SPROUT: 3D Structure Generation Using Templates

Paulina Mata,[†] Valerie J. Gillet,[‡] A. Peter Johnson,*[,‡] Jorge Lampreia,[†] Glenn J. Myatt,[‡]
Sandor Sike,[§] and Anna L. Stebbings[‡]

Departamento de Quimica, Faculdade de Ciencias e Tecnologia, Universidade Nova de Lisboa, Quinta da Torre, 2825 Monte da Caparica, Portugal, School of Chemistry, University of Leeds, Leeds LS2 9JT, U.K., and Eötvos Loránd University, Általános Számítástudományi Tanszék, Budapest, Bogdánfy u. 10/b, Hungary

Received July 2, 1994[®]

SPROUT is a computer program for the rational design of molecules for a range of applications in molecular recognition. Molecular graphs are built in a stepwise fashion by subgraph addition. Several heuristics are being explored to restrict the combinatorial explosion that is inherent in structure generation. These include the use of generalized molecular fragments, called templates, as building blocks. Structure generation consists of two stages: (i) the generation of skeletons from templates that satisfy steric constraints and (ii) the substitution of heteroatoms into skeletons to produce molecules that satisfy other constraints such as electrostatics. The choice and definition of the templates and template joining rules are described together with a description of the atom substitution process.

INTRODUCTION

SPROUT[1–3] is a program designed to generate molecules appropriate to a wide range of applications in molecular recognition, e.g., the *de novo* design of enzyme inhibitors, catalysts, or agents for asymmetric synthesis. One of the main problems to face in *de novo* design is the combinatorial explosion that is inherent in structure generation; attempts at finding solutions quickly lead to a large number of possibilities. This type of problem has been well studied in artificial intelligence, where methods have been devised for delaying and moderating the combinatorial explosion.[4] Generally, efficient solution methods require knowledge about the problem domain to direct the search, i.e., heuristics. In our approach several heuristics are being explored. This paper is concerned with heuristics that are derived from chemical knowledge about commonly occurring substructural fragments. These heuristics have lead to the definition of a set of templates that are used as building blocks for structure generation, and the definition of a set of rules that described how the templates can be joined in the process of structure generation.

STRUCTURE GENERATION

Structure generation techniques can be applied to the problem of *de novo* structure design where the aim is to build a wide range of molecules with a given set of steric and chemical properties. Structures can be generated in a brute-force approach by beginning with a single atom and then sequentially adding one atom and bond at a time to the growing partial structures. This approach has been used in a number of programs that have been described in the literature.[5–7] However, building all possible solution structures in this way is computationally impossible because of the combinatorial explosion of possibilities that would result. The computational effort can be reduced to some extent by using molecular fragments at each addition step rather than single atoms. The generation of chemical structures in 2D through the joining of substructures has already been used successfully in several domains.[8–10] In the case of 3D structures, the principle underlying the process is that the joining of a set of conformationally reasonable substructures can result in a molecule that also has a reasonable conformation. This approach has been used successfully in the WIZARD[11] and COBRA[12] programs for conformational analysis. For *de novo* structure design, however, an enormous number of fragments is required to enable a wide range of molecules to be produced, even when the fragments are restricted to low energy conformations. Thus exhaustive searching of structure space using molecular fragments is also prohibitive, and the programs described in the literature that use this method[13,14] use different techniques to sample structure space. A further class of programs for *de novo* structure design operate by positioning fragments at the interaction sites within the active site of an enzyme and then linking them by searching for fragments within a database.[15–17]

SPROUT

SPROUT[1–3] uses information about one molecule to constrain the design of others with which it can interact. SPROUT generates structures from fragments; however, heuristics are used to reduce the problem in order that a representative search can be made over all structure space and a wide range of different classes of structures can be generated. The main factor in reducing the problem to a manageable size is the use of generalized fragments, or templates, as building blocks, along with an associated set of rules for controlling the ways in which the templates can be joined. A brief outline of the program is given before focusing on the main topic of this paper, i.e., a detailed description of the templates and the template joining process.

In SPROUT, the receptor site is used to define a volume for structure generation and some target sites within the volume. Target sites are small regions of space where it is desirable to place a ligand atom in order to promote an interaction such as a hydrogen bond between the ligand and the receptor. In SPROUT structure generation has been

[†] Universidade Nova de Lisboa.
[‡] University of Leeds.
[§] Eötvos Loránd University.
[®] Abstract published in *Advance ACS Abstracts*, January 15, 1995.

0095-2338/95/1635-0479$09.00/0

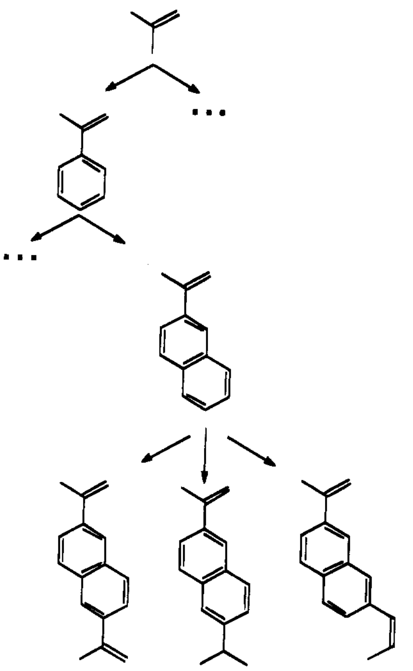

Figure 1. Skeletons are generated by the stepwise addition of fragments.

divided into two phases: (i) the generation of **skeletons** that satisfies the steric and geometric constraints (a skeleton is defined as a molecular graph whose vertices are labeled by hybridization state and whose edges are labeled by bond type) and (ii) the substitution of atoms in the skeletons to produce molecules that have the required properties, e.g., electrostatic and hydrophobic properties.

Skeletons are built in a stepwise fashion by subgraph addition (Figure 1). The subgraphs represent commonly occurring substructural fragments and are called **templates**. The process of adding templates to evolving skeletons is called **template joining**. The skeletons are built to satisfy the target site and volume constraints. The published version of SPROUT[2] builds skeletons by growing outwards from one target site, joining new templates onto the evolving skeleton until all the target sites are satisfied. A second version of SPROUT[18] has now been developed that builds skeletons from many target sites simultaneously. Partial skeletons that satisfy different target sites can be connected together. The definition of the templates and the template joining processes are the same in both versions of the program.

The aim of the atom substitution phase is not only to achieve chemical complementarity but also to confer certain properties on the molecules, e.g., solubility, stability, or ease of synthesis. Substituting one element type for another results in changes to the bond lengths and angles and hence the overall 3D shape of the molecule can change. However, it is assumed that substituting elements of the same hybridization state has only minor effect on 3D shape. Thus, the steric constraints are still satisfied when a skeleton is converted to a molecule, and the molecule produced also satisfies the less well defined constraints such as electrostatics and hydrophobicity.

The main features of the structure generation component of SPROUT are as follows:

1. **Templates** are used as building blocks where each template represents a number of substructures.

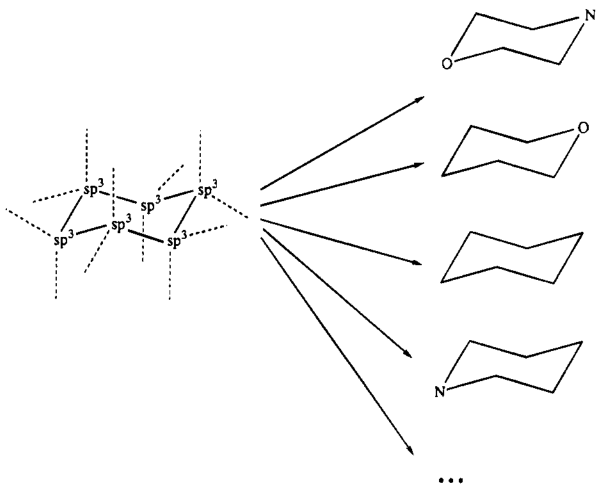

Figure 2. Each template represents several molecular fragments.

2. The set of templates is restricted to the more commonly occurring substructures.

3. The conformations available for each substructure are restricted to those with reasonable energy.

4. Graph searching techniques are used to explore the problem space.

5. Template joining routines have been implemented.

6. A set of rules has been developed to restrict the joining of the templates (**template joining rules**).

7. The steric similarities of the templates are used to reduce the number of joining operations that are required.

8. An atom substitution process is used to produce molecules that satisfy the steric and electrostatic constraints.

TEMPLATE DEFINITION

The underlying assumption in the use of templates for structure generation is that the difference in the overall shape of a molecule induced by the substitution of one atom for a different element type will be negligible. This assumption is valid for the more commonly occurring elements, C, N, and O, if atoms are substituted only by other atoms that have the same hybridization state and hence geometry. Some elements such as sulfur or phosphorus will have a larger effect on the bond angles and bond lengths, however, if the substitutions take place near the extremities of the skeletons these elements can also be considered to have a small effect on the overall shape of a structure.

Each template represents several molecular substructures in a single conformation (Figure 2). Templates are joined to form skeletons that satisfy steric constraints, i.e., a vertex of the skeleton should be positioned within each target site without the skeleton violating the boundary. Once a skeleton has been found that satisfies these constraints, then each of its constituent templates can be replaced by one of the molecular substructures it represents.

The use of templates is reasonable in the context of other approximations made in SPROUT, e.g., a template is assumed to be a rigid isolated substructure and its environment is not considered either at the intramolecular level (in the context of other templates which make up the molecule) or at the intermolecular level (within the receptor site) where steric and electronic effects can alter the shape of the bound molecule. Ignoring heteroatoms at this stage considerably

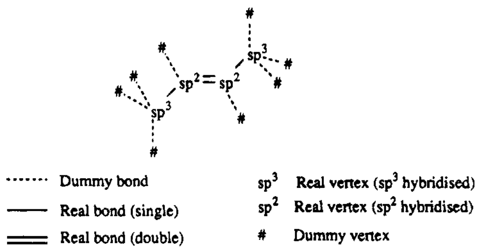

Figure 3. A template is made up of real vertices and dummy vertices. Real vertices represent non-hydrogen atoms, and they are connected to other real vertices by real edges. Dummy vertices define the direction of the free valencies of a real vertex.

