Sampling over Search Trees Using Abstractions

Rina Dechter
University of California, Irvine
Irvine, CA 92697
dechter@ics.uci.edu

Filjor Broka
University of California, Irvine
Irvine, CA 92697
fbroka@ics.uci.edu

Kalev Kask
University of California, Irvine
Irvine, CA 92697
kkask@ics.uci.edu

Alexander Ihler
University of California, Irvine
Irvine, CA 92697
ihler@ics.uci.edu

Abstract

We present a new sampling scheme for approximating hard to compute summation queries over graphical models (e.g., partition function). The scheme builds upon exact algorithms that traverse a weighted directed state-space tree representing a global probability function over a graphical model. With the aid of an abstraction function and randomization, the state space can be compacted to facilitate tractable computation, yielding a Monte Carlo Estimate that is unbiased.

1 Introduction and Background

Imagine that we want to compute a function over a weighted directed search tree where the graph is given implicitly, e.g., using a generative state-space search model whose explicit state tree is enormous and does not fit in memory. Knuth and Chen [4, 1] proposed a pioneering sampling scheme for estimating quantities that can be expressed as aggregates (e.g., sum) of functions defined over the nodes in the tree, focusing on estimating the number of nodes. Our work is inspired by their scheme and extends it to a class of functions defined over weighted directed graphs. In particular, since *reasoning tasks over graphical models (e.g., probability of evidence or partition function)* can be transformed into tasks over weighted directed state trees [2], our scheme (Abstraction Sampling) should apply.

A graphical model can be defined by a 4-tuple $\mathcal{M} = (\mathbf{X}, \mathbf{D}, \mathbf{F}, \prod)$, where $\mathbf{X} = \{X_i : i \in I\}$ is a set of variables indexed by a set I and $\mathbf{D} = \{D_i : i \in I\}$ is the set of finite domains of values for each X_i. $\mathbf{F} = \{\psi_\alpha : \alpha \in scopes(F)\}$ is a set of discrete functions, where $\alpha \subseteq I$ and $X_\alpha \subseteq X$ is the scope of ψ_α. A graphical model represents a global function $Pr(X) \propto \prod_\alpha \psi_\alpha(X_\alpha)$. A popular task is to compute the partition function $Z = \sum_X \prod_\alpha \psi_\alpha(X_\alpha)$. A graphical model can also be expressed via a weighted state tree (T). The states or nodes in the graph are partial assignments relative to a fixed variable ordering, where each layer corresponds to one variable.

2 Abstraction Sampling

Abstraction sampling uses an *abstraction function* that partitions the nodes in the search tree into subsets of *abstract* states under the assumption that nodes in the same abstract state represent similar subproblems and can therefore be *merged* into a "typical" single representative. Given an abstraction function over a weighted directed tree T, our algorithm generates a weighted directed subgraph $\tilde{T}$ using a generative randomized process. In particular, the scheme chooses randomly, using importance sampling probabilities, a single representative node from each encountered abstract state

Table 1: The effective variance is the empirical variance multiplied by the average number of nodes per probe. For j-level we compute the ratio between the effective variances of that level and the 0-level abstraction (r.e.v). The table displays the fraction of instances where r.e.v is less than x% percent. In parentheses results for j in (1, 2, 4, 6, 8). Benchmark statistics: ($\bar{n}$, $\bar{w}$, $\bar{k}$, $|\bar{F}|$, $\bar{s}$) are averages of number of variables, induced width, max domain size, number of functions, max scope size.

