$2D$-Structure Drawings of Similar Molecules *

J.D. Boissonnat, F. Cazals, and J. Flötotto

Projet Prisme, INRIA Sophia-Antipolis, 2004, route des Lucioles - BP 93, F-06902 Sophia Antipolis Cedex, France, {Firstname.Lastname}@sophia.inria.fr

Abstract. A common strategy in drug design and pharmacophore identification consists of evaluating large sets of molecular structures by comparing their $2D$ structure drawings. To simplify the chemists' task, the drawings should reveal similarities and differences between drugs. Given a family of molecules all containing a common template, we present an algorithm to compute standardised $2D$ structure drawings. The molecules being represented as a graph, we compute a structure called supertree in which all molecules can be embedded. Using the correspondences between atoms provided by the supertree, we are able to coordinate the drawings performed by a breadth-first traversal of the molecular graphs. Both parts of the problem are NP-hard. We propose algorithms of heuristic nature.

1 Introduction

1.1 Problem Statement

A *Molecule* is defined as an undirected graph of bounded degree with labelled nodes and edges. We restrict this definition to the case where molecules contain only induced cycles of length at most α —typically $\alpha = 6$, and two such cycles share at most one edge. This way, a molecule reduces to a tree with local cycles and blocks of cycles. [1] We define a *molecular family* as a set of molecules $\mathcal{M} = \{M_1, M_2, .., M_m\}$ sharing a connected subgraph T called the *template*. For a molecule M_k, the connected subgraphs of $M_k \backslash T$ are called its *appendices* and are denoted T_{t_i}, i.e. $M_k = T \cup \{T_{t_1}, T_{t_2}, .., T_{t_k}\}$. We further assume that each appendix is rooted at the unique node connecting it to the template. At last, all the molecules processed are assumed to have a planar drawing. Notice, that in the context of drug design, the assumptions are mostly satisfied.

The question we are interested in is related to the pharmacophore identification problem and was posed by a French pharmaceutical company as a follow-up to [4]. Using the previous definitions, we can state it as follows:

* A poster version of this work has been presented at JOBIM 2000 –Journées Ouvertes Biologie Informatique Mathématique, France, Mai 2000.

[1] Two cycles are adjacent if they share an edge. A block is a connected subgraph consisting of adjacent cycles. For further definitions concerning basic graph theory, we refer to [5, Chapter 1].

J. Marks (Ed.): GD 2000, LNCS 1984, pp. 115–126, 2001.

Given a molecular family and its template, draw the molecules in the plane in a standardised way to make the comparisons and interpretations of the chemist easier.

This is exemplified on Figure 1 which depicts two "similar" molecules with the drawings as they are stored in the database as well as the drawings automatically generated by our algorithm. The dotted edge served as template. [2]

Naturally, $2D$ drawings cannot capture all the subtleties of a molecule. (In particular the stereo-chemistry is not taken into account.) These $2D$ representations should therefore be considered as a necessary but not sufficient information: whenever they fall short from providing the precision required, they can be enriched by more precise facts such as the $3D$ Connolly surface, chemical measurements, etc.

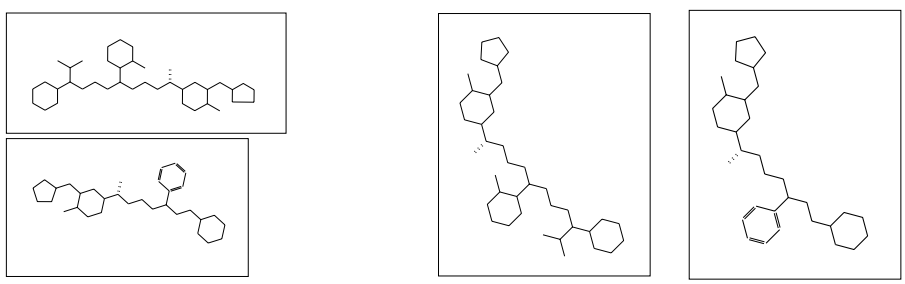

Fig. 1. (left) Database drawings, (right) Drawing algorithm "longest chain"

1.2 Paper Overview

In Section 2, we review relevant previous work. Section 3 outlines our drawing method. Its main ingredients are developed in Sections 4, 5, and 6. Implementation issues and results are presented in Section 7. Finally, Section 8 lists the future plans.