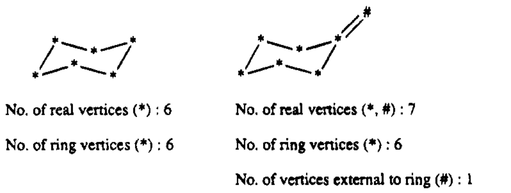
No. of real vertices (*) : 6
No. of ring vertices (*) : 6
No. of real vertices (*, #) : 7
No. of ring vertices (*) : 6
No. of vertices external to ring (#) : 1

Figure 4. Cyclic templates include rings systems that carry exocyclic double bonds.

reduces the number of templates that are required and thus reduces the combinatorial problem to a manageable size.

A **template** (Figure 3) is defined as a graph that has two different types of vertices: **real vertices** and **dummy vertices**. Real vertices represent non-hydrogen atoms in the template and they are labeled according to their hybridization and their environment but not by element type. Five different types of real vertices are defined: sp^3 vertices; sp^2 vertices occurring in either acyclic templates or within the ring in a cyclic template (Figure 4); exocyclic sp^2 vertices in cyclic templates; sp vertices; and aromatic vertices. The geometry of the vertices is tetrahedral, trigonal, or linear, according to the hybridization state. It is important to distinguish sp^2 vertices that belong to a ring from exocyclic sp^2 vertices that are directly attached to a ring since they can have different properties during template joining.

Dummy vertices are used to complete the valence of each of the real vertices and they define the orientation of the free valences. They are used during template joining to define the positions of the new vertices when a template is joined to a growing skeleton. Dummy vertices that remain in a solution skeleton are used to define the orientation of a hydrogen atom or an electron pair in the final molecule.

There are two kinds of edges in the template graphs: **real edges** and **dummy edges**. Real edges are edges between two real vertices and dummy edges correspond to edges between a real vertex and a dummy vertex. Dummy edges represent the direction of a new bond when a template is joined to an evolving skeleton or the direction of the hydrogen atoms or electron pairs in the final molecule.

Each real edge is labeled according to the type of bond it represents. The bond labels are as follows: single bonds in acyclic templates; single bonds in cyclic templates; double bonds in acyclic templates; double bonds as part of a ring; exocyclic double bonds in cyclic templates; triple bonds in acyclic templates; and aromatic bonds.

The length of an edge between two real vertices in a template is the corresponding carbon—carbon bond length. This value is used rather than one averaged over different element types, because in general the most common atoms in the generated molecules will be carbon atoms and carbon—carbon bonds are the most common bonds. Thus using carbon—carbon bond lengths will result in a more accurate shape for the final skeleton than would be found using averaged bond lengths.

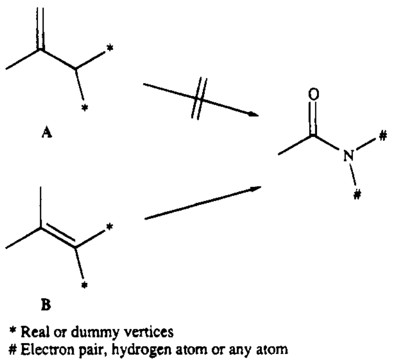

Figure 5. Skeleton B is in the correct conformation to produce the amide functional group by substituting the atoms and bonds as shown. Only minor changes in conformation are required. Skeleton A, however, is not in the correct conformation to produce the amide.

A value of 1.50 Å is used for the length of the edge between a real vertex and a dummy vertex. This represents an average bond length and its use facilitates template joining which involves overlapping dummy vertices from one template with real vertices from another.

There are some substructures for which the correspondence with the templates as defined is not straightforward, for example, the amide substructure (Figure 5). In fact this substructure cannot be properly represented by the template obtained by substituting the heteroatoms by carbon atoms (A). This is because all the atoms in the amide substructure are in the same plane. This problem is overcome by representing this substructure (and other similar substructures such as an ester group), by template B instead of A, so that the amide is formed during the atom substitution process by substituting into B and not A. Template B can be obtained by joining different combinations of templates.

There are also special templates for five- and seven-membered aromatic rings in which all the real edges have the same size and the dummy edges are all in the plane of the ring. This reduces the combinatorial problem, as the templates have a high degree of symmetry and allows the substitution of heteroatoms, if required, at any vertex.

CHOICE OF SUBSTRUCTURES

It is impractical to represent all possible substructures as templates even when the element type is ignored. This is because of the large number of possible combinations of atoms in different hybridization states and different bond types and because of the large number of conformations that are possible. Increasing the number of building blocks in SPROUT has an exponential effect on the size of the search graph. However, some of the hypothetical substructures and some conformations of a given substructure occur so infrequently that they can be safely ignored. Thus the number of substructures to include in the SPROUT template library is reduced considerably.

One objective for the design of SPROUT has been to maintain a distinction between the control strategy and the heuristics or knowledge used by the system. This allows the knowledge to be altered easily without requiring changes to the program. Thus the templates from which skeletons can be generated can be restricted by the user during a run. A module is currently under development that will allow the user to add new templates to the library. However the program is provided with a well balanced library of templates from which the user can select those he/she wants to use for each problem.

The template library was chosen by analyzing the frequency of occurrence of different substructures in a large database. This approach can be controversial since it will limit the type of skeletons obtained as solutions to a problem and also because it depends on the kind of molecules represented in the database, but it is inevitable that the risk of removing some interesting solutions is taken in order that the program can execute in reasonable time and that the number of solutions is manageable.

Selection of 2D Substructures. A study was undertaken of the relative occurrence of all possible bonding combinations of vertices and edges up to a given number of vertices in order to select common substructures represented by 2D graphs.

The database used for this search was the ORAC database,[19] version 7.7. This database consists of a large set of molecules (around 200 000) from different areas of chemistry so that it can be considered as representative in such a study. However, because it is a database of organic reactions, some relatively uncommon substructures can be selected due to the presence of unstable or strained molecules thus making the search less limited than if a database of commercially available compounds was used.

There is a trade off between the size of the fragments to include in the library and the number that must be included: larger fragments allow more atoms to be added in a single step; however, the number of substructures (and conformations) to consider increases exponentially as the number of atoms increases. Small fragments have the advantage that a smaller number of them is required to build the large majority of organic molecules. However, if the fragments are too small, then too many template joining operations are required. The building blocks used in SPROUT have therefore been limited to single rings and acyclic fragments that contain between one and four atoms.

A further reduction in the number of substructures to include is achieved if isolated sp^2 and sp vertices in the templates are forbidden. This considerably reduces the number of acyclic templates (by a factor of 5), simplifies the joining process since only single bonds need to be generated, and removes the need for a check for isolated sp and sp^2 vertices in the final skeleton.

Acyclic Fragments. In the limit, all acyclic substructures can be built from single atoms, if the set of building blocks consists of an sp^3 vertex, an sp^2 vertex, and an sp vertex. However, this would be very time consuming, and it was decided to include templates derived from a set of larger acyclic substructures.

Two factors were considered in choosing the maximum size for acyclic templates. Firstly, the number of conformers of an acyclic substructure increases exponentially when an additional atom is added. Secondly, usually it is desirable to limit the length of acyclic chains in molecules that are designed for binding to a receptor. This is because conformationally flexible molecules incur an entropic penalty on binding. Thus it was decided to limit the acyclic templates to represent substructures that have between one and four non-hydrogen atoms. Larger units can be built by joining these templates together. All the hypothetical substructures were considered that can be made by combining sp^3, sp^2, and sp vertices, except for those substructures that have terminal sp^2 and sp atoms (Figure 6). Of these, all except two were considered very common and not requiring any further analysis. The two exceptions are A=A=A=A and A#A−A#A, where A represents any atom and # represents a triple bond. A search was made in ORAC for these two substructures which proved that they are relatively uncommon and thus they have been excluded from the template library.

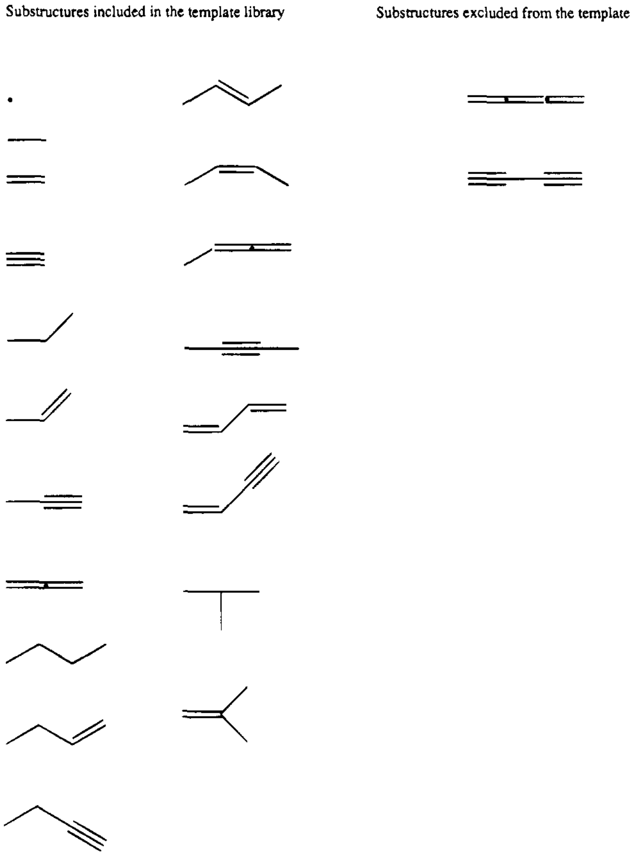
Substructures included in the template library
Substructures excluded from the template

Figure 6. The coverage of acyclic substructures in the template library.

Cyclic Substructures. Similar criteria were used to decide which cyclic substructures to include as templates. Initially the template library is restricted to single rings. Rings with three to seven atoms are included since they are the most commonly occurring ring sizes and because the number of conformations required for each ring is manageable.

All the hypothetical three−seven-membered rings were built by combining sp^3 and sp^2 vertices in all possible ways with all possible bond patterns. This resulted in 114 possible rings. Each of these rings was searched for in the ORAC database, with element types unspecified. A threshold of 0.2% was used (or 240 ORAC cards): any substructure that occurred more frequently was included in the template library; any substructure that was less frequent was omitted. More than half of the hypothetical rings were excluded in this way. The five-membered rings considered in the study are shown in Figure 7.

