Journal of Computer-Aided Molecular Design, 9 (1995) 65–86
ESCOM

J-CAMD 279

DBMAKER: A set of programs to generate three-dimensional databases based upon user-specified criteria

Chris M.W. Ho and Garland R. Marshall*

Center for Molecular Design, Washington University, St. Louis, MO 63130, U.S.A.

Received 9 April 1994
Accepted 30 August 1994

Keywords: Structure generation; Database searching; CONCORD; Drug design; De novo design; Molecular graphics

Summary

DBMAKER is a program that, in conjunction with CONCORD, generates three-dimensional structural databases. Numerous user-defined parameters monitor content, composition, size and connectivity information, but allow the program to generate random compounds within the scope of these constraints. SMILES string representations are generated, and conversion to 3D is performed by CONCORD. This assures high-quality 3D structures and portability to numerous proprietary storage formats. Methods are described to maintain compound registration, allowing database expansion as required without duplication.

Introduction

The majority of drugs are noncovalent ligands that have been designed to bind and modulate specific enzymes or receptors. When administered properly, desired physiological effects are initiated [1]. De novo development of such a drug necessitates a thorough understanding of the biochemistry underlying its action. A successful drug must bind its target with great affinity, yet retain specificity to limit side effects. This requires the medicinal chemist to produce compounds that exhibit exceptional steric and electrostatic complementarity with the biological target [2].

The process of ligand design and optimization is nontrivial [3]. To aid in this task, medicinal chemists have turned to computer-aided design tools [4]. In particular, the utility of 3D database search and retrieval methods has become recognized [5,6]. These techniques require a biochemical target that has been well characterized structurally. In particular, its ligand-binding mechanism must be known [7]. The 3D orientation of the key functional groups which a potential drug must contain for receptor recognition and association is termed the pharmacophore [8,9]. It is the goal of the medicinal chemist to develop novel chemical architecture (i.e. scaffolds) that position these pharmacophoric groups, or their bioisosteres, in the correct 3D arrangement for specific interaction with the therapeutic target.

To implement 3D searching, an investigator first transforms the active site geometry, or pharmacophore, into a 3D search query. The query specifies the relative position of each pharmacophoric element and acceptable atom types. Other traits are specified as well, including the steric fields the ligand must occupy together with spatial tolerances. The query is then used to search a 3D structural database for matching structures. Most software packages contain sophisticated features that permit fine discrimination of acceptable compounds. Desired structures not only fix the pharmacophoric components in the correct orientation for interaction with the binding site, but also possess novel (and potentially proprietary) scaffolding. A unique compound that contains the necessary elements for binding can be used as a starting point for further optimization.

A source of 3D information is required to begin query processing. Databases can be classified into two main types: (i) crystallographic, and (ii) structures whose geometry is calculated from chemical heuristics and statistical information.

The two most well-known crystallographic databases are the Cambridge Structural Database (CSD) [10], which contains approximately 100 000 structures, and the Brookhaven Protein Databank (PDB) [11]. Of these, the CSD is more useful to the medicinal chemist since it contains structural data for small molecules.

*To whom correspondence should be addressed.

Other chemical databases contain structures whose geometries have been calculated using statistics and heuristics derived from the laws of chemistry. The most widely used program available to generate such structures is CONCORD, written by Pearlman and co-workers [12]. This program accepts SMILES [13] string input and generates a single low-energy conformer per compound. Due to its ease of use and rapid operation, it is the standard program with which many pharmaceutical firms, as well as the Chemical Abstracts service [14], generate their 3D databases.

There are several commercially available databases which provide structures in CONCORD format. The most extensive of these is perhaps the Chemical Abstracts database [14], which contains nearly five million structures for which coordinates have been generated to date. Others include the CMD 3D database (Comprehensive Medicinal Chemistry), the MDDR 3D database (MACCS-II Drug Data Report, >25 000 compounds), the FCD 3D database (Fine Chemicals Directory, >65 000 compounds) and the CMC 3D database (>5000 compounds), all from Molecular Design Ltd. [15].

Commercial databases are also available from software developers who have written their own structure-generating routines. Chemical Design Ltd. produces a 3D database searching package which includes a proprietary structure-generating algorithm [16]. Databases available for use with their software include the Chapman and Hall Dictionary of Drugs (>13 000 compounds), the Dictionary of Fine Chemicals (>120 000 compounds) and the Dictionary of Natural Products (>54 000 compounds).

Bartlett et al. [17] have recently generated two databases, TRIAD (TRIcyclics for Automated Design) and ILLIAD (Inter-LInkers for Automated Design), that contain energy-minimized molecular structures representing a comprehensive collection of tricyclic and acyclic hydrocarbons. TRIAD contains over 400 000 unique molecules, while ILLIAD contains more than 100 000 structures representing low-energy conformations of 35 000 molecules. Both databases are designed to interface with CAVEAT, a 3D database searching program developed by Bartlett et al. [18].

The success of implementing 3D search technology depends greatly upon the quality and thoroughness of one's database. Large pharmaceutical firms may own databases containing hundreds of thousands of compounds. However, small companies or those in academia may have limited in-house 3D information. Thus, one may be forced to purchase a commercially available database such as those mentioned above. This has its own drawbacks. Many of these databases are integrally associated with specific, proprietary molecular design software. As such, one is usually not able to purchase one without the other. This can become quite expensive, and the software may not provide all the desired functionality. Furthermore, once a database is purchased, the investigator is obviously restricted to the compounds contained within it. Databases each have inherent strengths and weaknesses with regard to their inclusion of particular chemical classes. If available databases are deficient in the compounds needed to solve a particular problem, their utility is limited.

Previous investigators have addressed these matters. Martin and Van Drie described a system which 'mutates' SMILES strings and generates 3D structures using CONCORD [12]. Such a system can be used to diversify a given database by generating derivatives. Furthering this concept, we have developed a set of programs, collectively called DBMAKER, that allow investigators to generate their own 3D structural databases with the help of CONCORD. Numerous user-defined parameters monitor content, composition, size and connectivity information, but allow the program to generate random compounds within the scope of these constraints. SMILES string representations [13] are generated, and then converted to 3D structures with CONCORD [12]. This assures both the quality of the 3D structures produced, as well as the ability to generate them in a number of proprietary formats. Methods are described to maintain compound registration, thereby preventing duplication. This allows one to continually enlarge old databases by the addition of new compounds as they are required.

Computational Methods

Implementation

DBMAKER is written in C and presently runs on numerous platforms. The program requires CONCORD [12] for operation. Structures can be generated in many proprietary formats (as supported by CONCORD), including SYBYL [19], ALCHEMY [19], MDL/MACCS [15], MMP2 [20], MOPAC [21], GAUSSIAN [22], SAVOL [23], ORTEP [24], CSSR [16] and PDB [11].

DBMAKER consists of five modules: DBMAKER, DBEXTRACT, DBCOMPRESS, DBCROSS and DB-

I. **Generation of SMILES strings from user parameters:**
DBMAKER

II. **Processing of DBMAKER-generated SMILES strings by CONCORD to produce unique SMILES representations.**

III. **Extraction of these representations from CONCORD log file:**
DBEXTRACT

IV. **Comparison with previously generated strings in current database to determine duplication.**

V. **Isolation of new structures to be generated by CONCORD:**
DBCOMPRESS

VI. **Generation of 3D coordinates with CONCORD.**

VII. **Structure optimization and 3D database packing:**

Fig. 1. Overview of the database generation process.

```
Frequency of template usage                        0.50

Frequency of use for each template                 0.40   +1+++++1
                                                   0.20   N++++N
                                                   0.20   ++++C(=O)O
                                                   0.20   +++C(=O)N++

Number of backbone atoms              Frequency         Num
in non-cyclic structure                 1.00             7

Total number of sidechains              0.05             0
and associated frequency                0.35             1
                                        0.40             2
                                        0.20             3

Distribution of sidechain lengths       0.50             1
                                        0.40             2
                                        0.10             3

% Branching of side chains 3 C and up          0.33

MAKEUP
Backbone freq.      Sidechain freq.  Symbol     #atoms        # Connections
     0.80                0.70          C           1               4
     0.10                0.10          N           1               3
     0.05                0.10          O           1               2
     0.05                0.10          C(=O)       1               2

Requirements for BACKBONE:       Number      SMILES symbols
                                   = 0       NO ON NN OO

Requirements for SIDECHAINS:
                                   = 0       NO ON NN OO
                                   = 0       ((

Requirements for COMPLETE  STRING
          = 0         N(O  N(N  O(N  O(O     prevents bonded heteroatoms
          > 0         N O                    at least one heteroatom

Boundary parameters for program termination
     Number of failed attempts at producing a backbone                  100
     Number of failed attempts at producing a sidechain                 100
     Number of failed attempts at satisfying required elements          100
```

Fig. 2. User-defined parameters for DBMAKER.

CYCLE. The entire process of generating SMILES strings, converting them to structures with CONCORD, and appending them to a database is summarized in Fig. 1. Each step is discussed in detail below.

Structure generation

We implement the descriptive strategy of the SMILES chemical line notation [13] to afford the investigator the greatest control over the SMILES generating process. This permits the designation of numerous parameters, as shown in Fig. 2, that govern all component characteristics.

In principle, any chemical structure can be broken down into a single 'backbone' with assorted, attached appendages ('side chains'). Hence, the first step is to select a backbone length from a set of user-defined probabilistic ranges. The make-up of the backbone is established by choosing from user-specified atoms or functional groups with a given frequency. Side chains are then processed. First, the total number of side chains is established. Then, their lengths are also determined from a defined probabilistic distribution. Each side chain is created as above by choosing from authorized functional groups. Furthermore, a specified number of side chains containing three or more atoms are allowed to branch randomly.

It is important to note the advantage of using SMILES string designations with CONCORD. From just a few SMILES string characters, numerous structure classes can be generated, depending upon string order, position and syntax. This is illustrated in Fig. 3. For example, suppose side chains are being generated from the characters 'N', 'C(=O)', 'C' and 'O'. From these four components, one can generate SMILES representations that include alcohols 'CO', ethers 'COC', ketones 'CC(=O)C', esters 'C(=O)OC', aldehydes 'CC(=O)', acids 'CC(=O)O', amines 'CN' and amides 'C(=O)N'.

What remains is to determine the backbone position of each side chain. In order to maintain consistency, side chains cannot be placed haphazardly. Consider a backbone composed of six atoms. No side chain of any length may be placed at position (1), as this alters the backbone length. Similarly, only a single atom side chain may be placed at position (2). Thus, for any backbone, there is an associated maximum side-chain length, a maximum number of possible side-chain atoms and a specific distribution of lengths. Furthermore, each backbone atom or functional group has an inherent valence, which is also under user control. For example, an oxygen atom present in an ether linkage cannot be used to anchor a side chain. If any of these limitations are violated, the program restarts the side-chain selection process. Otherwise, each side chain is placed randomly within its allowable loci.

If the SMILES string passes occupancy criteria (described below), it is saved to a storage file. In turn, each backbone atom is written, followed by side chains in parentheses. If a particular backbone atom is a chiral

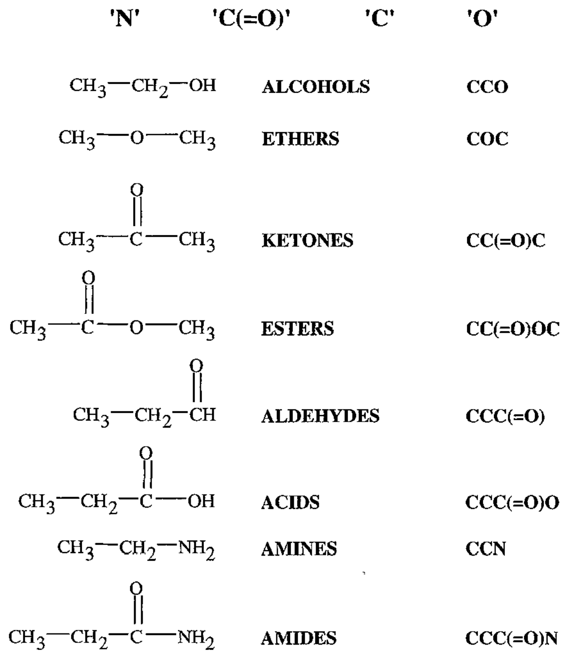


Fig. 3. Diversity of structures generated with just a few SMILES characters.

