}$ on the set $\hat{R}_r$. To realize this, we link by edges the "neighboring" nodes (with respect to the order of states in the initial task). The functions $\hat{g}_{(r,k)(r,k+1)}$, $k = 1 \dots |K| - 1$

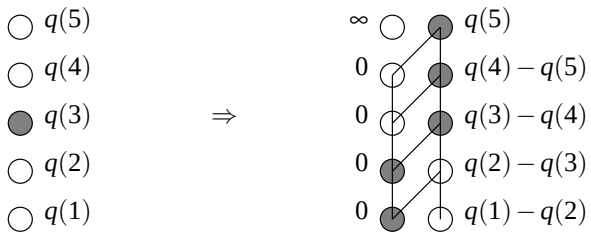

FIGURE 3. State qualities.

are defined as follows[2]:

$$\hat{g}_{(r,k)(r,k+1)} = \begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array},\ \text{if } k < |K|,$$

$$\hat{g}_{(r,|K|-1)(r,|K|)} = \begin{array}{|c|c|}\hline \infty & 0 \\ \hline \infty & 0 \\ \hline \end{array}.$$

It is easy to see, that only $|K|$ sublabelings in the new task have a finite quality (see fig. 2). "Forbidden" pairs of states i.e. those, for which the value of the function $\hat{g}$ is ∞ are not shown.

The states of the new task can be understood as follows. The value $\hat{f}(r,k) = 0$ means, that the state $f(r)$ of the initial task is located above the state k. The value $\hat{f}(r,k) = 1$ means, that the state $f(r)$ of the initial task is located below the state k or coincides with k. (The notations "above" and "below" are meant according to the introduced order of states in the initial task). Thus each pair (r,k^*) of the initial task corresponds to one sublabeling $\hat{f}_r(r,k) = 0,\ k < k^*,\ \hat{f}_r(r,k) = 1,\ k \geq k^*$ and vice versa.

The next step consists in choosing the state qualities for the new task – i.e. $\hat{q}_{(r,k)}(0)$ and $\hat{q}_{(r,k)}(1)$. These qualities should be defined in such a way, that the quality of each sublabeling will be the same as the quality $q_r(k^*)$ of the corresponding state k^* of the initial task. We define these qualities as follows:

$$\hat{q}_{(r,k)} = \begin{array}{|c|}\hline q_r(k) - q_r(k+1) \\ \hline 0 \\ \hline \end{array},\ \text{if } k < |K|$$

$$\hat{q}_{(r,|K|)} = \begin{array}{|c|}\hline q_r(|K|) \\ \hline \infty \\ \hline \end{array}.$$

It can be seen, that

$$\sum_k \hat{q}_{(r,k)}\big(\hat{f}_r(r,k)\big) = \sum_{k^* \leq k < |K|} \big[q_r(k) - q_r(k+1)\big] + q_r(|K|) = q_r(k^*) \tag{11}$$

holds for each sublabeling $\hat{f}_r(r,k) = 0,\ k < k^*,\ \hat{f}_r(r,k) = 1,\ k \geq k^*$ (see fig.3, where an example for $|K| = 5$ and $f(r) = 3$ is shown).

To build the set of edges $\hat{E}$ (as well as to define functions $\hat{g}$) we consider two nodes r and r' of the initial task, which are connected by an edge, i.e. $(r,r') \in E$. There are two corresponding subsets of nodes $\{(r,k)\}$ and $\{(r',k')\}$, $k,k' = 1 \ldots |K|$ in the second task.

[2]We write functions $\hat{q}$ and $\hat{g}$ in the form

$$\hat{q} = \begin{array}{|c|}\hline \hat{q}(1) \\ \hline \hat{q}(0) \\ \hline \end{array} \text{ and } \hat{g} = \begin{array}{|c|c|}\hline \hat{g}(1,0) & \hat{g}(1,1) \\ \hline \hat{g}(0,0) & \hat{g}(0,1) \\ \hline \end{array}$$

respectively for shortness. Sometimes we also omit indices writing e.g. $\hat{q}$ instead of $\hat{q}_{\hat{f}}$