The exclusion of templates with terminal sp^2 atoms requires the presence in the library of cyclic templates with

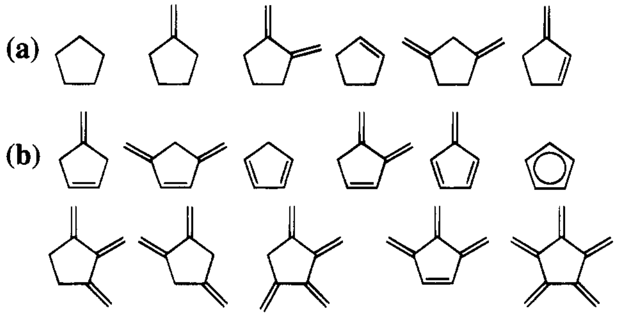

Figure 7. (a)The five-membered rings that are represented by templates. (b) The remaining five-membered rings that occur too infrequently to be included in the template library.

exocyclic sp^2 atoms. These templates are treated as cyclic templates, and it is possible to fuse templates using the bond to an exocyclic atom, see later.

SELECTION OF 3D CONFORMATIONS

Each template represents a single conformation of a substructure. Therefore each substructure must be analyzed to decide which of its conformers should be included in the library. Conformers other than the lowest energy conformers should be included since the templates will form components of larger molecules and steric or electronic constraints sometimes favor higher energy conformations of a substructure (for example, bulky substituents or intramolecular hydrogen bonds may favor a twist-boat conformation for a six-membered ring). Also, drugs often bind in conformations other than their lowest energy conformation. SPROUT should not, therefore, be restricted to generating minimum energy structures, and it should be possible to generate structures in a number of different conformations. In general, each substructure should be represented by a number of conformations and not only its low-energy conformations. However, in some cases this requires a large number of conformers; for example, consider the template representing cycloheptanone; if one boat, one chair, two twist boats, and two twist chair conformations are available, then the carbonyl double bond can be positioned in about 30 different positions. The inclusion of a large set of conformations for each substructure would rapidly result in an increase of the size of the combinatorial problem. Thus, there is a trade-off to be made between the number of conformations to include for each substructure and the variety of skeletons that can be generated.

Conformers can be selected by performing searches on a database of 3D structures such as the Cambridge Structural Database,[20] in a similar manner to that used for the 2D substructures. Other methods of selecting conformers include the use of systematic search techniques in molecular mechanics to find conformational minima. A good method would be to take into account the relative occurrence of the substructures and to include a more complete set of conformations for the most common substructures and just the low energy conformations for the less common substructures.

In the current version of SPROUT, the different conformations of the cyclic substructures were obtained by minimization using conformational analysis programs (MM2,[21] AMBER,[22] and CHARMm[23]). COBRA[12] was also used as it generates a range of conformations. Most of the conformations included are the low energy ones; however, for some of the most common substructures (for example cyclohexane) some others are also included. The work in this area is still under development.

For the acyclic substructures there is more conformation freedom, hence the number of conformers must be restricted. The following rules are applied: staggered conformations are included for bonds between sp^3-sp^3 vertices; the double bond is eclipsed between sp^2-sp^3 vertices; and s-E and s-Z conformations are included for sp^2-sp^2 bonds.

SEARCH GRAPH

The representation of the search process as a graph and the searching of the graph has already been described.[2] It is reviewed here briefly to provide context for the template joining processes and the template joining rules. The root of the graph represents the initial constraints. Goal nodes in the graph represent solution skeletons, i.e., skeletons that satisfy the constraints without violating the boundary. Intermediate nodes in the graph represent partial skeletons, i.e., skeletons that partially satisfy the steric constraints. The root node is expanded to nodes at the first level of the graph by positioning the templates that are specified as start templates at a given target site. Each orientation of a template gives rise to a node at the first level of the graph. Nodes at this level and subsequent levels are expanded as follows. A vertex of the partial skeleton represented by the node is selected; this vertex is called a seed vertex. The node is then expanded by joining all of the available templates to the partial skeleton at the seed vertex, according to the template joining rules, see later. A number of successor or child skeletons are produced, each one differing from its parent by the addition of a single template. Each successor is tested against the constraints and some user defined parameters, if it is accepted it is placed in the graph. After a node has been expanded, it is replaced in the graph with a different vertex selected as seed. This is repeated until all the available vertices have been used as seeds. The vertices that are used in a partial skeleton are determined by a user-defined parameter. The default is a single vertex called the **best vertex**; this is the vertex that is closest to an unsatisfied target site.

TEMPLATE JOINING

At each node expansion every template is joined to the partial skeleton represented by the node in all possible ways that involve the seed vertex. Each successor skeleton that is consistent with the constraints and the user-defined parameters is represented as a new node in the graph. Three types of join are considered in the current version of SPROUT (Figure 8): spiro, fusion, and new bond joining. The methods for spiro and fused joining templates are based on those used by the WIZARD and COBRA programs.[11,12] A new method has been developed for joining acyclic templates either to other acyclic templates or to cyclic templates; this join is called the **new bond** join.

Spiro joining and fuse joining are implemented in two steps in WIZARD and COBRA. The first step is a fitting step, where the two units are positioned ready for merging of the coordinates, but they remain as two separate units

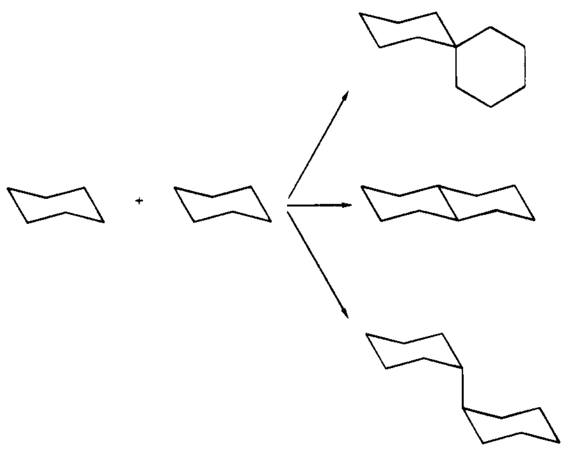

Figure 8. The three types of join implemented in SPROUT: spiro join, fuse join, and new bond join.

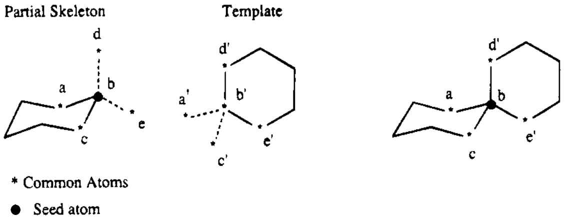

Figure 9. Spiro joining a cyclochexane template (containing the real vertices b′, d′, and e′ and the dummy vertices a′ and c′) to a partial skeleton (containing the real vertices a, b, and c and the dummy vertices d and e). The seed vertex, vertex b, is shown by the solid circle. All other vertices involved in the join are indicated by asterisks. The template is fitted to the partial skeleton so that b′ is superimposed on b and d′ and e′ are fitted to d and e, respectively. The resulting skeleton then has vertex coordinates given by a, b, c, d′, and e′. The dummy vertices d, e, a′, and c′ are removed. A second skeleton is generated by fitting b′ to b, d′ to e, and e′ to d.

without any conformational change. The second step is an assignment step where the coordinates for the resulting structure are chosen. The criteria for template joining in SPROUT are different from those used in WIZARD and COBRA. In SPROUT, the skeletons represent the approximate conformation of a final molecule, since they contain only carbon atoms. Optimum coordinates can be calculated following the atom substitution phase. During template joining, one of the units consists of a partial skeleton, the other unit is a template. An optimum fitting position is found for the template relative to the partial skeleton. Fitting involves finding the best superposition of the units so that vertices of the template are superimposed onto real and dummy vertices of the skeleton. Where there is overlap between real vertices in the skeleton and real vertices in the template the coordinates of the skeleton vertices are used. The coordinates of the remaining template vertices are used directly to form new vertices in the successor skeleton. Since the coordinates of the vertices in the parent partial skeleton are not altered during template joining, the process of identifying if one skeleton is a subskeleton of another is simplified.

Spiro joining involves superimposing one real vertex (the seed vertex) and two neighboring dummy vertices in the partial skeleton to one real vertex and two neighboring real vertices in the template, as shown in Figure 9. The template is joined in two different orientations. Both the real vertex in the template and in the skeleton must be sp^3 hybridized.

Fusion across one bond involves superimposing two real vertices and two neighboring dummy vertices in the partial skeleton to two real vertices and two neighboring real vertices in the template, see Figure 10. Each valid bond in the template is joined in each orientation to each ring bond that involves the seed vertex. There can be up to three bonds in the partial skeleton that involve the seed atom. A template bond is valid if the bond types and vertex types are compatible with the skeleton bond. Fusion can take place between single bonds if all the vertices are sp^3 hybridized, and a double bond can be fused to either a double bond or an aromatic bond. Ring templates bearing exocyclic sp^2 atoms can be fused to a double bond within a ring template, and fusion can take place across more than one bond as shown in Figure 11.

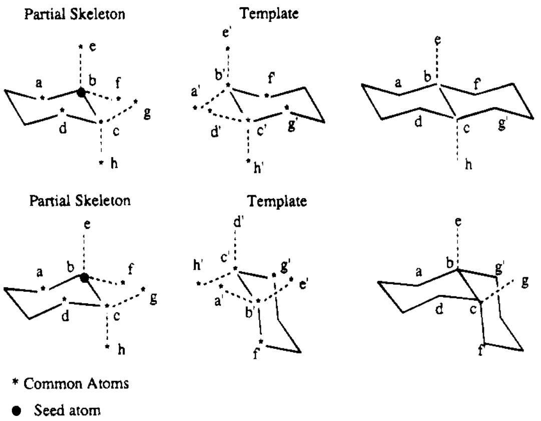

Figure 10. Fuse joining a cyclohexane template (containing the real vertices b′, c′, f′, and g′ and the dummy vertices a′, d′, e′, and h′) to the partial skeleton (containing the real vertices a, b, c, and d and the dummy vertices e, f, g, and h). The seed vertex, vertex b, is shown by the solid circle. All other vertices involved in the join are indicated by asterisks. First, the template is fitted so that the bond b′−c′ is superimposed on the bond b−c. Then f′ and g′ are fitted to f and g. The resulting skeleton has vertex coordinates given by a, b, c, d, f′, and g′. The dummy vertices f, g, a′, d′, e′, and h′ are removed. A second fused ring system is obtained by superimposing c′ to b and b′ to c. Two more fusions result when the second bond that includes the seed atom is processed, i.e., bond b−a.

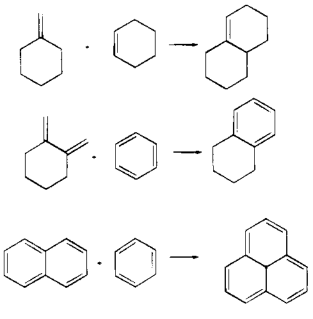

Figure 11. Examples of fused ring systems that result from template joining.