Benchmark (#instances, $\bar{n}$, $\bar{w}$, $\bar{k}$, $\|\bar{F}\|$, $\bar{s}$)	i-bound	$< 100\%$	$< 50\%$	$< 10\%$
BN	10	(5/49, 3/49, 3/49, 4/49, 3/49)	(3/49, 3/49, 1/49, 2/49, 2/49)	(0/49, 0/49, 0/49, 2/49, 1/49)
(49, 266.5, 24.3, 2, 266.5, 11.04)	15	(3/49, 0/49, 2/49, 3/49, 4/49)	(1/49, 0/49, 1/49, 1/49, 2/49)	(0/49, 0/49, 0/49, 1/49, 2/49)
DBN	10	(11/47, 16/47, 16/47, 9/47, 11/47)	(11/47, 15/47, 16/47, 9/47, 11/47)	(7/47, 10/47, 11/47, 5/47, 8/47)
(47, 750.3, 26.2, 2, 14847.9, 2)	15	(14/47, 16/47, 19/47, 19/47, 20/47)	(12/47, 14/47, 19/47, 16/47, 17/47)	(8/47, 10/47, 15/47, 15/47, 16/47)
Grids	10	(2/7, 2/7, 3/7, 3/7, 3/7)	(2/7, 2/7, 3/7, 3/7, 3/7)	(2/7, 2/7, 2/7, 2/7, 2/7)
(7, 271, 24.3, 2, 791.4, 4)	15	(3/7, 2/7, 2/7, 1/7, 2/7)	(1/7, 1/7, 1/7, 1/7, 2/7)	(1/7, 1/7, 1/7, 1/7, 1/7)
Linkage	10	(6/17, 6/17, 8/17, 7/17, 4/16)	(2/17, 6/17, 7/17, 6/17, 2/16)	(1/17, 3/17, 3/17, 3/17, 2/16)
(17, 949.9, 29.1, 4.9, 949.9, 4)	15	(6/15, 3/15, 2/15, 5/15, 2/14)	(1/15, 1/15, 2/15, 4/15, 1/14)	(0/15, 0/15, 0/15, 1/15, 0/14)
Promedas	10	(5/10, 6/10, 6/10, 4/10, 6/9)	(4/10, 6/10, 6/10, 4/10, 6/9)	(3/10, 6/10, 6/10, 3/10, 2/9)
(10, 670.9, 18.6, 2, 670.9, 3)	15	(4/10, 4/10, 4/9, 4/10, 6/10)	(2/10, 4/10, 3/9, 4/10, 6/10)	(2/10, 4/10, 2/9, 3/10, 2/10)
Segmentation	10	(0/6, 0/6, 0/6, 0/6, 0/6)	(0/6, 0/6, 0/6, 0/6, 0/6)	(0/6, 0/6, 0/6, 0/6, 0/6)
(6, 230.2, 18, 2, 858.3, 2)	15	(1/6, 1/6, 1/6, 1/6, 1/6)	(1/6, 1/6, 1/6, 0/6, 1/6)	(0/6, 0/6, 0/6, 0/6, 1/6)

and associates it with a *weight* that estimates the total contribution of all states that it represents. Importance sampling probabilities are chosen proportional to the current weight of a node, the cost of the path to the node, and an estimate of the total mass of the subtree rooted by this node, given by a heuristic function. An estimator to our query over T can be computed over the generated representative tree $\tilde{T}$, which is supposed to be far smaller. Clearly, if the number of abstract states is bounded, the generated tree is small and the estimator can be computed efficiently.

We show that like in the earlier schemes of Knuth and Chen [4, 1], our emerging estimator is unbiased. Moreover, both accuracy and time complexity of our proposed abstraction sampling scheme should depend on the quality of the abstraction function and the accuracy of the sampling process.

3 Experimental Results and Future Work

We evaluated our algorithm on instances from 6 benchmarks by running experiments using different configurations of heuristic and abstraction functions for one hour each. We used Weighted Mini-Bucket Elimination (WMBE) [3, 5] heuristic function, whose strength is controlled by a parameter called the i-bound. Higher i-bounds lead to stronger heuristics at the expense of higher computation and memory cost. For each problem instance we experiment with two i-bounds (10, 15). As abstraction function we used a family of abstraction functions called relaxed context-based abstraction, parametrized by a level. 0-level abstraction merges all nodes corresponding to a variable in a single abstract state, leading to a degenerate tree (path), and it thus corresponds to simple importance sampling. Higher level abstractions have increasing number of abstract states, thus leading in general to higher number of nodes expanded during sampling. We tested 6 different abstraction levels: 0, 1, 2, 4, 6, 8.

In Table 1 we present aggregate results comparing statistical efficiency of higher level abstraction and 0-level abstraction (simple importance sampling) as decribed in the caption. We notice that for several benchmarks (Linkage, Promedas, DBN), there are around 30-40% of the instances for which using a higher level abstraction outperforms using simple importance sampling.

Future work would include exploring and evaluating different abstraction function families, and extending the existing algorithm to AND/OR search spaces [2] to benefit from sampling on smaller search spaces.

References

[1] P.-C. Chen. Heuristic sampling: A method for predicting the performance of tree searching programs. *SIAM Journal on Computing*, 21:295–315, 1992.

[2] Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.

[3] Rina Dechter and Irina Rish. Mini-buckets: A general scheme for bounded inference. *J. ACM*, 50(2):107–153, 2003.

[4] D.E. Knuth. Estimating the efficiency of backtracking algorithms. *Math. Comput.*, 29:1121–136, 1975.

[5] Qiang Liu and Alexander T. Ihler. Bounding the partition function using holder's inequality. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 849–856, 2011.