Hierarchical Image Segmentation using Correlation Clustering

Amir Alush and Jacob Goldberger

Abstract—In this study we apply efficient implementations of integer linear programming to the problem of image segmentation. The image is first grouped into superpixels and then local information is extracted for each pair of spatially adjacent superpixels. Given local scores on a map of several hundred superpixels, we use correlation clustering to find the global segmentation, that is most consistent with the local evidence. We show that, although correlation clustering is known to be NP hard, finding the exact global solution is still feasible by breaking the segmentation problem down into subproblems. Each such subproblem can be viewed as an automatically detected image part. We can further accelerate the process by using the cutting-plane method, which provides a hierarchical structure of the segmentations. The efficiency and improved performance of the proposed method is compared to several state-of-the-art methods and demonstrated on several standard segmentation data sets.

Index Terms—image segmentation, integer linear programming, correlation clustering, hierarchical segmentation.

I. INTRODUCTION

Effective automatic grouping of objects into clusters is one of the fundamental problems in machine learning and other fields of study. In many approaches, the first step toward clustering a dataset is extracting a feature vector from each object. This reduces the problem to the aggregation of groups of vectors in a feature space. A commonly used algorithm in this case is the $k$-means. One drawback of $k$-means is that it can only find clusters that are linearly separable in the feature space. Furthermore, in many cases feature representation is not available and we are only given pairwise similarity information between data points. For example, in social networks, only binary neighborhood relations are given. In these cases feature based clustering algorithms cannot be applied in a straightforward way. Instead, we seek for a partition of the data based only on the similarity measure between the points. In this study we address pairwise clustering in the context of image segmentation which is a fundamental process in many image, video, and computer vision applications. It is essentially the partitioning of an image into several constituent components. Many segmentation algorithms have been proposed and studied in recent decades and new algorithms are continuously emerging. These segmentation algorithms are usually based on various combinations of local low-level features and global optimization methods. In this paper we focus on the global optimization side of image segmentation.

Many visual tasks including segmentation can benefit from the complexity reduction achieved by transforming an image with millions of pixels into a few hundred or thousand "superpixels". Superpixels, introduced in [1], are small, homogeneous regions preserving almost all boundaries between different regions and are obtained by a low-level process based on cues such as color, edges and texture. The use of superpixels as primitive objects for clustering significantly reduces computational costs and allows feature extraction to be conducted from a larger homogeneous region. In recent years many automatic superpixel algorithms have been proposed (see e.g. [1], [2], [3], [4]). Given a superpixel graph we can first extract a local similarity measure for each pair of spatially adjacent superpixels and then find the global segmentation that is the most consistent with the local similarity cues. This paradigm is common to many graph based image segmentation algorithms (e.g. [5], [6], [7], [8], [9], [10], [11], [12]). However, current segmentation approaches, even when applied to superpixels, do not aim to find a globally optimal segmentation. Instead, they utilize approximation methods such as greedy hierarchical superpixels merging [6], LP-relaxation [13], [14] and spectral clustering algorithms that find an approximation of the optimal normalized-cut [15]. Yarkony et al. [16] suggested a dual column-generating method, called PlanarCC, which solves a relaxed linear program by iteratively solving weighted two-coloring problems. Because these sub-problems require planarity to be tractable, PlanarCC is restricted to problems with planar structure. Another approximation approach is based on move-making methods that maintain a valid segmentation. In each step, a set of possible moves transforming the current segmentation is considered, and the move that leads to the maximal energy decrease is chosen until a local optimum is found [17].

In this study we define a probabilistic model for image segmentation given a superpixel map that is based on correlation clustering [18], [19] (also known as the multicut formulation [20]). Several studies have recently applied the correlation clustering framework to image segmentation. Since the problem is known to be NP-hard [19], there have been several attempts to find approximate solutions that provide good segmentation results. In [21], [14] the correlation clustering model was solved using higher order potentials and a relaxation of the exact optimization problem by a sequence of linear programs.

Andres et al. [22] used an integer linear programming (ILP) branch-and-cut strategy. Correlation clustering was also utilized for computing the ensemble segmentation from a given set of segmentations [23] based on the observation that seg-
mentations of the same image are expected to agree on image parts that are clearly separated from the rest of the image and when the segmentations are projected on a superpixel map, the correlation clustering problem can be broken down into non-overlapping parts and solved independently. The technique of decomposing image analysis into smaller subproblems is also related to dual decomposition optimization which was recently applied by [16] for image segmentation. In this work we show that unsupervised image segmentation based solely on local cues can also benefit from decomposing the segmentation problem into subproblems.