New bond joining involves superimposing a real vertex from the template to a dummy vertex that is bonded to the seed vertex in the skeleton. A dummy edge in the template must superimpose with a dummy edge in the skeleton, see Figure 12. Each vertex in the template is joined by each of its dummy edges to each dummy edge leading from the seed vertex in the skeleton. A new bond formed in this way is always a rotatable bond, and therefore a number of conformations can be produced about the bond. The conformations produced depend on the hybridization of the vertices: three staggered conformers are produced about a new bond between two sp^3 vertices; s-E and s-Z conformations are

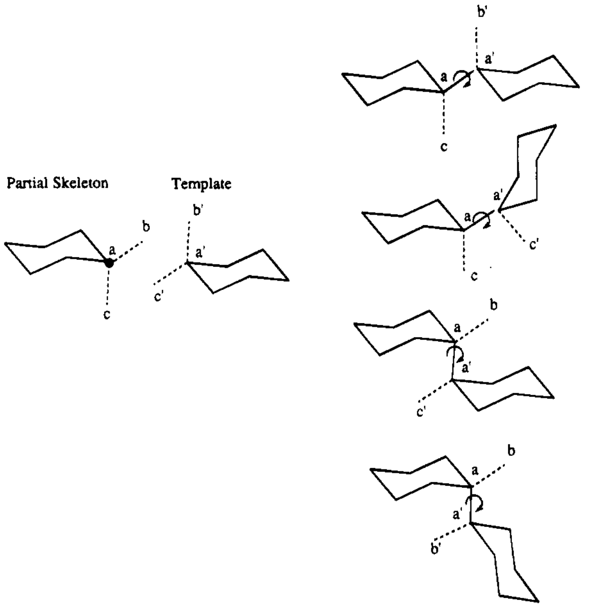

Figure 12. New bond joining a cylcohexane template (containing the real vertex a′ and the dummy vertices b′ and c′) to a partial skeleton (containing the real vertex a and the dummy vertices b and c). The seed vertex is vertex a, shown by the solid circle. Each time a new bond is formed a number of conformations are generated about the bond.

generated about a new bond between two sp^2 vertices; and when an sp^2 vertex is joined to an sp^3 vertex by a new bond, then the double bond is eclipsed to neighboring vertices of the sp^3 vertex. These are the most frequently occurring conformations about single bonds; however, other conformations do occur with relative frequency. It is also possible to relax these conditions to allow other conformations to be generated and then to select the most favorable conformations as successors. Different criteria can be used to sample these conformations, e.g., currently a Monte Carlo method has been implemented.[2]

Redundancy is avoided by taking account of the symmetry in a template. The vertices and bonds in a template are grouped according to their symmetry. All the vertices in a given group are identical, i.e., it is necessary to join the template to a partial skeleton using only one of these vertices; using any other vertex from the same group would generate an identical skeleton. The bonds are grouped similarly; only one bond from a group need be used for fuse joining to a partial skeleton; using any of the other bonds in the same group would result in an identical skeleton. One vertex is chosen arbitrarily from each group as representative; this is called a **representative vertex**. Similarly, one bond is chosen from each group and is called a **representative bond**. The representative vertices and bonds are flagged for a template, and only these are used during template joining. For example: the template representing benzene has one representative vertex, one representative bond within the ring, and one representative bond external to the ring (a dummy bond); and the template representing the chair conformation of cyclohexane has one representative vertex, two representative bonds within the ring, and two representative bonds external to the ring (dummy bonds). Symmetry considerations apply to the new template that is being joined to a partial skeleton and not to the partial skeleton itself.

Joining Rules. Several rules are used in the joining of templates: to reduce the time taken for a node expansion; to avoid the generation of unfavorable structures; and to avoid generating duplicate skeletons. The rules are encoded in a knowledge base that is separate from the program and can be modified easily.

Some general rules are as follows:

1. It is forbidden to superimpose any real vertex to a real vertex in an acyclic template. The absence of this rule would require longer acyclic templates for the same effectiveness.

2. Vertices from one cyclic template can be superimposed to vertices in another cyclic template, if they have compatible hybridization, to produce spiro and fused ring systems.

3. Double bond edges can be superimposed only to double or to aromatic edges, and sp^2 vertices can be superimposed only to sp^2 or aromatic vertices.

4. Aromatic edges can be superimposed to any double bond edge between two sp^2 vertices, and the vertices and edges involved become aromatic.

5. It is always forbidden to make an edge between two skeleton vertices if the number of vertices in the resulting ring is less than 8. This would result in the generation of a template which already exists in the library.

If a template is allowed to join to any other without restriction, the number of skeletons generated will become unmanageable. An example is used to illustrate the magnitude of the problem: if a seven-membered ring in a chair conformation is joined by a new bond to a vertex in a skeleton that has two free valences this results in 84 different skeletons, when only the staggered conformations are produced about the new bond. Therefore, heuristics have been introduced to restrict template joining by governing which templates can be joined together. These rules were derived using methods similar to those used for the selection of templates.

Some rules have been devised to prevent the joining of templates that produce unlikely substructures. Each of the templates was joined to every other template in turn by each of the joining methods. The resulting pairs of templates were grouped according to substructure. Searches were made in the ORAC database to find the frequency of occurrence of each of the substructures. This process is extremely time consuming, and it is very difficult to be exhaustive as the number of substructures that should be considered is enormous. The characteristics of the search process in the ORAC database, e.g., concerning aromaticity, and the limited time available means that it is possible that in a few cases some occurrences of substructures were missed. However, the results obtained are considered to represent the relative occurrence of the different substructures and have allowed the development of rules that are prescreens for template joining, allowing only those templates to be joined which can result in reasonable skeletons.

The searches suggested that a subset of the paired templates are sufficient to give interesting solutions. The searches were performed on a 2D database, but since each substructure exists in different conformations, and since some of them have a large set of representative vertices and bonds, it can be concluded that in fact a huge number of combinations have been excluded, most of which would give rise to poor solutions. This method can be extended to 3D to provide a more extensive set of rules. Work in this area has already begun and has shown that many more template combinations can be excluded.

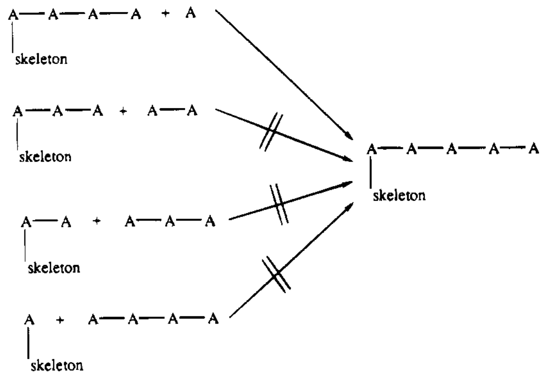

Figure 13. The template joining rules prevent the generation of duplicate skeletons by using different template.

Additional rules have been developed to prevent the generation of the same substructure by joining together different combinations of templates. This can occur frequently for acyclic templates.

The rules are summarized below for each of the join types.

New Bond Joining. Searches were performed for pairs of cyclic templates that are connected by a single acyclic bond. It was concluded that the most common combinations of rings joined in this way are six-membered rings to any other ring systems and five-membered rings to other five-membered rings. The other possibilities are excluded. It is assumed that any acyclic template can be joined to any cyclic template by a single acyclic bond.

For the joining of acyclic templates, rules have been developed that prevent the generation of duplicate skeletons by different routes. These are as follows:

1. It is forbidden to join any acyclic template to other acyclic templates consisting of from one to three vertices. Otherwise duplicated acyclic substructures would be generated, see Figure 13.

2. Any acyclic template can be joined to an acyclic template with four vertices (except the template representing $CH_3CH(CH_3)CH_3$). Acyclic substructures with more than four vertices are generated by this process.

3. The templates that can be joined to the template representing $CH_3CH(CH_3)CH_3$ are restricted to the single vertex acyclic template and any template containing double bonds. Joining any other templates, i.e., templates with more than one vertex where all the vertices are sp^3 hybridized, produces substructures that can be derived in other ways. In fact, all the acyclic substructures obtained by joining this template to a template with three sp^3 vertices can be generated using the template representing $CH_3CH_2CH_2CH_3$ and other templates with one to three sp^3 vertices (Figure 14).

4. There are some special templates, e.g., $CH_2{=}CH_2$ that are excluded from rule 1 above. This prevents the need for terminal acyclic sp^2 vertices, without the loss of solutions, see Figure 15.

The current template joining rules prevent the generation of the skeletons containing the substructures shown in Figure 16. However, this is not considered to be a problem since these substructures occur very infrequently in the ORAC database.

Bridged Joining. Searches were also performed in ORAC to analyze the frequency of occurrence of different bridged systems. The conclusion was that only a small subset of all the possible bridged systems are relatively common (using a cut-off of the presence of the substructure in 100 ORAC cards—equivalent to 10% of the occurrence of the most common bridge system).

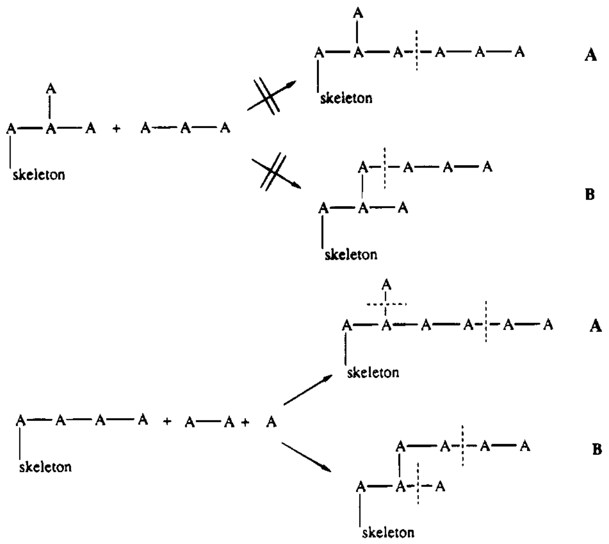

Figure 14. Some duplication is prevented by restricting the templates that can be joined to A–A(–A)–A. The joining of templates that have more than one vertex where all the vertices are sp^3 hybridized is forbidden since the same result can be achieved by starting with the straight chain template: A–A–A–A. "A" represents any element.

