Bounding Graphical Models Processing by Hypertree Width

Student: Lars Otten, Advisor: Rina Dechter

Bren School of Information and Computer Sciences,
University of California, Irvine, CA 92697-3425, U.S.A.
{lotten,dechter}@ics.uci.edu

Abstract. In 2000, Gottlob et al. [3] introduced a new graph parameter, the hypertree width, and showed that it provides a broader characterization of tractable constraint networks than the treewidth. In 2005 this observation was extended to general graphical models [5], showing that the hypertree width yields bounds on inference algorithms. This paper explores further the practical properties of the hypertree width parameter for bounding the complexity of constraint satisfaction and optimization tasks. To that end we study empirically the effectiveness of the treewidth vs. hypertree width over common network benchmarks.

1 Introduction

Constraint networks, Bayesian networks, Markov random fields and influence diagrams, commonly referred to as graphical models, are all languages for knowledge representation that use graphs to capture conditional independencies between variables. These independencies allow both the concise representation of knowledge and the use of efficient graph-based algorithms for query processing. Inference-based algorithms (e.g., variable-elimination, join-tree clustering) exploit the independencies captured by the underlying graphical model. They are known to be time and space exponential in the treewidth of the graph.

Graphical models algorithms, however, are often far more efficient than what is predicted by the treewidth, especially when the problem instances exhibit a significant amount of determinism. And indeed, the treewidth is a measure which is completely blind to the specific representation of the functions; in fact it assumes that a function defined on r variables will take $O(k^r)$ to specify, when k bounds the variables' domain size.

In 2000, Gottlob et al. [3] introduced another parameter called hypertree width and showed it to be more effective at capturing tractable classes of constraint networks. In [5], Kask et al. extend the applicability of the hypertree width to inference algorithms over general graphical models. In this paper we empirically explore the relevance of the hypertree width to graphical models, investigating whether the hypertree width yields a better bound than treewidth in practice.

In Section 2 we provide preliminaries, Section 3 gives an overview of tree decomposition and hypertree decomposition. Section 4 presents the empirical evaluation and Section 5 concludes.

2 Background

We assume the usual definitions of directed and undirected graphs.

Definition 1. *A* **hypergraph** *is a pair $H = (V, S)$ where $S = \{S_1, ..., S_t\}$ is a set of subsets of V, called hyper-edges. The* **primal graph** *of a hypergraph $H = (V, S)$ is an undirected graph $G = (V, E)$ such that there is an edge $(u, v) \in E$ for any two vertices $u, v \in V$ that appear in the same hyper-edge (namely, there exists S_i, s.t., $u, v \in S_i$). The* **dual graph** *of a hypergraph $H = (V, S)$ is an undirected graph $G = (S, E)$ that has a vertex for each hyper-edge, and there is an edge $(S_i, S_j) \in E$ when the corresponding hyper-edges share a vertex ($S_i \cap S_j \neq \emptyset$).*

Definition 2. *A hypergraph is a* **hypertree***, also called* **acyclic hypergraph***, if and only if its dual graph has an edge subgraph that is a tree, such that all the nodes in the dual graph that contain a common variable form a connected subgraph.*

Definition 3. *A* **graphical model** *$\mathcal{R}$ is a 4-tuple $\langle X, D, F, \otimes \rangle$ where: (1) $X = \{X_1, \ldots, X_n\}$ is a set of variables; (2) $D = \{D_1, \ldots, D_n\}$ is the set of their respective finite domains of values; (3) $F = \{f_1, \ldots, f_r\}$ is a set of discrete real-valued functions, each defined over a subset of variables $S_i \subseteq X$, called its scope. (4) $\otimes_i f_i \in \{\prod_i f_i, \sum_i f_i, \bowtie_i f_i\}$ is a combination operator. The graphical model represents the combination of all its functions: $\otimes_{i=1}^r f_i$. A* **reasoning task** *is based on a marginalization (elimination) operator, $\Downarrow$, and is defined by: $\Downarrow_{Z_1} \otimes_{i=1}^r f_i, \ldots, \Downarrow_{Z_t} \otimes_{i=1}^r f_i$, where $Z_i \subseteq X$.*

For example, for constraint optimization tasks one might have $\Downarrow_{Z_1} \otimes_{i=1}^r f_i = \min_X \sum_{i=1}^r f_i(X)$ as the reasoning task, where the f_i are the cost functions.

