Probability Distributions over Structured Spaces

Arthur Choi
Computer Science Department
University of California, Los Angeles
aychoi@cs.ucla.edu

Guy Van den Broeck
Department of Computer Science
KU Leuven, Belgium
guy.vandenbroeck@cs.kuleuven.be

Adnan Darwiche
Computer Science Department
University of California, Los Angeles
darwiche@cs.ucla.edu

Abstract

Our goal is to develop general-purpose techniques for probabilistic reasoning and learning in structured spaces. These spaces are characterized by complex logical constraints on what constitutes a possible world. We propose a tractable formalism, called probabilistic sentential decision diagrams, and show it effectively learns structured probability distributions in two applications: product configuration and preference learning.

1 Introduction

One of the long-standing goals of AI is to combine logic and probability in a coherent framework, and to enable high-level probabilistic reasoning. The bulk of the work with this goal has been towards first-order representations of uncertainty, and their reasoning and learning algorithms (Getoor and Taskar 2007; Kersting 2012; Milch et al. 2005). In this paper, we argue for a different, yet compatible direction of integration that has received much less attention: towards probability distributions over *structured spaces*. In traditional graphical models, the probability space is the Cartesian product of assignments to individual random variables, corresponding to the rows of a joint probability table. A structured probability space instead consists of complex objects, such as total and partial orders, trees, DAGs, molecules, pedigrees, product configurations, maps, plans, game traces, etc. Our goal is to develop a general-purpose framework for representing, reasoning with, and learning probability distributions over structured spaces. Performing these tasks in structured spaces has so far required special-purpose algorithms and techniques. For example, no single system is capable of learning distributions over both rankings and product configurations. This is in stark contrast with the state of general-purpose techniques for unstructured probability spaces, such as Bayesian networks.

As in the unstructured case, structured objects are conveniently represented by assignments to a set of variables. However, in a structured space, not every assignment represents a valid object. For example, not every sequence of actions represents a plan (the same truck cannot be loaded twice), and not every adjacency matrix represents a tree. Hence, probability distributions over such objects are not easily captured by the rows of a joint probability table. To effectively represent distributions over such objects, reason and learn, we believe it is important to explicitly deal with the structure of the probability space. This requires a *formal description* of the structure. For some of the above examples, a description language is already in widespread use, such as PDDL (McDermott et al. 1998) for planning problems, and GDL (Love et al. 2008) for games. For others, one can easily construct a theory in propositional logic that encodes the structure of the probability space. Section 2 details the encoding of our two running examples: student enrollment configurations and movie preferences.

Given a formal description of the structured space, the next challenge is to represent and reason with *probability distributions* over that space. There is work on facilitating logical constraints in probabilistic representations (Richardson and Domingos 2006; Mateescu and Dechter 2008), and within these representations, there has been considerable interest in probabilistic reasoning in the presence of logical constraints (Poon and Domingos 2006; Gogate and Dechter 2007). Despite these efforts, reasoning with constraints quickly becomes intractable, and many approximation algorithms fail when faced with the abundant determinism that is found in structured spaces. Therefore, we recently proposed a novel *tractable* representation of probability distributions over structured spaces, called probabilistic sentential decision diagrams (PSDDs) (Kisa et al. 2014). This representation is rooted in a long tradition of tractable representations of logical knowledge bases (Darwiche 2011). Section 3 describes PSDDs and their inference algorithms.

Finally, we are interested in *learning* distributions over structured spaces from data. Machine learning in structured spaces is investigated in structured prediction (Bakır et al. 2007). Again, different spaces often require special-purpose learning algorithms, and the structure of the probability space is rarely taken into account explicitly. Moreover, the probabilistic representations that support logical constraints do not support the learning of such constraints, with the exception of Chang et al. (2008). We believe that learning probabilistic models over structured spaces is an important and challenging open problem. As a first step, we developed an algorithm for learning PSDD parameters from complex data, given a formal description of the structured

L	K	P	A	Students
0	0	1	0	6
0	0	1	1	54
0	1	1	1	10
1	0	0	0	5
1	0	1	0	1
1	0	1	1	0
1	1	0	0	13
1	1	1	0	8
1	1	1	1	3

Table 1: Student enrollment data.

space (Choi, Van den Broeck, and Darwiche 2015). Section 4 describes our learning algorithm and its application to our running examples.