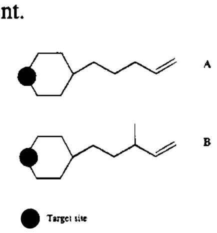

Figure 15. The acyclic joining rule 1 prevents the generation of skeleton A, grown out from the target site, since there is no 4 vertex chain with a terminal sp^2 atom. Skeleton B, however, can be generated, by first joining A–A–A–A to the ring and then joining A=A to this chain. The exclusion of template A=A from rule 1 allows it to be joined to chains with fewer than four vertices so that skeleton A can be generated by first joining A–A–A to the ring and then joining in A=A.

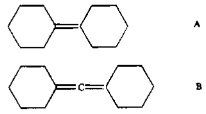

Figure 16. Substructures that cannot be generated as a consequence of the template joining rules.

Bridged joining is the most difficult type of join to implement, because it requires the overlapping of a large number of vertices and in some cases a considerable distortion of the templates involved. Therefore, it was decided that bridged ring systems should be included in the library rather than generating them by implementing a bridged joining method. In fact, the number of substructures to include is not very large (Figure 17), and these are quite rigid substructures so that in most cases each of them can be represented in the library by only one conformation.

Spiro Joining. A similar analysis using ORAC was made for substructures involving spiro joining. Using a threshold of occurrence in 74 ORAC cards (equivalent to 10% of the occurrence of the most common spiro system) allows the number of spiro joined pairs to be considerably reduced (Figure 18). The possibility of including spiro systems as

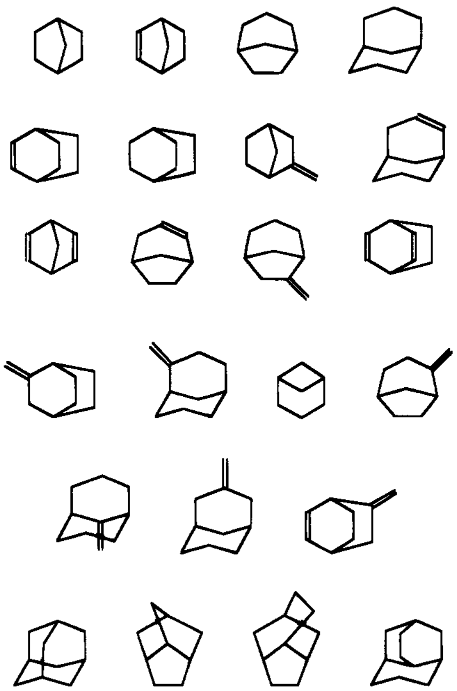

Figure 17. Simple and complex bridged ring systems shown in decreasing order of their frequency in the ORAC database.

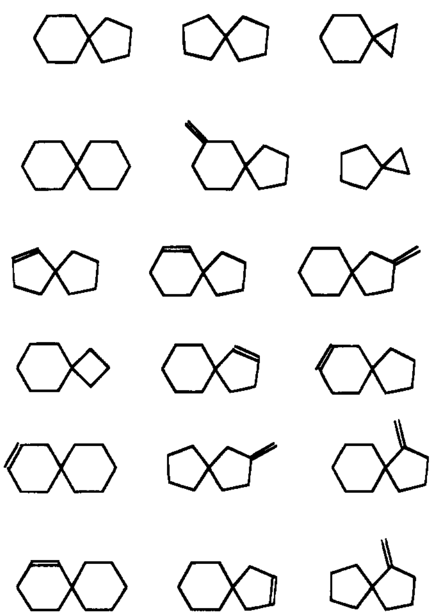

Figure 18. The most commonly occurring spiro rings systems found in the ORAC database, shown in decreasing order of frequency.

templates in the library and avoiding implementing the spiro joining method was considered. However, these substructures are much more flexible than the bridged ones, and this would imply the addition of a large number of templates to the library. Also the implementation of the spiro joining method does not present the same difficulties as the bridged join.

Some rules were developed in order to restrict the spiro join to generate only the most common spiro systems. Examples of these rules are as follows:

1. Six-membered rings with only sp^3 vertices can be spiro joined to three-, four-, five-, and six-membered rings with only sp^3 vertices, five-membered rings with one sp^2 vertex, and five- or six-membered rings with one double bond edge.
2. No spiro joining can be made to bridged systems.

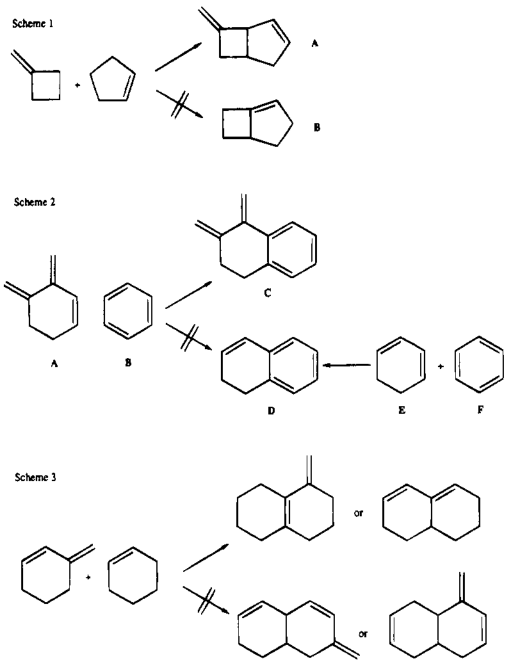

Figure 19. Application of some of the template joining rules for ring fusions.

Fused Joining. Searching for the occurrence of substructures consisting of two fused rings is more difficult compared with the other joins since this is a very common joining method. The search was extremely time consuming, and it was almost impossible to be exhaustive; however, it is considered that the results obtained are representative of the relative occurrence of the different substructures. The threshold used was occurrence in 120 ORAC cards, that is, 0.1% of the total of ORAC cards and about 1% of the number of cards that contain the most common fused substructure.

Some rules were also devised to avoid generating the less common substructures. Most of the rules are similar to those used for spiro joining, for example:

Aromatic six-membered rings can be fused to

1. four-, five-, six-, and seven-membered rings with only two sp^2 vertices and one double bond edge,
2. five-, six-, and seven-membered rings with three sp^2 vertices not separated by more than one sp^3 vertex and one ring double bond edge, and
3. aromatic five-, six-, and seven-membered rings.

Other rules were developed to avoid the generation of uncommon substructures or duplication:

1. It is forbidden to overlap more than two vertices in four-membered cyclic templates. This allows the generation of substructure A in Scheme 1 of Figure 19 but forbids the generation of B.
2. It is forbidden to overlap more than three vertices from one cyclic template. This prevents the generation of D in Scheme 2 of Figure 19 from A and B but allows it from E and F and thus avoids duplication.
3. If one of the templates has two or more double bond edges, if possible one of them should always be overlapped with another one in the other ring template (Figure 19, Scheme 3).

In general, the inclusion in the library of templates representing fused pairs of rings does not look very interesting as the number of conformations available is very large, and also because complex fused systems are very common and thus this joining method must always be implemented,

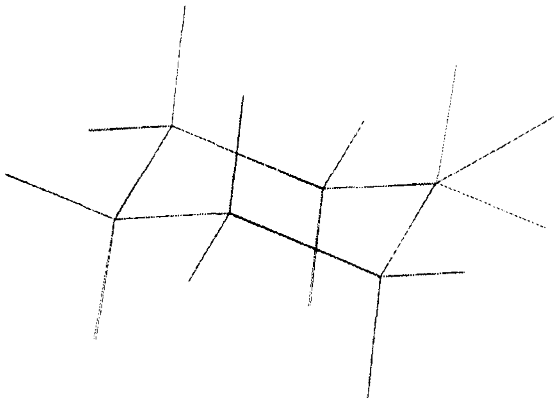

Figure 20. The templates representing the chair conformations of cyclohexane and cyclohexanone are superimposed by their ring vertices.

also it would be difficult to avoid duplication. However the inclusion of these templates in the library for very strained systems, for example, pairs of fused rings in which one of the rings is a three- or four-membered ring, should be considered. In fact, due to strain these systems are not very flexible, and thus the number of conformations to include in the library is reduced. Also, complex fused ring systems with more than one three- or four-membered ring fused to the same ring are not very common, and the joining of these strained cases which require a great distortion can be avoided.

CLASSES OF SIMILARITY

Each successor of a node must undergo a series of tests before it can be included in the search graph. One of these tests compares the partial skeleton with the boundary. If any of its vertices violate the boundary, then the partial skeleton is not included in the search graph. The relative shapes and sizes of the templates can be used in some cases to avoid generating partial skeletons that will violate the boundary, thus saving on processing time.

The concept of similarity is different for cyclic and acyclic templates and is described in the next paragraphs.

Cyclic Templates. The templates that represent cyclic substructures are grouped into classes of similarity according to the number of vertices in the ring, the similarities of the shapes of the rings (by comparing the signs of the torsion angles), and their orientation when joined to the skeleton by ring vertices. Cyclic templates that represent rings of the same size, with or without exocyclic vertices, can be included in one class of similarity. The orientation of the ring when it is joined to the skeleton is of particular importance for this classification.

The template representing cyclohexane in the chair conformation belongs to a different class of similarity than the template representing cyclohexane in the boat conformation because it has a different shape. However, the templates representing cyclohexane and cyclohexanone in the chair conformation belong to the same class of similarity because their ring shapes and the direction of free valences of the ring vertices are quite similar (Figure 20). It is also possible to find a good superimposition of the templates representing cyclohexene and 1,3-cyclohexadiene when only the ring vertices are considered because the shape of the rings is similar (Figure 21). However the direction of the free valences at the vertices that are overlapped is quite different, some are sp^3 in one ring and sp^2 in the other. This difference results in different orientations when the templates are added to a skeleton, and the volume occupied by them in each case can be completely different. A similar situation occurs when the templates for cyclohexane in a boat and a twist boat conformation are overlapped (Figure 22). Although all the vertices are sp^3, the difference in direction of the free valences can be enough to result in a big difference in the orientation of the template when it is joined to the skeleton.