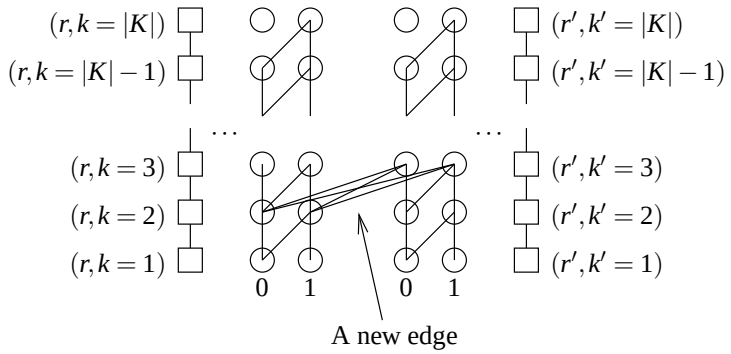

FIGURE 4. Construction of the set of edges

We introduce edges $\big((r,k),(r',k')\big)$ in the new task, i.e. we connect each node (r,k) with each node (r',k') by an edge (see fig. 4, where only one of the introduced edges is shown). Furthermore let us consider a pair of states (k^*,k'^*) on the edge (r,r') of the initial task. There is a corresponding pair of sublabelings in the new task. Consequently the goal is to define qualities for the introduced edges in such a way, that the quality of this pair of sublabelings (without taking into account the $\hat{q}$-part) will be equal to the quality $g_{rr'}(k^*,k'^*)$ of the initial pair of states (k^*,k'^*), i.e.

$$\begin{aligned}
&\sum_{k,k'} \hat{g}_{(r,k)(r',k')}\big(\hat{f}(r,k),\hat{f}(r',k')\big) = \\
&\qquad \sum_{k<k^*}\sum_{k'<k'^*} \hat{g}_{(r,k)(r',k')}(0,0) \;+ \sum_{k<k^*}\sum_{k'\geq k'^*} \hat{g}_{(r,k)(r',k')}(0,1) + \\
&\qquad \sum_{k\geq k^*}\sum_{k'<k'^*} \hat{g}_{(r,k)(r',k')}(1,0) \;+ \sum_{k\geq k^*}\sum_{k'\geq k'^*} \hat{g}_{(r,k)(r',k')}(1,1) = \\
&\qquad = g_{rr'}(k^*,k'^*). \qquad (12)
\end{aligned}$$

This should hold for each pair (k^*,k'^*) of states in the initial task. Such qualities can be chosen as

$$\hat{g}_{(r,k)(r',k')} = \begin{array}{|c|c|}\hline 0 & \alpha_{rr'}(k,k') \\ \hline 0 & 0 \\ \hline \end{array},$$

where

$$\alpha_{rr'}(k,k') = \begin{cases} g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) - g_{rr'}(k+1,k') - g_{rr'}(k,k'+1), \text{ if } k,k' < |K|, \\ g_{rr'}(k,k') - g_{rr'}(k+1,k'), \text{ if } k < |K|,\ k' = |K|, \\ g_{rr'}(k,k') - g_{rr'}(k,k'+1), \text{ if } k = |K|,\ k' < |K|, \\ g_{rr'}(k,k'), \text{ if } k = |K|,\ k' = |K|. \end{cases}$$

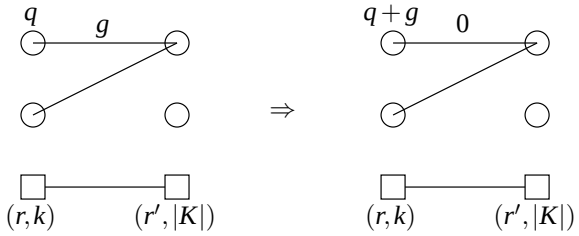

FIGURE 5. Equivalent transformation

It is easy to show, that in doing so, the equations (12) are satisfied. Let us substitute $\hat{g}$ in (12) by the values defined above.

$$0+0+0+\sum_{k\geq k^*}\sum_{k'\geq k'^*}\alpha_{rr'}(k,k') =$$
$$\Big[g_{rr'}(k^*,k'^*)-g_{rr'}(k^*,|K|)-g_{rr'}(|K|,k'^*)+g_{rr'}(|K|,|K|)\Big]+$$
$$\Big[g_{rr'}(k^*,|K|)-g_{rr'}(|K|,|K|)\Big]+\Big[g_{rr'}(|K|,k'^*)-g_{rr'}(|K|,|K|)\Big]+g_{rr'}(|K|,|K|)=$$
$$=g_{rr'}(k^*,k'^*). \tag{13}$$

Let us consider finally a labeling f of the initial task. The corresponding labeling $\hat{f}$ in the new task is

$$\hat{f}(r,k)=\begin{cases}1 & \text{if } k\geq f(r),\\ 0 & \text{otherwise.}\end{cases}$$

It can be seen from (11) and (13), that the qualities $\hat{G}(\hat{f})$ (the quality of the labeling $\hat{f}$ in the new task) and $G(f)$ (the quality of the labeling f in the initial one) are equal.

