Geometric Constraint Graphs Decomposition Based on Computing Graph Circuits

Robert Joan-Arinyo, Marta Tarrés-Puertas, and Sebastià Vila-Marta

Grup d'Informàtica a l'Enginyeria, Universitat Politècnica de Catalunya, Barcelona, Catalunya

Abstract. Geometric constraint solving is a growing field which plays a paramount role in industrial applications and that is deeply rooted in automated deduction in geometry. In this work we report on an algorithm to solve geometric constraint-based problems by decomposing biconnected graphs. The algorithm is based on recursively splitting the graph through sets with three vertices located on fundamental circuits of the graph. Preliminary practical experiments suggest that the algorithm runtime is at worst quadratic with the total number of vertices in the graph.

1 Introduction

Constraint-based parametric geometric models are data structures designed to represent and describe objects by encoding geometric shape and topological properties. They are at the core of a number of paramount industrial applications, say computer-aided design, robot path planning, molecular design, user interaction with virtual reality systems.

A central issue found in parametric geometric modeling is the constraint solving problem which can be roughly summarized as follows: Given a set of geometric elements and a set of constraints between them, place each geometric element in such a way that the constraints are fulfilled.

In this work, we consider 2D geometric constraint problems defined by a set of geometric elements like points, lines, line segments, circles and circular arcs, along with a set of constraints like distance, angle, incidence and tangency between any two geometric elements. The algorithms that solve geometric constraint problems are named solvers. The reader is referred to the work in [1, 5, 8, 13] for an extensive review on geometric constraint solving algorithms.

Among the existing solving methods we focus on constructive techniques. In these techniques the input is a geometric constraint problem represented as a geometric constraint graph. The output is a constructive plan, that is, a sequence of basic steps that describe how to build a solution to the constraint-based geometric problem. Basic steps correspond to elemental operations which are solved with dedicated algorithms.

In this paper we introduce a new algorithm based on the tree decomposition technique reported in [11]. The algorithm directly computes a graph decomposition from which a constructive plan can be easily derived.

In what follows we assume the reader is familiar with basic terminology of graph theory, the concept of geometric constraint graph associated to a geometric problem defined by constraints, and some definitions related to geometric constraint graphs. For more information we refer the reader to the works by Even, [2], Gao et al., [4], Hoffmann et al., [5], Joan-Arinyo et al., [9, 11],Owen [12], Thulasiraman and Swamy, [15]. and Whitney, [16],

2 The Algorithm

Let $G = (V, E)$ be a geometric constraint graph where V represents the set of geoms and E the set of constraints defined between them. Given a set of hinges $\{a, b, c\} \subseteq V$, a set decomposition of G can be trivially computed. Moreover, a recursive application of set decompositions yields a tree decomposition of G.

The goal is now to compute a set of hinges of a constraint graph G. We consider two distinct cases according the connectivity of G. For 0 and 1 connected graphs, there is a smooth approach. We refer to [9] for the details. For biconnected graphs, the approach is far more difficult. In what follows we focus on this class of graphs. Figure 1 outlines our algorithm.

```
INPUT: biconnected constraint graph G = (V,E), |V| ≥ 3
OUTPUT: a set of hinges {v1,v2,v3} ⊆ V, if one exists

Compute a spanning tree T of G
Compute the set of fundamental circuits C of G
    according to T
foreach Ci ∈ C do
    Compute the set of bridges B of G with respect to Ci
    Compute the collapsed graph G'
    Compute the merged graph G''
    Compute the planar embedding D of G''
    foreach F ∈ D do
        foreach {v1,v2,v3} ⊆ F do
            if {v1,v2,v3} ∈ Ci then
                return {v1,v2,v3}
            endif
        endfor
    endfor
endfor
return ∅
```

Fig. 1. Decomposition of a biconnected graph.

The algorithm proceeds as follows. First a spanning tree for the graph G is computed by applying a depth-first search. Then the associated fundamental circuits $\{C_1, \ldots, C_n\}$

are identified. From previous work, [9], we know that any set of hinges of G must be a subset of the vertices of some fundamental circuit C_i of G. Therefore we restrict the search for hinges to the set of fundamental circuits.