Definition 4. *The set of variables X and the scopes $S = \{S_1, \ldots, S_r\}$ of a graphical model defines the graphical model's hypergraph (X, S). If this hypergraph is a hypertree the graphical model is called* **acyclic***.*

The special cases of reasoning tasks which we have in mind are constraint networks, belief networks or mixed networks that combine both [2]. The primary tasks for constraint networks are finding or counting solutions, they are defined using relations as functions, and relational join and project as the combination and marginalization operators. The primary tasks over belief networks are belief updating and finding the most probable explanation. They are specified using conditional probability functions defined on each variable and its parents in a given directed acyclic graph, and use multiplication and summation or maximization as the combination and marginalization operators [5].

3 Tree and Hypertree Decompositions

Tree clustering schemes have been widely used for constraint processing and for probabilistic reasoning. The most popular variants are join-tree and junction-tree algorithms. The schemes vary somewhat in their graph definitions as well as in the way tree decompositions are processed [1, 7, 3, 8]. However, they all involve a decomposition of a hypergraph into a hypertree.

Definition 5. *[5] Let $\mathcal{P}$ be a reasoning task over a graphical model $\langle X, D, F, \bigotimes \rangle$. A* **tree decomposition** *for $\mathcal{P}$ is a triple $\langle T, \chi, \psi \rangle$, where $T = (V, E)$ is a tree and χ and ψ are labeling functions that associate with each vertex $v \in V$ two sets, $\chi(v) \subseteq X$ and $\psi(v) \subseteq F$, that satisfy the following conditions:*

1. *For each $f_i \in F$, there is exactly one vertex $v \in V$ such that $f_i \in \psi(v)$.*
2. *If $f_i \in \psi(v)$, then $scope(f_i) \subseteq \chi(v)$.*
3. *For each variable $X_i \in X$, the set $\{v \in V | X_i \in \chi(v)\}$ induces a connected subtree of T. This is also called the running intersection or the connectedness property.*

The **treewidth** *of a tree decomposition $\langle T, \chi, \psi \rangle$ is $w = \max_v |\chi(v)| - 1$. The treewidth of $\mathcal{P}$ is the minimum treewidth over all its tree decompositions.*

Algorithm *Cluster-Tree Elimination (CTE)* is a message-passing algorithm, where each vertex of the tree sends a function to each of its neighbors. If the tree contains m edges, then a total of $2m$ messages will be sent as follows. For each neighbor v, node u takes all the functions in $\psi(u)$ and all the messages received by u from all adjacent nodes, and generates their *combined* function which is *marginalized* over the separator with v and sent to v. (For further discussion and other styles of algorithms such as *join-tree clustering (JTC)* see [5].)

Theorem 1. *[5] Given a reasoning task $\mathcal{P}$ over a graphical model $\langle X, D, F, \bigotimes \rangle$, and a tree decomposition $\langle T, \chi, \psi \rangle$, let m be the number of vertices in the tree decomposition, w its treewidth, sep its maximum separator size, r the number of input functions in F, deg the maximum degree in T, and k the maximum domain size of a variable. The time complexity of CTE is $O((r + m) \cdot deg \cdot k^{w+1})$ and its space complexity is $O(m \cdot k^{sep})$.*

3.1 Hypertree Decomposition

One issue with the treewidth is its sole dependence on the primal graph, ignoring its hypergraph structure completely. For example, an acyclic problem whose scopes have high arity would have a high treewidth even though it can be processed in linear time. In particular, Bayesian networks which are *polytrees* [6] are acyclic, yet they have treewidth equal to the maximum family size, which can be arbitrarily large. For example, the noisy-OR and noisy-AND specifications of polytrees are linear time, yet their treewidth can be arbitrarily large [6].

The hypertree width introduced by [3] for constraint networks and extended in [5] for general graphical models, relies on the notion of *hypertree decompositions.* As a subclass of tree decompositions, it provides a stronger indicator of tractability than the treewidth.

Definition 6. *[3, 5] Let $\mathcal{T} = \langle T, \chi, \psi \rangle$, where $T = (V, E)$ be a tree decomposition of a reasoning problem $\mathcal{P}$ over a graphical model $\langle X, D, F, \bigotimes \rangle$. $\mathcal{T}$ is a* **hypertree decomposition** *of $\mathcal{P}$ if the following additional condition is satisfied:*

4. For each $v \in V$, $\chi(v) \subseteq scope(\psi(v))$.

The **hypertree width** *of a hypertree decomposition is $hw = \max_v |\psi(v)|$. The hypertree width of $\mathcal{P}$ is the minimum hypertree width over all its hypertree decompositions.*

In [3] the complexity of processing hypertree decomposition for solving a constraint satisfaction problem is analyzed as a function of the hypertree width hw, it is shown that a hypertree decomposition of a constraint problem can be processed in time $O(m \cdot hw \cdot logt \cdot t^{hw})$.