2 Previous Work

The bibliographic work consists of three parts: literature concerning the drawing of chemical structures, graph drawing methods in general, and the graph theoretic question of subgraph isomorphisms.

To the best of our knowledge, there exist very few publications concerning automatic drawing of $2D$ chemical structures. Several chemists confirmed that no automatic drawing software exists on the market. Nevertheless, in the Seventies and Eighties several approaches were proposed. First attempts by Zimmerman

[2] Examples are taken from our partner's database. For confidentiality reasons, they are slightly modified.

and Feldman [8] use a library of basic substructures and combine them according to a set of rules. The main criticism to this approach is lack of generality. In [3], a three-dimensional model is projected in the plane, but cyclic structures are not drawn with correct angles. In [14], Shelley presents a promising heuristic method. Similar to our approach, it is based on a breadth-first traversal of the molecule with the successive positioning of atoms. However, the heuristic is quite involved, and the final choice for atom positions is not completely described. A more recent genetic algorithm by D.B.Hibbert [10] generates and displays chemical structures. Angles are fixed following the valence of the atoms, and a drawing is evaluated with respect to the distance between non-adjacent atoms.

Concerning graph drawing literature and to the best of our knowledge, efficient algorithms for planar drawing of trees with unitary edge length and several fixed angles do not exist. However, hardness results for drawing trees are particularly interesting. We use the logic machine approach of Whitesides and Eades [5, Chapter 11.2] to show the hardness of our drawing problem.

The comparison of graphs and trees are well known questions in graph theory. Problems involving isomorphism testing of graphs are in general NP-hard problems [15, Chapter 2.6]. For molecular graphs, i.e. node labelled graphs, a simplification is shown in [13]. The computational complexity is lowered down by an important factor, but the problem still remains NP-complete. Polynomial-time algorithms exist for trees: The embedding problems for subgraph isomorphism and topological embedding can be solved in $O(n^{2.5} \log n)$ time for two trees of size $O(n)$ [9]. The problem is NP-hard for more than two trees, as shown in [1].

3 Algorithm Outline

3.1 Drawing Conventions and Constraints

A chemical drawing must satisfy several properties. The drawing is planar, and the degree of the node determines the angle between its adjacent edges. However, assuming one edge is fixed, the positioning of its adjacent nodes is not fully determined. The possible choices are shown in Figure 2. The angle between two edges incident to a node of degree 3 and a node of degree 2 that does not have two double bonds is $\alpha = \pm\frac{2\pi}{3}$. For a node of degree 2 with two double bonds there is no choice, $\alpha = \pi$. There are six possibilities to draw the adjacent nodes of a node of degree 4: The angles are multiples of $\alpha = \frac{\pi}{2}$, and the order among the edges is not fixed. If the edge belongs to a cycle, the angle is given by the type of cycle (i.e. for a hexagon $\alpha = \frac{2\pi}{3}$).

3.2 Automated Molecular Drawing: A Three Stages Process

From the previous discussion, a satisfactory solution to our problem needs to be able to:

1. identify the sub-structures common to several molecules,
2. perform the drawing of the molecules.

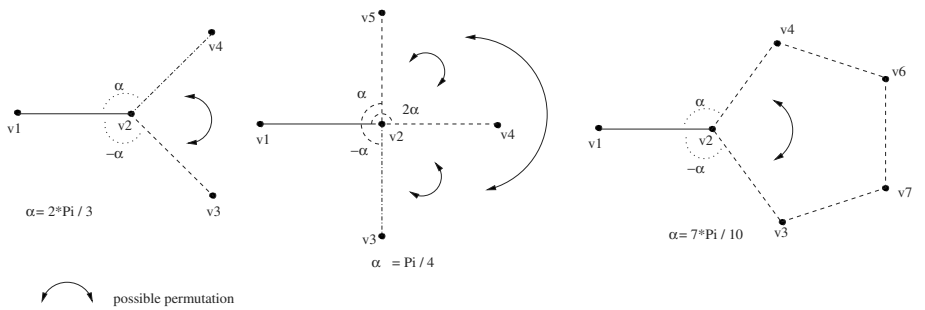

Fig. 2. Angles between bonds

To solve the first part of the problem, we construct a graph, the *supertree*, which "includes" all the molecules of the family. It is a fictive molecule into which each molecule is embedded. In general, the supertree itself does not admit a planar drawing. It is simply used to record and coordinate the drawings of the molecules.
The second stage of the drawing process consists of effectively drawing the molecules. The criteria to be respected are twofold. First, the constraints of Section 3.1 must hold. Second, in order to improve readability, some attention has to be devoted to aesthetic aspects. In particular, we shall be concerned with the diameter of the drawing — the maximum distance between two atoms, as well as the total area of the drawing.