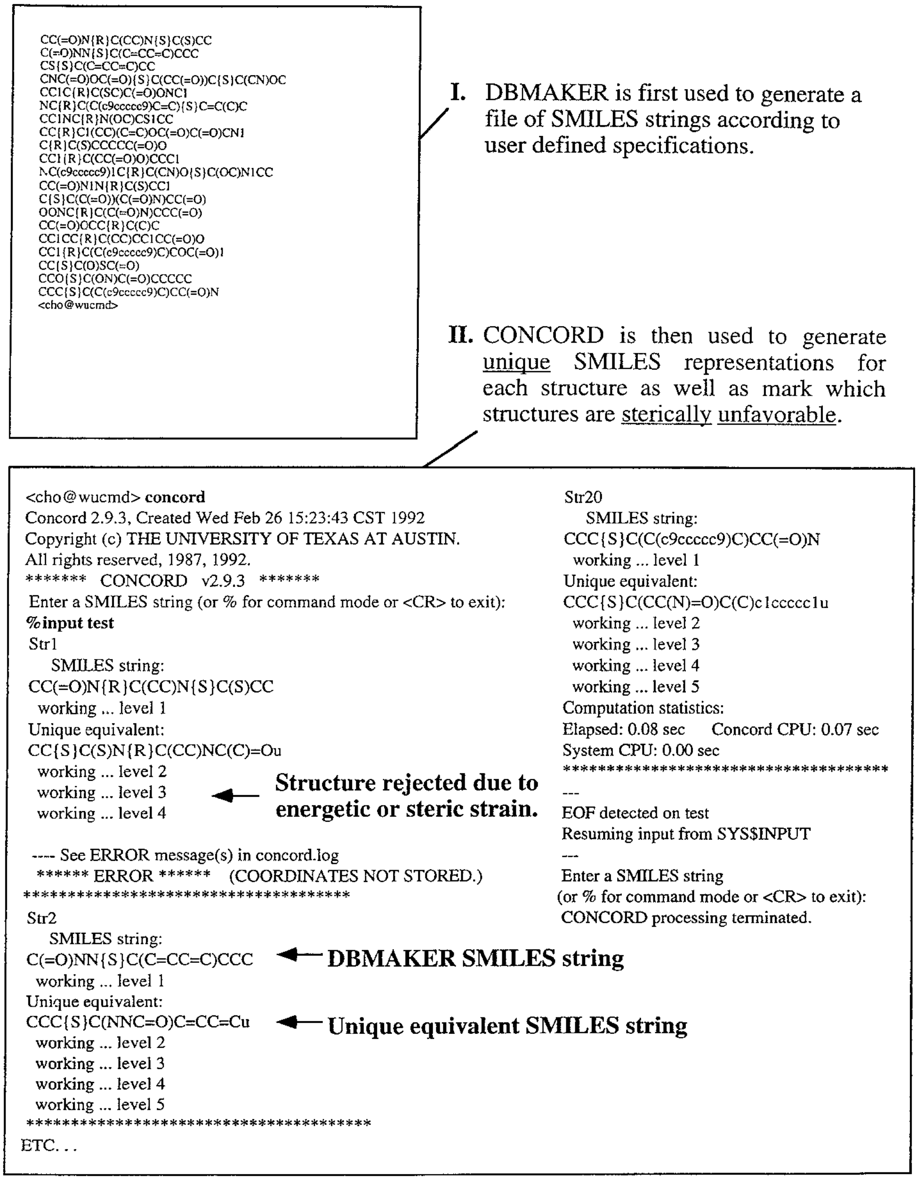
center, it is preceded by a randomly determined '{R}' or '{S}' designation. This method is repeated ad finitus and can produce a wide variety of linear compounds.

Occupancy criteria

Once the backbone and side-chain compositions have been determined, DBMAKER allows the user to screen each component for the presence or absence of specific SMILES string patterns. Since structures are generated randomly, albeit within the context of our parameters, certain groups of atoms may be undesirable due to chemical instability or synthetic difficulty. The user can eliminate these atom combinations by specifying them in the DBMAKER parameter file (Fig. 2), with the requirement that *none* be present for any acceptable compound. For example, if a database under construction is to be devoid of esters, the patterns C(=O)O and OC(=O) should be selected against. Conversely, the user may want *all* structures to contain a specific substructure or a given distribution of functional groups. These groups can also be designated, with the appropriate occupancy requirements. For example, to ensure at least three potential hydrogen-bond donors or acceptors in every structure, the requirement 'N & O > 2' is implemented.

Validation and uniqueness

The random nature of this process produces at least two problems which must be dealt with at this stage. First, due to the random combination of atoms, a given structure may be physically impossible because of steric



```
CC(=O)N{R}C(CC)N{S}C(S)CC
C(=O)NN{S}C(C=CC=C)CCC
CS{S}C(C=CC=C)CC
CNC(=O)OC(=O){S}C(CC(=O))C{S}C(CN)OC
CC1C{R}C(SC)C(=O)ONC1
NC{R}C(C(c9ccccc9)C=C){S}C=C(C)C
CC1NC{R}N(OC)CS1CC
CC{R}C1(CC)(C=C)OC(=O)C(=O)CN1
C{R}C(S)CCCCC(=O)O
CC1{R}C(CC(=O)O)CCC1
NC(c9ccccc9)1C{R}C(CN)O{S}C(OC)N1CC
CC(=O)N1N{R}C(S)CC1
C{S}C(C(=O))(C(=O)N)CC(=O)
OONC{R}C(C(=O)N)CCC(=O)
CC(=O)OCC{R}C(C)C
CC1CC{R}C(CC)CC1CC(=O)O
CC1{R}C(C(c9ccccc9)C)COC(=O)1
CC{S}C(O)SC(=O)
CCO{S}C(ON)C(=O)CCCCC
CCC{S}C(C(c9ccccc9)C)CC(=O)N
<cho@wucmd>
```

I. DBMAKER is first used to generate a file of SMILES strings according to user defined specifications.

II. CONCORD is then used to generate unique SMILES representations for each structure as well as mark which structures are sterically unfavorable.

```
<cho@wucmd> concord
Concord 2.9.3, Created Wed Feb 26 15:23:43 CST 1992
Copyright (c) THE UNIVERSITY OF TEXAS AT AUSTIN.
All rights reserved, 1987, 1992.
******* CONCORD  v2.9.3 *******
 Enter a SMILES string (or % for command mode or <CR> to exit):
%input test
 Str1
   SMILES string:
 CC(=O)N{R}C(CC)N{S}C(S)CC
  working ... level 1
 Unique equivalent:
 CC{S}C(S)N{R}C(CC)NC(C)=Ou
  working ... level 2
  working ... level 3    <--  Structure rejected due to
  working ... level 4         energetic or steric strain.

 ---- See ERROR message(s) in concord.log
  ****** ERROR ******  (COORDINATES NOT STORED.)
*************************************
 Str2
   SMILES string:
 C(=O)NN{S}C(C=CC=C)CCC    <-- DBMAKER SMILES string
  working ... level 1
 Unique equivalent:
 CCC{S}C(NNC=O)C=CC=Cu     <-- Unique equivalent SMILES string
  working ... level 2
  working ... level 3
  working ... level 4
  working ... level 5
***************************************
ETC. . .
```

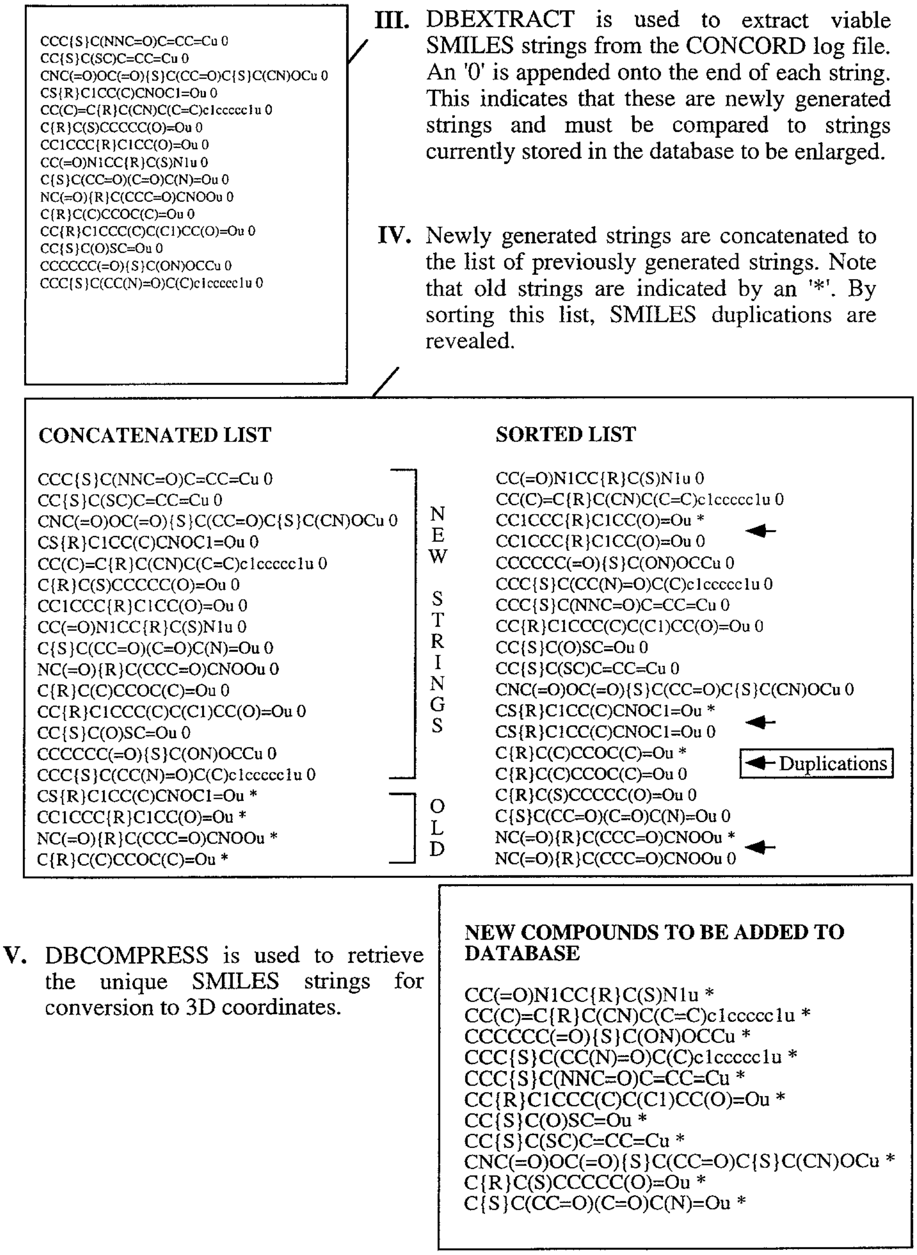
```
Str20
   SMILES string:
 CCC{S}C(C(c9ccccc9)C)CC(=O)N
   working ... level 1
 Unique equivalent:
 CCC{S}C(CC(N)=O)C(C)c1ccccc1u
   working ... level 2
   working ... level 3
   working ... level 4
   working ... level 5
 Computation statistics:
 Elapsed: 0.08 sec     Concord CPU: 0.07 sec
 System CPU: 0.00 sec
 *************************************
 ---
 EOF detected on test
 Resuming input from SYS$INPUT
 ---
 Enter a SMILES string
(or % for command mode or <CR> to exit):
 CONCORD processing terminated.
```

Fig. 4. Process of structure validation and database registration.

interference. Second, structural duplication can occur. Not only can two identical SMILES strings result, but due to degeneracy, two different strings may describe the exact same chemical structure. Thus, the list of SMILES strings generated by DBMAKER must be processed to determine the nondegenerate standardized representation for each structure according to rules established by Weininger [25]. Both these concerns are dealt with by CONCORD. Our method of structure validation and database registration is shown in Fig. 4. After processing the DBMAKER-generated output with CONCORD, a log file results, listing each input string, its nondegenerate unique representation, and whether or not a structure could be generated. The module DBEXTRACT then processes the log file and extracts the unique SMILES string of each structure that could be generated without chemical error.