To find the optimal segmentation, based on the correlation clustering model, we need to solve an Integer Linear Program (ILP). The ILP problem is known to be NP hard which has prevented the algorithm from being applied to the image segmentation problem. The main contribution of this study is showing that finding the optimal global segmentation that is consistent with the local cues, is still tractable. This can be done by a careful analysis of the implementation of the general ILP formulation to the image segmentation task.

The rest of this paper is organized as follows. In the next section we review correlation clustering and previous attempts to apply it to image segmentation. Section 3 presents an efficient method to solve the ILP problem and the method is extended to the problem of hierarchical segmentation in Section 4. Experimental results are shown in Section 5. A preliminary version of this work appears in [24].

II. ILP FORMULATION FOR IMAGE SEGMENTATION

Assume \( V = \{1, ..., n\} \) is a set of data points that we want to cluster. For each pair \( i, j \) we are given a probabilistic pairwise information \( p_{ij} \in [0, 1] \) on whether \( i \) and \( j \) are in the same cluster. A higher value indicates a local tendency to group \( i \) and \( j \) into the same cluster and vice versa. The goal of clustering is to find a global grouping of the set \( V \) that is most consistent with the local probabilistic indications \( \{p_{ij}\} \).

A clustering of a set \( \{1, ..., n\} \) can be represented as a set of \( n \)-over-two binary decisions \( x = \{x_{ij} | 1 \leq i < j \leq n\} \) such that \( x_{ij} = 1 \) if \( i \) and \( j \) are in the same cluster and 0 otherwise. The given probability \( p_{ij} \) can be viewed as a probabilistic information \( p_{ij} = p(x_{ij} = 1) \) on the true value of \( x_{ij} \). The correspondence between clusterings and binary decision sets is not one-to-one. Each clustering is represented by a different set of binary decisions but not every set of binary decisions corresponds to a valid clustering. The pairwise relation ‘\( i \) and \( j \) are in the same cluster’ is a transitive relation. If \( i, j, k \) are in the same cluster then necessarily \( i, k \) should be in the same cluster. It can be easily verified that the correspondence between clusterings and transitive binary decision sets is one-to-one.

Assuming a uniform prior over the clusterings, the posterior probability of a clustering \( x \) is:

\[
p(x) \propto \prod_{i<j} p(x_{ij}) \tag{1}
\]

Note that in this simplified probabilistic model the binary local information cues are assumed to be independent. The optimal global clustering which is most consistent with the local pairwise evidence, can be found by computing \( \arg \max_x p(x) \) over all possible clusterings. It can be easily verified that:

\[
\log p(x_{ij}) = x_{ij} \log p_{ij} + (1-x_{ij}) \log (1-p_{ij}) \tag{2}
\]

\[
= x_{ij} \log \frac{p_{ij}}{1-p_{ij}} + \log (1-p_{ij})
\]

Hence,

\[
\log p(x) = \sum_{i<j} \log p(x_{ij}) = \sum_{i<j} w_{ij}x_{ij} + \text{const} \tag{3}
\]

such that ‘const’ is a scalar that is not dependent on \( x \) and

\[
w_{ij} = \log \frac{p_{ij}}{1-p_{ij}} \tag{4}
\]

For each pair \( i, j \in V \) the weight \( w_{ij} \in (-\infty, \infty) \) is a symmetric pairwise similarity measure such that a positive weight indicates a local tendency to group \( i \) and \( j \) into the same cluster and vice versa. The best clustering is:

\[
\hat{x} = \arg \max_x p(x) = \arg \max_x \sum_{i<j} w_{ij}x_{ij} \tag{5}
\]

such that the maximization is done over all the sets of transitive binary decisions \( x \). The clustering score \( \sum w_{ij}x_{ij} \) is the weight summation over all data pairs that are in the same cluster. Observing that the transitivity constraints are linear, the optimal clustering is obtained by solving the following Integer Linear Program (ILP):

\[
\max_x \sum_{i<j} w_{ij}x_{ij} \tag{6}
\]

s.t. \( x_{ij} + x_{jk} - x_{ik} \leq 1 \quad \forall i, j, k \)

\( x_{ij} \in \{0,1\} \quad \forall i, j \)

The linear constraint \( x_{ij} + x_{jk} - x_{ik} \leq 1 \) on the binary variables, enforces transitivity on the binary decisions, i.e., \( x_{ij} = x_{jk} = 1 \) implies that \( x_{ik} = 1 \). Hence, the most likely clustering \( \hat{x} = \arg \max_x p(x) \) is obtained as the solution of the ILP maximization problem (6).

We can easily incorporate prior knowledge on the clustering \( x \) into the ILP framework. Let \( q \) be a prior probability that any two points are in the same cluster. For large values of \( q \) the optimal clustering tends to have a small number of clusters and vice versa. The modified weight function for the posterior probability is:

\[
w_{ij} = \log \frac{p_{ij}}{1-p_{ij}} + \lambda \tag{7}
\]

such that \( \lambda = \log \frac{q}{1-q} \).

