A Graphical Model for Simultaneous Partitioning and Labeling

Philip J. Cowans
Cavendish Laboratory, University of Cambridge,
Cambridge, CB3 0HE, United Kingdom
pjc51@cam.ac.uk

Martin Szummer
Microsoft Research
Cambridge, CB3 0FB, United Kingdom
szummer@microsoft.com

Abstract

In this work we develop a graphical model for describing probability distributions over labeled partitions of an undirected graph which are conditioned on observed data. We show how to efficiently perform exact inference in these models, by exploiting the structure of the graph and adapting the sum-product and max-product algorithms. We demonstrate our approach on the task of segmenting and labeling hand-drawn ink fragments, and show that a significant performance increase is obtained by labeling and partitioning simultaneously.

1 INTRODUCTION

Probabilistic models are usually defined over the Cartesian product of a number of discrete or continuous one-dimensional spaces. For example, models performing joint binary classification of N objects are defined over $\{-1,+1\}^N$. While in many cases it is intractable to explicitly enumerate all possible configurations, in the case of graphical models where the probability distribution factors according to the structure of an undirected graph, message passing techniques such as the sum-product and max-product algorithms can be used to render the computation feasible.

In this work, we extend the graphical model formalism to the case of probability distributions defined over labeled partitions of an undirected graph; in other words, possible divisions of the graph into sets of vertices referred to as *parts*, where each part is assigned a label. An example of a labeled partition is given in Figure 1. Note that the number of parts varies between partitions and is usually unknown in advance. Our method represents partitions directly, rather than incorporating part identifiers into the labels. We thereby avoid the degeneracy that different permutations of part identifiers represent the same partition (see Section 3.4 for a comparison of the two approaches). In this work we restrict ourselves to binary labels, but the method can be generalized straightforwardly to larger label sets. Conversely, unlabeled partitioning may be viewed as a special case with just one label. Our model is similar to the Conditional Random Field (CRF) [2], and allows the probability distribution to be conditioned on arbitrary observed data. This model is widely applicable to joint segmentation and classification tasks, which are common in computer vision, handwriting recognition, speech and natural language processing. The Markov Random Field (MRF), which is an undirected graphical model whose potential functions do not depend on observed data, is for the purposes of this paper a special case of the CRF, and can also be extended in the way described below.

Previously, probabilistic models have been used for graph partitioning, but by using Monte Carlo techniques rather than exact inference [1]. Liu [4] has performed partitioning, but not using a probabilistic framework. Other work [7] has extended the CRF to perform multiple inference tasks simultaneously, but has not considered partitioning of non-linear graphs.

We begin by describing the full probabilistic model, then consider representations for labeled partitions and efficient algorithms for performing the necessary

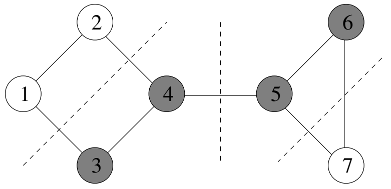


Figure 1: An example of a labeled partition. Vertices are partitioned as follows: $(1,2,+)$, $(3,4,-)$, $(5,6,-)$, $(7,+)$, where the last symbol in each group indicates the label assigned to that part.

inference tasks. Finally, we describe the application of our model to the task of parsing hand-drawn ink diagrams.

2 THE PROBABILISTIC MODEL

Let $\mathcal{G}$ be an undirected graph consisting of vertices $\mathcal{V}$ and edges $\mathcal{E}$. We assume that $\mathcal{G}$ is triangulated, so that every cycle of length greater than three is spanned by a chord. This can always be achieved by adding edges, but usually at the expense of increasing the maximum clique size, and therefore computational complexity.

Let S be a partition of $\mathcal{G}$, that is, a set of non-empty subsets of $\mathcal{V}$, such that each vertex in $\mathcal{V}$ is a member of precisely one subset. Each subset is referred to as a *part* of $\mathcal{G}$. In this paper, the term *partition* will always refer to a *contiguous* partition:

Definition 1. *A partition of $\mathcal{G}$ is* ***contiguous*** *if and only if all parts are internally connected. In other words, if i and j are vertices contained within the same part, there exists a path on $\mathcal{G}$ between i and j entirely contained within that part.*

A labeled partition of $\mathcal{G}$ is represented by $\mathcal{Y} = (\mathsf{S}, \boldsymbol{y})$, where S describes the partition and $\boldsymbol{y} \in \{-1,+1\}^M$ is a vector containing the labels associated with each part. For example, a partition of three elements into two parts could be $\mathsf{S} = \{\{1\}\{2,3\}\}, \boldsymbol{y} = [+1,-1]$. Let $\mathbb{Y}$ be the set of all possible labeled partitions of $\mathcal{G}$. Note that M, the length of $\boldsymbol{y}$, is dependent on S. Let t_i be the index of the part to which vertex i is assigned, so that y_{t_i} is the label given to that vertex.