To summarise, the outline of the algorithm is the following. First, cycles and blocks are detected in all the molecules. Next, a representation of the whole family of molecules is computed, the so-called supertree. At last, the drawing algorithm is applied to each molecule separately, but drawing decisions are coordinated using the supertree. The heuristics applied in the drawing procedure are based on the supertree.

4 Preprocessing Step: Cycle and Block Detection

The first step is the detection of cycles and blocks of cycles in all molecules. For each appendix, a Breadth-First Search (BFS) algorithm stores parent-child relations and *non-tree edges*, i.e. the last edge of each cycle to be traversed by the BFS. The corresponding cycle to a non-tree edge is found by collecting the ancestors of the nodes incident to the non-tree edge until a common ancestor is found. Adjacent cycles are grouped to a block. The first node of a block in BFS order is called its *entry node*. It is easy to see that the detection of cycles in a molecular graph of size $O(n)$ with c induced cycles of maximum size α has $O(n + \alpha c)$ running time.

5 Identification of Common Substructures

5.1 Supertree under Topological Embedding

The first part of the problem is to identify substructures that are isomorphic in several molecules of the family. This is necessary to be able to draw them the same way, up to a rigid motion. To give an illustration, Molecule 1 and Molecule 2 shown in Figure 3 have the template (dotted) but also the dashed parts in common. An embedding of a graph G into a graph S is a one-to-one

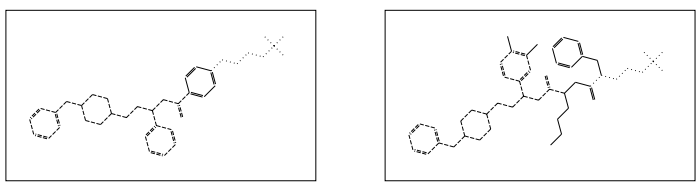

Fig. 3. Identified substructures, (a) Molecule 1 (b) Molecule 2

mapping of all nodes of G to the nodes (not necessarily all) of S. The type of embedding defines the condition under which two nodes can be mapped.[3] The smallest common embeddable supertree (SCES), thus, contains all atoms of all molecules, but several identical atoms may be associated to a single node.

We choose topological embedding which allows matching an edge to a path. This way, similar substructures are matched even if they are connected by paths of different length. This is, for example, employed in Figure 3. Figure 4 presents a simple supertree example.

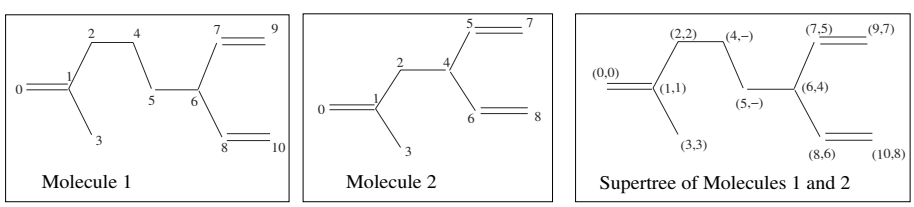

Fig. 4. Simple example to demonstrate the supertree

5.2 Supertree Computation

In this section, we show how to compute the supertree $S(r(M), r(M'))$ of two rooted trees M and M' with roots $r(M)$ and $r(M')$. In the sequel, $S(r(M), r(M'))$

[3] For definitions in this section, we refer to [9].

stands for the supertree as well as its size. Recall, that the roots are provided by the template. M and M' are appendices of the molecules rooted at the same template node.

Algorithm of Gupta and Nishimura. We adapt the algorithm proposed by Gupta and Nishimura in [9] which is based on a dynamic programming scheme. Without giving the details of the proofs, the results of [9] are briefly illustrated and later revisited for the treatment of cycles and blocks.