2 Structured Spaces

Consider a set of Boolean variables $X_1, \ldots, X_n$. The term *unstructured space* refers to the 2^n instantiations of these variables. The term *structured space* refers to a subset of these instantiations, which is determined by some complex, application-specific criteria. We will encode these criteria as a theory in propositional logic.

Configuration To provide a concrete example of a structured space, consider the following *configuration* problem. A computer science department organizes four courses: Logic (L), Knowledge Representation (K), Probability (P), and Artificial Intelligence (A). Students are asked to enroll for these courses under the following restrictions:

– A student must take at least one of Probability or Logic.

– Probability is a prerequisite for AI.

– The prerequisite for KR is either AI or Logic.

The department may have data on student enrollments, as in Table 1, and may wish to learn a probabilistic model for reasoning about student preferences. For example, the department may need to know whether students are more likely to satisfy the prerequisite of KR using AI or using Logic. In this example, we can encode the states of this domain (i.e., valid combinations of courses), as the following structured space, expressed in CNF:

$$(P \vee L) \wedge (A \Rightarrow P) \wedge (K \Rightarrow A \vee L) \tag{1}$$

Even though there are 16 combinations of courses, only 9 of them represent are valid. This task is an example of a problem with known, and possibly complex, domain constraints in propositional logic that induce our structured space.

Ranking As a second example, consider a preference learning domain, where we want to represent distributions over rankings (i.e., permutations). Say we want to rank n different items. Consider then the Boolean variables A_{ij} for $i, j \in \{1, \ldots, n\}$. Here, the index i represents an *item* and the index j represents its *position* in a total ranking of n items. The unstructured space consists of the 2^{n^2} instantiations of our n^2 Boolean variables. A structured space of interest consists of the subset of instantiations that correspond to *total rankings* over n items. The size of this structured space is only $n!$ as the remaining instantiations do not correspond to valid, total rankings (e.g., an instantiation with two items in the same position, or one item in two positions).

One can define the structured space of rankings using a Boolean formula, whose models induce the space. Assuming that we want to rank $n = 3$ items, we can define the structured space using two types of Boolean constraints:

- Each item i is assigned to exactly one position, leading to three constraints for $i \in \{1, 2, 3\}$: $(A_{i1} \wedge \neg A_{i2} \wedge \neg A_{i3}) \vee (\neg A_{i1} \wedge A_{i2} \wedge \neg A_{i3}) \vee (\neg A_{i1} \wedge \neg A_{i2} \wedge A_{i3})$.
- Each position j is assigned exactly one item, leading to three constraints for $j \in \{1, 2, 3\}$: $(A_{1j} \wedge \neg A_{2j} \wedge \neg A_{3j}) \vee (\neg A_{1j} \wedge A_{2j} \wedge \neg A_{3j}) \vee (\neg A_{1j} \wedge \neg A_{2j} \wedge A_{3j})$.

The Boolean formula defining the structured space will then correspond to a conjunction of these six constraints (more generally, $2n$ constraints).

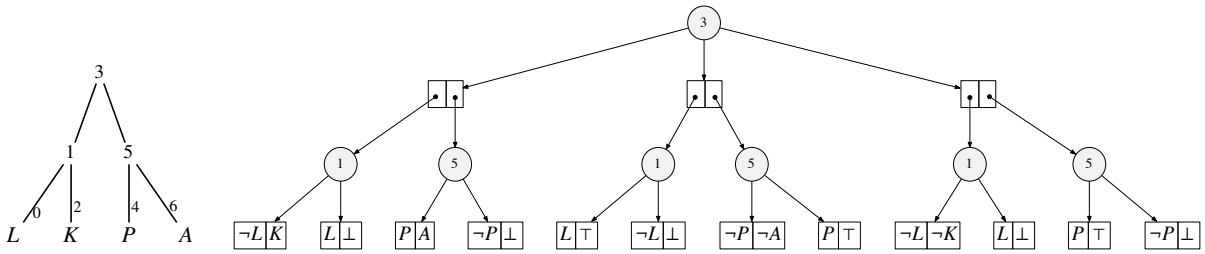
Partial Ranking To consider an even more complex example, let us consider the structured space of *partial rankings.* For defining this space, we will use Boolean variables A_{ij} with $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, t\}$. Here, the index i represents an *item,* and the index j represents the *tier* it is assigned to. The semantics is that we would prefer an item that appears in a higher tier (smaller j) over one appearing in a lower tier (larger j), but we will not distinguish between items within a tier. The sizes of tiers can vary as well. For example, the first tier can represent a single best item, the first and second tiers can represent the top-2, the first three tiers can represent the top-4, and so on. This type of partial ranking is analogous to one that would be obtained from a single-elimination tournament, where a 1st and 2nd place team is determined (in the finals), but where the 3rd and 4th place teams may not be distinguished (the two losers of the semi-finals). We can also define this structured space using a Boolean formula, which is a conjunction of two types of constraints. The first type of constraint ensures that *each item i is assigned to exactly one tier.* The second type of constraint ensures that *each tier j has exactly m_j items.*