Elimination of redundancy

We now have a list of unique strings; thus, if two structures are identical, they will have the exact same SMILES notation. Not only must we eliminate duplicates in the current list of strings, but we must also exclude structures previously stored in the database. In short, we need to scan our input list, rapidly compare it to structures currently in the database, and isolate the novel



```
CCC{S}C(NNC=O)C=CC=Cu 0
CC{S}C(SC)C=CC=Cu 0
CNC(=O)OC(=O){S}C(CC=O)C{S}C(CN)OCu 0
CS{R}C1CC(C)CNOC1=Ou 0
CC(C)=C{R}C(CN)C(C=C)c1ccccc1u 0
C{R}C(S)CCCCC(O)=Ou 0
CC1CCC{R}C1CC(O)=Ou 0
CC(=O)N1CC{R}C(S)N1u 0
C{S}C(CC=O)(C=O)C(N)=Ou 0
NC(=O){R}C(CCC=O)CNOOu 0
C{R}C(C)CCOC(C)=Ou 0
CC{R}C1CCC(C)C(C1)CC(O)=Ou 0
CC{S}C(O)SC=Ou 0
CCCCCC(=O){S}C(ON)OCCu 0
CCC{S}C(CC(N)=O)C(C)c1ccccc1u 0
```

III. DBEXTRACT is used to extract viable SMILES strings from the CONCORD log file. An '0' is appended onto the end of each string. This indicates that these are newly generated strings and must be compared to strings currently stored in the database to be enlarged.

IV. Newly generated strings are concatenated to the list of previously generated strings. Note that old strings are indicated by an '*'. By sorting this list, SMILES duplications are revealed.

CONCATENATED LIST

```
CCC{S}C(NNC=O)C=CC=Cu 0
CC{S}C(SC)C=CC=Cu 0
CNC(=O)OC(=O){S}C(CC=O)C{S}C(CN)OCu 0
CS{R}C1CC(C)CNOC1=Ou 0
CC(C)=C{R}C(CN)C(C=C)c1ccccc1u 0
C{R}C(S)CCCCC(O)=Ou 0
CC1CCC{R}C1CC(O)=Ou 0
CC(=O)N1CC{R}C(S)N1u 0
C{S}C(CC=O)(C=O)C(N)=Ou 0
NC(=O){R}C(CCC=O)CNOOu 0
C{R}C(C)CCOC(C)=Ou 0
CC{R}C1CCC(C)C(C1)CC(O)=Ou 0
CC{S}C(O)SC=Ou 0
CCCCCC(=O){S}C(ON)OCCu 0
CCC{S}C(CC(N)=O)C(C)c1ccccc1u 0
CS{R}C1CC(C)CNOC1=Ou *
CC1CCC{R}C1CC(O)=Ou *
NC(=O){R}C(CCC=O)CNOOu *
C{R}C(C)CCOC(C)=Ou *
```

SORTED LIST

```
CC(=O)N1CC{R}C(S)N1u 0
CC(C)=C{R}C(CN)C(C=C)c1ccccc1u 0
CC1CCC{R}C1CC(O)=Ou *
CC1CCC{R}C1CC(O)=Ou 0
CCCCCC(=O){S}C(ON)OCCu 0
CCC{S}C(CC(N)=O)C(C)c1ccccc1u 0
CCC{S}C(NNC=O)C=CC=Cu 0
CC{R}C1CCC(C)C(C1)CC(O)=Ou 0
CC{S}C(O)SC=Ou 0
CC{S}C(SC)C=CC=Cu 0
CNC(=O)OC(=O){S}C(CC=O)C{S}C(CN)OCu 0
CS{R}C1CC(C)CNOC1=Ou *
CS{R}C1CC(C)CNOC1=Ou 0
C{R}C(C)CCOC(C)=Ou *
C{R}C(C)CCOC(C)=Ou 0
C{R}C(S)CCCCC(O)=Ou 0
C{S}C(CC=O)(C=O)C(N)=Ou 0
NC(=O){R}C(CCC=O)CNOOu *
NC(=O){R}C(CCC=O)CNOOu 0
```

V. DBCOMPRESS is used to retrieve the unique SMILES strings for conversion to 3D coordinates.

NEW COMPOUNDS TO BE ADDED TO DATABASE

```
CC(=O)N1CC{R}C(S)N1u *
CC(C)=C{R}C(CN)C(C=C)c1ccccc1u *
CCCCCC(=O){S}C(ON)OCCu *
CCC{S}C(CC(N)=O)C(C)c1ccccc1u *
CCC{S}C(NNC=O)C=CC=Cu *
CC{R}C1CCC(C)C(C1)CC(O)=Ou *
CC{S}C(O)SC=Ou *
CC{S}C(SC)C=CC=Cu *
CNC(=O)OC(=O){S}C(CC=O)C{S}C(CN)OCu *
C{R}C(S)CCCCC(O)=Ou *
C{S}C(CC=O)(C=O)C(N)=Ou *
```

Fig. 4. (continued).

compounds. Figure 4 shows how this is accomplished. The SMILES strings of structures currently stored in the database are marked with an appended asterisk (*) to denote their status. Strings in our new list generated from DBEXTRACT contain an appendage (0). The two lists are concatenated into a single file, sorted using the UNIX 'sort' command, and then written to a new file. In this sorted file, duplicate SMILES strings will be located adjacent to one another. The module DBCOMPRESS then processes the sorted file and retrieves all unique SMILES strings that contain an appendage (0). These strings represent all the novel structures that will be added to the database. Notice that the new compounds listed in this file have an asterisk appended to each structure since these will become a part of the augmented database. These strings are added to the compound registry by simply concatenating them to the old database SMILES registry.

CONCORD is then utilized to generate the actual 3D structures. Coordinates can be produced in a number of proprietary formats; however, we normally direct CONCORD to generate compounds in SYBYL mol format [19]. Following execution, a single file is generated that contains the structural information for all compounds stored serially. A SYBYL macro is then employed to systematically retrieve the molecular data for each compound, minimize if necessary, and then store to a database format via a converter program.

Databases of structures that contain specific common elements

At times, databases of structures that contain specific common elements are desired. For example, if a laboratory is adept at synthesizing certain classes of compounds, one would want derivatives of these compounds incorporated into databases for searching. As another example, X-ray crystallographic analysis of a binding site may reveal that potential ligands must contain a double bond at position X or a branched main chain at position Y. Accordingly, structures in the databases should reflect these requirements.

This is accomplished by employing SMILES *templates*. The templates enable the user to define constant regions to maintain desired elements, yet allows DBMAKER to generate diversity in the compounds produced. They are also defined in the parameter file as shown in Fig. 2. For example, if a database of diketo compounds is needed, the SMILES template '***C(=O)C(=O)***' is employed. The '*' symbols of each template indicate where DBMAKER is allowed to place backbone atoms and appended

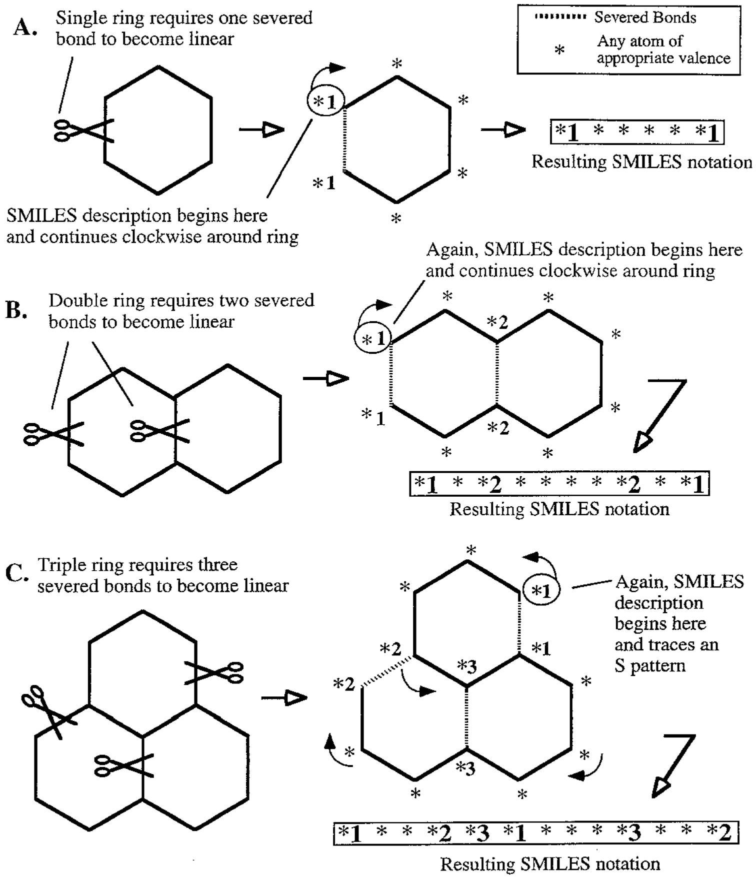

A. Single ring requires one severed bond to become linear
Severed Bonds
* Any atom of appropriate valence
*1
*
*
*
*1
*
*1 * * * * *1
Resulting SMILES notation
SMILES description begins here and continues clockwise around ring
Again, SMILES description begins here and continues clockwise around ring
B. Double ring requires two severed bonds to become linear
*1
*2
*1
*2
*1 * *2 * * * * *2 * *1
Resulting SMILES notation
C. Triple ring requires three severed bonds to become linear
*1
Again, SMILES description begins here and traces an S pattern
*2
*1
*3
*2
*3
*1 * * *2 *3 *1 * * * *3 * * *2
Resulting SMILES notation

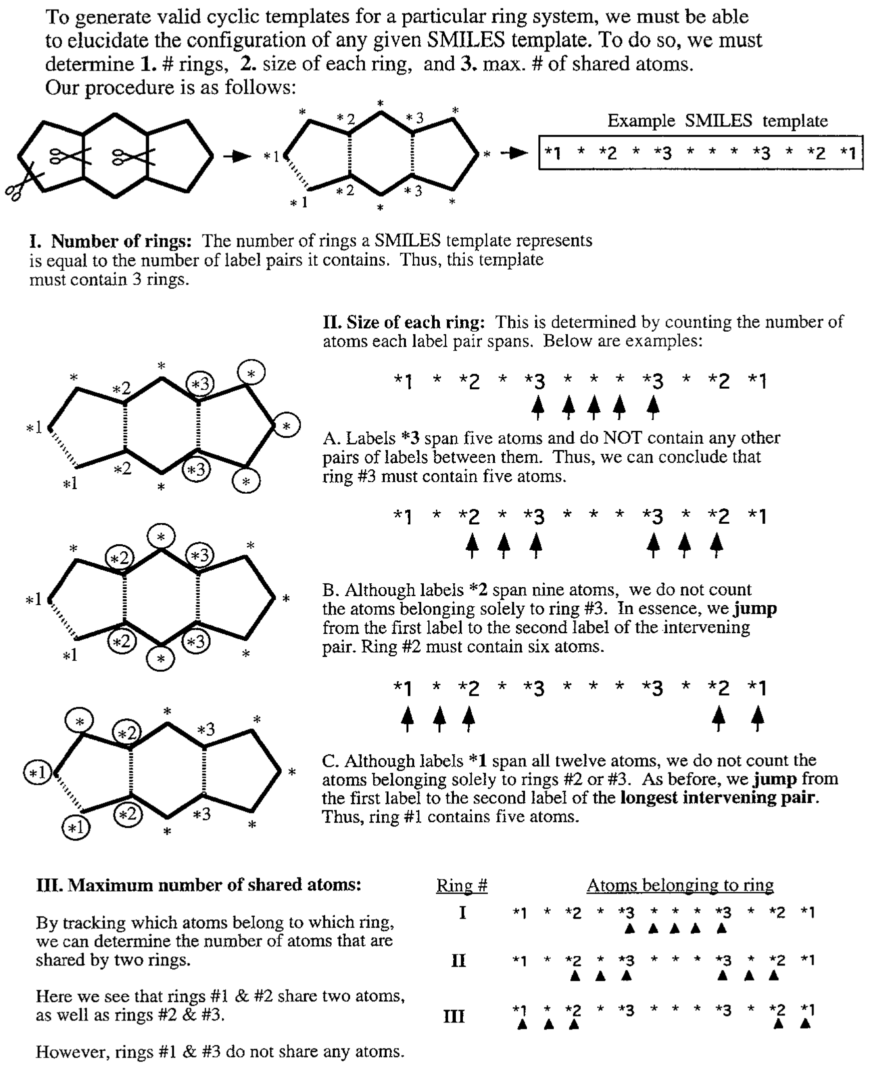
Fig. 5. Representation of cyclic structures using SMILES.

side chains. All other template symbols are then transferred to the generated SMILES string in the positions designated by the template. A database of dicarboxylic acids would be generated with the template 'OC(=O)*** etc. **C(=O)O'. Similarly, a database of compounds centered about a double bond would be produced with the template '****=****'. As shown, these templates provide the user with a very flexible means of generating structures containing invariant functional groups or chemical structures. However, their greatest utility is in the generation of cyclic compounds.