The algorithm is based on the following observation. An embedding can either map the two roots $r(M)$ and $r(M')$ to the root of the supertree and some children of $r(M)$ to distinct children of $r(M')$, or one of the roots can be mapped to a descendant of the other root. In [9, Lemma 7.4], the three possibilities are formalised as follows:

> For any node a of M with children $b_1, ..., b_k$ and for any node u of M' with children $v_1, ..., v_l$, three quantities are defined:
>
> $$M_1 = \min_{1 \le i \le l} \left\{ S(a, v_i) + 1 + \sum_{j \neq i} S(\emptyset, v_j) \right\};$$
>
> $$M_2 = \min_{1 \le i \le k} \left\{ S(b_i, u) + 1 + \sum_{j \neq i} S(b_j, \emptyset) \right\}; \text{ and}$$
>
> $$M_3 = MinWM(\{b_1, ..., b_k\}, \{v_1, ..., v_l\}) + 1.$$
>
> Then $S(a, u) = \min \{M_1, M_2, M_3\}$.

As in [9], $MinWM(\{b_1, ..., b_k, n_1, ..., n_l\}, \{v_1, ..., v_l, m_1, ..., m_k\})$ is the minimum weight perfect matching of the bipartite graph built from the two sets of nodes adjacent to a and u and a copy of each node in the party node set. The weight of the edge (b_i, v_j) is the size of the minimum supertree $S(b_i, v_j)$. Edges (b_i, m_i) and (n_j, v_j) correspond to a matching of a node with its copy. They have weight $S(b_i, \emptyset)$ and $S(\emptyset, v_j)$, respectively.
The running time of the algorithm is $O(n^{2.5} \log n)$ for two trees of size $O(n)$. This cost is dominated by the bipartite matching calculations. For details, we refer to [9].

Dealing with Cycles and Blocks. Unfortunately, a dynamic programming scheme does not apply to general graphs. A reduction from trees with blocks of cycles to trees is necessary in order to avoid NP-hardness. Two solutions come to mind: The reduction of a block to one node, or the destruction of the cycles by "hiding" an edge.

The first solution has the disadvantage that only complete blocks of cycles can be matched. The matching of a path that is not part of a cycle to some edges of a cycle is, yet, desirable for some chemical structures. The second one has a different drawback: since the choice of the hidden edge, e.g. the non-tree

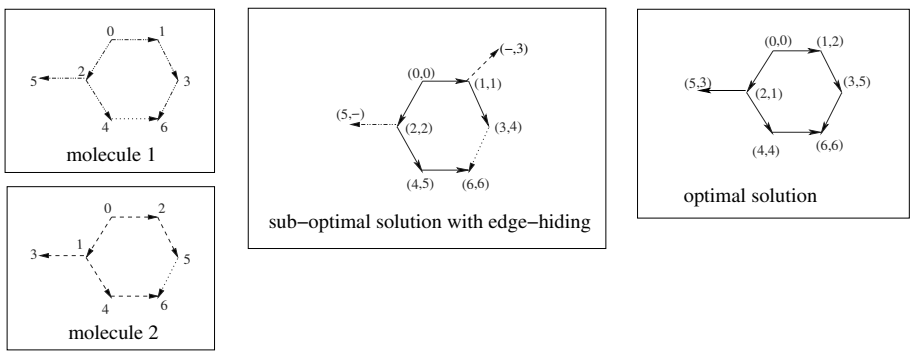

Fig. 5. Hiding non-tree edges (the node labels follow the BFS-order)

edge, is not uniquely determined by the graph-structure, the result might not be optimal as illustrated in Figure 5.

We choose a hybrid solution to avoid these inconveniences: For the matching of two blocks with entry nodes a or u, we do not regard their internal structure but consider a block as an atomic node —solution 1. In the case where either a or u is not part of a block, the non-tree edges of the blocks are hidden as proposed in the second solution, and the normal procedure applies.

Adaptation of the Algorithm to Labelled Trees. Only a slight modification is necessary to take atom types into account. The matching of nodes that are not of the same type is punished by counting the corresponding supertree node twice.