Assume we are given a weighed undirected graph \( G = (V, E) \) such that \( V \) is the data points \( \{1, ..., n\} \) that we want to cluster and \( w_{ij} \) is the weight of the edge \( ij \in E \). This graph clustering problem (6) is known as “correlation clustering” [18], [19] and has several advantages. It is derived from a precise probabilistic modeling and it does not require users to specify a parametric form for the clusters, or to pick the number of clusters. The main drawback of the ILP approach is its high complexity which impedes its applicability for clustering large sets. The ILP problem (6) is known to be NP-hard [19].
Assume we are given a superpixel map of an image and a similarity measure between each two neighboring superpixels. We can form the segmentation problem as an instance of correlation clustering and solve the ILP (6) to find the optimal segmentation. This segmentation approach, however, is NP-hard and is not tractable for a graph of hundreds or more superpixels. Most of previously suggested graph-based methods for image segmentation try, explicitly or implicitly, to handle this NP-hardness of the ILP problem by either approximate solutions to the ILP clustering problem (e.g. greedy incremental superpixel merging [6]) or by finding optimal solutions to modified problems (e.g. minimal normalized cut [25]).

A simple approximation approach is to delete all the edges between dissimilar superpixels (i.e., with weights below a predefined threshold), and then look for connected components in the remaining graph. This approach, however, is too local since a single edge with a weight above threshold is sufficient to cause two almost separate regions to be merged. Felzenszwalb and Huttenlocher [26] refined this strategy by proposing an agglomerative global approach based on constructing a minimum spanning tree. A standard approximate solution of the global ILP problem (6) is obtained by an LP relaxation that replaces the binary constraint \( x_{ij} \in \{0, 1\} \) with the linear constraint \( 0 \leq x_{ij} \leq 1 \) [27], [28], [13], [14]. The LP solution, however, is not binary and it is not clear how to convert it into a binary solution that satisfies transitivity. Given the solution of the relaxed LP problem, the segmentation can be found by considering the connected components obtained by eliminating edges with \( x_{ij} \) values below a specified threshold.

In the next sections we show that finding the exact solution for the NP-hard ILP problem (6) is still tractable for image segmentation applied to superpixels.

### III. Efficient Strategies for Solving an ILP

In this section we describe an efficient method for solving the ILP problem (6) by breaking it into small sub-problems and by incrementally adding transitivity constraints that are not satisfied by the current solution.

We use the following notation. Let \( V_1, \ldots, V_k \) be a partition of the dataset we want to cluster \( V \). For each \( i, j \in \{1, \ldots, k\} \), denote \( E \cap (V_i \times V_j) \) by \( E_{ij} \). For \( i \neq j \), an edge in \( E_{ij} \) is called a crossing edge; otherwise the edge is called an internal edge. Denote the set of all the crossing edges by \( E_{cross} \).

**Theorem 1:** Assume \( V \) can be divided into disjoint subsets \( V_1, \ldots, V_k \) such that there is no edge with a positive weight between members of different subsets (i.e., \( w_{ij} \leq 0 \) for every \( i, j \in E_{cross} \)). Then the data clustering, which is the optimal solution of the ILP problem (6), is a refinement of the partition \( V_1, \ldots, V_k \) and is obtained by separately applying the ILP optimization on each subset.

**Proof.** The cost function (6) can be written as a sum of two components:

\[
\sum_{ij \in E} w_{ij} x_{ij} = \sum_{ij \in E_{cross}} w_{ij} x_{ij} + \sum_{t=1}^{k} \sum_{ij \in E_{it}} w_{ij} x_{ij}
\]

where \( E_{it} \) are all the graph edges \( (i, j) \in E \) such that \( i, j \in V_t \). Eq. (8) decomposes the variables that appear in the cost function (6) into two disjoint subsets. The first set contains the crossing edges and the second set contains the internal edges. Hence, by separately maximizing each one of the two sub-problems, we get an upper bound on the solution of the ILP problem (6). Since we assume that \( w_{ij} \leq 0 \) for all \( (i, j) \in E_{cross} \), the optimal zero-one solution of:

\[
\max \sum_{ij \in E_{cross}} w_{ij} x_{ij} \text{ is obtained by setting } x_{ij} = 0 \text{ for all } (i, j) \in E_{cross}.
\]

Solving an ILP problem on each sub-graph \( G_t = (V_t, E_{tt}) \), \( t = 1, \ldots, k \) separately:

\[
\max \sum_{ij \in E_{tt}} w_{ij} x_{ij}
\]

s.t. \( x_{ij} + x_{jk} - x_{ik} \leq 1 \quad \forall i, j, k \in V_t \)

\( x_{ij} \in \{0, 1\} \quad \forall i, j \in V_t \)

we get an upper bound on the optimal global solution. It can be easily verified that the combined solution (with \( x_{ij} = 0 \) for all crossing edges) satisfies all the transitivity constraints in (6) and hence it is optimal.