In [5] it was shown that the time complexity bound of Theorem 1 can be extended straight-forwardly to any graphical model that is absorbing relative to 0. A graphical model is absorbing relative to a 0 element if its combination operator has the property that $x \bigotimes 0 = 0 \;\; \forall x$; for example, multiplication has this property while summation does not. It has been shown that:

Theorem 2. *[5] A hypertree decomposition of a reasoning problem that is absorbing relative to 0 can be processed[1] in time $O(m \cdot deg \cdot hw \cdot logt \cdot t^{hw})$ and space $O(t^{hw})$, where m is the number of edges in the hypertree decomposition, hw its hypertree width, and t bounds on the size of the relational representation of each function in $\mathcal{R}$.*

Note that if $t = k^w$ then Equation 2 becomes: $m \cdot hw \cdot w \cdot logk \cdot (k^w)^{hw}$. Clearly, in this case, for any tree decomposition, the treewidth-based bound is far superior than the one provided by its hypertree width. The question we thus ask is, under what conditions would the complexity bound generated by the hypertree width be tighter than the bound generated by the treewidth? And how often are those conditions met in practice?

4 Experiments

In this section we evaluate empirically the treewidth and hypertree width bounds on various practical probabilistic networks such as coding networks, dynamic Bayesian networks, genetic linkage instances and CPCS networks used in medical diagnosis.

Since computing the minimal hypertree width is NP hard in general, we employed a heuristic method for generating hypertree decompositions that were recently proposed in [4]. It uses Bucket Elimination (BE) and produces a tree decomposition of the primal graph given an elimination order and extends it to a hypertree decomposition by means of set covering heuristics. This results in upper bounds for both the treewidth and hypertree width.

We looked at about 100 problem instances. Table 1 displays a summary of a subset of 23 Bayesian networks from the UAI'06 Evaluation Repository[2] which

[1] The algorithms for processing decompositions assumed in [3] and [5] are slightly different

[2] http://ssli.ee.washington.edu/ bilmes/uai06InferenceEvaluation/

instance	class	n	a	k	t	$\bar{k}$	$\bar{t}$	w	hw	R	$\bar{R}$
BN_20		2843	4	91	910	18.92	213.32	7	4	1.88	-0.38
BN_22	DBN	2425	4	91	910	18.74	193.39	6	3	2.88	0.78
BN_24		1819	4	91	910	20.99	269.41	5	2	3.88	1.75
BN_26		3025	7	10	3645	6.00	1222.67	10	2	2.88	1.61
BN_70		2315	5	36	128	5.38	22.92	86	42	45.34	5.72
BN_71	linkage	1740	5	36	128	6.17	24.23	52	39	-1.25	-12.89
BN_72		2155	5	36	128	6.18	23.94	66	53	-8.97	-20.89
BN_73		2140	4	36	128	5.86	21.57	92	55	27.28	-2.72
BN_74		749	4	36	128	6.44	24.26	28	22	-2.78	-7.82
BN_75		1820	5	36	128	5.76	24.32	100	51	48.16	5.36
BN_76		2155	5	36	128	7.01	28.07	72	40	27.77	2.96
BN_77		1020	4	45	162	9.21	42.48	24	14	8.74	0.35
BN_80		360	12	2	4096	2.00	134.26	22	4	-7.83	-1.89
BN_82	CPCS	360	12	2	4096	2.00	134.26	23	4	-7.53	-1.59
BN_84		360	12	2	4096	2.00	134.26	21	4	-8.13	-2.19
BN_86		422	18	2	262144	2.00	2200.63	23	4	-14.75	-6.45
BN_88		422	18	2	262144	2.00	2200.63	24	5	-19.87	-9.49
BN_126		512	5	2	16	2.00	6.00	58	21	-7.83	1.12
BN_127	coding	512	5	2	16	2.00	6.00	55	22	-9.93	-0.56
BN_128		512	5	2	16	2.00	6.00	56	21	-8.43	0.52
BN_129		512	5	2	16	2.00	6.00	56	22	-9.63	-0.26
BN_130		512	5	2	16	2.00	6.00	54	21	-9.03	-0.09
BN_131		512	5	2	16	2.00	6.00	54	21	-9.03	-0.08