3 Structured Representations and Reasoning

We will now describe our representations for structured spaces and their probability distributions.

Tractable Structured Spaces To enable efficient inference and learning, our approach requires that the Boolean formula for the structured space is in a tractable form, known as a *sentential decision diagram* (SDD) (Darwiche 2011). Figure 1 depicts an SDD for the Boolean formula given in Equation 1, for our student enrollment problem.

A circle in an SDD is called a *decision node* and its children (paired boxes) are called *elements.* Literals and constants ($\top$ and $\bot$) are called *terminal nodes.* For element (p, s), p is called a *prime* and s is called a *sub.* A decision node n with elements $(p_1, s_1), \ldots, (p_n, s_n)$ is interpreted as $(p_1 \wedge s_1) \vee \cdots \vee (p_n \wedge s_n)$. SDDs satisfy some strong properties that make them tractable for certain tasks, such as probabilistic reasoning. For example, the prime and sub of an element do not share variables. Moreover, if $(p_1, s_1), \ldots, (p_n, s_n)$ are the elements of a decision node, then primes $p_1, \ldots, p_n$ must form a partition ($p_i \neq \mathsf{false}$, $p_i \wedge p_j = \mathsf{false}$ for $i \neq j$, and $p_1 \vee \cdots \vee p_n = \mathsf{true}$).



(a) Vtree (b) SDD

Figure 1: A vtree and SDD for the student enrollment problem. Numbers in circles correspond to vtree nodes.

Each SDD is *normalized* for some *vtree*: a binary tree whose leaves are in one-to-one correspondence with the formula variables. The SDD of a Boolean formula is unique once a vtree is fixed. Figure 1 depicts an SDD and the vtree it is normalized for. More specifically, each SDD node is normalized for some vtree node. The root SDD node is normalized for the root vtree node. If a decision SDD node is normalized for a vtree node v, then its primes are normalized for the left child of v, and its subs are normalized for the right child of v. As a result, terminal SDD nodes are normalized for leaf vtree nodes. Figure 1 labels decision SDD nodes with the vtree nodes they are normalized for.

For partial ranking spaces, we report below the SDD sizes for $n = 64$ items and $t = 4$ tiers, where each tier grows by a factor k, i.e., top-1, top-k, top-k^2 and top-k^3 items. The SDD size counts the sizes of all decision nodes.

		Size		
n	k	SDD	Structured Space	Unstructured Space
8	2	443	840	$1.84 \cdot 10^{19}$
27	3	4,114	$1.18 \cdot 10^{9}$	$2.82 \cdot 10^{219}$
64	4	23,497	$3.56 \cdot 10^{18}$	$1.04 \cdot 10^{1233}$
125	5	94,616	$3.45 \cdot 10^{31}$	$3.92 \cdot 10^{4703}$
216	6	297,295	$1.57 \cdot 10^{48}$	$7.16 \cdot 10^{14044}$
343	7	781,918	$4.57 \cdot 10^{68}$	$7.55 \cdot 10^{35415}$

As we shall see next, this size corresponds roughly to the number of parameters needed to induce a distribution over the structured space. For example, for $n = 64$ items, and $k = 4$, one needs about $23,497$ parameters to induce a distribution over a structured space of size $3.56 \cdot 10^{18}$.

Tractable Distributions over Structured Spaces To represent distributions over a structured space, we proposed *probabilistic sentential decision diagrams* (PSDDs) (Kisa et al. 2014). The Boolean formula defining the structured space is first compiled into an SDD, and then parameterized to induce a probability distribution. Figure 2 depicts a PSDD, that was found by parameterizing the SDD of Figure 1.