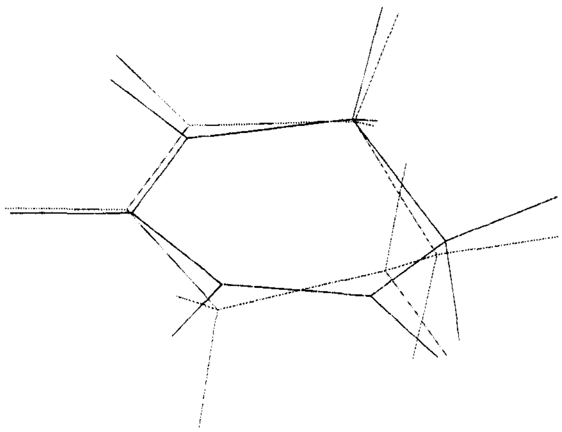

Figure 21. The templates representing cyclochexene and 1,3-cyclohexadiene are superimposed by their ring vertices. The rings are similar in shape—the maximum distance between corresponding ring vertices is 0.159 Å. However, the rings belong to different similarity classes because the directions of their free valencies are quite different.

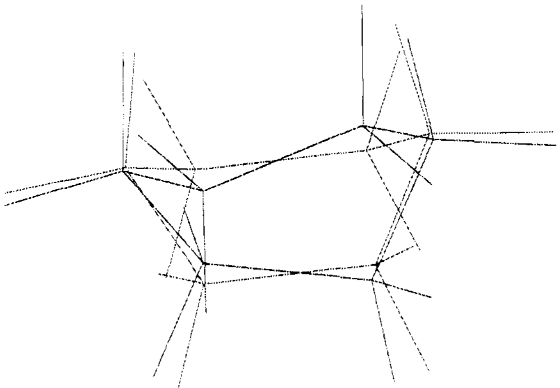

Figure 22. The templates representing cyclohexane in the boat and a twist boat conformation are superimposed. The rings are similar in shape—the maximum distance between corresponding ring vertices is 0.210 Å. However, the difference in the directions of the free valencies results in the templates being assigned to different classes.

Each class of similarity has a template called the *parent* template. An unsuccessful template joining operation with this template because of a boundary violation allows operations with the remaining templates in the class to be excluded from consideration. Therefore the parent template of the class is always the first template to be processed by the joining algorithm. The template with the smallest number of real vertices is chosen as the parent template of a class, this is the one that has minimum steric bulk. Thus, if joining this template to the skeleton causes the skeleton to violate the steric constraints, then the same will be true for all other members of the class when they are joined in a similar way, i.e., using ring vertices and in the same orientation in space. The templates representing cyclohexane and cyclohexanone

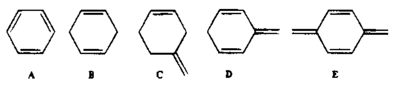

Figure 23. Templates belonging to the same similarity class. Template B is the parent template.

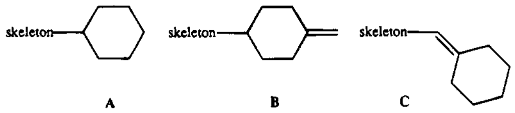

Figure 24. The similarity classes can be used to improve the efficiency of the program by preventing some template joining. If skeleton A is pruned from the search graph because it violates the boundary, then it is certain that skeleton B will also violate the boundary, since the cyclohexanone ring joined in the same way will occupy a large volume. Skeleton C however must be generated and tested against the boundary since the cyclohexanone is joined by the vertex that is external to the ring. Similarly, if skeleton D violates the boundary, then skeleton E must also violate it whereas skeleton F must still be generated and tested.

in the chair conformation belong to the same class and cyclohexane is the parent template.

The parent template should also have the maximum number of ring vertices with free valences in different hybridization states. This means that in spite of having a shape similar to the other templates in the class the range of directions to explore for its joining must be as large as possible. The templates shown in Figure 23, all belong to the same class of similarity, and template B is the parent template. In fact, template B is the same size as template A and is smaller than any of the others, it has vertices with free valences in two different hybridization states (sp^3 and sp^2), whereas A has vertices in only one hybridization state (sp^2). Thus B can be used to check all the possible orientations the templates can adopt when joined to the evolving skeleton by ring vertices, and if there is insufficient space for it, then it is possible to exclude all the others from consideration.

Assigning templates to classes according to their similarity can increase the efficiency of the template joining step: if the joining of a parent template to a partial skeleton, by a given joining method that involves ring vertices, fails for steric reasons, then it is no longer necessary to attempt to join the rest of the class. In the situation presented in Figure 24 if it is not possible to add the template representing cyclohexane to the skeleton (A), because it violates the boundary, then it is not necessary to consider the template representing cyclohexanone when joined as in B because it will occupy the same volume. However, if template representing cyclohexanone is added to the skeleton as in C, it will occupy a different volume, and this way of adding the template to the skeleton should be considered. Thus the concept of similarity classes of templates cannot be used to restrict the joining operations when vertices outside the ring are involved in the joining process.

Some of the classes even allow the exclusion of other classes from consideration in the joining process. If any ring with a planar shape and having free valences in the plane is excluded, this allows the exclusion of all planar rings that have more vertices (Figure 25). Also, if it is impossible to join cyclohexane in a chair conformation, it will also be impossible to join bridged systems having just cyclohexane rings in a chair conformation.

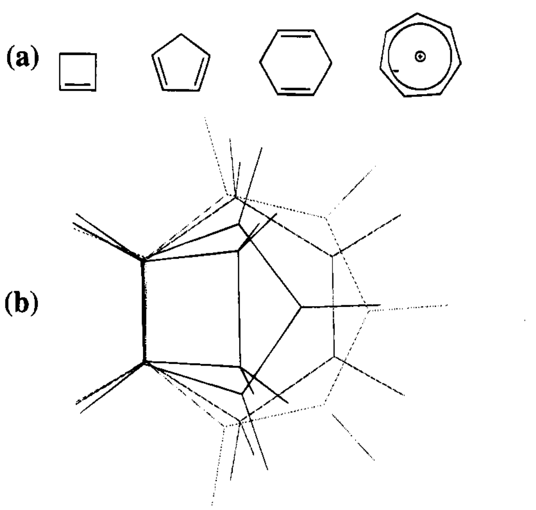

Figure 25. (a) A set of planar cyclic templates of different ring sizes. (b) The planar templates shown in (a) are superimposed by a ring double bond and one of the dummy bonds attached to the double bond. The templates are joined in order of increasing size starting with the smallest template. When addition of one of these templates cause a skeleton to violate the steric constraints, all large planar templates will also violate the constraints and therefore they can be excluded from consideration.

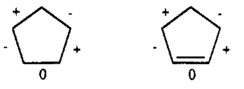

Figure 26. The signs of the torsion angles are shown for the templates representing cyclopentane and cyclopentene in the envelope conformation.

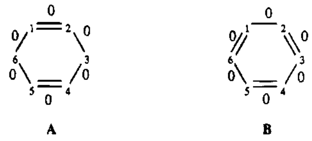

Figure 27. The signs of the torsion angles shown for the templates representing the planar conformation of 1,4-cyclohexadiene and benzene.

Templates are assigned to similarity classes by analyzing the torsion angles that include ring edges and the possibility of superimposing the template to the parent of the class by superimposing edges with similar torsion angles and vertices with similar hybridization states. Considering the templates representing cyclopentane and cyclopentene in the envelope conformation (Figure 26), if the ring edges are superimposed according to the signs of the torsion angles, there is no way of superimposing vertices with similar hybridization (two sp^2 vertices would be superimposed to two sp^3 vertices), so these two templates should be included in different classes of similarity. Although the shapes of the rings are quite similar, they can be added to the skeleton in different orientations. However, the same rule allows templates A and B in Figure 27 to be assigned to the same class. In fact, if vertices [1 2 3 4 5 6] in template B are overlapped to the vertices with the same labels in A, the torsion angles are in good agreement and vertices 1, 2, 4, and 5 in ring B are overlapped to vertices that have the same hybridization in ring A but not vertices 3 and 6. However, if the overlap is made between vertices [1 2 3 4 5 6] in template B and vertices [2 3 4 5 6 1] in template A, there will also be good agreement between the torsion angles in both rings, and 3 and 6 in ring B are overlapped with vertices having the same hybridization in ring A. Thus for each of the vertices in template B, there is a way of overlapping it with a vertex

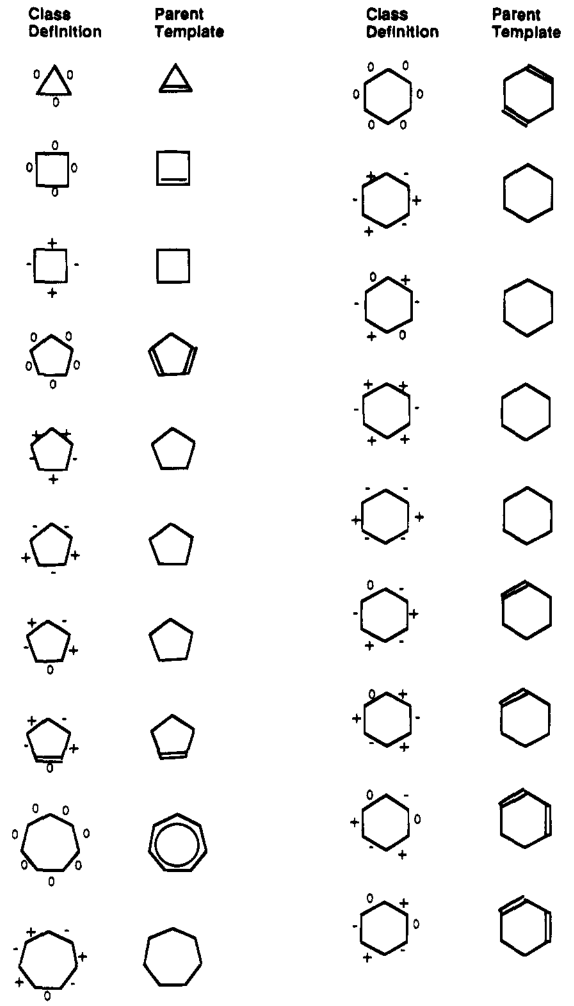

Figure 28. The similarity classes for three-, four-, five-, and six-membered rings, and two of the classes for seven-membered ring are shown along with the parent template of each class.

that has the same hybridization in ring A while having a good agreement of the torsion angles.

From the cases analyzed it was concluded that comparing the sign of the torsion angle is sufficient for assigning templates to a similarity classes. In all cases analyzed it was verified that this value was sufficient, and even a difference of about 20 degrees in the value of the torsion angle has a small effect on the shape of the ring and in the orientation of the free valences. However the analysis must take into account small deviations from a torsion angle of zero, and torsion angles between +5 and −5 are considered as zero.

Some of the available classes of similarity are presented in Figure 28 along with the parent template of each class.