Data-Structures and Implementation. For every node a of M proceeding from leaves to root and for every node u of M' proceeding from leaves to root, $S(a, u)$ can be recursively computed following the scheme:

$$S(a,u) = \begin{cases} 1 & \text{if they are leaves and type}(a) = \text{type}(u), \\ 2 & \text{if they are leaves and type}(a) \neq \text{type}(u), \\ \max(\text{size}(block_1), \text{size}(block_2)) & \text{if } a \text{ root of } block_1 \text{ and } u \text{ root of } block_2, \text{and the blocks are leaves,} \\ \min\{M_1, M_2, M_3\} & \text{otherwise.} \end{cases}$$

Of course, to be able to actually construct the supertree, one has to keep track of the subtrees as well as their sizes. To do so, for each pair of nodes (a, u), a potential supertree node is created. Depending on whether $S(a, u)$ was the

result of M_1, M_2, or M_3, the corresponding edges are inserted. For example, in the case of $S(a,u) = M_3$, the edges inserted are $((a,u),(b_j,v_i))$ for each pair (b_j,v_i) matched in the computation of M_3, $((a,u),(b_l,\emptyset))$ and $((a,u),(\emptyset,v_k))$ for the non-matched nodes (matched with its own copy).
The tree rooted at the node $(r(M),r(M'))$ is the smallest supertree of M and M'. This is due to the fact that at each node insertion, we insert the edges that connect the node to its children in the supertree.

5.3 Supertree for the Family of Molecules

Theorem 1. *The computation of the smallest common embeddable supertree of m rooted trees of bounded degree is NP-hard and hard to approximate within a factor of $n^{1-\epsilon}$, for any $\epsilon > 0$.*

Proof. The problem is reduced from the INDEPENDENT SET problem following the result of [1]. For details see [2].

In this context, we content ourselves to an approximate solution and compute the supertrees two by two in a binary fashion. At first, the family is divided into pairs of molecules for which the supertree is computed. This is repeated recursively until we are left with a single supertree. Notice that the optimality of the result over several recursive calls is not guaranteed.

The size of the supertrees is in the worst case the sum of the size of the molecules —a rough upper bound. Thus, for m molecules of size $O(n)$, the size of the supertree of the family is in the worst case of order $O(m \times n)$. The following property is easily checked.

Proposition 1. *The computation of the supertree of a family of m molecules of sizes $O(n)$ has a worst case running time of $O((mn)^{2.5}\log(mn))$*

6 Drawing Molecules

The algorithm is based on the observation that all possible drawings of the molecule can be enumerated. Once an edge has been drawn, there is only a limited number of possibilities to draw the adjacent edges – see Section 3.1. The drawing algorithm consists of determining the choice of angles that leads to a permissible drawing which, in addition, fulfils as much as possible the aesthetic criteria.

6.1 NP-Hardness Proof

Before describing the algorithm, we state that it is an NP-hard problem to decide whether there exists a permissible drawing of a molecule following the constraints described in Section 3.1. The proof mimics the "logic engine" of [7]. See [2] for details.

Theorem 2. *It is an NP-hard problem to decide whether there exists a permissible drawing of a molecule.*

6.2 Basic Drawing Procedure

The basic procedure to draw a molecule consists of traversing the nodes in breadth-first-order starting at a template node and assigning a coordinate to each adjacent node. In case of a cycle or a block of cycles, the breadth-first order is not followed, and the whole block is processed at once. Before assignment, each position is checked for conflicts, i.e. intersections of the new edge with already drawn nodes and edges. If the assignment of coordinates to an adjacent node is impossible, we backtrack to the preceding node (in the breadth-first traversal), and try another permutation of node/position pairs. In case of failure, we backtrack even further.

To complete the description, one needs to define an order of preference:

P1. on the adjacent nodes of the last node drawn.
Two options referred to as "largest" and "longest" subtree are implemented. Nodes are ordered with respect to the depth or size of the subtree they are root of.

P2. on the possible positions of the nodes to be drawn.
The positions maximising the sum over the distances to already drawn nodes or the continuation of the longest chain are preferred.

The combination of the possibilities for P1 and P2 yield 4 heuristics. (But we disregard the combination "largest subtree" – "development of longest chain' which does not make sense.)