The constructed task can be simplified in the following way. One can see, that certain "elements" (qualities on certain nodes and edges) give a constant contribution for the goal function. "Constant contribution" means a quantity, which does not depend on labeling. Such elements are the "highest" nodes $(r,|K|)$ for all $r\in R$ and the "highest" edges $\big((r,|K|),(r',|K|)\big)$ for all $(r,r')\in E$. This holds, because in the nodes $(r,|K|)$ really only one state is allowed: $\hat{f}(r,|K|)=1$. These elements can be zeroed out without changing the location of the best labeling. The values of all labelings will be changed by a value C, which is the sum of all zeroed elements:

$$C=\sum_{r\in R}q_r(|K|)\ +\sum_{(r,r')\in E}g_{rr'}(|K|,|K|).$$

The above simplification is in fact an uniform transformation considered in the previous section.

A further simplification is based on the fact, that the contribution of certain edges for the goal function depends only on the state in *one* node. All those edges, which are connected to the "highest" nodes – $\big((r,k),(r',|K|)\big)$ for all $(r,r')\in E,\ k<|K|$, have this property. The contribution of these edges depends only on the states in corresponding nodes (r,k). These contributions can be zeroed out by equivalent transformations, shown in fig.5. Each state $\big((r,k),1\big)$ gets after that transformation an additional quality

$$\triangle\,\hat{q}_{(r,k)}(1)=\sum_{r':(r,r')\in E}\hat{g}_{(r,k)(r',|K|)}(1,1)=\sum_{r':(r,r')\in E}\Big[g_{rr'}(k,|K|)-g_{rr'}(k{+}1,|K|)\Big].$$

After these two transformations the "highest" nodes $(r,|K|)$ can be simply removed from the set of nodes $\hat{R}$ without changing the task.

Let us summarise all considerations made so far. Let $\mathscr{A} = (V,K,q,g)$ be a task with the set of $|K|$ states: $K = \{1\dots|K|\}$. The corresponding task $\hat{\mathscr{A}} = (\hat{V},\{0,1\},\hat{q},\hat{g})$ with the set of only two states is constructed as follows:

- The set of nodes is
$$\hat{R} = \{(r,k) \mid r \in R,\ k = 1\dots|K|-1\}.$$
- The set of edges is
$$\begin{aligned}\hat{E} =& \{\big((r,k),(r,k+1)\big) \mid r \in R,\ k = 1\dots|K|-2\} \cup \\ & \{\big((r,k),(r',k')\big) \mid (r,r') \in E,\ k,k' = 1\dots|K|-1\}.\end{aligned}$$
- State qualities are
$$\hat{q}_{(r,k)}(0) = 0,$$
$$\hat{q}_{(r,k)}(1) = q_r(k) - q_r(k+1) + \sum_{r':(r,r')\in E} \Big[g_{rr'}(k,|K|) - g_{rr'}(k+1,|K|)\Big].$$
- Qualities of state pairs are
$$\hat{g}_{(r,k)(r,k+1)} = \begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array}$$
$$\hat{g}_{(r,k)(r',k')} = \begin{array}{|c|c|}\hline 0 & \alpha_{rr'}(k,k') \\ \hline 0 & 0 \\ \hline \end{array}, \text{ with}$$
$$\alpha_{rr'}(k,k') = g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) - g_{rr'}(k+1,k') - g_{rr'}(k,k'+1).$$
- Each labeling $f : R \to K$ of the initial task corresponds to exactly one labeling $\hat{f} : \hat{R} \to \{0,1\}$ in the new task. This labeling is
$$\hat{f}(r,k) = \begin{cases} 1 & \text{if } k \geq f(r), \\ 0 & \text{otherwise.} \end{cases}$$
Only such labelings of the new task have a finite quality.
- The qualities of the labeling f and the corresponding labeling $\hat{f}$ differ by a constant, which does not depend on the labeling:

$$G(f) = \hat{G}(\hat{f}) + \sum_{r\in R} q_r(|K|) + \sum_{(r,r')\in E} g_{rr'}(|K|,|K|). \tag{14}$$

We would like to note, that the transformation "K to 2" can be made for an arbitrary MinSum Task. If the initial task is in addition submodular, the constructed task is submodular as well. This can be easily proved. There are only edges of type

$$\hat{g} = \begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array} \quad \text{or} \quad \hat{g} = \begin{array}{|c|c|}\hline 0 & \alpha \\ \hline 0 & 0 \\ \hline \end{array}$$

in the new task (let us call them "type 1" and "type 2" respectively). Edges of the first type are submodular. If the initial task is submodular, all α-s are not positive. Therefore all edges of the second type are submodular as well.

