A Critique of Bag-Connected Tree-Width

J. Yancey
LCSEE,
West Virginia University,
Morgantown, WV
{jayancey@mix.wvu.edu}

Abstract

While tree decomposition methods have proven useful in solving constraints networks (CSPs), the problem of computing tree-decompositions has concentrated primarily on a single parameter, the tree-width. Experimental studies have shown that other parameters must also be considered when a decomposition is used to solve a CSP. It has been observed that in certain cases a bad decomposition, in terms of tree-width, may be more efficient in solving the underlying CSP. The authors show experimentally that the clusters appearing in decompositions may have several connected components when minimization of the width is done in order to produce a "good" decomposition. This lack of connectedness may cause the solving method to waste effort by finding the solution to unnecessary subproblems related to these non-connected clusters. To avoid this problem, the authors introduce a new graph parameter called *Bag-Connected Tree-Width* which considers tree decompositions for which each cluster is connected. They demonstrate that finding the smallest width bag-connected tree-decomposition is **NP-hard** and additionally introduce a polynomial time algorithm that produces bag-connected tree-decompositions of any graph.

1 Introduction

The authors study improved methods of solving Constraint Satisfaction Problems (CSPs), which are of interest as they provide an efficient way in which to formulate a variety of problems within computer science, especially in Artificial Intelligence.

Formally, A *constraint satisfaction problem* is a triple (X, D, C), where $X = \{x_1, \cdots x_n\}$ is a set of n variables, $D = (D_{x_1}, \cdots, D_{x_n})$ is a list of finite domains of values, one per variable, and $C = \{C_1, \cdots, C_e\}$ is a finite set of e constraints. Each constraint C_i is a pair $(S(C_i), R(C_i))$, where $S(C_i) = \{x_{i_1}, \cdots, x_{i_l}\} \subseteq X$ is the scope of C_i, and $R(C_i) \subseteq D_{x_{i_1}} \times \cdots \times D_{x_{i_k}}$ is its *compatibility relation*. The *arity* of C_i is $|S(C_i)|$. A CSP is called *binary* if all constraints are of arity 2. A constraint network is represented by a hypergraph called the constraint graph, whose vertices correspond to variables and edges to the constrains scopes. Tthe authors examine only binary CSPs, for simplicities sake. Further, without loss of generality, we assume that the network is connected. To simplify the notations, in the sequel, we denote the graph $(X, \{S(C_1), \cdots, S(C_e)\})$. An assignment on a subset of X is said to be *consistent* if it does not violate any constraint.

Testing whether a CSP has a *solution* is known to be NP-complete. Accordingly. many efforts have been made to make the solving of instances more efficient in practice by using optimized backtracking algorithms, heuristics, constraint learning, non-chronological backtracking, filtering techniques based on constraint propagation and numerous other techniques. The time complexity for these approaches is exponential, at least $O(n.d^n)$ where n is the number of variables and d is the maximum size of domains.

In their paper the authors show that solving CSPs using decomposition is inefficient and show that the reason can be found in the nature of the decompositions, for which w^+ is close to w, where w is the tree-width of the constraint graph and w^+ is an approximation of the tree-width. Minimizing w^+ can lead to decompositions such that some clusters have several connected components. However, a lack of connectedness may lead the solving method to waste effort solving subproblems related to these non-connected clusters, by passing many times from a connected component to another.

To avoid this problem, the authors introduce a new graph invariant, a parameter called *Bag-Connected Tree-Width*. This parameter is equal to the minimal width over all the tree decompositions for which each cluster has a single connected component. In other words, the Bag-Connected Tree-Width will be the minimum width for all Bag Connected Tree-Decompositions. The authors prove that the computation is NP-hard and propose a polynomial time algorithm in order to approximate this parameter and the associated decompositions. The time complexity of their algorithm is $O(n(n+e))$.

The next section states the problem and introduces the notations and principles of tree-decomposition and examines the motivation and related work. The third section introduces the notion of Bag-Connected Tree-Decomposition and states the problem. The fourth section offers a critique of the Bag-Connected Tree-Width algorithm. The final section presents a conclusion.

2 Motivation and Related Work

In this section we examine Tree-Decomposition methods and the impact of non-connected clusters on the efficiency of some of these methods.

2.1 Tree-Decomposition Methods

Tree-Clustering (TC) is the reference method for solving binary CSPs by exploiting the structure of its constraint graph. It is based on the notion of tree-decomposition of graph [Rob86].

Definition 2.1 *A tree-decomposition of a graph $G = (X, C)$ is a pair (E, T), where $T = (I, F)$ is a tree, and $E = \{E_i : i \in I\}$ is a family of subsets of X such that*

(i) $\cup_{i \in I} E_i = X$,

(ii) for each edge $(x, y) \in C$, there is $i \in I$, such that $\{x, y\} \subseteq E_i$,

(iii) for all $i, j, k \in I$ if k is on a unique $i - j$ path of T, then $E_i \cap E_j \subseteq E_k$.