Cyclic structures

In order to represent a cyclic structure in SMILES notation, the structure must first be linearized by breaking strategically chosen bonds. It is then written as if it were linear, except that the atoms whose bonds were broken are designated numerically. This is illustrated in Fig. 5. In example A, we see the structure of a generic six-membered ring. To linearize this ring, we must sever one bond. The SMILES representation then begins at one of the atoms of the severed bond and continues clockwise around the ring. Therefore, the SMILES notation for this ring is *1*****1, where the '*' symbols denote any atom of satisfactory valence. In essence, this representation is a template which designates the backbone atoms that must be followed by severed bond numerical labels to generate a valid cyclic structure.

Example B shows a generic structure for two six-membered rings fused along one edge. In this case two bonds must be severed to linearize the structure. Starting at the label for the first severed bond, *1, we again proceed around the ring in a clockwise manner to generate the template, '*1**2*****2**1'. As a chemical example, substituting c's for the asterisks, we generate the SMILES notation for a naphthalene group: c1cc2ccccc2cc1. Example C furthers this concept by addressing a fused, three-ring system.


To generate valid cyclic templates for a particular ring system, we must be able to elucidate the configuration of any given SMILES template. To do so, we must determine **1.** # rings, **2.** size of each ring, and **3.** max. # of shared atoms. Our procedure is as follows:

Example SMILES template

*1 * *2 * *3 * * * *3 * *2 *1

I. Number of rings: The number of rings a SMILES template represents is equal to the number of label pairs it contains. Thus, this template must contain 3 rings.

II. Size of each ring: This is determined by counting the number of atoms each label pair spans. Below are examples:

*1 * *2 * *3 * * * *3 * *2 *1

A. Labels ***3** span five atoms and do NOT contain any other pairs of labels between them. Thus, we can conclude that ring #3 must contain five atoms.

*1 * *2 * *3 * * * *3 * *2 *1

B. Although labels ***2** span nine atoms, we do not count the atoms belonging solely to ring #3. In essence, we **jump** from the first label to the second label of the intervening pair. Ring #2 must contain six atoms.

*1 * *2 * *3 * * * *3 * *2 *1

C. Although labels ***1** span all twelve atoms, we do not count the atoms belonging solely to rings #2 or #3. As before, we **jump** from the first label to the second label of the **longest intervening pair**. Thus, ring #1 contains five atoms.

III. Maximum number of shared atoms:

By tracking which atoms belong to which ring, we can determine the number of atoms that are shared by two rings.

Here we see that rings #1 & #2 share two atoms, as well as rings #2 & #3.

However, rings #1 & #3 do not share any atoms.

Ring #	Atoms belonging to ring
I	*1 * *2 * *3 * * * *3 * *2 *1
II	*1 * *2 * *3 * * * *3 * *2 *1
III	*1 * *2 * *3 * * * *3 * *2 *1

Fig. 6. Elucidation of ring configuration for cyclic SMILES templates.

DBMAKER generates cyclic structures by employing user-defined *cyclic templates*, which designate the backbone atoms that should be followed by appropriate numbers in the SMILES string. However, we must make the distinction between aromatic and nonaromatic compounds. For example, consider the template '*1*****1'. If upper-case 'C's are substituted for each '*', we produce the SMILES string for cyclohexane: C1CCCCC1. However, should we wish to generate the aromatic compound benzene, lower-case 'c's must be used instead: c1ccccc1. Therefore, we make this distinction in DBMAKER by using minus signs '–' for aromatic compounds and plus signs '+' for nonaromatic compounds. As such, the template for cyclohexane derivatives is '+1+++++1' while the template for benzene derivatives is '–1–––––1'.

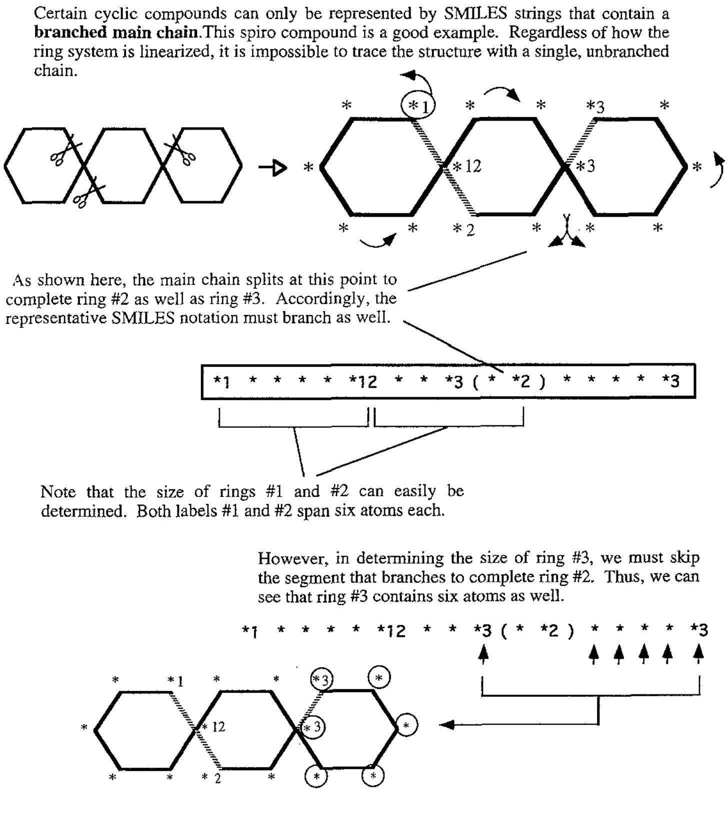
Each template listed in the parameter file depicts a separate cyclic entity, and the user specifies the frequency with which each cyclic template should be utilized. Again, proper atom or functional group valences must be maintained. For the discussion below, we will adopt the convention of using an asterisk '*' as a generic representation of any atom.

Generating cyclic templates: DBCYCLE

Although DBMAKER allows cyclic compounds to be generated, the user must supply the appropriate cyclic templates. Unfortunately, a potentially large number of complex structures can exist that all contain a given number of rings and specific ring sizes. Therefore, we developed another program, called DBCYCLE, which produces these templates given a desired number of rings and sizes. The first part of the program combinatorially generates potential cyclic templates. The second part empirically determines whether each potential template can produce the desired ring configuration. To develop this procedure, we studied the SMILES representations of numerous cyclic compounds from the Chemical Abstracts Ring Systems Handbook (ring systems file I: RF1–RF32530 [14]). Several observations could be generalized to the majority of the ring systems studied. These observations were then incorporated into a method to determine the ring configuration of each valid SMILES template. This is illustrated in Fig. 6.

First, in order to linearize a cyclic structure that contains N rings, N bonds must be broken. Accordingly, all SMILES templates representing a structure with N rings must contain N pairs of numerical labels. This is shown in Figs. 5 and 6.

Second, given that every pair of numerical labels in a SMILES string represents the severed bonds of a particular ring, the number of atoms 'included' by each pair of labels must equal the number of atoms in the corresponding ring. An example can best illustrate this point. Figure 6 shows a tricyclic compound containing five, six and five atoms, respectively. Labels #3 span exactly five atoms; thus, the atoms designated by this pair of labels



Certain cyclic compounds can only be represented by SMILES strings that contain a **branched main chain**.This spiro compound is a good example. Regardless of how the ring system is linearized, it is impossible to trace the structure with a single, unbranched chain.

As shown here, the main chain splits at this point to complete ring #2 as well as ring #3. Accordingly, the representative SMILES notation must branch as well.

*1 * * * * *12 * * *3 (* *2) * * * * *3

Note that the size of rings #1 and #2 can easily be determined. Both labels #1 and #2 span six atoms each.

However, in determining the size of ring #3, we must skip the segment that branches to complete ring #2. Thus, we can see that ring #3 contains six atoms as well.

*1 * * * * *12 * * *3 (* *2) * * * * *3

Fig. 7. SMILES representation of cyclic structures that contain branched main chains.

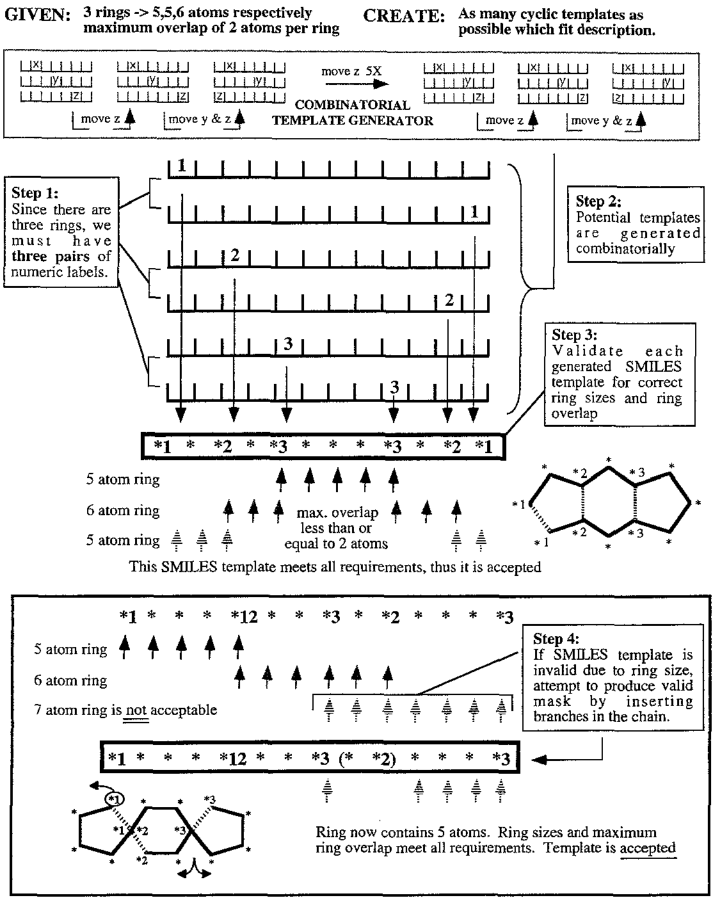
are contained within ring III. Although labels #2 span nine atoms, we see that the atoms between the #3 labels should not be counted since they belong exclusively to ring III. Therefore, we jump from the first #3 label to the second. As such, there are six atoms contained in ring II. Similarly, by jumping from the first #2 label to the second, there are five atoms contained within ring I.

Third, by keeping track of the atoms contained within each ring, we can easily determine the number of overlapping atoms between any two rings. In the example shown in Fig. 6, we see that rings I and II overlap by two atoms. In addition, rings II and III overlap by two atoms as well. However, we see that there are no shared atoms between rings I and III.

Certain cyclic compounds can only be represented by SMILES strings that contain a branched main chain. The significance of this is illustrated in Fig. 7, using spiro compounds as an example. In this figure, it is impossible to trace the structure with a single, unbranched chain, regardless of how the system is linearized. Thus, its representative SMILES template will also contain a branch. Note that the size of rings I and II can easily be validated since they both span six atoms. However, in determining the size of ring III, we must skip the segment that branches to complete ring II. In doing so, we find that ring III also contains six atoms.

Figure 8 demonstrates how the DBCYCLE program generates the cyclic templates for a given ring configuration. In this example, suppose that we wish to generate templates for tricyclic structures containing five, five and six atom rings, respectively. Furthermore, we require that all structures contain a maximum of two overlapping atoms between any two rings.

First, since we are dealing with compounds containing three rings, we know that all templates must employ three pairs of numerical labels (six in total). Using these six numerical labels, we combinatorially generate potential templates, as shown in the figure. We then employ the validation method described above to determine whether the template fits our requirements. As shown, the current template contains two five-atom rings and one six-atom ring. Furthermore, we see that the maximum overlap between any two rings is two atoms. Therefore, we conclude that the current template is valid and we store it to a file. The actual cyclic structure corresponding to this template is depicted as well. The combinatorial process then continues, and validation is performed on the next template.