An SDD is parameterized by providing distributions for its decision nodes and its terminal nodes, $\top$. A decision SDD node $n = (p_1, s_1), \ldots, (p_k, s_k)$ is parametrized using a distribution $(\theta_1, \ldots, \theta_k)$, which leads to a decision PSDD node $(p_1, s_1, \theta_1), \ldots, (p_k, s_k, \theta_k)$. The parameters θ_i are notated on the edges outgoing from a decision node; see Figure 2. A terminal SDD node $n = \top$ is parameterized by a distribution $(\theta, 1 - \theta)$, leading to terminal PSDD node $X : \theta$. Here, X is the leaf vtree node that n is normalized for; see Figure 2.

The semantics of PSDDs is based on the notion of a *context,* γ_n, for PSDD node n. Intuitively, this is a Boolean formula which captures all variable instantiations under which the decision diagram will branch to node n. We will not describe the distribution Pr_r induced by a PSDD r, but will stress the following local semantics of PSDD parameters. For a decision PSDD node $n = (p_1, s_1, \theta_1), \ldots, (p_k, s_k, \theta_k)$, we have $Pr_r(p_i|\gamma_n) = \theta_i$. Moreover, for a terminal SDD node $n = X : \theta$, we have $Pr_r(X|\gamma_n) = \theta$. This local semantics of PSDD parameters is the key reason behind their many well-behaved properties (e.g., the existence of a closed form for learning maximum likelihood parameters under complete data). We defer the reader to Kisa et al. (2014) for a more thorough exposition of PSDD syntax, semantics and properties.

PSDDs admit a variety of queries. For example, in preference learning tasks, we can compute the most likely total ranking $\sigma^\star = \text{argmax}_\sigma \, Pr(\sigma)$, which corresponds to the most probable explanation (MPE) in Bayesian networks. We can also compute the expected rank of each item i: $E[j] = \sum_j j \cdot Pr(A_{ij})$. As a tractable representation, such algorithms typically require a time complexity that is only linear in the size of the PSDD.

4 Learning in Structured Spaces

Kisa et al. (2014) provide an algorithm for learning the parameters of a PSDD given a *complete* dataset. This algorithm identified the (unique) maximum likelihood parameters, and further, in closed-form. It is used to estimate the parameters in Figure 2 from the dataset in Table 1. Choi, Van den Broeck, and Darwiche (2015) provide an EM algorithm for learning PSDD parameters from an *incomplete* dataset. This algorithm applies to a general type of incomplete dataset, allowing one to learn distributions from datasets based on arbitrary constraints, such as pairwise preferences in preference learning tasks; cf. (Lu and Boutilier 2011).

Ranking The standard approaches for dealing with distributions over rankings employ specialized distributions, such as the Mallows (1957) model. In a Mallows model, we assume a *central* total ranking σ, with probabilities of other rankings σ' decreasing as their distance from σ increases. The PSDD approach we utilize, however, is quite different as it allows one to induce any distribution over rankings, or any other structured space. For an empirical evaluation of

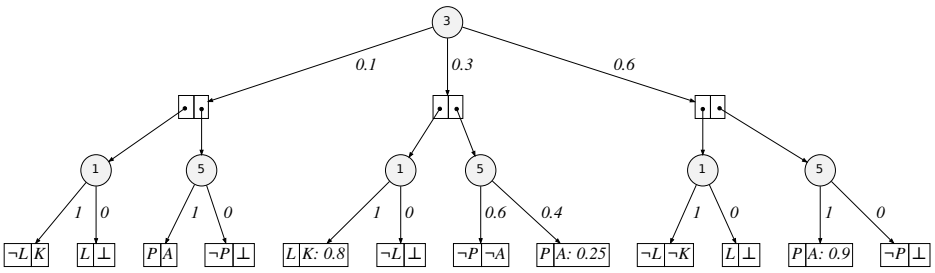


Figure 2: A PSDD for the student enrollment problem, learned from the dataset in Table 1, and SDD in Figure 1.

PSDDs in the preference learning task, see Choi, Van den Broeck, and Darwiche (2015), where PSDDs were shown to compare favorably to Mallows mixture models, and where more complex queries were highlighted.