Acyclic Templates. Templates representing acyclic substructures are divided into classes of similarity according to the hybridization of the vertices that have free valences (Table 1). For example, the templates representing CH_3-CH_3 and $CH_3CH_2CH_3$, that have only sp^3 vertices, belong to the same class of similarity. The template that represents $CH_3C\#CCH_3$ also belongs to the same class since all its free valences are in sp^3 vertices even though it has vertices with a different hybridization state. The template that represents $CH_2{=}CH_2$ and has only sp^2 vertices belongs to a different class of similarity, and the template representing CH_3-$CH{=}CH_2$ belongs to yet another class as it has free valences in sp^3 and sp^2 vertices. In fact it is the hybridization of the vertices that defines the direction of the free valences and thus the way in which templates are joined to the skeleton.

These classes of similarity allow the grouping of templates so that if the program tries to join the parent templates in classes 1 (just sp^3 vertices), 2 (just sp^2 vertices), or 4 (just sp vertices) to a skeleton and fails for steric reasons, it will not consider the remaining templates in the same class. In fact, they will have vertices in the same volume and require an even larger space. The exclusion of these classes also allows the exclusion of other classes, for example, if it is impossible to join ethane (the parent template in class 1), and ethene (the parent template in class 2), this allows the exclusion of propene and all other templates in class 3 (3 = 1 + 2 − sp^3 and sp^2 vertices).

In the future, the concept, definition, and use of the classes of similarity can be developed in order to use the similarity between different templates and the results of unsuccessful joins more efficiently and effectively.

ATOM SUBSTITUTION

The skeletons resulting from structure generation do not contain information about element type: the vertices are described by hybridization state alone, and the connections between them are described by bond type. There are a number of reasons for introducing different element types into a skeleton, these are as follows:

1. to promote binding of the ligand to the receptor via electrostatic interactions;

2. to stabilize certain intramolecular bonding situations or conformations, e.g., an enol is normally unstable, but substitution of an additional O gives a stable carboxylic acid;

3. to confer certain physical properties, e.g., solubility;

Table 1. Classes of Similarity for Acyclic Templates

classes	templates included	parent template	templates excluded
class 0	CH_4	CH_4	If it is impossible to add this template, all the other templates are excluded.
class 1	templates with all free valences at sp^3 vertices	$CH_3{-}CH_3$	If it is impossible to join the parent template, any other template in this class is excluded.
class 2	templates with all free valences at sp^2 vertices	$CH_2{=}CH_2$	If it is impossible to join the parent template, any other template in this class is excluded.
class 3	templates with free valences at sp^3 and sp^2 vertices	none	If it is impossible to add the parent templates in class 1 and 2, any template in this class is excluded.
class 4	templates with all free valences at sp vertices	CH#CH	If it is impossible to join the parent template, any other template in this class is excluded.
class 5	templates with all free valences at sp^3 and sp vertices	none	If it is impossible to add the parent templates in class 1 and 4, any template in this class is excluded.
class 6	templates with all free valences at sp^2 and sp vertices	none	If it is impossible to add the parent templates in class 2 and 4, any template in this class is excluded.

Table 2. Properties That Can Be Assigned at Target Sites

hydrogen bonding	charge
donor	neutral
acceptor	positive
donor or acceptor	negative
not specified	not specified

4. to facilitate ease of synthesis, e.g., from a synthetic viewpoint two rings connected by a hydrocarbon chain can be simplified by substituting heteroatoms into the chains to act as cleavage sites in a retrosynthetic plan.

The atom substitution process currently implemented in SPROUT addresses the first of these points. Atoms are substituted around the vertices of the skeleton that satisfy the target sites. Heteroatoms can be substituted for vertices that have the same hybridization state and therefore geometry. The target sites are designed to model localized interactions such as hydrogen bonding and charge−charge interactions. In practice, the substitutions are performed at the level of functional groups, rather than as single atoms or as complete templates. Functional groups are used since these provide an adequate description of the electrostatic properties of atoms in different environments without requiring an enormous number of different substructural fragments.[24−26]

Properties are associated with the target sites prior to structure generation. The hydrogen bonding and charge values that are available are shown in Table 2. Currently these are assigned manually via a graphical interface, however, an automated method of identifying interaction sites within a receptor site is under development. When a vertex satisfies a target site during skeleton generation, it is labeled with the properties that were specified at the target site.

The method of atom substitution is knowledge-based and uses a library of functional groups. Each entry in the library consists of a description of the vertex and bond features that must be found in the skeleton in order for the functional group to be a valid substitution and a rule that describes the necessary atom and bond substitutions to be performed. The pattern of vertex and bond features are described using a language called PATRANII that is similar to SMILES[27] (PATRANII is based on the PATRAN notation used in the LHASA program[28]). PATRANII is capable of describing any chemical substructure. The element types of the vertices are not specified in the PATRANII pattern since the patterns themselves correspond to subskeletons rather than substructures. A linear string details the vertices and bonds alternately, and each vertex and bond can have features associated with it. Examples of the vertex features in use are listed in Table 3, and some bond features are shown in Table 4. The substructural information of the functional group is stored within its associated rule. The rule specifies the actions to be performed as a result of matching the pattern with a skeleton. An example entry is for the secondary amide functional group shown in Figure 29.

```
CHEMICAL-LABEL <Secondary Amide Sub>
...STARTP
...X[HS=3];HACCEPTOR]-X=X[HS=1];[HBOTH]-X
...ENDP

RULE
IF Secondary Amide Sub
THEN
        substitute-atom 1 with O
        substitute-bond 1 with =
        substitute-atom 2 with C
        substitute-bond 2 with -
        substitute-atom 3 with N
        substitute-atom 4 with C
END-THEN
```

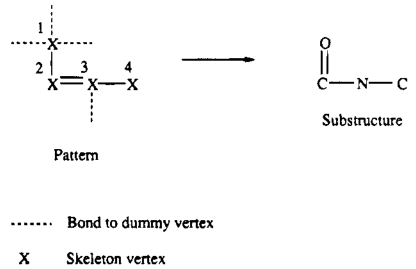

Figure 29. The functional group library entry for a secondary amide. The PATRAN string describes the pattern of vertices and bonds that must be present in the skeleton. Vertex 1 must be a hydrogen bond acceptor, and vertex 3 must be either a hydrogen acceptor or donor. The dashed lines represent dummy vertices that must be present in the skeleton. The rule describes the conversion of the pattern to the secondary amide substructure shown on the right.

The steps involved in performing atom substitution on a skeleton are shown in Figure 30 and are described below.

Perceive Chemical Features. Information concerning the input skeleton is extracted. The information includes the following: (i) a list of the vertices that satisfy the target sites; (ii) a list of all synthetically significant rings, i.e., all three- to seven-membered rings and the smallest set of smallest rings for larger rings; (iii) a list of fused, spiro, and bridged ring systems; (iv) a list of aromatic atoms and bonds; and (v) the number of hydrogens is calculated for each vertex. This information is necessary for some functional groups where the environment of the group must be defined for a valid substitution. For example, the PATRANII string for the pattern that represents the hemiacetal substructure is

X-[RINGS=5,6]X[HS=2];[HACCEPTOR]-[RINGS=5,6]X-X[HS=3];[HBOTH]

This states that the bonds between vertices 1 and 2 and between vertices 2 and 3 must be in a five- or six-membered

Table 3. Examples of the Vertex Features That Can Be Set for a Functional Group Pattern

feature	description	feature values
HS	the number of hydrogens on the vertex	0,1,2,3
ARYL	whether the vertex is aromatic or not	YES, NO, EITHER
RINGS	size of the ring containing the vertex	*Any number*, YES, NO, EITHER
HACCEPTOR	hydrogen bond acceptor	NONE
HDONOR	hydrogen bond donor	NONE
HBOTH	hydrogen bond acceptor or donor	NONE
NEUTRAL	neutral vertex	NONE
POSITIVE	positively charged vertex	NONE
NEGATIVE	negatively charged vertex	NONE
LIPO-YES	vertex has lipophilic character	NONE
LIPO-NO	vertex does not have lipophilic character	NONE

Table 4. Examples of the Bond Features That Can Be Set for a Functional Group Pattern

feature	description	feature values
RINGS	the size of the ring containing the vertex	*Any number*, YES, NO, EITHER
FUSION	the bond is fused	YES, NO, EITHER

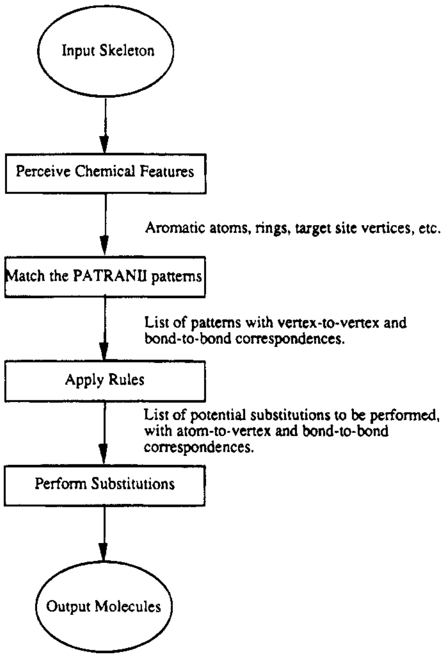
Input Skeleton
Perceive Chemical Features
Aromatic atoms, rings, target site vertices, etc.
Match the PATRANII patterns
List of patterns with vertex-to-vertex and bond-to-bond correspondences.
Apply Rules
List of potential substitutions to be performed, with atom-to-vertex and bond-to-bond correspondences.
Perform Substitutions
Output Molecules

Figure 30. Flow chart for the atom substitution process.

ring; vertex 2 must have a connectivity of 2; vertex 4 must have a connectivity of 1; vertex 2 will map to a hydrogen bond acceptor; and vertex 4 will map to either a hydrogen bond acceptor or donor.

Match the PATRANII Patterns. Each of the PATRANII patterns is matched against the skeleton. Matches that contain the target site vertices of the skeleton are stored in a list with the pattern vertex to skeleton vertex and pattern bond to skeleton bond correspondences. There is a correspondence between an atom of a functional group and a vertex of a skeleton if the hybridization state of the atom matches the hybridization of the vertex, they have the same connectivity, and the properties match.

Apply Rules. Each pattern in the list found above is transformed to a substitution pattern by application of its associated rule. The substitutions list consists of functional groups with the functional group atom to skeleton vertex and functional group bond to skeleton bond correspondences.

Perform Substitutions. A list of substitutions is built up for each target site. If there are no matches for a target site, then that target site is ignored during the substitutions phase. All combinations of the functional groups that are possible at each target site are produced, without overlap of the functional groups. Identical molecules are removed. Any vertices that remain unsubstituted default to carbon.