The most refined partition \( V_1, \ldots, V_k \) that satisfies the requirement of Theorem 1 (no positive weight on crossing edges) can be found by utilizing a greedy approach. We begin with some vertex \( v \in V \) defining the initial set of vertices \( V_1 = \{v\} \). Then, at each iteration, we look for a positive weight edge \((u, v)\), where \( u \in V_1 \) and \( v \not\in V_1 \). Then vertex \( v \) is brought into \( V_1 \). This process is repeated until no vertex can be added to \( V_1 \). We next choose a vertex outside of \( V_1 \) and start constructing \( V_2 \) from the remaining vertices, etc. We call the members of the obtained partition the “positively connected components” (they are actually the connected components of the graph obtained by eliminating all the non-positive weight edges in the original graph). The complexity of the algorithm applied to a \( n \)-vertex graph is \( O(n^2) \). As a result of Theorem 1, we can solve the ILP problem (6) for each positively connected component separately.

For each positively connected component we still need to solve an NP-hard ILP problem that corresponds to correlation clustering restricted to that component. In the case of image segmentation the graph we want to partition is sparse since it is planar and each node has only a small number of spatially adjacent nodes. In case of a sparse graph we can formulate the

---

**Algorithm 1** The GDIS segmentation algorithm.

**Input:** A superpixel map (viewed as a graph) with local similarity weights \( \{w_{ij}\} \) between neighbour superpixels.

**Output:** A segmentation of the image (via clustering of the superpixels).

Break the graph into its positively connected components. Assume there are \( k \) components denoted by \( V_1, \ldots, V_k \).

**for** \( i = 1, \ldots, k \) **do**

Solve the ILP problem restricted to the subset \( V_i \) using edge-based variables and the cutting-plane method.

**end for**

The obtained image segments are the union of the superpixel clusters of the positively connected graph components.
ILP problem (6) much more compactly by associating binary variables only to edges of the graph instead of all the node pairs [22]. The cost function we optimize remains exactly the same. The edge labeling transitive consistency constraint can be enforced by adding a linear constraint for each existing edge that prevents a situation where the two adjacent nodes belong to different clusters but there is a path connecting them in which all the nodes along the path are labeled as connected. Formally for each edge \( ij \in E \) and for each path in the graph \( i = v_0, v_1, ..., v_k = j \), from \( i \) to \( j \) we add the linear constraint:

\[
x_{v_0,v_1} + \cdots + x_{v_{k-1},v_k} - x_{ij} \leq k - 1 \tag{10}
\]

The exponential number of such constraints can be implemented using the cutting plane method [29]. In our approach we apply this edge-based ILP separately for each positively connected component.

The success of applying the graph partitioning approach described above to image segmentation depends on the existence of image parts that can be separated from the rest of the image. Figures 1 and 3 demonstrates that this is indeed a common situation (implementation details are described in Section 4). In these images we show the positively connected components and the final segmentation obtained by solving an ILP problem for each component separately. Therefore, the obtained segmentation is a refinement of the positively connected components partition. We dub the proposed segmentation algorithm “Graph Decomposition ILP Segmentation” (GDIS). The GDIS algorithm was implemented in C. We used the Gurobi software (www.gurobi.com) to solve the ILP optimization subproblems. Applying the GDIS algorithm on an image where the size of the largest positively connected component is 1000 takes a few seconds. The GDIS segmentation algorithm is summarized in Algorithm-Box 1.

IV. HIERARCHICAL SEGMENTATION

Image segmentation divides an image into homogenous regions or objects according to a particular perceptual feature, such as homogeneity in color tone. Hierarchical segmentation algorithms simultaneously analyze the image at several different scales of analysis. Their output is not a single partition, but rather a hierarchy of regions or data structures that captures different partitions for different scales of analysis.

There are two strategies to obtain a segmentation hierarchy. Agglomerative algorithms are based on a “bottom up” approach where each point starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. A divisive strategy is a “top down” approach: we start with one cluster, then the clusters are more refined as we move down the hierarchy. In this section we show that, as a by-product of applying the cutting-plane method to solve the ILP problem (6), we automatically obtain a top-down hierarchical segmentation.

The cutting-plane method is used to solve the ILP problem (6) in the following way. In the first round we ignore all the linear constraints and obtain a solution. If this solution describes a partition of the graph nodes, then it satisfies all the constraints and therefore it is the optimal solution. Otherwise, on each iteration we add all the linear constraints that are not satisfied by the current solution and obtain a new solution. This process is continued until we obtain a solution that satisfies all the constraints. This solution is the optimal segmentation. The computational advantage of the cutting-plane method is that we usually obtain the optimal solution after a small number of iterations using a only a small fraction of the huge set of constraints.