GIVEN: 3 rings -> 5,5,6 atoms respectively
maximum overlap of 2 atoms per ring
CREATE: As many cyclic templates as possible which fit description.
x
y
z
move z 5X
move z
move y & z
COMBINATORIAL TEMPLATE GENERATOR
1
2
3
*1 * *2 * *3 * * * *3 * *2 *1
5 atom ring
6 atom ring
5 atom ring
max. overlap less than or equal to 2 atoms
*1 * * * *12 * * *3 * *2 * * * *3
7 atom ring is not acceptable
*1 * * * *12 * * *3 (* *2) * * * *3
*1
*2
*3

Step 1: Since there are three rings, we must have **three pairs** of numeric labels.

Step 2: Potential templates are generated combinatorially

Step 3: Validate each generated SMILES template for correct ring sizes and ring overlap

This SMILES template meets all requirements, thus it is accepted

Step 4: If SMILES template is invalid due to ring size, attempt to produce valid mask by inserting branches in the chain.

Ring now contains 5 atoms. Ring sizes and maximum ring overlap meet all requirements. Template is accepted

Fig. 8. DBCYCLE method to systematically generate cyclic templates.

As shown in the box at the bottom of Fig. 8, the majority of the templates will be invalid due to a ring size infraction. In this example, a template has been generated that contains seven atoms in ring III. According to the rules described in Fig. 6, this template would be rejected. However, we can attempt to rectify the template by inserting branches at various loci along the main chain. We see that by inserting a branch at the designated position, ring III now correctly contains five atoms. Note that with the selection of this branch point, both rings I and II remain unchanged. The branched cyclic template is now acceptable since both ring sizes and overlaps are correct. The actual structure corresponding to this template is also illustrated.

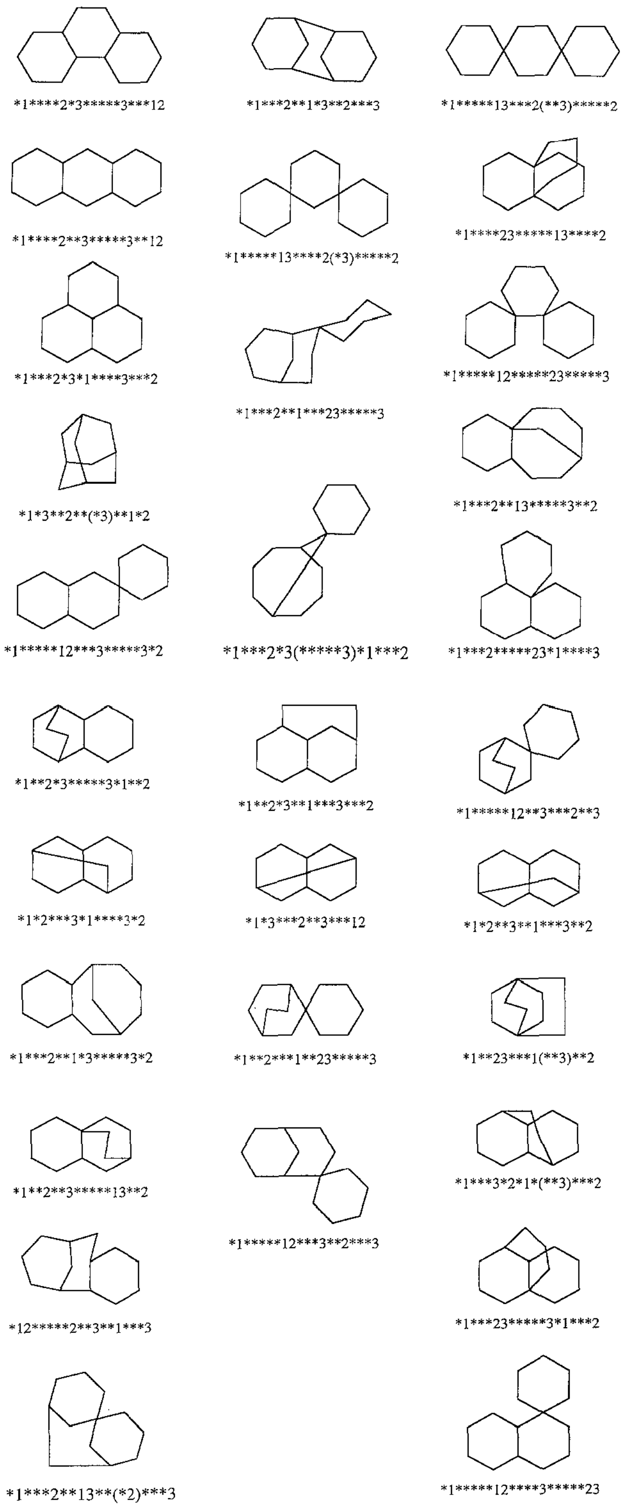


Fig. 9. Cyclic templates generated by DBCYCLE for tricyclic 6,6,6-ring systems with a maximum of four overlapping atoms between any two rings.

Generating structural diversity

Although DBMAKER allows the user to produce a variety of structures, it is less able to generate large, more complex ligands, especially those involving combinations of rings and aliphatic chains. These structures require long and complicated SMILES expressions. We have found that generating such lengthy strings outright with DBMAKER results in a much higher percentage of rejections, due to steric violations. This may result from the random nature of atom placement. Invariably, the efficiency of this endeavor decreases with string complexity. A solution to this problem, however, as well as an effective means of generating ligand diversity is to form *combinations* of strings that are known to be structurally sound. In a chemical sense, this is akin to taking separate compounds, removing a valence atom from each structure at an appropriate site, and then joining them by means of a new bond. To achieve this, we utilize the module DBCROSS. This program takes two lists of SMILES strings and randomly 'crosses' them in a genetic sense. For example, a database of cyclic structures can be bred with a database of aliphatic or linear structures. Specific ratios of parental 'gametes' can also be defined. Thus, we could specify that three random cyclic structures be attached to every chosen aliphatic fragment, or vice versa. Compounds containing complex mixtures of cyclic and linear components could be formed after repeated generations of inbreeding.

Application and discussion

To demonstrate the capabilities of DBCYCLE, we generated cyclic templates for tricyclic 6,6,6-atom and 5,5,6-atom systems. We then compared these templates to the cyclic structures found in the Chemical Abstracts Ring Systems Handbook (ring systems file I: RF1–RF32530 [14]) to see how many cyclic systems we had matched.

Figures 9 and 10 depict the cyclic templates produced by DBCYCLE for the tricyclic 6,6,6-atom and 5,5,6-atom ring systems, respectively. Upon comparison to the ring systems found in the Chemical Abstracts Ring Systems Handbook (file I), we found that DBCYCLE reproduced all of the 6,6,6 systems (30) and all of the 5,5,6 systems (40). Rare, complex ring constructions that are beyond the scope of our simple validation scheme described

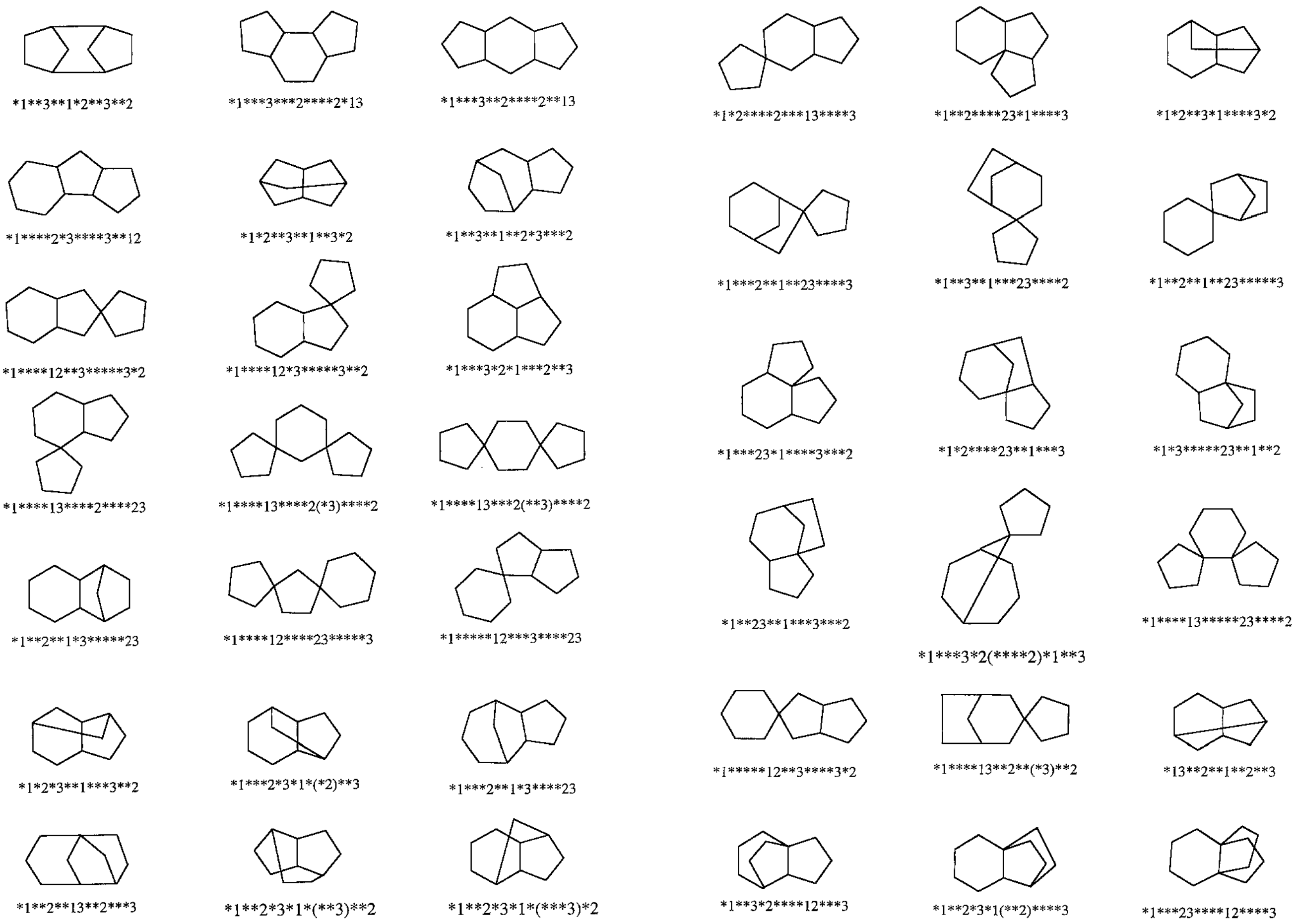


Fig. 10. Cyclic templates generated by DBCYCLE for tricyclic 5,5,6-ring systems with a maximum of four overlapping atoms between any two rings.

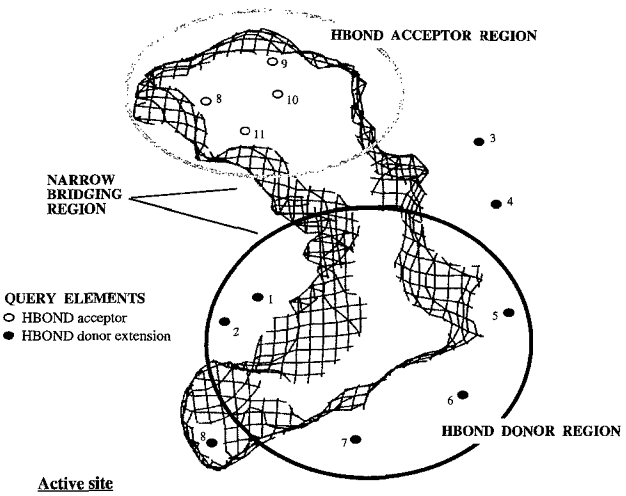
above may be missed. However, by comparing the output of DBCYCLE to the Ring Systems Handbook, we can ensure that the most common cyclic systems for which synthetic means are known are included.

Because of the combinatorial nature of the DBCYCLE template-generating procedure, the program takes exponentially longer to produce templates for systems containing larger numbers of rings. As shown in Fig. 8, our method generates SMILES templates by brute force. Thus, the worst-case running time of our algorithm can be approximated by $(nq)^{2n}$ for a system containing n rings of q average size. Furthermore, redundant SMILES strings are generated. As a result, the output must be processed by CONCORD to determine unique structures. Since the number of generated strings also increases exponentially with larger ring systems, even more time is required. On an R4000 SG Indigo, SMILES strings for three-ring systems can be generated in a few seconds. Four-ring systems require about a minute; five-ring systems about 6 h. Without further optimization of the code (i.e., pruning functions to improve efficiency), the combinatorial generation and extraction of unique six-ring templates requires about a week of CPU time. A factor of two speed-up can be realized by eliminating the 'branched chain' processing of cyclic templates as described in step 4 of Fig. 8. However, all cyclic systems that cannot be traced with an unbroken backbone will be missed.