In this work, the conditional probability distribution over $\mathbb{Y}$ has the form $P(\mathcal{Y} \mid \boldsymbol{x}, \boldsymbol{\theta}) =$

$$\frac{1}{Z(\boldsymbol{\theta})} \prod_{i \in \mathcal{V}} \psi_i^{(1)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}) \prod_{i,j \in \mathcal{E}} \psi_{ij}^{(2)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}), \quad (1)$$

where $\boldsymbol{x}$ is the observed data, $\boldsymbol{\theta}$ is a vector representing the model parameters collectively, and $Z(\boldsymbol{\theta})$ is a normalization constant. $\psi_i^{(1)}$ are unary potentials defined for each vertex, and $\psi_{ij}^{(2)}$ are pairwise potentials defined for each edge. The unary potentials introduce a data-dependent bias towards assigning one label or the other to each vertex. The pairwise potentials model the compatibility between the parts and labels of neighboring vertices, and are also data dependent. The dependence of these potentials on $\boldsymbol{x}$ is through feature vectors, $\boldsymbol{g}_i$ and $\boldsymbol{f}_{ij}$, defined for each vertex i and edge (i,j) respectively. The potentials then have the form

$$\psi_i^{(1)}(\mathcal{Y}, \boldsymbol{x}, \boldsymbol{\theta}) = \begin{cases} \phi(\boldsymbol{w}_+ \cdot \boldsymbol{g}_i(\boldsymbol{x})) & \text{if } y_{t_i} = +1 \\ \phi(\boldsymbol{w}_- \cdot \boldsymbol{g}_i(\boldsymbol{x})) & \text{if } y_{t_i} = -1 \end{cases}, \quad (2)$$

where $\phi(\cdot)$ is a non-linear mapping, and $\boldsymbol{w}_+$ and $\boldsymbol{w}_-$ are vectors of feature weights depending on the label of the appropriate vertex. In this work, we will always use an exponential non-linearity, $\phi : x \mapsto \exp(x)$, although in general other functions may be used. The pairwise potentials are defined by

$$\psi_{ij}^{(2)}(\mathcal{Y}, \boldsymbol{x}, \boldsymbol{\theta}) = \begin{cases} \phi(\boldsymbol{v}_{\text{ss}} \cdot \boldsymbol{f}_{ij}(\boldsymbol{x})) & \text{if } t_i = t_j,\ y_{t_i} = y_{t_j} \\ \phi(\boldsymbol{v}_{\text{sd}} \cdot \boldsymbol{f}_{ij}(\boldsymbol{x})) & \text{if } t_i \neq t_j,\ y_{t_i} = y_{t_j} \\ \phi(\boldsymbol{v}_{\text{dd}} \cdot \boldsymbol{f}_{ij}(\boldsymbol{x})) & \text{if } t_i \neq t_j,\ y_{t_i} \neq y_{t_j} \end{cases} \quad (3)$$

where $\boldsymbol{v}_{\text{ss}}$, $\boldsymbol{v}_{\text{sd}}$ and $\boldsymbol{v}_{\text{dd}}$ are vectors of feature weights to be used when i and j belong to the same part, different parts with the same label, and different parts with different labels respectively. The fourth case, corresponding to vertices with different labels in the same part, does not occur by definition. The parameters in $\boldsymbol{\theta}$ are therefore $(\boldsymbol{w}_+, \boldsymbol{w}_-, \boldsymbol{v}_{\text{ss}}, \boldsymbol{v}_{\text{sd}}, \boldsymbol{v}_{\text{dd}})$. Note that there is a redundancy in the weight vectors. In practice, $\boldsymbol{w}_-$ and $\boldsymbol{v}_{\text{dd}}$ were constrained to be $\mathbf{0}$.

2.1 TRAINING

The overall goal of the model above is to predict labeled partitions of unseen data. In order to do this, we must first estimate the model parameters, $\boldsymbol{\theta}$. These parameters are learned from example data. Given a labeled training examples, $(\boldsymbol{x}, \mathcal{Y})$, the posterior probability of the parameters is given using Bayes' rule,

$$P(\boldsymbol{\theta} \mid \boldsymbol{x}, \mathcal{Y}) \propto P(\mathcal{Y} \mid \boldsymbol{x}, \boldsymbol{\theta}) \cdot P(\boldsymbol{\theta}), \quad (4)$$

where $P(\boldsymbol{\theta})$ is a prior distribution over the weights. The model is trained by finding the maximum *a posteriori* weights using a quasi-Newton gradient ascent algorithm (specifically, the BFGS method). A significant advantage is that the model is convex in the parameters, meaning that we are guaranteed to find the global maximum using gradient ascent. The gradient of the log posterior, $\mathcal{LP}$, with respect to a parameter θ_k is given by

$$\frac{\partial}{\partial \theta_k} \mathcal{LP} = \sum_{i \in \mathcal{V}} \left(\frac{\partial}{\partial \theta_k} \log \psi_i^{(1)} - \left\langle \frac{\partial}{\partial \theta_k} \log \psi_i^{(1)} \right\rangle \right) + \frac{\partial}{\partial \theta_k} \log(\mathrm{P}(\boldsymbol{\theta})) \quad (5)$$

if θ_k is a parameter of the unary potentials. The gradients with respect to parameters of the pairwise potentials have a similar form. It is straigtforward to generalize this expression to handle multiple training examples. The brackets, $\langle \cdots \rangle$, in the second terms represent expectations with respect to the distribution over $\mathbb{Y}$ given by the current parameter values. This requires the computation of marginal probability distributions for individual vertices and pairs of vertices connected by an edge. Furthermore, the optimization algorithm needs to evaluate (1) explicitly, which in turn requires evaluation of the partition function,

$$Z(\boldsymbol{\theta}) = \sum_{\mathcal{Y}} \prod_{i \in \mathcal{V}} \psi_i^{(1)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}) \prod_{i,j \in \mathcal{E}} \psi_{ij}^{(2)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}) . \quad (6)$$

Both of these tasks involve summations over subsets of possible labeled partitions. This summation can be performed efficiently by message passing using a modified version of the sum-product algorithm. The details of this algorithm will be given in Section 3 below.