Note that each skeleton may give rise to many heterosubstituted derivatives because of the many possible combinations of functional groups.

Worked Example. The steps involved in atom substitution are illustrated for the skeleton shown in Figure 31. The skeleton was generated to satisfy four target sites. Two of the target sites have been assigned as hydrogen bond donors and two as hydrogen bond acceptors, as shown. Initially, the skeleton is analyzed to extract important features such as the six-membered aromatic ring and the number of hydrogens on each vertex, assuming that all the vertices represent carbon atoms. This information is used when the PATRANII patterns are matched against the skeleton. PATRANII patterns representing 11 functional groups are found as matches when the properties assigned to the vertices in the skeleton are taken into account. The functional groups are listed in Table 5 along with the mapping to skeleton vertices. A list of potential substitutions is then built from the rules for atom and bond substitutions that are associated with each pattern. The functional groups are then combined to produce molecules; all combinations are considered but some are omitted, for example, when the functional groups overlap. A total of eight molecules is produced, shown in Figure 32. No molecules are produced using the ketone functional group mapped to vertices 5 and 6 of the skeleton since all of the mappings of vertex 4 also include vertex 5. This condition prevents the formation of a carboxy group, which is already contained in the functional group library,

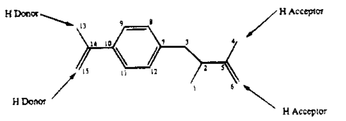

Figure 31. A skeleton that satisfies four target sites. Two of the target sites are labeled as hydrogen bond donors and two are labeled as hydrogen bond acceptors.

Table 5. Functional Groups Found as Matches Together with the Skeleton Vertices That They Map to

functional group	skeleton vertices
nitro	6 5 4
ketone	6 5
hydroxy	13 14 4 5
carboxy	15 14 13 6 5 4
guanidino	15 14 13 6 5 4
primary amine	13 14 4 5
primary amide	4 5 6

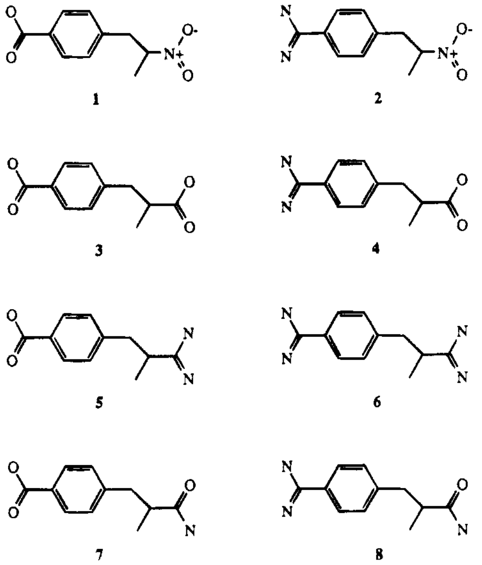

Figure 32. The eight molecules produced after performing atom substitutions on the skeleton shown in Figure 32.

from a ketone and hydroxy. Molecule 7 is produced by substituting a carboxy and a primary amide functional group onto the skeleton.

CONCLUSIONS

This paper is mainly concerned with the definition and selection of the templates and template joining rules that are applied to the problem of structure generation in the SPROUT program. The potential of SPROUT for generating novel structures has already been demonstrated for a range of problems.[1-3]

Currently efforts are being directed toward the development of SPROUT in a number of areas. These include developing a more flexible approach for specifying both hydrogen bonding and hydrophobic target sites; optimizing the conformations of the molecules following atom substitutions; developing of a more flexible interface that will allow the user to interact with the search graph during processing; improving the efficiency of the algorithm so that it can be applied more effectively to large receptor sites; and developing methods for ranking the output, e.g., by estimating their synthetic accessibility and by scoring their fit to the receptor. The building of the template library and template joining rules is also under continuing development as the variety and quality of the structures that can be generated is heavily determined by the templates that are available and the ways they are joined together.

ACKNOWLEDGMENT

We gratefully acknowledge STRIDE, FEDER, and JNICT for financial support to P.M. and J.L. and NATO for financial support to P.M. and S.S.

REFERENCES AND NOTES

(1) Gillet, V. J.; Newell W.; Mata, P.; Myatt, G.; Sike, S.; Zsoldos, Z.; Johnson, A. P. SPROUT: Recent Developments in the *De Novo* Design of Molecules. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 207–217.

(2) Gillet, V. J.; Johnson, A. P.; Mata, P.; Sike, S.; Williams, P. SPROUT: A Program for Structure Generation. *J. Comput.-Aided Mol. Des.* **1993**, *7*, 127–153.

(3) Gillet, V. J.; Johnson, A. P.; Mata, P.; Sike, S. Automated Structure Design in 3D. *Tetrahedron Comput. Method.* **1990**, *3*(6C), 681–696.

(4) Nilsson, N. J. *Principles of Artificial Intelligence*; Springer-Verlag: 1982.

(5) Nishibata, Y.; Itai, A. Automatic Creation of Drug Candidate Structures Based on Receptor Structure. Starting Point for Artificial Lead Generation. *Tetrahedron* **1991**, *47*, 8985–8990.

(6) Rotstein, S. H.; Murcko, M. A. GenStar: A Method for *De Novo* Drug Design. *J. Comput.-Aided Mol. Des.* **1993**, *7*, 23–43.

(7) Pearlman, D. A.; Murcko, M. A. CONCEPTS: New Dynamic Algorithm for *De Novo* Drug Suggestion. *J. Comput. Chem.* **1993**, *14*(10), 1184–1193.

(8) Lindsay, R. K.; Buchanan, B. G.; Feigenbaum, E. A.; Lederberg, J. *Applications of Artificial Intelligence for Organic Chemistry—The DENDRAL Project*; McGraw-Hill: 1980.

(9) Carhart, R. E.; Smith, D. H.; Gray, N. A. B.; Nourse, J. G.; Djerassi, C. GENOA: A Computer Program for Structure Elucidation Utilizing Overlapping and Alternative Substructures. *J. Org. Chem.* **1981**, *46*, 1708–1718.

(10) Abe, H.; Yamasaki, T.; Fujiwara I; Sasaki, S. Computer-Aided Structure Elucidation Methods. *Anal. Chim. Acta* **1981**, *133*, 499–506.

(11) Dolata D. P.; Carter R. E. WIZARD—Applications of Expert System Techniques to Conformational-Analysis. 1. The Basic Algorithms Exemplified on Simple Hydrocarbons. *J. Chem. Inf. Comput. Sci.* **1987**, *27*, 36–47.

(12) Leach, A. R.; Prout, K.; Dolata, D. P. An Investigation into the Construction of Molecular Models by the Template Joining Method. *J. Comput.-Aided Mol. Des.* **1988**, *2*, 107–123.

(13) Moon, J. J.; Howe, W. J. Computer Design of Bioactive Molecules: A Method for Receptor-Based *De Novo* Ligand Design. *Proteins: Struct., Funct., Genet.* **1991**, *11*, 314–328.

(14) Rostein, S. H.; Murcko, M. A. GroupBuild: A Fragment-Based Method for *De Novo* Drug Design. *J. Med Chem.* **1993**, *36*, 1700–1710.

(15) Böhm, H. J. LUDI: Rule-Based Automatic Design of New Substituents for Enzyme Inhibitor Leads. *J. Comput.-Aided Mol. Des.* **1992**, *6*, 593–606.

(16) Eisen, M.; Wiley, D. C.; Karplus, M.; Hubbard, R. E. HOOK: A Program for Finding Novel Molecular Architectures that Satisfy the Chemical and Steric Requirements of a Macromolecule Binding Site. Submitted to *Proteins: Struct., Funct., Genet.*

(17) Tschinke, V.; Cohen, N. C. The NEWLEAD Program: A New Method for the Design of Candidate Structures from Pharmacophoric Hypotheses. *J. Med. Chem.* **1993**, *36*, 3863–3870.

(18) SPROUT2. Manuscript in preparation.

(19) Hopkinson, G. A; Cook, T. P.; Buchan, I. P. Computer Treatment of Chemical Reactions and Synthetic Problems. In *Chemical Information Systems: Beyond the Structure Diagram*; Bawden, D., Mitchell, E. M., Lipkowitz, K. B., Eds.; Ellis Horwood Limited: 1990.

(20) Allen, F. H.; Bellard, S.; Brice, M. D.; Cartwright, B. A.; Doubleday, A.; Higgs, H.; Hummelink, T.; Hummelink-Peters, B. G.; Kennard, O.; Motherwell, W. D. S.; Rodgers, J. R.; Watson, D. G. The Cambridge Crystallographic Data Centre: Computer-Based Search, Retrieval, Analysis and Display of Information. *Acta Crystallogr.* **1979**, *B35*, 2331–2339.

(21) Buckert, U.; Allinger, N. L. *Molecular Mechanics*; American Chemical Society: Washington, DC, 1982.

(22) Weiner, S. J.; Kollman, P. A.; Nguyen, D. T.; Case, D. A. An All Atom Force Field for Simulations of Proteins and Nucleic-Acids. *J. Comput. Chem.* **1986**, *7*, 230.

(23) Momany, F. A.; Rone, R. Validation of the General Purpose QUANTA 3.2/CHARMm Force Field. *J. Comput. Chem.* **1992**, *13*(7), 888–900.

(24) Chau, P.-L.; Dean, P. M. Automated Site-Directed Drug Design: The Generation of a Basic Set of Fragments to be used for Automated Structure Assembly. *J. Comput.-Aided Mol. Des.* **1992**, *6*, 385–396.

(25) Chau, P.-L.; Dean, P. M. Automated Site-Directed Drug Design: Searches of the Cambridge Structural Database for Bond Lengths in Molecular Fragments to be used for Automated Structure Assembly. *J. Comput.-Aided Mol. Des.* **1992**, *6*, 397–406.

(26) Chau, P.-L.; Dean, P. M. Automated Site-Directed Drug Design: An Assessment of the Transferability of Atomic Residual Charges (CNDO) for Molecular Fragments. *J. Comput.-Aided Mol. Des.* **1992**, *6*, 407–426.

(27) Weininger, D. Smiles. 3. Depict. Graphical Depiction of Chemical Structures. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 237–243.

(28) Hopkinson, G. A. *Computer-Assisted Organic Synthesis Design*. Ph.D. Thesis, Leeds University, 1985.

CI940300Y