To demonstrate the utility of DBMAKER and DBCROSS, we will describe four scenarios in which a database of potential ligands can be generated to solve a particular ligand design problem.

The first scenario is illustrated in Figs. 11A–D. In this example, we will attempt to generate ligands capable of binding within the active site of DHFR [26]. A cast of the active site [27] is shown in Fig. 11A, superimposed upon the pharmacophore. The pharmacophore was elucidated by studying the active site, noting where potential ligand hydrogen-bond donor and acceptor groups would best interact with the receptor. Several observations can be made.

The DHFR active site is shaped much like a boomerang, with a narrow bridging region spanning two larger domains. Though larger, these two areas are relatively flat. We conclude that a *cyclic, planar, six-membered* compound would best fill these regions and provide a stable platform to anchor complementary functional groups. In



Active site

Extreme polarity in pharmacophore - each region dominated by either hydrogen bond donor or acceptor loci.

Each region sterically constrained - planar ring best used as a foundation upon which functional groups can be anchored.

Narrow bridging region - a small chain would best be used to bridge the two groups of pharmacophoric elements

Plan

2 sub-databases will be generated:
- HBOND donor cyclic database
- HBOND acceptor cyclic database

→ These databases will then be crossed with a small linker fragment to generate the final search database.

Fig. 11. (A) DBMAKER scenario #1. Attempt to generate ligands capable of binding with DHFR active site.

order to span the bridging region, it would be best to use a small chain to connect the two domains.

An extreme polarity exists in the pharmacophore. The pharmacophoric elements present in the top domain are exclusively hydrogen-bond acceptors. In contrast, throughout the bridging region and the lower domain, hydrogen-bond donor elements are abundant.

Based on these observations, our plan is to generate two sub-databases: a hydrogen-bond donor cyclic database and a hydrogen-bond acceptor cyclic database. Each database will target their respective portion of the active site. By then crosssing a randomly chosen structure from each database with a small linker chain, we can generate a database of compounds tailored to this active site. These compounds will then be searched against a 3D query containing the pharmacophore using the 3D database search program FOUNDATION-FX (Ho, C.M.W. and Marshall, G.R., manuscript in preparation). This program performs similar functions as previously published [28], however, incorporating full conformational flexibility as well [29].

The parameters used to develop the *HB acceptor* database are shown in Fig. 11B. To force an aromatic six-membered ring, the cyclic template '–1––––1' is used. The four components 'C', 'N', 'O' and 'C(=O)' are employed as described above, and their backbone and side-chain utilization frequencies are listed. The #atoms designation tells DBMAKER that each of the four SMILES symbols should be treated as a single atom entity when filling backbone and side-chain sites. The #connections designation in the parameter file allows the user to govern how many atoms/functional groups each component is able to bond with. For example, aromatic carbons are able to anchor one side chain. Since each carbon must bond to two neighbors in the ring, only a maximum of three associations is possible. Thus, its '#connections designation' is set at three. Backbone aromatic nitrogens cannot anchor any side chains. Thus, their maximum number of associations is set at two.

Two to four side chains are specified per structure, each containing one to three atoms or functional groups. Side-chain components are listed in the parameter file together with their frequency of utilization. As described above, a large number of different side chains can be generated from just a few SMILES characters, depending upon the number, order and syntax of their use.

The occupancy requirements are the key to successful database generation. The backbone requirements are used to limit the number and location of heteroatoms used in the aromatic ring. The first limitation, 'nn o = 0', prevents formation of consecutive aromatic nitrogens. It also prevents use of oxygen in the ring. The backbone utilization frequency (0.0) should prevent this; however, we specify this requirement just in case. The second limitation, 'n < 3', limits the number of aromatic nitrogens to two or less. The side-chain requirement 'NO ON NN OO = 0' prevents the occurrence of consecutive heteroatoms. The 'O > 2' limitation guarantees that at least three oxygen atoms will be present in the side chains. This is to ensure that at least three hydrogen-bond acceptor elements will be present in every structure.

The complete SMILES requirements apply to the fully constructed SMILES representation of each structure. It is enforced after backbone and side-chain elements have been assembled. The 'n(n1(= 0' requirement prevents the use of an aromatic nitrogen as an anchor for a side chain. As discussed above, the #connections specification of (2) should prevent this; however, it is included just in case. The 'n1 < 2' requirement also prevents the occurrence of consecutive aromatic nitrogens in the ring. Under the right conditions, the backbone 'n1ccccn1' may result. This satisfies the 'nn = 0' backbone specification; however, since both nitrogens carry the '1' labels, they are bonded to one another. To prevent this from occurring, we use the requirement 'n1 < 2'.

The requirement 'N) C) = 0' allows us to further tailor this database. Since we are attempting to generate hydrogen-bond acceptors, this requirement prevents the side chains of each structure from ending in nitrogen, -NH_2 or carbon. Thus, side chains are designed to terminate as alcohols, aldehydes or carboxylic acids. This increases the exposure of hydrogen-bond acceptor elements. Finally, the 'O > 2' requirement again ensures the presence of at least three hydrogen-bond acceptor elements in the final structure.

An initial group of 1000 SMILES strings was randomly produced within the scope of the defined parameters. Subsequently, processing and extraction of unique SMILES strings were carried out, leaving 215 different structures. These strings were then converted to SYBYL multi_MOL format with CONCORD. The entire process required approximately 75 s of CPU time (R4000 Indigo). Structures generated with these parameters are displayed. As shown in Fig. 11B, a variety of compounds are possible. Each contains at least two hydrogen-bond acceptor groups.

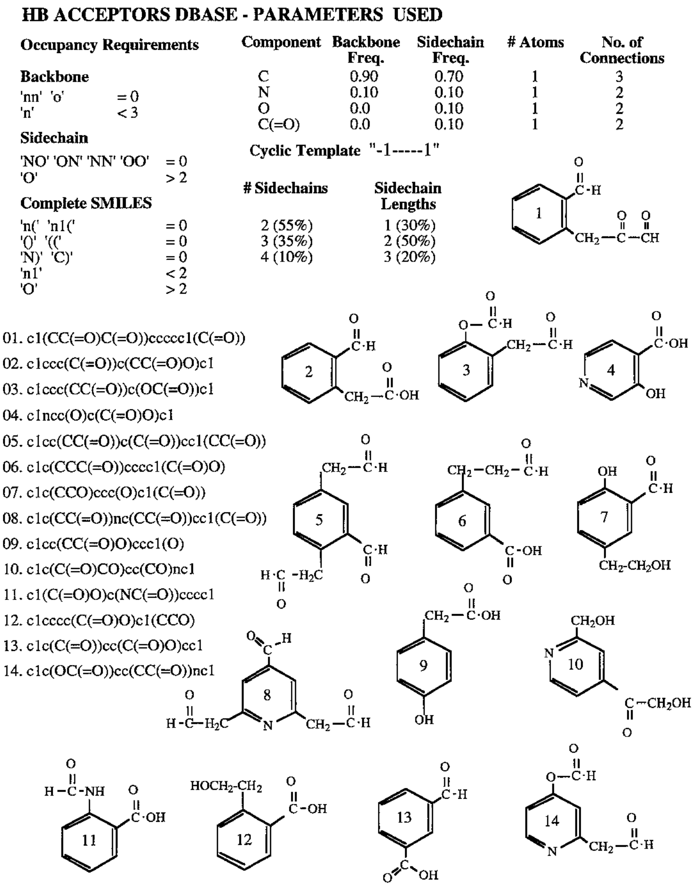
The *Hb donor* database uses the same four SMILES components as discussed above. As shown in Fig. 11C, the utilization frequencies for both backbone and side-chain selection are also the same. However, the parameters are slightly skewed towards a large number and greater length of side chains. These frequencies were chosen because the hydrogen-bond donor region of the active site is larger.

The occupancy requirements for both backbone and side chain are also nearly identical. Note that nitrogen has been added to the 'N O > 2' side-chain requirement. Since both $-NH_2$ and -OH are hydrogen-bond donors, this guarantees that each structure should contain at least two hydrogen-bond donors.

Between these two databases, the parameters differ in the complete SMILES requirements. In this database we wanted to make sure that the side chains terminate in hydrogen-bond donors. Thus, the requirement 'OC(=O) =O)) = 0' prevents side chains from terminating with a carbonyl group. The requirement 'C(=O)C(=O) = 0' eliminates diketo structures. The remaining two requirements prevent the occurrence of ethers and esters.

Again, an initial group of 1000 SMILES strings was randomly produced within the scope of the defined parameters. Processing and extraction of unique SMILES strings left 523 different structures. These strings were then converted to SYBYL multi_MOL format with CONCORD. The entire process required approximately 2.5 min of CPU time (R4000 Indigo). Structures generated with these parameters are displayed. As shown in Fig. 11C, a variety of compounds are possible. Each contains at least two hydrogen-bond donor groups.

DBCROSS was then used to form the *combination database*, as depicted in Fig. 11D. For each structure in this database, a compound from both the acceptor and



HB ACCEPTORS DBASE - PARAMETERS USED

Occupancy Requirements

Backbone

'nn' 'o'	= 0
'n'	< 3

Sidechain

'NO' 'ON' 'NN' 'OO'	= 0
'O'	> 2

Complete SMILES

'n(' 'n1('	= 0
'()' '(('	= 0
'N)' 'C)'	= 0
'n1'	< 2
'O'	> 2

Component	Backbone Freq.	Sidechain Freq.	# Atoms	No. of Connections
C	0.90	0.70	1	3
N	0.10	0.10	1	2
O	0.0	0.10	1	2
C(=O)	0.0	0.10	1	2

Cyclic Template "-1-----1"

# Sidechains	Sidechain Lengths
2 (55%)	1 (30%)
3 (35%)	2 (50%)
4 (10%)	3 (20%)

01. c1(CC(=O)C(=O))ccccc1(C(=O))
02. c1ccc(C(=O))c(CC(=O)O)c1
03. c1ccc(CC(=O))c(OC(=O))c1
04. c1ncc(O)c(C(=O)O)c1
05. c1cc(CC(=O))c(C(=O))cc1(CC(=O))
06. c1c(CCC(=O))cccc1(C(=O)O)
07. c1c(CCO)ccc(O)c1(C(=O))
08. c1c(CC(=O))nc(CC(=O))cc1(C(=O))
09. c1cc(CC(=O)O)ccc1(O)
10. c1c(C(=O)CO)cc(CO)nc1
11. c1(C(=O)O)c(NC(=O))cccc1
12. c1cccc(C(=O)O)c1(CCO)
13. c1c(C(=O))cc(C(=O)O)cc1
14. c1c(OC(=O))cc(CC(=O))nc1

Fig. 11. (B) DBMAKER scenario #1. Generation of hydrogen-bond acceptor database.