Denote the intermediate results obtained at the \( t \)-th iteration of the cutting-plane method by \( x(t) = \{x_{ij}(t)|1 \leq i < j \leq n\} \). For example, the result of the first iteration, where no constraints are used, is:

\[
x_{ij}(1) = \begin{cases} 
1 & \text{if } w_{ij} > 0 \\
0 & \text{if } w_{ij} < 0 
\end{cases} \tag{11}
\]

We can construct a valid segmentation from \( x(t) \) by defining a new graph \( y(t) \) as the connected components of the graph \( x(t) \). More explicitly, \( y_{ij}(t) = 1 \) if there is a path between nodes \( i \) and \( j \) in the graph defined by \( x(t) \).
Algorithm 2 Hierarchical Segmentation

Input: A weighted undirected graph \( G = (V, E) \) with weights \( \{w_{ij}\} \).
Output: A hierarchical clustering of the graph nodes.

Solve ILP (6) with no constraints.

repeat
   Add constraints that are violated by the current solution.
   Solve the ILP problem using the current constraint subset.
   Compute connected components of the ILP solution to obtain a clustering.
until no constraint is violated.
The obtained clustering set forms a hierarchical clustering of \( G \).

and \( j \) on the graph defined by \( x(t) \). It can be easily verified that the connected components graph \( y(1) \) obtained from \( x(1) \) is exactly the ‘positive connected components’ graph defined in the previous section. Let \( \hat{x} \) be the optimal solution of the ILP problem (6), i.e.,

\[
S(\hat{x}) = \sum_{i<j} w_{ij} \hat{x}_{ij}
\]

is the largest score among all solutions corresponding to a valid segmentation. Let

\[
S(x(t)) = \sum_{i<j} w_{ij} x_{ij}(t)
\]

be the score of the intermediate solution \( x(t) \). In a similar way we define

\[
S(y(t)) = \sum_{i<j} w_{ij} y_{ij}(t)
\]

to be the score of the segmentation \( y(t) \). The graph \( y(t) \) is a valid segmentation but not necessarily the optimal one, hence \( S(y(t)) \leq S(\hat{x}) \). On the other hand, \( x(t) \) is a solution of a maximization problem with fewer constraints than (6) and therefore \( S(\hat{x}) \leq S(x(t)) \). Combining the two inequalities we obtain easily computed lower and upper bounds for the solution of the ILP (6):

\[
S(y(t)) \leq S(\hat{x}) \leq S(x(t)).
\]

The cutting plane iterations terminate when the solution \( x(t) \) is a valid segmentation. In that case \( x(t) = y(t) = \hat{x} \).

Note that always \( S(x(t+1)) \leq S(x(t)) \) since \( x(t+1) \) is a solution of an ILP problem with more constraints than the ILP problem whose solution is \( x(t) \). However, it is not necessarily true that \( S(y(t+1)) \geq S(y(t)) \). Nevertheless, we empirically observed that in most cases we do obtain the inequality \( S(y(t+1)) \geq S(y(t)) \). We can view the sequence \( y(1), y(2), \ldots, \hat{x} \) as a hierarchical segmentation set composed of gradually more refined segmentations. The hierarchical segmentation algorithm, derived from the cutting-plane method, is summarized in Algorithm-Box 2.

V. EXPERIMENTAL RESULTS

We present visual and quantitative results of our algorithm for five different datasets:

1) BSDS500 [6]: 200 training images, 100 validation images and 200 test images.
2) SBD [30]: Stanford Background Dataset, 715 outdoor scene images.
3) MSRC [31]: 591 images and 23 object classes. For ground truth images, we used the ones described in [15].
5) Weizmann Two-Objects dataset [33]: 200 images; 100 images with a single object and 100 images with two objects.

We also show the effect of the efficient ILP algorithm on the segmentation procedure.

A. Extracting superpixels and local weights

We used a state-of-the-art superpixel map suggested by Arbelaez et al. [6]. The first step is shifting from pixels to superpixels. The Oriented Watershed Transform (OWT) [6] is used to produce an over-segmentation of the image into a few hundred superpixels. It was observed in [34] that on average it is enough to represent an image with a few hundred superpixels to obtain almost full boundary recall for low enough thresholds.