The search is performed in a planar embedding D of a graph G' resulting from transforming the given graph G according to the *bridges*, [9, 15], defined in G by the fundamental circuit under study. If the algorithm fails finding a fundamental circuit with a set of hinges, the input graph is not decomposable.

3 Experimental Results

To gain insight on the algorithm behavior and to perform a preliminary assessment of the algorithm runtime behavior, we have implemented it in the `SolBCN` framework which can be downloaded under a GNU General Public License (see [14]).

The tests have been conducted on a standard desk computer with a Pentium IV at 3GHz processor and 1GB of core memory. The algorithm is implemented in Java using the Sun JDK. The tests were planned as follows. Using the methodology defined in [10] to generate random geometric constraint graphs, two datasets were defined:

1. D_1: A set of 1000 randomly generated geometric constraint graphs with sizes ranging from 3 to 200 vertices. All the graphs were well-constrained but not necessarily tree-decomposable, that is not necessarily solvable by the tree decomposition approach.
2. D_2: A set of 1000 randomly generated of geometric constraint graphs with sizes ranging from 3 to 200 vertices. All the graphs were under-constrained but not necessarily tree-decomposable.

We also defined four versions of the decomposition algorithm:

1. A_1: this is a *brute-force* algorithm that performs an exhaustive search for hinges.
2. A_2: In this version first vertices of degree two are removed. Then the brute-force algorithm is applied.
3. A_3: This version is an improvement of A_2 with specific treatment for 0-connected and 1-connected graphs.
4. A_4: is the algorithm presented in this work.

Let SG_s denote the set of graphs G such that $s = |V(G)|$. Notice that $3 \leq s \leq 200$. We applied each algorithm version to each dataset. For each algorithm and each graph in a data set, we recorded the algorithm runtime $t(G)$. Then for each graph size s, we averaged the runtime values as

$$T(s) = \frac{\sum_{\forall G \in SG_s} t(G)}{s}$$

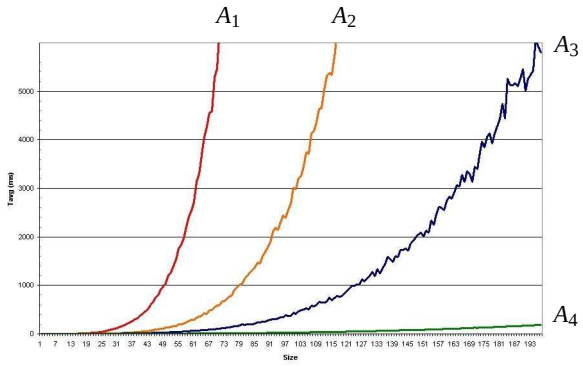

Fig. 2. Behavior of the algorithms A_1, A_2, A_3, and A_4 on the dataset D_1.

The results yielded by these tests are represented in Figure 2 for dataset D_1 and in Figure 3 for dataset D_2.

These results show that for both datasets the algorithm A_4 introduced in this paper exhibits a noticeable improved behavior. For graphs G with $|V(G)| \approx 200$, the runtime for the algorithm A_4 is of about 200ms what allows interactive use in the SolBCN framework.

4 Summary and Future Work

We have presented a new algorithm to compute a decomposition tree of a geometric constraint graph which makes interactive geometric constraint solving feasible. This algorithm is based on finding three hinges for the geometric constraint graph. The hinges are always found in a fundamental circuit of the graph and can be directly found by inspecting a particular planar embedding of it. The algorithm has been implemented and the empirical tests show a significant improvement with respect to a brute force approach.

In the near future we plan to work following three main directions. The first one is to proof the algorithm correctness. The second one is to carry out an in depth study of the algorithm complexity. We conjecture that the algorithm shown in this paper has quadratic runtime behavior. This time complexity is as good as the ones exhibited by [3], and [6, 7], algorithms. However, we believe that the running time complexity of our algorithm can be improved by reusing intermediate results in subsequent steps if convenient data structures are provided. This is the third direction to further explore.