The constructed task (as described above) is in certain sense not unique. Obviously the new task can be changed by equivalent and uniform transformations in any way. Consequently, the transformation "K to 2" transforms not only *one* task $\mathscr{A}$ into *one* new task $\hat{\mathscr{A}}$, but rather an equivalence class $\mathscr{E}(\mathscr{A})$ into another equivalence class $\mathscr{E}(\hat{\mathscr{A}})$. This property

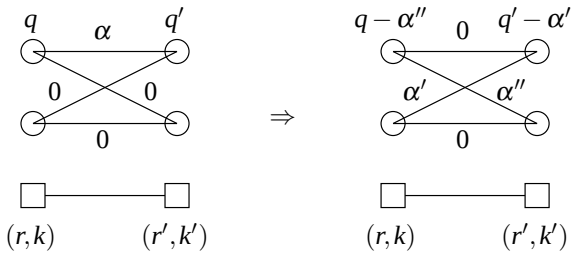

FIGURE 6. Equivalent transformation

can be used to represent all functions $\hat{q}$ and $\hat{g}$ of a task in an unified manner. For $\hat{g}$-s we prefer to use the representation

$$\hat{g} = \begin{array}{|c|c|}\hline \alpha'' & 0 \\ \hline 0 & \alpha' \\ \hline \end{array}. \tag{15}$$

The edges of the first type have already this form (i.e. $\alpha' = 0$ and $\alpha'' = \infty$). The edges of the second type can be transformed into this form using the equivalent transformation shown in fig. 6 (with $\alpha' + \alpha'' = -\alpha$). In addition, if the task is submodular all edges can be transformed into this form with non-negative α' and α''. A further possibility to simplify a task is applying the uniform transformation

$$\hat{q}_{(r,k)}(0) = \hat{q}_{(r,k)}(0) - c,$$
$$\hat{q}_{(r,k)}(1) = \hat{q}_{(r,k)}(1) - c,$$

where $c = \min[\hat{q}_{(r,k)}(0), \hat{q}_{(r,k)}(1)]$. After that, all values of $\hat{q}$-s become non negative.

Summarising, a general MinSum task $\mathscr{A}(V,K,q,g)$ can be transformed into a corresponding binary MinSum task $\hat{\mathscr{A}}(\hat{V},\{0,1\},\hat{q},\hat{g})$ with the following properties:

- The functions $\hat{q}$ are non-negative.
- The functions $\hat{g}$ have the form (15). If the initial task is submodular, both α' and α'' can be chosen non-negative.

We will call the above representation "the canonical form".

Remark 7. Obviously the transformation "K to 2" described above, is possible only, if there are no infinite qualities in the initial task. Otherwise the operation "$-$" is not always well defined. In practice it is however always possible to substitute infinite qualities by some appropriately chosen big values, so that the location (as well as the value) of the solution remains surely unchanged. It is even always possible if the submodularity of the initial task should be preserved.

Remark 8. The transformation "K to 2" can be straightforward generalized for such semirings and tasks, where the "inverse" of $\otimes$ is well defined operation for all occurring qualities, i.e. if the equation $a \otimes x = b$ is uniquely solvable with respect to x for all values a and b, occurring in the task. For example, for SumProd problems this transformation can be easily done, if there are no zero qualities in the task. The functions $\hat{g}$ in the constructed task have again the form (15) in that case (up to replacing 0 by the unit element of multiplication).