2.2 INFERENCE

In general, we are interested in using the trained model to group and label unseen data. This is achieved by finding the most probable configuration,

$$\mathcal{Y}^{\mathrm{MAX}} = \arg\max_{\mathcal{Y}} \prod_{i \in \mathcal{V}} \psi_i^{(1)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}) \prod_{i,j \in \mathcal{E}} \psi_{ij}^{(2)}(\mathcal{Y}, \boldsymbol{x}; \boldsymbol{\theta}) . \quad (7)$$

As with the partition function, this maximization can be performed efficiently using a version of the max-product algorithm.

3 OPERATIONS OVER LABELED PARTITIONS

In Section 2, it was shown that an important part of both the training and inference processes is the enumeration of all possible labeled partitions, in order to either sum or maximize over them. As in the more usual case of labeling vertices, explicit enumeration of all possible values is prohibitively expensive. However, as we show below, we are able to exploit the structure of the graph to significantly reduce the computational cost, rendering exact inference tractable in many cases. The derivation below follows the conditions for the possibility of local computation provided by Shenoy and Shafer [5]. An alternative derivation however is possible following Lauritzen [3].

If G is a subset of $\mathcal{V}$, we use $\mathcal{Y}_G \in \mathbb{Y}_G$ to denote a labeled partition of the corresponding induced subgraph. We define *consistency* as follows:

Definition 2. *Labeled partitions $\mathcal{Y}_G$ and $\mathcal{Y}_H$, of subgraphs G and H respectively, are* ***consistent****, denoted $\mathcal{Y}_H \backsim \mathcal{Y}_G$, if and only if:*

1. *All vertices appearing in $G \cap H$, are assigned the same label by $\mathcal{Y}_G$ and $\mathcal{Y}_H$, and*
2. *All pairs of vertices appearing in $G \cap H$ are in the same part in $\mathcal{Y}_G$ if and only if they are in the same part in $\mathcal{Y}_H$.*

The notation $\hat{\mathcal{Y}}_G(\mathcal{Y}_{G \cup H})$ is used to denote the unique labeled partition of G which is consistent with $\mathcal{Y}_{G \cup H}$. The maximal cliques of $\mathcal{G}$ are defined in the usual way, and are denoted $C_1, \ldots, C_N$. If b and t are two cliques, and b contains all vertices from t which appear in cliques other than t, then b is said to be a *branch* and t is the corresponding *twig*.

Following the framework of Shenoy and Shafer, we introduce the notion of a *valuation* ψ on a subset of $\mathcal{V}$. In the case of standard belief propagation, valuations are functions assigning a real, non-negative value to possible configurations of subsets of the variables. In this work, a valuation on a subset G will be defined as a function mapping $\mathbb{Y}_G$ to the non-negative real numbers. $\mathbb{V}_G$ is the set of all valuations on G. In the case where the valuation is over the whole of $\mathcal{G}$, the range of the valuation will be interpreted as being proportional the probability of the corresponding labeled partition. In the case of valuations defined over subsets of $\mathcal{V}$ the valuations are referred to as potentials of which those defined in (1) are an example. We define two operations on valuations:

1. **Combination:** Suppose G and H are subsets of $\mathcal{V}$ and ψ_G and ψ_H are valuations on those subsets. The operation of combination defines a mapping $\otimes : \mathbb{V}_G \times \mathbb{V}_H \mapsto \mathbb{V}_{G \cup H}$, such that

$$\psi_G \otimes \psi_H (\mathcal{Y}_{G \cup H}) \triangleq \psi_G \left(\hat{\mathcal{Y}}_G (\mathcal{Y}_{G \cup H}) \right) \cdot \psi_H \left(\hat{\mathcal{Y}}_H (\mathcal{Y}_{G \cup H}) \right) . \quad (8)$$

2. **Marginalization:** Suppose G and H are subsets of $\mathcal{V}$ such that $G \subseteq H$, and ψ_G and ψ_H are valuations as before. Marginalization is a mapping $\downarrow : \mathbb{V}_H \mapsto \mathbb{V}_G$ such that

$$\psi_H^{\downarrow G}(\mathcal{Y}_G) \triangleq \sum_{\mathcal{Y}_H \backsim \mathcal{Y}_G} \psi_H (\mathcal{Y}_H) . \quad (9)$$