Table 1. Results for experiments with Bayesian networks from the UAI'06 Evaluation Repository.

we view as more interesting or representative of the overall set. For each test instance we report the number of variables n, the maximum and average domain size (k, $\bar{k}$), the maximum arity a of the conditional probability tables (CPTs) and the maximum and average tightness (t, $\bar{t}$), defined as the number of tuples with non-zero probability. We also record the treewidth w and hypertree width hw obtained for the BE decomposition heuristic. To quantify the difference between the two bounds (in orders of magnitude), we define the following ratios:

$$R = log_{10}\left(\frac{k^w}{t^{hw}}\right) \quad \text{and} \quad \bar{R} = log_{10}\left(\frac{\bar{k}^w}{\bar{t}^{hw}}\right).$$

When R is negative the treewidth bound is tighter than the hypertree width bound. We observe that for problem instances with large domain sizes and relatively tight CPTs the hypertree width bound improves upon the treewidth by several orders of magnitude. For instance, on problems BN_70 and BN_75, the difference is more than 40 orders of magnitude in favor of the hypertree width bound. These networks belong to the genetic linkage analysis domain and are known to contain a lot of deterministic CPTs.

In contrast, and as expected, on problem instances with small domain sizes and relatively little determinism, such as coding and CPCS networks, the traditional treewidth based bound provides a much tighter measure of computational complexity. We also experimented with deterministic grid networks and networks derived from the ISCAS'85 digital circuit benchmark. All these networks have maximum domain sizes of 2 and exhibit a low degree of determinism, therefore the treewidth bound is indeed superior.

5 Conclusion

It is well known that a graph treewidth provides bounds for many computational tasks over graphical models (e.g., satisfiability, counting, belief updating, finding the most likely explanation.). All these tasks are bounded exponentially by the graph treewidth.

In this paper we show that the hypertree width bound, which was shown to provide a broader tractability characterization for constraint networks, can be extended to search and inference algorithms for general graphical models, such as Bayesian networks, Markov networks, cost networks and so on when functions are specified as *relations*.

We explored empirically the practical benefit of hypertree width, compared to treewidth, in bounding the complexity of algorithms over given problem instances. Statistics collected over 100 instances Bayesian networks (only 23 are reported for space reasons) provided interesting, yet somewhat sobering information. We showed that while the hypertree width is always smaller than the treewidth, the complexity bound it implies is often inferior to the bound suggested by the treewidth. However, when problem instances possess substantial determinism and when the functions have high arity, the hypertree width can provide bounds that are many orders of magnitude tighter and therefore far more informative than the treewidth. This demonstrates the added sensitivity of the hypertree width to the hypergraph structure and to the functions' specification.

In summary, our study suggests that the hypertree width is a useful and informative parameter for characterizing the complexity of graphical models task, beyond the treewidth. In practice it can yield a tighter bound on complexity only when functions are defined on large scopes and when the function representation is quite sparse.

References

1. R. Dechter and J. Pearl: Tree Clustering for Constraint Networks. *Artificial Intelligence* **38** (1989): 353–366.
2. R. Dechter and R. Mateescu: Mixtures of deterministic-probabilistic networks and their AND/OR search space. In *Proceedings of UAI'04*: 120–129.
3. G. Gottlob, N. Leone, and F. Scarcello: A comparison of structural CSP decomposition methods. *Artificial Intelligence* **124** (2000): 243–282.
4. A. Dermaku, T. Ganzow, G. Gottlob, B. McMahan, N. Musliu, M. Samer: Heuristic Methods for Hypertree Decompositions. *Technical Report DBAI-TR-2005-53*, Vienna University of Technology, 2005
5. K. Kask, R. Dechter, J. Larrosa, and A. Dechter: Unifying tree decompositions for reasoning in graphical models. *Artificial Intelligence* **166** (2005): 165–193.
6. J. Pearl: Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, 1988.
7. S. L. Lauritzen and D. J. Spiegelhalter: Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society. Series B* **50(2)** (1988): 157–224.
8. P. P. Shenoy: Binary join trees for computing marginals in the Shenoy-Shafer architecture. *Int. J. Approx. Reasoning* **17** (1997): 239–263.