Remark 9. The transformation "K to 2" can be straightforward generalized for labeling problems of order greater than two. The key idea is, that the values of the constructed functions $\hat{q}$ and $\hat{g}$ can be seen as a kind of partial derivative $\partial G(f)/\partial f(r)$ and $\partial^2 G(f)/\partial f(r)\partial f(r')$ respectively [6]. It is easy to see, that for problems of second order

such derivatives of order higher than two are zero, which is not the case for problems of higher order. Derivatives of higher order can be however "modelled" in the new task by binary functions of form $g(k_1,k_2\dots k_l)=\alpha\cdot k_1\cdot k_2\dots k_l$ with ($k_i\in\{0,1\}$). Consequently the constructed binary task will have the same order as the initial one. Unfortunately, the number of functions g in the constructed task grows as $O(|K|^L)$, where L is the order.

4. Construction of a MinCut problem, exact solution of submodular MinSum problems

A general MinCut problem is formulated as follows. Let $\mathscr{G}(\mathscr{V},\mathscr{E})$ be an oriented graph. There are two special nodes called *source* (denoted by s) and *target* (denoted by t) in the set of nodes. Each edge $(u,v)\in\mathscr{E}$ has its quality $c(u,v)$. A *cut* $\mathscr{C}$ is a subset of edges, so that there is no directed path from the source to the target in the "remaining" graph $\mathscr{G}(\mathscr{V},\mathscr{E}/\mathscr{C})$. In addition this set should be "minimal" in the following sense. There is no proper subset of $\mathscr{C}$ which has the same property. The quality of a cut is the sum of all edge qualities in this subset. The task is to find the cut with minimum quality.

Without going deep into details we give the rules for transformation of a binary MinSum problem $\mathscr{A}(V,\{0,1\},q,g)$ (given in canonical form) into a corresponding MinCut task. These rules are very similar to those given in [14]. A slight difference is only, that we consider also functions g, which can have infinite qualities.

- The set of nodes in the MinCut task is the set of nodes of the MinSum problem with two additional nodes s and t (source and target):

$$\mathscr{V}=R\cup\{s,t\}.$$

- The set of edges is the "doubled" set of edges of the MinSum task with additional edges going from the source to each node and from each node to the target:

$$\mathscr{E}=\big\{(r,r')\mid(r,r')\in E\big\}\cup\big\{(r',r)\mid(r,r')\in E\big\}\cup\big\{(s,r)\mid r\in R\big\}\cup\big\{(r,t)\mid r\in R\big\}.$$

- The edge qualities are:

$$\begin{aligned} c(s,r)&=q_r(1), & c(r,t)&=q_r(0),\\ c(r,r')&=\alpha'_{rr'}, & c(r',r)&=\alpha''_{rr'}. \end{aligned}$$

Let us summarise all above transformations, namely: the transformation "K to 2", transformation of the functions g into canonical form and construction of the MinCut task.
Let $\mathscr{A}=(V,K,q,g)$ be a task given by its components:

the base graph $V=(R,E)$ is arbitrary,
the set of states $K=\{1\dots|K|\}$ is an ordered set,
the state qualities are q_r,
the qualities of state pairs are $g_{rr'}$.

The corresponding MinCut task $(\mathscr{G}=(\mathscr{V},\mathscr{E}),s\in\mathscr{V},t\in\mathscr{V},c)$ is built as follows:

- The set of nodes is

$$\mathscr{V}=\{(r,k)\mid r\in R,\ k=1\dots|K|-1\}\cup\{s\}\cup\{t\}.$$

- The set of edges is

$$\begin{aligned}\mathscr{E} = &\big\{\big((r,k),(r,k+1)\big) \mid r \in R,\ k = 1\dots|K|-2\big\}\cup\\ &\big\{\big((r,k+1),(r,k)\big) \mid r \in R,\ k = 1\dots|K|-2\big\}\cup\\ &\big\{\big((r,k),(r',k')\big) \mid (r,r') \in E,\ k,k' = 1\dots|K|-1\big\}\cup\\ &\big\{\big((r',k'),(r,k)\big) \mid (r,r') \in E,\ k,k' = 1\dots|K|-1\big\}\cup\\ &\big\{\big(s,(r,k)\big) \mid r \in R,\ k = 1\dots|K|-1\big\}\cup\\ &\big\{\big((r,k),t\big) \mid r \in R,\ k = 1\dots|K|-1\big\}.\end{aligned}$$