A valuation over the whole graph is said to factor if it can be written as the combination of valuations on the cliques,

$$\psi(\mathcal{Y}) = \bigotimes_{i=1}^{N} \psi_i (\mathcal{Y}_i) , \quad (10)$$

where i runs over the cliques in $\mathcal{G}$. As combination allows products of valuations over subsets of a clique to be written in terms of a single valuation over the whole clique, the model given in (1), excluding the partition function, is in this form. Before demonstrating the possibility of efficient local computation, we first demonstrate that three axioms due to Shenoy and Shafer are satisfied:

Axiom 1. Commutativity and associativity of combination. *If G, H and K are subsets of $\mathcal{V}$, for any valuations ψ_G, ψ_H and ψ_K, we have $\psi_G \otimes \psi_H = \psi_H \otimes \psi_G$ and $\psi_G \otimes (\psi_H \otimes \psi_K) = (\psi_G \otimes \psi_H) \otimes \psi_K$.*

Proof. Follows directly from the definition of combination. □

Axiom 2. Consonance of marginalization, *If G, H and K are subsets of $\mathcal{V}$ such that $K \subseteq G \subseteq H$, for any valuations ψ_G, ψ_H and ψ_K,*

$$\left(\psi_H^{\downarrow G}\right)^{\downarrow K} = \psi_H^{\downarrow K}. \tag{11}$$

Proof. Writing the marginalization explicitly,

$$\begin{aligned}\left(\psi_H^{\downarrow G}\right)^{\downarrow K} &= \sum_{\mathcal{Y}_G \backsim \mathcal{Y}_K} \sum_{\mathcal{Y}_H \backsim \mathcal{Y}_G} \psi_H\left(\mathcal{Y}_H\right) \\ &= \sum_{\mathcal{Y}_H \backsim \mathcal{Y}_K} \psi_H\left(\mathcal{Y}_H\right) = \psi_H^{\downarrow K}, \qquad (12)\end{aligned}$$

where the second line follows as for any $\mathcal{Y}_H \backsim \mathcal{Y}_K$ there is a unique $\mathcal{Y}_G$ such that $\mathcal{Y}_G \backsim \mathcal{Y}_K$ and $\mathcal{Y}_H \backsim \mathcal{Y}_G$, and for any $\mathcal{Y}_H \not\backsim \mathcal{Y}_K$, no such $\mathcal{Y}_G$ exists. □

Axiom 3. Distributivity of marginalization over combination, *If G and H are subsets of $\mathcal{V}$, for any valuations ψ_G and ψ_H, $(\psi_G \otimes \psi_H)^{\downarrow G} = \psi_G \otimes (\psi_H^{\downarrow G\cap H})$.*

Proof. Performing an explicit expansion gives

$$\begin{aligned}(\psi_G \otimes \psi_H)^{\downarrow G} &= \sum_{\mathcal{Y}_{G\cup H} \backsim \mathcal{Y}_G} \psi_G\left(\hat{\mathcal{Y}}_G\left(\mathcal{Y}_{G\cup H}\right)\right) \cdot \\ &\qquad \psi_H\left(\hat{\mathcal{Y}}_H\left(\mathcal{Y}_{G\cup H}\right)\right) \\ &= \psi_G\left(\mathcal{Y}_G\right) \cdot \sum_{\mathcal{Y}_{G\cup H} \backsim \mathcal{Y}_G} \psi_H\left(\hat{\mathcal{Y}}_H\left(\mathcal{Y}_{G\cup H}\right)\right) \\ &= \psi_G\left(\mathcal{Y}_G\right) \cdot \sum_{\mathcal{Y}_H \backsim \hat{\mathcal{Y}}_{G\cap H}(\mathcal{Y}_G)} \psi_H\left(\mathcal{Y}_H\right), \qquad (13)\end{aligned}$$

which is equal to $\psi_G \otimes (\psi_H^{\downarrow G\cap H})$ by definition. □

3.1 THE SUM-PRODUCT ALGORITHM

In the next two sections we develop an extension of the sum-product algorithm suitable for probability distributions over partitions. As with the more usual form of this algorithm, our method exploits the known structure of $\mathcal{G}$ by passing messages containing the results of local computations. Our goal is to compute sums over a subset of all possible partitions, such as those needed for the partition function, as given in (6). This task should be contrasted with that of the usual sum-product algorithm [3], which sums over assignments of labels to the vertices. Since we sum over a different domain we will need to modify the messages passed and the ranges of summation. Later, in Section 3.3, we will also adapt the max-product algorithm for labeled partitions. Consider a sum of form:

$$f_s\left(\mathcal{Y}_1\right) = \sum_{\mathcal{Y} \backsim \mathcal{Y}_1} P^*\left(\mathcal{Y}\right) = \left(P^*\left(\mathcal{Y}\right)\right)^{\downarrow C_1}, \tag{14}$$

where P^* is a (possibly unnormalized) probability distribution[1] over labeled partitions of $\mathcal{G}$. Let the cliques be numbered $C_1, \ldots, C_N$, such that C_1 is the clique containing the vertices onto which we wish to marginalize and such that for all k, C_k is a twig in the graph $C_1 \cup C_2 \cup \ldots \cup C_k$. Such an ordering is always possible if $\mathcal{G}$ is triangulated. According to Axiom 2, this can be expressed as