For each pair of spatially adjacent superpixels we need to obtain (based on the image content) the probability that they are part of the same segment. Arbelaez et al. [6] proposed a similarity measure that combines multiple local cues into a globalization framework based on spectral clustering. The similarity measure takes the form of a logistic-regression that is optimized using an annotated training set. The outcome of this approach is an OWT superpixel map in which each arc pixel (a pixel separating two neighboring superpixels) has a score of being a boundary pixel (a pixel separating two neighboring segments). They refer to this score as the ‘globalized probability of boundary’ (gPb-owt) [35]. This pixel-level score can be converted into a score between adjacent superpixels by
averaging all the scores of the pixels on the corresponding arc. The values of the gPb-owt score increase monotonically with the probability that a segmentation boundary exists but they are not probabilities in the strict sense. Monotonicity is enough for agglomerative clustering that iteratively merges the most similar regions [6]. However, for our approach which avoids agglomerative clustering and is based instead on a global optimization, we need the score to have a strictly probabilistic interpretation. To convert the gPb-owt score of an arc into a probability to be on a segment boundary, we apply the following procedure. For each arc pixel, using a ground-truth annotation, we can check whether it is on a segment boundary or not. Next, for each value of the gPb-owt score we compute the relative number of arc pixels that have that gPb-owt score and are part of a segmentation boundary. The result of this analysis, performed using the training part of Weizmann database [33], is shown in Fig. 2. As can be seen, the gPb-owt score indeed increases monotonically but it does not coincide with the exact boundary probability. The graph in Fig. 2 can be used to convert the gPb-owt score into meaningful probability values. Using Eq. (4), the probabilities are converted into the weights that are used for the ILP optimization (6) to obtain the final image segmentation.

Although beyond the scope of this work, it should be noted that there are many other features [7], [34], [5] and learning methods [14], [27] to compute a similarity measure between two neighboring superpixels. Our efficient ILP optimization procedure is also applicable for all these cases.

B. Segmentation results on BSDS500

Before applying our method on the test portion of the BSDS500 dataset, we converted the gPb-owt scores [6] into probabilities based on the train set of the BSDS500 as explained in section 4.1. We used several standard methods for objective segmentation evaluation: the probabilistic Rand index (PRI) [36], the variation of information (VOI) [37], the boundary-based F-measure [38] and the Covering score.

Using the training set we chose the parameters set that had the highest F-measure scores for each algorithm. Using the same parameter set, all four measures mentioned above were recorded for the testing set. The results for our algorithm and the UCM [6] are shown in Table I. Table I shows that the GDIS also outperforms two other recently introduced approximate graph optimization methods [16], [14]. Compared to the UCM, the GDIS generated similar results on F and

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Covering</th>
<th>VOI</th>
<th>PRI</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDIS</td>
<td>0.59</td>
<td>1.95</td>
<td>0.83</td>
<td>0.73</td>
</tr>
<tr>
<td>Positive connected components</td>
<td>0.53</td>
<td>1.89</td>
<td>0.74</td>
<td>0.65</td>
</tr>
<tr>
<td>UCM [6]</td>
<td>0.58</td>
<td>1.97</td>
<td>0.83</td>
<td>0.73</td>
</tr>
<tr>
<td>PlanarCC [16]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.72</td>
</tr>
<tr>
<td>Kim [14]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.70</td>
</tr>
</tbody>
</table>

TABLE I
Comparison of our method and the UCM segmentations on the BSDS500 test set using four measures: F, PRI, VOI (lower is better) and Covering (higher is better).

Fig. 4. Example of an image from the Weizmann Two-Objects dataset. The F-measure scores of the CPMC and GDIS algorithms are 0.771 and 0.834 respectively.
Fig. 5. Histogram of the ILP problem size as a fraction of the number of superpixels in the image. Statistics for the BSDS500 test part.

Fig. 6. Average running time statistics, as a function of the size of the largest positively connected component of the superpixel map, computed on the BSDS500 test part.

PRI and only slightly better results with respect to VOI and Covering. We also show the results obtained by applying only the positive connected component procedure which is the first step of the GDIS algorithm. These results are, by definition, coarser than those obtained by the GDIS algorithm. To alleviate any confusion, when comparing the UCM results to the ones mentioned in [6], in [6] the different measures were recorded while optimizing for each measure separately using different result sets. Sample results for the BSDS500 test set are shown in Fig. 3. The fact that the GDIS results are very close to those of the UCM on the BSDS500 is because we used the same superpixel maps and the same underlying similarity score that was tuned on the BSDS500 dataset. It should be emphasized that unlike the UCM which is based on greedy iterative merging, we obtain the exact global maximum.