- The edge qualities are

$$\begin{aligned}&c\big((r,k),(r,k+1)\big) = 0, && r \in R,\ k = 1\dots|K|-2,\\ &c\big((r,k+1),(r,k)\big) = \infty, && r \in R,\ k = 1\dots|K|-2,\\ &c\big((r,k),(r',k')\big) = -\alpha_{rr'}(k,k')/2, && (r,r') \in E,\ k,k' = 1\dots|K|-1,\\ &c\big((r',k'),(r,k)\big) = -\alpha_{rr'}(k,k')/2, && (r,r') \in E,\ k,k' = 1\dots|K|-1,\\ &c\big(s,(r,k)\big) = \max[q'_{rk},0], && r \in R,\ k = 1\dots|K|-1,\\ &c\big((r,k),t\big) = \max[-q'_{rk},0], && r \in R,\ k = 1\dots|K|-1,\end{aligned}$$

  where

$$\begin{aligned}\alpha_{rr'}(k,k') &= g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) - g_{rr'}(k+1,k') - g_{rr'}(k,k'+1),\\ q'_{rk} &= q_r(k) - q_r(k+1) +\\ &\quad 1/2 \sum_{(r,r')\in E} \big[g_{rr'}(k,1) + g_{rr'}(k,|K|) - g_{rr'}(k+1,1) - g_{rr'}(k+1,|K|)\big].\end{aligned}$$

If the task $\mathscr{A}$ is submodular, all edge qualities in the constructed MinCut problem are non-negative. In this case the MinCut task can be solved in polynomial time using MaxFlow algorithms (see e.g. [1]).

5. Conclusion

Labeling problems have attracted attention of many researchers in the last years. Very promising results were obtained especially for binary labeling problems, either for MinSum (see e.g. [14]) or for SumProd [18]. There were also many attempts to generalize these results for the case of more than two states. Some examples are: [3], where an approximative approach is proposed, [9], where an exact solvable subclass of MinSum problems is described, etc.

We presented in this report a scheme to construct a corresponding binary labeling problem for an arbitrary one. Consequently all results obtained before for binary labeling problems are extended to the general case.

At this place we would like to discuss the relevance of minimization of a general submodular functional for practical applications. The situation in this field seems to be rather complex. At the moment we see many possibilities to use our approach for certain computer vision tasks, like e.g. stereo reconstruction, motion estimation, segmentation, image restoration etc. (for examples see [7, 8, 10, 12, 13, 20, 21]). Actually such problems are often formulated (and more or less successfully solved) as energy minimization tasks with "a-priori energy" (or "modell term") which is already covered by [9] and/or [3, 14]. It is not easy to explain without going deep into details, why one can need something more,

namely the minimization of a general submodular functional. Let us consider the following example. It might be useful to penalize depth/disparity jumps in stereo reconstruction depending on depth/disparity itself. This leads immediately to general submodular penalty functions. Obviously, this is profitable for those scenes (and only for those), which fulfill this assumption. It means, that the use of a general submodular functional is allways very task specific. Summarizing, we can not give a concrete example of a task, where the minimization of a general submodular functional can surely improve existing results. In this situation we can only give some general ideas, where the method can be used.

Considering a particular application, it is often necessary simply to choose one of those known energy functionals, which would be hopefully suitable for the given task. Our result essentially enlarges "the set of possibilities" for practical applications, allowing to incorporate "more a-priori knowledge" in prior models.

Actually there are many approximative methods for energy minimization for the general case – [2, 3, 11, 15] etc. Many of them are based on minimization of a function over binary variables. For instance, in [3] an iterative procedure is described, which solves in each iteration a binary submodular MinSum problem. It seems to be possible to extend such approaches so that the used "auxiliary tasks" are not necessarily binary. Besides of this, all the methods for approximative minimization of general functionals over binary variables can be used now directly for MinSum problems with many states (using "Transformation K to 2", described in the work). Another way to solve general MinSum problems approximatively is to approximate the initial task by an submodular one and solve it then exactly, rather then to apply approximative algorithms to the initial task.

In statistical pattern recognition it is often the case, that we can not assume that all parameters of the prior model are known. Consequently they should be learned (often even in an unsupervised manner). In many cases the final task is posed as the maximum a-posteriori decision, that is just an energy minimization task. When trying to keep MAP solvable in the learning phase, it is necessary to restrict the parameters of the prior modell. It would be obviously useful to be able to restrict the parameters as weak as possible (to keep the modell adequate). Thus any enlargement of the class of exactly solvable MinSum tasks is of interest from this point of view as well.

There are also many other open question in the field of labeling problems. Unfortunately, the scheme "K to 2" presented in this report gives no direct way *to solve* labeling problems (besides of submodular MinSum tasks). For instance, the question of an exact solvable (and practically relevant) subclass of SumProd problems is still open even in the binary case.