6.3 Rigid Blocks

The analysis of the decision tree which describes all possible drawings immediately leads to the notion of rigid blocks. We define a *rigid block* to be a part of the molecule whose embedding depends on only one decision. In case of a block, for example, the choice of the position of one node determines the placement of the whole block and of its adjacent edges. In the drawing procedure, this is taken into account by processing an entire block immediately after the placement of the first node, as already mentioned above. For the adjacent edges, the angle is determined by the angle inside the cycle. An example of a molecule with a block of 5 cycles is given in Figure 6(c).

6.4 Coordination of the Drawings

This section describes how drawing decisions are coordinated using the supertree. The first molecule is drawn as described above, with the only difference that the heuristics (depth or size of subtrees) are based on the supertree structure. For the following molecules, the heuristic consists of trying the choice that has been successful in the majority of the molecules already processed. If it fails, another angle is chosen with the usual heuristics. Already drawn molecules are not corrected in order to avoid further backtracking.

The data-structure used for the coordination of the angles consists of a priority queue for each edge that contains all angles used for the edge. The priority is the number of molecules that employed this solution. It is updated each time a molecule has been processed. In fact, it is a persistent data structure as proposed in [6] using the "fat node model".

7 Implementation and Results

The implementation consists of about 10000 lines of C++ code. It uses the Graph Template Library GTL (http://infosun.fmi.uni-passau.de/GTL) and the STL library (Standard Template Library) [12]. The GTL library implements a graph data-structure together with basic algorithms such as graph traversal algorithms. It is based on the generic data-structures of STL.Molecules are given as molfiles or structure-data files, two connection table (CTAB) formats that are standard in the chemical software industry (see http://www.mdli.com). The list of atoms and bonds in CTAB format is transformed into the graph description language GML [11] by a perl script. It is possible to display the former drawing of the molecule (that was done by the chemist) to be able to compare the results. The graphical output is written in OpenGL (http://reality.sgi.com/opengl). A CTAB file is produced for further use in standard chemical set-ups. For printing, a Postscript file is dumped.

The drawing algorithm can be applied to single molecules separately from the supertree computation. Some of the results are shown as examples in the paper. Usually, there is not much difference between the different heuristics, neither in running time nor in the drawings. For the supertree program, some molecules are entered – either with specified templates or without (default: the template consists of the first edge). Again, the options for the drawing algorithm are valid.

The running time on a SUN computer with a SPARC processor at 300 MHz and 250 MB of RAM is about 25 seconds for 12 molecules of average size of 35 atoms. So far, the code has not been optimised. More experimental facts will be provided in [2].

We tested examples from two molecular databases. The examples of database 1 were selected by the pharmacologist as good and typical examples for their application. In the second molecular database, the drawings are given as 3D projections of the molecular structures –see Figure 6(b).

8 Conclusion and Future Work

The calculation of the supertree is a promising solution to coordinate drawing decisions in the layout of $2D$-chemical structures of several similar molecules. In addition to the fact that the drawings reveal similarities and differences between several molecules, the supertree speeds up the drawing procedure. Whenever a

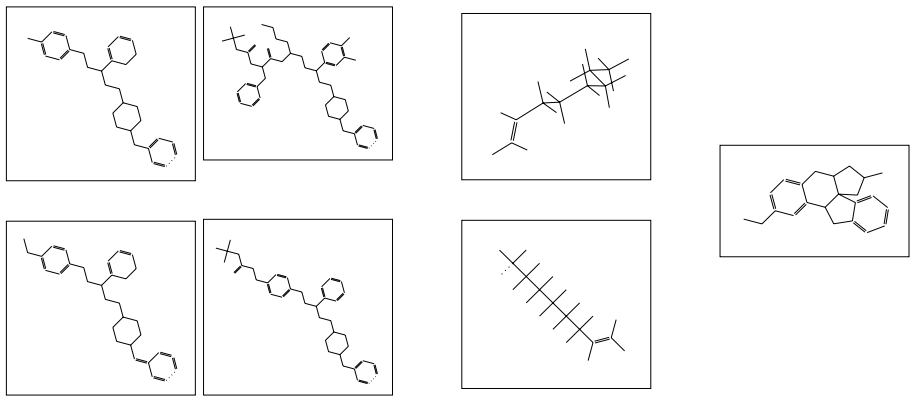

Fig. 6. (a) Family of 4 molecules, (b) Example from database 2, (c) Molecule with block of cycles

backtracking is necessary in the first molecule, copying its result for the other molecules prevents from testing the misleading decisions again.