$$\begin{aligned}f_s\left(\mathcal{Y}_1\right) &= \left(\left(P^*\left(\mathcal{Y}\right)\right)^{\downarrow \mathcal{V}\setminus C_N}\right)^{\downarrow C_1} \\ &= \left(\left(\bigotimes_{i=1}^{N} \psi_i\left(\mathcal{Y}_i\right)\right)^{\downarrow \mathcal{V}\setminus C_N}\right)^{\downarrow C_1} \qquad (15) \\ &= \left(\left(\bigotimes_{i=1}^{N-1} \psi_i\left(\mathcal{Y}_i\right)\right) \otimes \left(\psi_N\left(\mathcal{Y}_N\right)^{\downarrow C_N \cap \mathcal{V}}\right)\right)^{\downarrow C_1}.\end{aligned}$$

In the last step, Axiom 3 has been used. C_N is a twig by construction. Let C_B be a corresponding branch, then $C_N \cap \mathcal{V} = C_N \cap C_B$, hence

$$\begin{aligned}f_s\left(\mathcal{Y}_1\right) = \Bigg(\Bigg(\bigotimes_{\substack{i=1\\ i\neq B}}^{N-1} &\psi_i\left(\mathcal{Y}_i\right)\Bigg) \otimes \psi_B\left(\mathcal{Y}_B\right) \otimes \\ &\left(\psi_N\left(\mathcal{Y}_N\right)^{\downarrow C_N \cap C_B}\right)\Bigg)^{\downarrow C_1}. \qquad (16)\end{aligned}$$

In other words, the problem can be converted to an equivalent marginalization over a graph with one less clique in which the potential for C_B has been replaced according to:

$$\psi_B \leftarrow \psi_B \otimes \left(\psi_N^{\downarrow C_N \cap C_B}\right). \tag{17}$$

By repeatedly eliminating cliques in this way we can systematically remove cliques until there is only one remaining, C_1. Any further summation which is required (either to give marginals over a smaller subset of vertices, or to calculate the partition function) can be performed explicitly.

3.2 MESSAGE PASSING

The result of the elimination illustrated in (17) can be interpreted in terms of a message passed from C_N

[1] While our method is applicable to any summation of this form, we will focus on the application to probability distributions in this paper.

to the rest of the graph. Messages are passed between cliques along edges in a *junction tree* [3]. Let $\mu_{i\to j}(\mathcal{Y}_j)$ be the message passed from C_i to C_j. The form of the message is a list of labeled partitions of the intersection $C_i \cap C_j$, each of which has an associated scalar value. The messages are updated iteratively according to the rule:

$$\mu_{i\to j}(\mathcal{Y}_j) \leftarrow \sum_{\mathcal{Y}_i \backsim \mathcal{Y}_j} \psi_i(\mathcal{Y}_i) \prod_{\substack{k\in\mathcal{N}(i)\\ k\neq j}} \mu_{k\to i}(\mathcal{Y}_i), \qquad (18)$$

with the outgoing messages from a clique being updated once all incoming messages from the other neighboring cliques $\mathcal{N}(\cdot)$ have been received. As the junction tree has no cycles, this process will terminate after a finite number of iterations. Having updated all of the messages, it is then possible to find f_s using

$$f_s(\mathcal{Y}_1) = \psi_1(\mathcal{Y}_1) \prod_{k\in\mathcal{N}(1)} \mu_{k\to 1}(\mathcal{Y}_1). \qquad (19)$$

Having defined the algorithm formally, it is useful to also give an intuitive interpretation. The message passed from C_i to C_j can be interpreted as a statement summarizing the values of the 'upstream' potentials for labeled partitions which are consistent with each labeled partition of the separator between C_i and C_j. See Figure 2 for an example of the message passing process. As is the case with the usual form of the sum-product algorithm, the same messages are used in computing different marginals. Marginal distributions for all cliques can be found simultaneously with a single bidirectional pass of the message update rule.

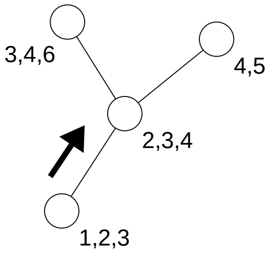
3.3 THE MAX-PRODUCT ALGORITHM

Just as is the case for the usual form of the sum-product algorithm, it is possible to replace the summation in (14) with a maximization to obtain the max-product algorithm. This is equivalent to a redefinition of marginalization to represent the maximum valuation consistent with the sub-partition rather than the sum over all valuations. This algorithm is used to compute maximizations, for example the configuration of C_1 in the most probable labeled partition,

$$\mathcal{Y}_1^{\mathrm{MAX}} = \arg\max_{\mathcal{Y}_1} \max_{\mathcal{Y}\backsim\mathcal{Y}_1} P^*(\mathcal{Y}). \qquad (20)$$

In the context of probabilistic inference, this is necessary when searching for the most probable configuration. Message passing is done in the same way as described above, with a modified message update rule.

$$\mu_{i\to j}(\mathcal{Y}_j) \leftarrow \max_{\mathcal{Y}_i \backsim \mathcal{Y}_j} \psi_i(\mathcal{Y}_i) \prod_{\substack{k\in\mathcal{N}(i)\\ k\neq j}} \mu_{k\to i}(\mathcal{Y}_i). \qquad (21)$$