REFERENCES

[1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin, *Network flows: theory, algorithms, and applications*, Prentice-Hall, 1993.

[2] Endre Boros and Peter L. Hammer, *Pseudo-boolean optimization*, Tech. report, RRR 48-2001, September 2001.

[3] Y. Boykov, O. Veksler, and R. Zabih, *Fast approximate energy minimization via graph cuts*, ICCV, 1999, pp. 377–384.

[4] R. Burkard, B. Klinz, and R. Rudolf, *Perspectives of monge properties in optimization*, DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science **70** (1996).

[5] B. Flach, *Strukturelle Bilderkennung: Habilitationsschrift*, Technische Universität Dresden, 2003.

[6] B. Flach and D. Schlesinger, *Best labeling search for a class of higher order gibbs models*, Pattern Recognition and Image Analysis **14** (2004), no. 2, 249–254.

[7] B. Flach, D. Schlesinger, E. Kask, and A. Skulisch, *Unifying registration and segmentation for multi-sensor images*, DAGM 2002 (Luc Van Gool, ed.), LNCS 2449, Springer, 2002, pp. 190–197.

[8] D. M. Greig, B. T. Porteous, and A. H. Seheult, *Exact maximum a posteriori estimation for binary images*, J. R. Statist. Soc. **51** (1989), no. 2, 271–279.
[9] H. Ishikawa, *Exact optimization for markov random fields with convex priors*, IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003), no. 10, 1333–1336.
[10] Hiroshi Ishikawa and Davi Geiger, *Segmentation by grouping junctions*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998.
[11] Wainwright M. J., Jaakkola T. S., and Willsky A. S., *Map estimation via agreement on (hyper) trees: Message-parsing and linear programming approaches*, Tech. report, MSR-TR-2004-90, 2004.
[12] E. Kask and S. Fuchs, *Ct-basierte 3d-Analyse von Baustoffen*, 5. Anwendungsbezogener Workshop zur Erfassung, Verarbeitung, Modellierung und Auswertung von 3D-Daten, Dezember 2002, pp. 35–42.
[13] Vladimir Kolmogorov and Ramin Zabih, *Computing visual correspondence with occlusions via graph cuts*, International Conference on Computer Vision, 2001, pp. 508–515.
[14] ———, *What energy functions can be minimized via graph cuts?*, ECCV 2002 (Berlin Heidelberg) (A. Heyden et al., ed.), LNCS, no. 2352, Springer-Verlag, 2002, pp. 65–81.
[15] I. Kovtun, *Partial optimal labeling search for a np-hard subclass of (max,+) problems*, Pattern Recognition (Gerald Krell Bernd Michaelis, ed.), LNCS, vol. 2781, Springer, September 2003, pp. 402–409.
[16] Schlesinger M.I. and Hlavác V., *Ten lectures on statistical and structural pattern recognition*, Kluwer Academic Publishers, Dordrecht, May 2002.
[17] G. Monge, *Déblai et Remblai*, Mem. de l'Academie des Sciences, 1781.
[18] James Gary Propp and David Bruce Wilson, *Exact sampling with coupled markov chains and applications to statistical mechanics*, Random Structures Algorithms **9** (1996), no. 1, 223–252.
[19] A. Rosenfeld, R.A. Hummel, and S.W. Zucker, *Scene labeling by relaxation operations*, IEEE Trans. SMC **6(6)** (1976), 420–433.
[20] S. Roy and I. Cox, *A maximum-flow formulation of the n-camera stereo correspondence problem*, International Conference on Computer Vision, 1998.
[21] D. Schlesinger, *Gibbs probability distributions for stereo reconstruction.*, DAGM 2003 (B. Michaelis and G. Krell, eds.), LNCS, vol. 2781, 2003, pp. 394–401.
[22] M.I. Schlesinger, *Mathematical methods of image processing*, Naukova Dumka, Kiev, 1989.
[23] M.I. Schlesinger and B. Flach, *Some solvable subclasses of structural recognition problems*, Czech Pattern Recognition Workshop 2000 (Tomáš Svoboda, ed.), 2000, pp. 55–62.
[24] D. Shlezinger, *Strukturelle Ansätze für die Stereorekonstruktion*, Ph.D. thesis, Technische Universität Dresden, 2005, http://nbn-resolving.de/urn:nbn:de:swb:14-1126171326473-57594.