Unfortunately, the method is restricted to the case where a template or at least one common atom is known in all molecules of the family. The choice of the template is crucial to the quality of the result. For now, a generalisation of the supertree algorithm to the case of non-rooted trees seems difficult to realize without a significant slow down in computation time (a factor of n^2 per call). Notice, however, that template queries to data-bases yield in general the necessary information. Second, the optimality of the size of the supertree after successive recursive calls is not guaranteed. It might be interesting to derive bounds on the error and to propose more sophisticated strategies to avoid it. Last but not least, one might look at a further use of the supertree representation in this context. The size of the supertree as a measurement of similarity, for example, could be considered.

The results of the drawing algorithm are in general correct and readable to the chemist. The heuristics avoid backtracking in most cases which makes the computation very fast. Problems occur when no permissible drawing can be found. Should this happen, we return the original database drawings. A mechanism to allow compromises in the drawing such as the changing of angles or variances of the edge length would allow more flexibility. Another problem is the orientation of ring systems such as steroids. Such chemical structures must be drawn with a certain orientation. In order to draw them in the right manner, we propose to supply a dictionary of critical ring systems with their predefined drawings.

Acknowledgements. The authors wish to thank Sue Whitesides for interesting discussions on rigid blocks and on the complexity of the drawing algorithm during her visit at INRIA in May 1999.

References

1. Tatsuya Akutsu and Magnús M. Halldórsson. On the approximation of largest common subtrees and largest common point sets. *Theoretical Computer Science*, to appear 2000.
2. Jean-Daniel Boissonnat, Fréderic Cazals, and Julia Flötotto. 2*d*-structure drawings of similar molecules. Technical report, INRIA Sophia Antipolis, to appear 2000.
3. Raymond E. Carhart. A model-based approach to the teletype printing of chemical structures. *Journal of Chemical Information and Computer Sciences*, 16:82–88, 1976.
4. F. Cazals. Effective nearest neighbours searching on the hyper-cube, with applications to molecular clustering. In *14th ACM Symposium on Computational Geometry*, pages 222–230, 1998.
5. G. di Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing*. Prentice Hall, 1999.
6. J. R. Driscoll, N. Sarnak, D. D. Sleator, and R. E. Tarjan. Making data structures persistent. *Journal of Computer and System Sciences*, 38:86–124, 1989.
7. Peter Eades and Sue Whitesides. The logic engine and the realization problem for nearest neighbor graphs. *Theor. Comput. Sci.*, 169(1):23–37, 1996.
8. R. J. Feldmann, S. R. Heller, E. Hyde, and W. T. Wipke (Ed.). *Computer Representation and Manipulation of Chemical Information*, pages 55–60. Wiley, New York, 1974.
9. A. Gupta and N. Nishimura. Finding largest subtrees and smallest supertrees. *Algorithmica*, 21:183–210, 1998.
10. D. Brynn Hibbert. Generation and display of chemical structures by genetic algorithms. *Chemometrics and Intelligent Laboratory Systems*, 20:35–43, 1993.
11. M. Himsolt. Gml: A portable graph file format. Technical report, Universität Passau, 1997. cf. http://www.fmi.uni-passau.de/himsolt/Graphlet/GML.
12. D.R. Musser and Atul Saini. *STL Tutorial and Reference Guide*. Addison-Wesley Publishing Company, 1995.
13. V. Nicholson, C.-C. Tsai, M. Johnson, and M. Naim. A subgraph isomorphism theorem for molecular graphs. In *Graph theory and topology in chemistry, Collect. Pap. Int. Conf.*, volume 51 of *Stud. Phys. Theor. Chem.*, pages 226–230, Athens/GA, 1987.
14. Craig A. Shelley. Heuristic approach for displaying chemical structures. *J. Chem. Inf. Comput. Sci.*, 23:61–65, 1983.
15. J. van Leeuwen (Ed.). *Handbook of Theoretical Computer Science*, volume A. Elsevier, 1990.