(a)

Potential	Value
$\psi_{ij}(t_i = t_j, y_i = y_j)$	0.6
$\psi_{ij}(t_i \neq t_j, y_i = y_j)$	0.4
$\psi_{ij}(t_i \neq t_j, y_i \neq y_j)$	0.2

(b)

Partition	Label	Value
(123)	+	0.216
(123)	-	0.216
(12)(3)	+, +	0.096
(12)(3)	+, -	0.024
⋮	⋮	⋮
(1)(2)(3)	-,-,-	0.064

(c)

Partition	Label	Value
(23)	+	0.336
(23)	-	0.336
⋮	⋮	⋮
(2)(3)	-,-	0.272

(d)

Figure 2: An example of message passing. (a) The junction tree corresponding to $\mathcal{G}$. (b) The potentials, in this case uniform and independent of data for clarity. (c) The clique potential for the clique consisting of vertices 1, 2 and 3. (d) The message passed from (123) to (234), concerning labeled partitions of vertices 2 and 3.

Having updated all of the messages, $\mathcal{Y}_1^{\mathrm{MAX}}$ can be found using

$$\mathcal{Y}_1^{\mathrm{MAX}} = \arg\max_{\mathcal{Y}_1} \psi_1(\mathcal{Y}_1) \prod_{k\in\mathcal{N}(1)} \mu_{k\to 1}(\mathcal{Y}_1). \qquad (22)$$

To find the global maximum configuration, we repeat the above for all possible roots, and reconstruct the global partition as the union of the local configurations (which will be consistent with one another).

Again, it is instructive to consider the intuitive meaning of the messages. In this case they can be interpreted as statements about the maximum value that can be achieved 'upstream' as a function of the clique separator configuration. When the next cluster computes its maximum configuration, the contribution of downstream potentials can therefore be incorporated from the messages rather than having to be recomputed from scratch each time.

3.4 EDGE-DUAL REPRESENTATION

Let us consider two alternative representations which cast the inference task so that it can be solved using the standard forms of the sum-product and max-product algorithms. In the first of these techniques, rather than working with partitions, a 'part ID' is assigned to each vertex. The corresponding partition is therefore defined so that contiguous regions with the same part ID are assigned to the same part. To allow for labeled partitions, a separate set of part IDs must be reserved for each label.

This approach has several problems. Firstly, we must ensure that enough part IDs are available to realize all possible partitions. Depending on the structure of $\mathcal{G}$, a lower bound on the minimum number required is the size of the largest clique. In practice the required number will be greater than this. In general, this means that inference will be significantly slower than the equivalent binary labeling problem.

A more serious drawback of this approach is that it introduces bias into the results; finding the most probable assignment of part IDs is not equivalent to finding the most probable partition; the latter marginalizes over the multiple assignments of IDs which correspond to the same partition.

An alternative representation which avoids these problems is to use indicator variables, $\check{\boldsymbol{x}}\left(\mathcal{Y}\right)$, for each edge in $\mathcal{G}$. For binary labels, these variables are over a set of six values: two states corresponding to segments belonging to the same part with each label, and four corresponding to different parts with all four combinations of labels. To construct a graphical model for these variables, we define the *edge-dual graph*:

Definition 3. *For any graph $\mathcal{G}$, the* ***edge-dual graph****, $\check{\mathcal{G}} = \left(\check{\mathcal{V}}, \check{\mathcal{E}}\right)$ contains one vertex for each edge in $\mathcal{G}$. Vertices in $\check{\mathcal{G}}$ are connected by an edge if and only if all vertices connected to their corresponding edges in $\mathcal{G}$ belong to the same clique.*

An example of an edge-dual graph is shown in Figure 3. Every labeled partition of $\mathcal{G}$ corresponds to a unique configuration of the edge-dual vertices, but there are configurations of the edge-dual vertices which do not correspond to labeled partitions. Hence,

Definition 4. *A configuration of the edge-dual vertices is* ***valid*** *if and only if it corresponds to a labeled partition of $\mathcal{G}$.*

Invalid configurations arise when pairwise constraints yield contradictory information; following one path between two vertices on $\mathcal{G}$ indicates that they are in the same part, whereas another path indicates that they are not, or their labels disagree. It is possible to establish the validity of a configuration using only calculations local to cliques on $\check{\mathcal{G}}$.

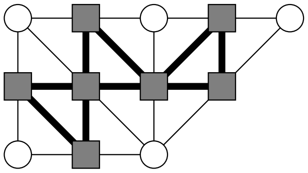


Figure 3: An example of an undirected graph (circular vertices and light lines) and the corresponding edge-dual graph (square vertices and heavy lines).