C. Segmentation results on the MSRC, SBD & VOC2012 datasets

Following Ren and Shakhnarovich [9], in these experiments we only used the BSDS500 train part for training. We report the results in a way similar to [6] by optimizing the scale parameter in two ways: jointly for the entire training dataset (ODS), and separately for each test image (OIS). For the GDIS we tune the score bias parameter which “sets” the segmentation scale. We compared our results to UCM algorithm [6] and to ISCRA algorithm [9] that recently introduced a cascaded boundary classifier trained sequentially using different color, textures and geometric appearance features. The results for the three datasets are shown in Tables II, III and IV. It is clear that overall the proposed GDIS algorithm performs favorably in comparison to UCM on all 3 datasets. We achieved similar results to the ISCRA algorithm [9] on the SBD and MSRC datasets and better results than [9] on the VOC2012 dataset.

### Table II

<table>
<thead>
<tr>
<th>Method</th>
<th>Covering</th>
<th>VOI</th>
<th>PRI</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCM</td>
<td>0.74</td>
<td>0.65</td>
<td>1.09</td>
<td>1.28</td>
</tr>
<tr>
<td>GDIS</td>
<td>0.74</td>
<td>0.65</td>
<td>1.07</td>
<td>1.22</td>
</tr>
<tr>
<td>ISCRA</td>
<td>0.74</td>
<td>0.66</td>
<td>1.02</td>
<td>1.17</td>
</tr>
</tbody>
</table>

**Table II**

**RESULTS FOR MSRC: OPTIMAL SCALE PER (TEST) IMAGE, ODS: OPTIMAL SCALE FOR ENTIRE TEST DATA SET.**

### Table III

<table>
<thead>
<tr>
<th>Method</th>
<th>Covering</th>
<th>VOI</th>
<th>PRI</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCM</td>
<td>0.62</td>
<td>0.56</td>
<td>1.42</td>
<td>1.66</td>
</tr>
<tr>
<td>GDIS</td>
<td>0.65</td>
<td>0.58</td>
<td>1.40</td>
<td>1.61</td>
</tr>
<tr>
<td>ISCRA</td>
<td>0.66</td>
<td>0.60</td>
<td>1.39</td>
<td>1.57</td>
</tr>
</tbody>
</table>

**Table III**

**RESULTS FOR SBD: OPTIMAL SCALE PER (TEST) IMAGE, ODS: OPTIMAL SCALE FOR ENTIRE TEST DATA SET.**

### Table IV

<table>
<thead>
<tr>
<th>Method</th>
<th>Covering</th>
<th>VOI</th>
<th>PRI</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCM</td>
<td>0.59</td>
<td>0.57</td>
<td>1.58</td>
<td>1.59</td>
</tr>
<tr>
<td>GDIS</td>
<td>0.66</td>
<td>0.60</td>
<td>1.14</td>
<td>1.22</td>
</tr>
<tr>
<td>ISCRA</td>
<td>0.57</td>
<td>0.56</td>
<td>1.64</td>
<td>1.65</td>
</tr>
</tbody>
</table>

**Table IV**

**RESULTS FOR VOC2012: OPTIMAL SCALE PER (TEST) IMAGE, ODS: OPTIMAL SCALE FOR ENTIRE VALIDATION DATA SET.**

### Table V

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDIS</td>
<td>0.84</td>
</tr>
<tr>
<td>CPSC [7]</td>
<td>0.84</td>
</tr>
<tr>
<td>UCM [6]</td>
<td>0.78</td>
</tr>
<tr>
<td>Alp [5]</td>
<td>0.68</td>
</tr>
<tr>
<td>SWA V1 [39]</td>
<td>0.66</td>
</tr>
<tr>
<td>SWA V2 [39]</td>
<td>0.61</td>
</tr>
<tr>
<td>Mean Shift [2]</td>
<td>0.61</td>
</tr>
<tr>
<td>N-Cuts [40]</td>
<td>0.58</td>
</tr>
</tbody>
</table>

**Table V**

**AVERAGE SINGLE SEGMENT COVERAGE TEST RESULTS ON THE WEIZMANN TWO-OBJECTS DATA SET. HIGHER IS BETTER.**

D. Segmentation results on the Weizmann Two-Objects Dataset

The Weizmann dataset consists of gray-valued images having either a single or two foreground objects. The goal is to generate coverage of the entire spatial support of the object.
Fig. 7. From left to right: original image followed by three intermediate valid segmentations created as a result of adding more cutting plane constraints (moving to the right). The intermediate segmentations become more refined as we add constraints. Example images were taken from the BSDS500.

in the image using a single segment, and as accurately as possible.