Here, we highlight some of the qualitative empirical results for partial rankings, which considered the `movielens` dataset, consisting of over 1 million ratings of roughly 3,900 movies and 6,040 users. We took the top 64 most frequently rated movies, and used the ratings of the 5,891 users who rated at least one of these movies. Ratings are integers from 1 to 5, and for each user, we compiled an SDD representing their pairwise preferences: if a user gives movie i a higher rating than another movie j, we assert a constraint that the movie is at least as highly ranked as the other (or appears in at least as high a tier).[1] We thus obtain an SDD for each user by conjoining together all such pairwise preferences, which, over all users, gives us an incomplete dataset of partial rankings. We compile the SDD for the structured space itself, and learn the resulting PSDD's parameters from our dataset of SDDs, using EM.

We consider in particular queries that are based on diversified recommendations—a type of conditional query that is permitted by the proposed framework. To our knowledge, no current framework for preference distributions can handle such queries (at least not exactly and efficiently). First, we may ask for the top-5 movies (by expected tier), given the constraint α that "the highest ranked movie is Star Wars V":

1	Star Wars: Episode V - The Empire Strikes Back (1980)
2	Star Wars: Episode IV - A New Hope (1977)
3	Godfather, The (1972)
4	Shawshank Redemption, The (1994)
5	Usual Suspects, The (1995)

We see that another Star Wars movie is also highly ranked. However, if we wanted to use this ranking to recommend a movie, suggesting Star Wars IV is not very helpful, as someone who likes Star Wars V is likely to have already seen Star Wars IV. Say we condition on another constraint β, that "no other Star Wars movie appears in the top-5," and further, a third constraint γ, that "at least one comedy appears in the top-5." Conditioning on these three constraints α, β and γ, we obtain the new ranking, where Star Wars IV is replaced by the comedy/drama American Beauty:

[1]For partial rankings of 64 movies, we assumed four tiers representing the top-1, top-5, top-25 and top-64 movies.

1	Star Wars: Episode V - The Empire Strikes Back (1980)
2	American Beauty (1999)
3	Godfather, The (1972)
4	Usual Suspects, The (1995)
5	Shawshank Redemption, The (1994)

This example illustrates the powerful and open-ended types of queries allowed by the proposed framework.

Acknowledgments This work was supported by ONR grant #N00014-12-1-0423, NSF grant #IIS-1118122, and the Research Foundation-Flanders (FWO-Vlaanderen).

References

Bakır, G.; Hofmann, T.; Schölkopf, B.; Smola, A.; Taskar, B.; and Vishwanathan, S. 2007. *Predicting structured data*. MIT press.

Chang, M.-W.; Ratinov, L.-A.; Rizzolo, N.; and Roth, D. 2008. Learning and inference with constraints. In *AAAI*, 1513–1518.

Choi, A.; Van den Broeck, G.; and Darwiche, A. 2015. A tractable framework for learning and querying preference distributions. Under review.

Darwiche, A. 2011. SDD: A new canonical representation of propositional knowledge bases. In *IJCAI*, 819–826.

Getoor, L., and Taskar, B. 2007. *Introduction to statistical relational learning*. MIT press.

Gogate, V., and Dechter, R. 2007. SampleSearch: A scheme that searches for consistent samples. In *IJCAI*, 147–154.

Kersting, K. 2012. Lifted probabilistic inference. In *ECAI*.

Kisa, D.; Van den Broeck, G.; Choi, A.; and Darwiche, A. 2014. Probabilistic sentential decision diagrams. In *KR*.

Love, N.; Hinrichs, T.; Haley, D.; Schkufza, E.; and Genesereth, M. 2008. General game playing: Game description language specification.

Lu, T., and Boutilier, C. 2011. Learning Mallows models with pairwise preferences. In *ICML*, 145–152.

Mallows, C. L. 1957. Non-null ranking models. *Biometrika*.

Mateescu, R., and Dechter, R. 2008. Mixed deterministic and probabilistic networks. *Annals of Mathematics and AI* 54(1-3):3–51.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL-the planning domain definition language.

Milch, B.; Marthi, B.; Russell, S.; Sontag, D.; Ong, D. L.; and Kolobov, A. 2005. BLOG: Probabilistic models with unknown objects. In *IJCAI*, 1352–1359.

Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *AAAI*.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62(1):107–136.