Suppose $P^*\left(\check{\boldsymbol{x}}\left(\mathcal{Y}\right)\right)$ is a probability distribution over labeled partitions of $\mathcal{G}$ as represented by the edge-dual variables. We are generally interested in operations such as the summation of P^* over all partitions. Rather than expressing the summation in terms of partitions, we can work directly with $\check{\boldsymbol{x}}$, provided that the summation is limited to those configurations which are valid. This can be achieved by introducing an indicator function, $\mathbb{I}\left(\check{\boldsymbol{x}}\right)$, which takes the value 1 if $\check{\boldsymbol{x}}$ is valid and 0 otherwise,

$$\sum_{\mathcal{Y}} P^*\left(\check{\boldsymbol{x}}\left(\mathcal{Y}\right)\right) = \sum_{\check{\boldsymbol{x}}} \mathbb{I}\left(\check{\boldsymbol{x}}\right) \cdot P^*\left(\check{\boldsymbol{x}}\right). \tag{23}$$

There is a one-to-one correspondence between cliques in $\mathcal{G}$ and $\check{\mathcal{G}}$, so functions which factor according to $\mathcal{G}$ also factor according to $\check{\mathcal{G}}$. If P^* factors, we can write

$$\sum_{\mathcal{Y}} P^*\left(\check{\boldsymbol{x}}\left(\mathcal{Y}\right)\right) = \sum_{\check{\boldsymbol{x}}} \left(\prod_i \mathbb{I}_i\left(\check{\boldsymbol{x}}_i\right) \cdot \check{\psi}_i\left(\check{\boldsymbol{x}}_i\right) \right), \tag{24}$$

where i ranges over the cliques of $\check{\mathcal{G}}$. In (24), the local nature of $\mathbb{I}$ has been used to factor it as well as P^*. The result is a sum over a function which factors according to $\check{\mathcal{G}}$, so it can be found using the standard sum-product algorithm.

As there is a one-to-one correspondence between valid edge-dual configurations, and labeled partitions of $\mathcal{G}$, this algorithm is in many respects equivalent to that presented in Section 3.1. However, in two important respects it is less efficient. Firstly, as the sum includes edge-dual configurations which are invalid, the number of terms in the sum is significantly greater. Secondly, it is necessary to determine the validity of the current configuration for each term, which introduces additional overhead. The algorithm presented in Section 3.1 may be regarded as an efficient implementation of this algorithm, where the validity of configurations is precomputed, and only those which are valid are included in the sum.

3.5 COMPLEXITY

The dominant factor in the complexity of the message passing algorithm is the time taken to process all possible partitions of the largest clique. Table 1 lists the number of possible configurations for the various cases. It can be seen from the table that the method described in Section 3 offers a considerable improvement in the complexity of the calculations.

Table 1: Sizes of the message tables for each of the methods. (a) Unlabeled Partitions (these are the Bell numbers). (b) Binary labeled partitions (c) Binary labeled edge-dual representation. (d) Binary labeled part IDs (lower bound).

	Clique Size					
	2	**3**	**4**	**5**	**6**	**n**
(a)	2	5	15	52	203	Bell no. B_n
(b)	*6*	*22*	*94*	*454*	*2430*	A001861 [6]
(c)	6	216	46656	6.0×10^7	4.7×10^{11}	$6^{n(n-1)/2}$
(d)	16	216	4096	1.0×10^5	3.0×10^6	$(2n)^n$

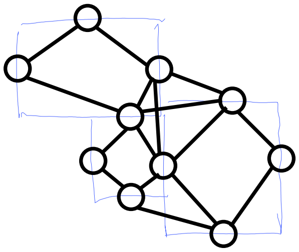


Figure 4: An example of an undirected graph constructed from the input data in which each vertex represents an ink fragment.

4 APPLICATION TO INK DATA

In this section we apply the algorithm developed in Section 3 to the task of parsing hand-drawn ink diagrams, focusing on the particular problem of grouping electronic ink strokes into perceptually salient objects and labeling the resulting groups. We demonstrate our approach on organization charts such as that shown in Figure 5, where objects are labeled as either containers or connectors. However, the method is general and may be applied to a wide variety of related problems.

4.1 PRE-PROCCESSING

The input data is a set of ink strokes, which may span multiple objects. The first stage is to split the strokes into fragments, which are assumed to belong to a single object, by dividing each stroke into sections which are straight to within a given tolerance.

Having fragmented the strokes, we build an undirected graph, $\mathcal{G}$, containing one vertex for each ink fragment (See Figure 4). This is the graph which will be partitioned to obtain the grouping of ink fragments. In our algorithm, $\mathcal{G}$ is constructed by first building a candidate graph (which is not necessarily triangulated) by connecting all pairs of fragments satisfying an appropriate distance constraint. Additional edges are added to create a triangulated graph, and pairwise feature vectors are generated for all edges on the new graph, including those which were added during triangulation. This approach gave a mean tree-width of 4.0 when applied to our training database. By modifying the algorithm to constrain the tree-width, an adjustable compromise between speed and accuracy can be obtained.

Table 2: Labeling errors for the three models. Results are the mean of three cross-validation splits. Relative differences are shown between models L and LI, and between LI and PLI. The mean relative differences are aggregations of the differences for each split, rather than the differences between the means for individual models. This is to reduce the effect of systematic variation between splits.