The width of the tree-decomposition (E, T) is equal to $\max_{i \in I} |E_i| - 1$. The tree-width $w(G)$ of a graph G is the minimum width over all tree-decompositions of G.

The original version of TC [Dec89] begins by computing a tree-decomposition. The second step clusters are solved independently, considering each cluster as a subproblem, and then enumerating all its solutions. Next, a global solution of the CSP, if one exists, can be found efficiently by exploiting the tree structure of the decomposition. The complexity of this algorithm is $O(n.d^{w^+ + 1})$ where $w^+ + 1$ is the size of the larges cluster $(w + 1 \leq w^+ + 1 \leq n)$.

The Backtracking on Tree-Decomposition method (BTD) has shown to be more efficient in practical terms [Jeg05]. BTD does not need to solve each cluster completely to find its solution, in contrast to TC. A backtrack search is implemented, which exploits a variable ordering induced by a depth first traversal of the tree-decomposition. The author notes that while this approach has had good practical results, it is the worst case from a theoretical viewpoint.

In order to make structural methods more efficient, therefore, we must a priori minimize the values of w^+ and s, where s is the size of the largest minimal separator.

3 Statement of Problem

The notion of Bag-Connected Tree-Decomposition is defined, which corresponds to tree-decompositions for which each cluster E_i is connected.

Definition 3.1 *Given a graph $G = (X, C)$, a tree-decomposition (E, T) of G is connected if for all $E_i \in E$, the subgraph $G[E_i]$ of G induced by E_i is a connected graph. the width of a tree-decomposition (E, T) is equal to $max_{i \in I}|E_i| - 1$. The bag-connected tree-width w_c is the minimal width over all the bag-connected tree-decompositions of G.*

Given a graph $G = (X, C)$ of tree-width w, we see that necessarily $w \leq w_c$. If G is a chordal graph, $w = w_e$. If not, the bag-connected tree-width would be $\frac{k}{2}$, for cycles of length k without chords.

Our problem is related to construction of the optimal Bag-Connected Tree-Decomposition of width w_e, which can be shown to be **NP-hard**.

We have been seen that it is not necessary to find an optimal tree-decomposition for the solving of CSPs, and in many cases is not even desirable. Accordingly, the authors propose a polynomial time algorithm that finds a bag-connected tree-decomposition with no guarantee concerning the optimality of the result.

4 Critique

The problem that the authors seek to address in their paper is relatively new to the field of computer science. As such, the authors give a full background of related problems, providing detailed analyses of their solutions and impact on the problem the authors introduce. As a result of this detailed history, this paper is quite successful at painting the landscape of the problem and the many ideas that are required to understand the context of the *Bag-Connected Tree-Decomposition* problem.

The authors clearly introduce the notations and princples of tree-decomposition methods, and thereafter examine some problems inherent in the computing of "good" tree-decompositions. Both of these sections are rather thorough, clearly citing the evolution of the techniques and their limitations. This approach works well in preparing for the introduction of the notion of a Bag-Connected tree-decomposition, which is introduced by the authors. Further the algorithm introduced by the authors is easy to follow and has a figure to aid in comprehension.

Throughout the reading of the paper several areas could be improved for the reader by being slightly more thoughful of the formatting. While the paper is decomposed into sections, and in some cases subsections, portions are rather dense and algorithms aren't presented in a manner that immediately makes the structure apparent. Readability could be improved, perhaps by strategically using bold text, adding some more white space, and breaking down some of the larger and dense paragraphs present throughout the article.

5 Conclusion

The authors introduced the concept of *Bag-Connected Tree-Decomposition* and a new graph invariant associated with this decomposition, *Connected-Tree-Width*. This notion was introduced to make the solving of CSPs via decomposition

methods an easier task. From experiments the authors observed that the best tree-decompositions to solve CSPs of contain collections of clusters with several connected components, a feature which may decrease the efficiency of finding a solution. As the problem of finding a Bag-Connected Tree-Decomposition of minimum width is **NP-hard**, the authors introduced the first polynomial time algorithm which computes Bag-Connected Tree-Decompositions. The complexity of their algorithm is $O(n(n+e))$ where n is the number of vertices and e is the number of edges.

References

[Arn87] Arnborg, S.; Corneil, D.; and Proskuroswki, A. 1987. Complexity of finding embeddings in a k-tree. *SIAM Journal of Discrete Mathematics* 8:277-284.

[Dec89] Dechter, R. and Pearl, J. 1989. Tree-Clustering for Contraint Networks. *Artificial Intelligence* 125:93-118.

[Jeg05] Jegou, P.; Ndiaye, S.; and Terrioux, C. 2005. Computing and exploiting tree-decompositions for solving constraint networks. In *Proceedings of CP* 112-117.

[Rob86] Robertson, N., and Seymour, P. 1986. Graph minors II: Algoritmic aspects of treewidth. *Algorithms* 7:309-322.

[Ros06] Rossi, F.; van Beek, P.; and Walsh, T. 2006. *Handbook of Constraint Programming.* Elsevier.