We used the single object images as our training set for learning the true probability mapping as explained above. The two object images were used as the test set. We followed the testing procedure described in [5] using their publicly available testing code [33]. We also compared our algorithm to the CPMC algorithm using their publicly available code [7]. The segmentation results were assessed by their consistency with ground truth segmentation using the F-measure [38]. Following [5] and [7], the results are reported using the average best F-measure criterion. For the GDIS we created 20 segmentations for each image by linearly varying the value of $\lambda$ (see Eq. (7)) from 0 to 3. For the UCM we created 100 segmentations using different scale factors. For the CPMC we used all available segments produced with their public available code [7]. For each ground truth object-pair the segmentation that best fit the foreground with respect to F-measure was selected and the value of the similarity was recorded. These top similarities were then averaged. The averaged results for both objects are reported in Table V. As can be seen, the GDIS algorithm scored the highest (obtaining same result as the CPMC algorithm). Note that the only differences between the implementation of our optimization approach and the UCM [6] are the similarity weight scaling (Fig. 2) and the global ILP optimization that we apply instead of the greedy superpixel merging procedure in [6]. Fig. 4 shows an example of segmentation results on an image from
the Weizmann dataset.

E. Efficiency analysis of the ILP algorithm

Here we present an efficient method for solving the ILP problem (6) by breaking it down into small sub-problems. Next, we demonstrate the efficiency contribution of these two components when applied to an image segmentation task. The complexity of our ILP algorithms depends on the size of the largest component in the decomposition. We computed the following statistics. We constructed the positively connected components and measured the size of the largest component. Fig. 5 shows, for images in the BSDS500 dataset, a count histogram of the relative size of the largest component (i.e. the relationship between the number of superpixels in the largest component and the number of superpixels in the images). As can be seen, the average size of the largest component is smaller than the number of superpixels in the images. The average size of the superpixel graph for the BSDS500 is 1160 whereas the average number of superpixels in the largest component is 830.

Fig. 6 shows the runtime statistics (measured on an Intel Duo-Core, 2.5GHz, 4GB RAM) of the ILP Gurobi software combined with the cutting-plane method applied to positively connected components taken from the BSDS500 images. Note that no multi-processing optimization was done so effectively the mean running time of GDIS on a single core was 1.6 seconds.

F. Cutting Plane Hierarchical Segmentation Results

The cutting plane algorithm produces an intermediate non-consistent solution. Fig. 7 demonstrates on several examples from the BSDS500 the valid segmentations produced by computing the connected components of the intermediate solutions. For each intermediate segmentation $y = \{y_{ij}\}$, the ILP score $S(y) = \sum_{i<j} w_{ij} y_{ij}$ was less than the score of the optimal solution. As a result of adding more constraints, the intermediate segmentations usually become more refined at each iteration and as such can be considered as a hierarchical map of segmentations.

G. Failure Cases

Fig. 8 shows several failure examples of our method. We recall that the underlying arc scores are provided by gPb-owt. When gPb-owt produces either too coarse (Fig. 8a) or too fine detections (Fig. 8c) our method fails at these tasks since the connected components are either too coarse and cannot capture the image objects or too fine, over breaking the scene. This failures emphasise that our method is limited by the underlying arc probabilities produced at the preprocessing step.

VI. DISCUSSION AND CONCLUSION

In this study we presented a probabilistic modeling of image segmentation based on correlation clustering and an efficient algorithm of the ILP optimization problem. We showed that, given local scores on a map of several hundred superpixels, finding the global segmentation that is the most consistent with the local evidence, is still tractable. We also showed that by using cutting-plane method to solve the ILP optimization we automatically obtain a hierarchical segmentation structure. The coarsest segmentations are the positively connected components described in Section 3 and the most refined segmentation is the solution of the ILP problem. We then applied the method to a dataset with manually segmented images and compared its performance to several recent algorithms. In recent years there was a lot of effort to extract better region based features between neighboring superpixels and developing novel machine learning methods to extract better informative similarity weights from these features. In this study we focused on the global optimization aspect of image segmentation, based on a given superpixel map and local similarly scores between adjacent superpixels. In our implementation we used the probabilistic information score extracted from the gPb-owt score. Exploiting additional content based features from the superpixels as shown in [7], [34], [5] would also be beneficial. The ideas presented in this study can be combined with recent approaches (e.g. [14], [7], [34]) to further improve segmentation and object detection results.

REFERENCES


Amir Alush received the B.Sc. degree (cum laude) in bio-medical engineering in 2007 and the M.Sc. degree in bio-medical engineering in 2009 both from the Tel-Aviv university. He is currently a Ph.D. student at the Bar-Ilan University working on discrete optimization problems and their applications in image and video data and medical imaging.

Jacob Goldberger received the B.Sc. degree (cum laude) in mathematics and computer science from the Bar-Ilan University, Ramat Gan, Israel, the M.Sc. degree (cum laude) in mathematics, and the Ph.D. degree in Engineering, both from the Tel-Aviv University, Tel-Aviv, Israel. In 2004 he joined the Faculty of Engineering, Bar-Ilan University, Ramat Gan, Israel, where he is currently an associated professor. His main research interests include machine learning with applications to information theory, computer vision, medical imaging, speech recognition and natural language processing.