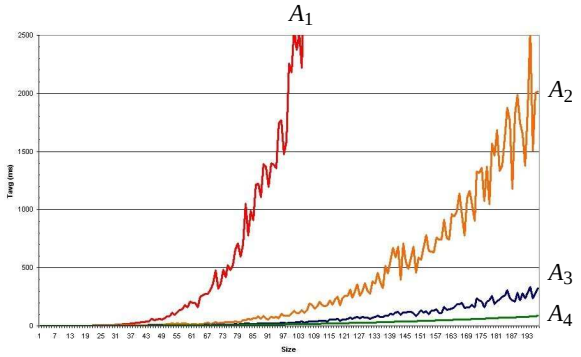

Fig. 3. Behavior of the algorithms A_1, A_2, A_3, and A_4 on the dataset D_2.

5 Acknowledgments

This research has been partially funded by the Spanish Ministerio de Educación y Ciencia and by FEDER under grant TIN2007-67474-C03-01.

References

1. C. B. Durand. *Symbolic and numerical techniques for constraint solving*. PhD thesis, Computer science department, Purdue University, 1998. Major Professor-C. M. Hoffmann.
2. S. Even. *Graph Algorithms*. Computer Science Press, Potomac, Md., 1979.
3. I. Fudos and C. M. Hoffmann. A graph-constructive approach to solving systems of geometric constraints. *ACM Trans. Graph.*, 16(2):179–216, 1997.
4. X.S. Gao, Q. Lin, and G. Zhang. A C-tree decomposition algorithm for 2D and 3D geometric constraint solving. *Computer-AIded Design*, 38(1):1–13, 2006.
5. C. M. Hoffmann and R. Joan-Arinyo. A brief on constraint solving. *Computer-Aided Design and Applications*, 5(2):655–663, 2005.
6. C.M. Hoffmann, A. Lomonosov, and M. Sitharam. Decomposition plans for geometric constraint systems, Part I: Performance measures for CAD. *Journal of Symbolic Computation*, 31(4):367–408, April 2001.
7. C.M. Hoffmann, A. Lomonosov, and M. Sitharam. Decomposition plans for geometric constraint systems, Part II: New algorithms. *Journal of Symbolic Computation*, 31(4):409–427, April 2001.
8. R. Joan-Arinyo, A. Soto, and S. Vila. Resolución de restricciones geométricas. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 20:121–136, 2003.
9. R. Joan-Arinyo, A. Soto-Riera, M. Tarrés-Puertas, and S. Vila-Marta. Decomposition of geometric constraint graphs based on computing fundamental circuits. Research report LSI-07-31-R, Department de Llenguatges i Sistemes Informatics, Technical University of Catalunya, 2007.
10. R. Joan-Arinyo, A. Soto-Riera, and S. Vila-Marta. Constraint-based techniques to support collaborative design. *Journal of Computing and Information Science in Engineering*, 6:139–148, 2006.

11. R. Joan-Arinyo, A. Soto-Riera, S. Vila-Marta, and J. Vilaplana-Pastó. Revisiting decomposition analysis of geometric constraint graphs. *Computer-Aided Design 36*, pages 123–140, 2004.
12. J.C. Owen. Algebraic solution for geometry from dimensional constraints. In *SMA'91: Proceedings of the first ACM symposium on Solid modeling foundations and CAD/CAM applications*, pages 397–407, New York, NY, USA, 1991. ACM, ACM Press.
13. A. Soto-Riera. *Geometric Constraint Solving in 2D*. PhD thesis, Departament de Llenguatges i Sistemes Informàtics, UPC, 1998.
14. A. Soto-Riera, S. Vila-Marta, D. Silva, and M. Freixa. The SolBCN geometric constraint solver. Source code from `http://floss.lsi.upc.edu`, 2007. Under GNU-GPL license.
15. K. Thulasiraman and N.N.S. Swamy. *Graphs: Theory and Algorithms*. John Wiley & Sons, 1992.
16. H. Whitney. Non-separable and planar graphs. *Proceedings of the National Academy of Sciences of the United States of America*, 17(2):125–127, February 1931.