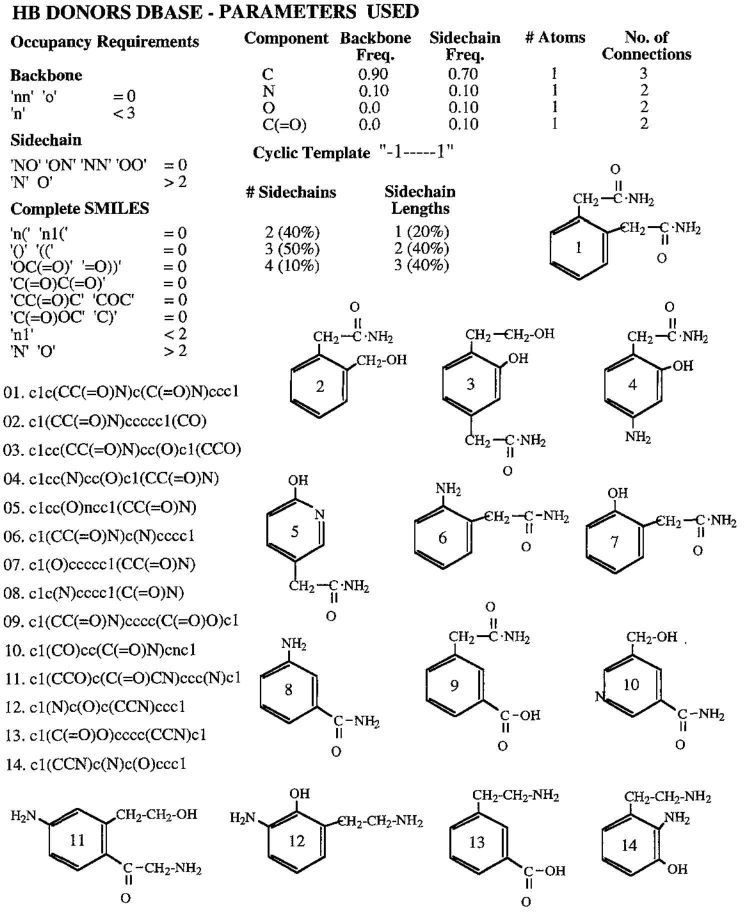
donor databases was selected at random. These two components were then spliced onto the ends of a two-carbon linker fragment. DBCROSS was used to initially generate 5000 SMILES strings in this manner. Processing and extraction of unique strings left 4822 different structures. These strings were then converted to SYBYL multi_MOL format with CONCORD. The entire process required approximately 25 min of CPU time (R4000 Indigo).

To determine how well these structures complemented the active site for which they were designed, a 3D database search was conducted against the DHFR pharmacophore using the program FOUNDATION-FX (Ho, C.M.W. and Marshall, G.R., manuscript in preparation). This program allows the retrieval of partial query solutions and employs full conformational flexibility [29]. A number of constraints were used to limit the number of hits retrieved. First, a maximum of three hits per structure (different conformations) was specified. Second, the coordinates of each retrieved hit were required to differ from any previous hit by greater than 0.500 Å rms deviation. Third, a maximum error of 0.500 Å rms deviation from the query was tolerated. Fourth, a maximum number of 4000 search conformations was allowed per database structure. Finally, any hit was required to fit within the active site using a reduced atom radius factor of 0.700.

Given the query of 11 hydrogen-bond donor and acceptor sites, 35 five-query element hits were retrieved (15 different structures). The best six are shown in Fig. 11D together with the pharmacophoric elements they complement. The group of retrieved compounds was diverse, each engineered to complement specific regions of the DHFR active site. In addition, 50 four-query element hits were retrieved in the first 350 structures scanned.

Figure 12 details the second scenario. In this example, we will attempt to generate ligands capable of binding with the P1/P2 region of the HIV-1 protease active site [30]. As detailed in this figure, the P1/P2 region is a crescent-shaped structure incorporating both P1 and P2 sidechain binding pockets. A ligand-binding pharmacophore deduced from the crystal structure of the protease is shown superimposed upon a cast of the active site. This region has numerous hydrogen-bond donor and acceptor sites scattered throughout. As shown in the side view, this area is also relatively flat.

From our observations of this region, we can conclude that a planar, cyclic structure would again best serve as



HB DONORS DBASE - PARAMETERS USED

Occupancy Requirements

Backbone

'nn' 'o'	= 0
'n'	< 3

Sidechain

'NO' 'ON' 'NN' 'OO'	= 0
'N' O'	> 2

Complete SMILES

'n(' 'n1('	= 0
'()' '(('	= 0
'OC(=O)' '=O))'	= 0
'C(=O)C(=O)'	= 0
'CC(=O)C' 'COC'	= 0
'C(=O)OC' 'C)'	= 0
'n1'	< 2
'N' 'O'	> 2

Component	Backbone Freq.	Sidechain Freq.	# Atoms	No. of Connections
C	0.90	0.70	1	3
N	0.10	0.10	1	2
O	0.0	0.10	1	2
C(=O)	0.0	0.10	1	2

Cyclic Template "-1-----1"

# Sidechains	Sidechain Lengths
2 (40%)	1 (20%)
3 (50%)	2 (40%)
4 (10%)	3 (40%)

01. c1c(CC(=O)N)c(C(=O)N)ccc1
02. c1(CC(=O)N)ccccc1(CO)
03. c1cc(CC(=O)N)cc(O)c1(CCO)
04. c1cc(N)cc(O)c1(CC(=O)N)
05. c1cc(O)ncc1(CC(=O)N)
06. c1(CC(=O)N)c(N)cccc1
07. c1(O)ccccc1(CC(=O)N)
08. c1c(N)cccc1(C(=O)N)
09. c1(CC(=O)N)cccc(C(=O)O)c1
10. c1(CO)cc(C(=O)N)cnc1
11. c1(CCO)c(C(=O)CN)ccc(N)c1
12. c1(N)c(O)c(CCN)ccc1
13. c1(C(=O)O)cccc(CCN)c1
14. c1(CCN)c(N)c(O)ccc1

Fig. 11. (C) DBMAKER scenario #1. Generation of hydrogen-bond donor database.

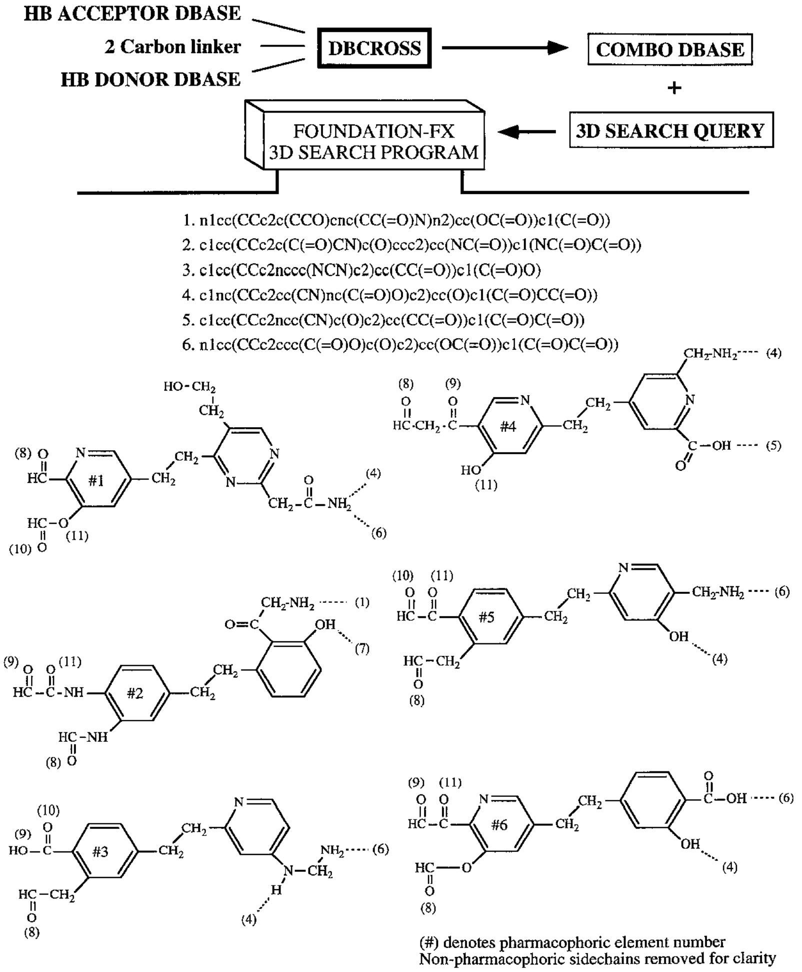
a foundation upon which complementary functional groups could be anchored. Although this portion of the active site is fairly flat, it is also quite extensive in area. Thus, we can use five-membered, six-membered and fused multi-ring assemblies as a foundation for ligand construction. This is accomplished by using the three templates shown in Fig. 12. However, one cannot simply generate combinations of atoms haphazardly and then plug them into the template. The reason for this is that combinations of capital-letter SMILES symbols, such as C1CCCC1 and C1NCCN1, produce substituted aliphatic cyclic structures that are *nonplanar*. Converting these to aromatic cyclic templates, i.e. '1––––1', where lower-case letters would substitute for the '–' symbols, can be done. However, five-membered aromatic rings such as c1cccc1 are not chemically possible (although CONCORD will generate them).

The trick lies in the selection of backbone components. Figure 12 lists those that are employed. We find that specific combinations of C=N, N=C, C=C, C(=O), N and O will produce planar rings when written to SMILES strings with the cyclic templates. All carbon atoms in the backbone must be sp^2 hybridized to assure planarity, hence our choice of elements. With these components, the generated rings are planar and contain numerous sites to which substituent groups may be attached.

The '#connections' designations are especially important in this example. All backbone carbons are sp^2, thus, they are able to bond with three neighbors (two ring atoms and a side chain). As such, all carbons will have a designation of three. Any backbone nitrogen can only bond with its two neighbors, thus its designation is two. Note also that the 'two-atom' components (C=C, C=N, N=C) contain two '#connections' designations, since DBMAKER can substitute the double bond freely.

Two to four side chains are specified per structure, each containing one to four atoms or functional groups. The same four side-chain components as implemented previously are listed in the parameter file, together with their frequencies of utilization.

The backbone and side-chain requirements are again designed to eliminate bonded pairs of heteroatoms. As

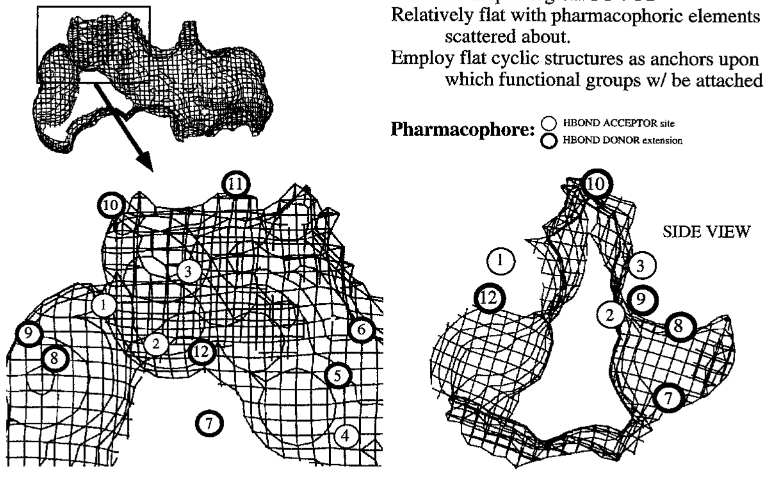


1. n1cc(CCc2c(CCO)cnc(CC(=O)N)n2)cc(OC(=O))c1(C(=O))
2. c1cc(CCc2c(C(=O)CN)c(O)ccc2)cc(NC(=O))c1(NC(=O)C(=O))
3. c1cc(CCc2nccc(NCN)c2)cc(CC(=O))c1(C(=O)O)
4. c1nc(CCc2cc(CN)nc(C(=O)O)c2)cc(O)c1(C(=O)CC(=O))
5. c1cc(CCc2ncc(CN)c(O)c2)cc(CC(=O))c1(C(=O)C(=O))
6. n1cc(CCc2ccc(C(=O)O)c(O)c2)cc(OC(=O))c1(C(=O)C(=O))

(#) denotes pharmacophoric element number
Non-pharmacophoric sidechains removed for clarity

Fig. 11. (D) DBMAKER scenario #1. Generation of combo database and resulting hits from FOUNDATION-FX search.

HIV-1 Protease Active Site



Crescent-shaped region: P1 + P2
Relatively flat with pharmacophoric elements scattered about.
Employ flat cyclic structures as anchors upon which functional groups w/ be attached.

Pharmacophore: ○ HBOND ACCEPTOR site
○ HBOND DONOR extension

HIV "CRESCENT" DATABASE - PARAMETERS USED

Occupancy Requirements

Backbone

'NO' 'ON' 'NN' 'OO'	= 0

Sidechain

'NO' 'ON' 'NN' 'OO'	= 0
'N' 'O'	> 1

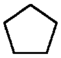
Complete SMILES

'()' '(('	= 0
'C)'	= 0
'N(O' 'N(N'	
'N1(N' 'N1(O'	= 0
'O(' 'O1' 'O2'	= 0
'1(' '2('	= 0
'N1'	< 2
'N2'	< 2

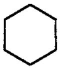
Component	Backbone Freq.	Sidechain Freq.	# Atoms	No. of Connections
C=N	0.20	0.00	2	3=2
C=C	0.20	0.00	2	3=3
N=C	0.20	0.00	2	2=3
C(=O)	0.20	0.10	1	2
C	0.00	0.30	1	4
N	0.10	0.30	1	3
O	0.10	0.30	1	2

TEMPLATES

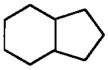
"+1++++1"



"+1+++++1"



"+1+++2+1++++2"



# Sidechains	Sidechain Lengths
2 (40%)	1 (30%)
3 (50%)	2 (30%)
4 (10%)	3 (30%)
	4 (10%)

Fig. 12. DBMAKER scenario #2. Attempt to generate ligands capable of binding with P1/P2 of the HIV-1 protease active site.

before, the 'NO, ON, NN and OO = 0' requirement restricts the generation of bonded heteroatoms in either the backbone or side chain. Furthermore, the 'N O > 1' requirement guarantees that at least two hydrogen-bond donating and/or accepting groups will be present in the side chains.