L	8.5%
LI	4.5%
% Δ LI/L	−48.9% ± 24.9%
PLI	2.6%
% Δ PLI/LI	−42% ± 8%

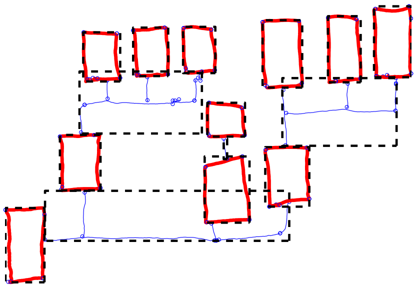


Figure 5: Example labelings and groupings: the most probable partition and labeling using model PLI. Heavy lines indicate fragments which have been classified as containers and lighter lines indicate connectors. Groups of fragments which belong to the same part are outlined using a dashed box. (Image rotated from original.)

4.2 FEATURES AND PRIORS

We chose features to reflect the spatial and temporal distribution of ink strokes, for example lengths and angles of fragments, whether two fragments were drawn with a single stroke, and the temporal ordering of strokes. We also used a number of 'template' features which were designed to capture important higher level aspects of the ink, such as the presence of T-junctions.

We use Gaussian priors, with correlations specified between the priors for weights corresponding to related features. In total 61 unary features and 37 pairwise features were used.

4.3 RESULTS

To test the performance of the method, we used a database of 40 example diagrams, consisting of a total of 2157 ink fragments. Three random splits were generated, each consisting of 20 examples used for training and 20 used for evaluation. Training was performed by finding the MAP weights as described in Section 2.1. The models were tested by finding the most probable partition and labeling as described in Section 2.2, and counting errors made against ground-truth data.

For comparison, we also consider two related models which model labeling only, without considering partitioning. The first of these models has a similar form to that described in Section 2, but uses pairwise potentials given by

$$\psi_{ij}^{(2)}\left(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}\right)=\begin{cases}\phi\left(\boldsymbol{v}_{\mathrm{s}}\cdot\boldsymbol{f}_{ij}\left(\boldsymbol{x}\right)\right) & \text{if } y_i=y_j\\ \phi\left(\boldsymbol{v}_{\mathrm{d}}\cdot\boldsymbol{f}_{ij}\left(\boldsymbol{x}\right)\right) & \text{if } y_i\neq y_j\end{cases}, \quad (25)$$

where $\boldsymbol{v}_{\mathrm{s}}$ and $\boldsymbol{v}_{\mathrm{d}}$ are weights corresponding to vertices i and j having the same and different labels respectively. The second related model does not use pairwise potentials at all — ink fragments are labeled independently of the other labelings. In the following, we refer to the full model performing labeling and partitioning as model PLI. LI is the model performing labeling only with pairwise potentials, and L is the model with unary potentials only.

Labeling error rates are shown in Table 2. Figure 5 shows the output of the algorithm on an example diagram. Further examples are available online at `http://research.microsoft.com/~szummer/aistats05/`.

5 DISCUSSION

The results given in Section 4.3 show that our approach is capable of providing high-quality labeled partitions. The data also illustrate an important point; simultaneous labeling and partitioning produces a significant improvement in labeling performance. This is easily understandable — the constraint that vertices within the same part must be labeled identically provides strong evidence for the labeling part of the algorithm, and the boundaries between regions of different labels are strong candidates for part boundaries. Hence the two aspects of the algorithm reinforce each other.

There are a number of extensions to the model which have not been discussed in this paper. The most straightforward is the incorporation of other local constraints, such as known labels of particular vertices, or information concerning the relationship of two vertices in the partition. These can easily be included through additional potentials which assign zero probability to configurations violating the constraints, and in the context of the ink parsing provide a valuable method for incorporating user feedback. It seems that more complex information, such as priors over the number of parts, can be incorporated by increasing the amount of information passed in the messages.

In some applications the maximum clique size may be too large for exact inference to be feasible, motivating approximate methods. Monte Carlo techniques have already been applied to problems of this sort [1], but it is desirable to apply alternative approximations such as loopy belief propagation, variational inference or expectation propagation.

6 CONCLUSION

We have presented a probabilistic model over labeled partitions of an undirected graph, and have shown that the structure of the graph may be used to efficiently perform exact inference with message passing algorithms. We have demonstrated the application of the model to the task of parsing hand-drawn diagrams. Our experiments illustrate that it is possible to obtain high-quality results using this technique. The results obtained prove that in our applications, labeling accuracy is improved by performing partitioning at the same time.

Acknowledgements

We would like to thank Thomas Minka, Yuan Qi and Michel Gangnet for helpful advice and discussion, and for providing excellent software that allowed the work presented in this paper to be completed. We are also grateful to Hannah Pepper for collecting our ink database.

References

[1] A. Barbu and S. Zhu. Graph partition by Swendsen-Wang cuts. In *ICCV*, 2003.

[2] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[3] S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[4] X. Liu and D. Wang. Perceptual organization based on temporal dynamics. In *NIPS*, volume 12, 2000.

[5] P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Readings in uncertain reasoning*, Morgan Kaufmann, pages 575–610, 1990.

[6] N. Sloane. The On-Line Encyclopedia of Integer Sequences, 2004. `http://www.research.att.com/projects/OEIS?Anum=A001861`

[7] C. Sutton, K. Rohanimanesh, and A. McCallum. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *ICML*, 2004.