The complete SMILES requirements need a bit more explanation. The 'C) = 0' specification prevents occurrence of side chains terminating in a methyl group. By doing so, we force side chains to terminate with functional groups which are then better positioned to interact with the receptor. This also enables the side chain to better search the available conformational space. Solitary N or O atoms may also be chosen as backbone components. Valence restrictions prevent backbone oxygens, but not backbone nitrogens, from anchoring a side chain. As such, the 'N(O N(N N1(O N1(N = 0' requirement prevents any side-chain heteroatoms from branching off of any backbone nitrogen atom.

In using the fused 6-5 ring template '+1+++2+1++++2', oxygen atoms may not be selected to fill backbone positions preceding the middle two numeric labels. This is because any atom in these positions is bonded to three other neighbors. This violates the oxygen valence of two. Thus, the requirement 'O(O1 O2 = 0' is specified. In fact, any atom preceding the middle two numeric labels is unable to bind any side chains. Thus, the '1(2(= 0' specification is used. Finally, if a SMILES string contains two 'N1' or 'N2' symbols, then the two nitrogens are bonded to one another by definition. The last two requirements prevent this occurrence.

An initial group of 5000 SMILES strings was randomly produced within the scope of the defined parameters. Processing and extraction of unique SMILES strings left 4125 different structures. These strings were then converted to SYBYL multi_MOL format with CONCORD. The entire process required approximately 15 min of CPU time (R4000 Indigo).

To determine how well these structures complemented the active site for which they were designed, a 3D database search was again conducted against the HIV-1 protease pharmacophore using the program FOUNDATION-FX. The search was performed to retrieve structures containing any combination of four or more query elements. A number of constraints were used to limit the number of hits retrieved. First, a maximum of four hits per structure (different conformations) was specified. Second, the coordinates of each retrieved hit were re-

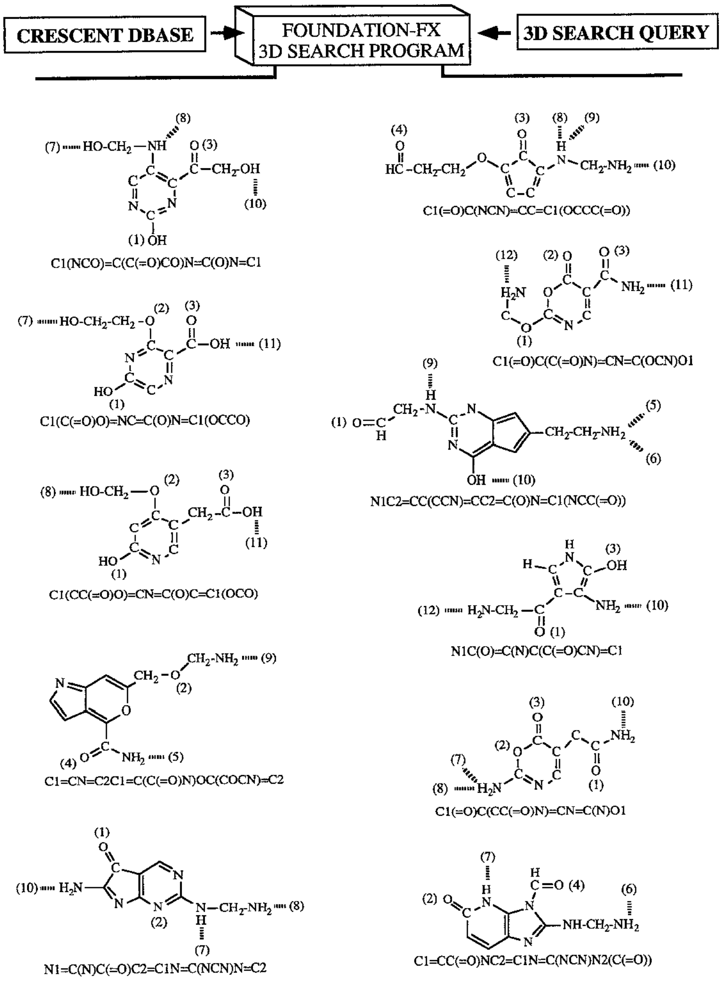

CRESCENT DBASE
FOUNDATION-FX
3D SEARCH PROGRAM
3D SEARCH QUERY
C1(NCO)=C(C(=O)CO)N=C(O)N=C1
C1(=O)C(NCN)=CC=C1(OCCC(=O))
C1(=O)C(C(=O)N)=CN=C(OCN)O1
C1(C(=O)O)=NC=C(O)N=C1(OCCO)
N1C2=CC(CCN)=CC2=C(O)N=C1(NCC(=O))
C1(CC(=O)O)=CN=C(O)C=C1(OCO)
N1C(O)=C(N)C(C(=O)CN)=C1
C1=CN=C2C1=C(C(=O)N)OC(COCN)=C2
C1(=O)C(CC(=O)N)=CN=C(N)O1
N1=C(N)C(=O)C2=C1N=C(NCN)N=C2
C1=CC(=O)NC2=C1N=C(NCN)N2(C(=O))

Fig. 12. (continued).

quired to differ from any previous hit by more than 0.500 Å rms deviation. Third, a maximum error of 0.500 Å rms deviation from the query was tolerated. Fourth, a maximum number of 4000 search conformations was allowed per database structure. Finally, any hit was required to fit within the active site using a reduced atom radius factor of 0.700.

Given the query of 12 hydrogen-bond donor and acceptor sites, 25 five-query element hits were retrieved (nine different structures) and 140 four-query element hits were found. Selected structures are shown in Fig. 12 together with the pharmacophoric elements they complement. The group of retrieved compounds was diverse, each engineered to complement the P1-P2 region of the HIV-1 protease active site.

The third scenario is shown in Fig. 13. This example highlights the ability of DBMAKER to customize a database directed at a particular chemical construction. Suppose we have an HIV-1 protease inhibitor lead compound which we are interested in derivatizing to improve activity. We would like to generate a 3D structural database of derivatives for further analysis.

Barrish et al. recently published the design, synthesis and preliminary SAR of a series of HIV protease inhibitors containing a novel C2-symmetrical 'aminodiol' core structure [31]. The lead compound 'template' is shown in Fig. 13 together with the substituents tested and their positions. To generate the database of derivatives, we proceed by manually creating two databases: one containing the structures of various lead compounds, the other containing the possible substituent structures. These two databases are then crossed using the DBCROSS program to form the database of derivatives. The exact backbone positions for substituent additions may be specified together with the number of elements to place. Due to the combinatorial nature of this process, a large number of structures can be produced from various functional groups and a few substitution sites. The inhibitory activity of the structures in the resulting database can then be studied using numerous analytical methods.

The last scenario is described in Fig. 14. Here, we attempt to generate ligands that mimic the structure of NADPH. This example shows how a database of complex structures can be assembled iteratively from a few simple components. The structure of NADPH is shown in the figure. Schematically, we can divide this structure into two nucleosides joined by a small linker chain. Each nucleoside is composed of a heterocyclic base attached to a sugar molecule. Thus, a total of five components is present: a short *linker chain* that joins two *sugar* moieties, each attached to a *heterocyclic base*. Using this molecule as a model, we will generate a database of similar chemical structures.

Our plan is as follows. First, two databases must be generated: one containing heterocyclic bases and the other composed of sugars. We then use the program DBCROSS to 'cross' these two groups of parental structures in a 1:1 ratio. This produces a new database of nucleosides, each containing one sugar and one base. We then generate a small group of linker chains. Finally, the linker chains are crossed with the collection of nucleosides to produce the final product. This last crossing is performed with a 2:1 ratio of nucleosides to chains.

The heterocyclic database is generated using the same parameters and occupancy requirements employed for scenario #2 above. Again, three different cyclic templates are employed. The '+1++++1' template generates five-membered heterocycles such as the substituted imidazoles described above. Adding one atom, the '+1+++++1' six-membered template produces compounds similar to pyrimidines. Lastly, the two rings are fused together by the '+1+++2+1++++2' template, generating compounds structurally related to purines. Using these parameters, a heterocyclic database containing 500 structures was generated in approximately 3 min of CPU time.

Far more planning is involved in generating the carbohydrate database. First of all, sugars are nonaromatic compounds; thus, our cyclic templates must contain '+'s instead of '–'s. In the diagram of NADPH in Fig. 14, we see that backbone atoms 1 and (N–1) of the sugar are the atoms to which the other components are attached. In generating our SMILES representations, we will leave those atoms unsubstituted. We see this in the cyclic template 'C1++CO1' which represents a furanose sugar. In this five-membered ring, carbon atoms 1 and 4 are written into the template. Thus, side-chain substitution is only permitted in the remaining carbons. For a six-membered pyranose sugar, 'C1+++CO1' is the corresponding template. Again, substitution is allowed only in the middle three carbons.

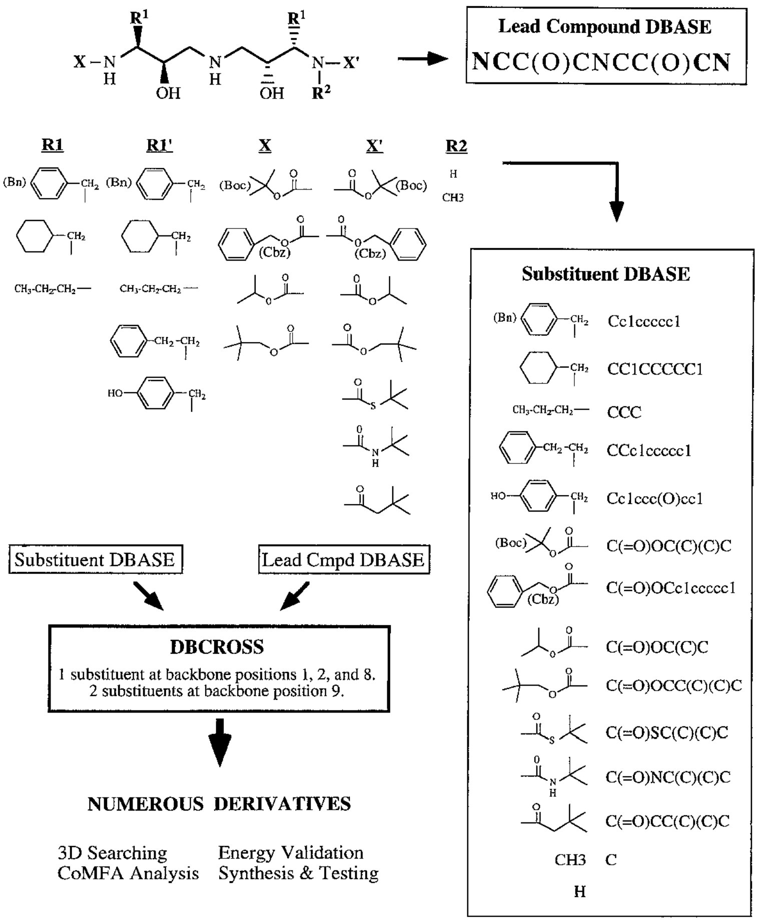


Fig. 13. DBMAKER scenario #3. Use of DBCROSS to generate derivatives from lead compounds and substituents [31].

Side-chain lengths are kept short (1–2 groups). Side-chain constituents are relatively simple. Hydroxyl groups, alcohols, simple ethers, amine and amide derivatives are specified. These are listed in the parameters together with their utilization frequencies. With these parameters, DBMAKER and CONCORD were used to generate a carbohydrate database containing 150 compounds. This required approximately 2 min of CPU time (R4000 Indigo). Due to the constraints placed on the structure-generating process, the total number of possible structures is low.

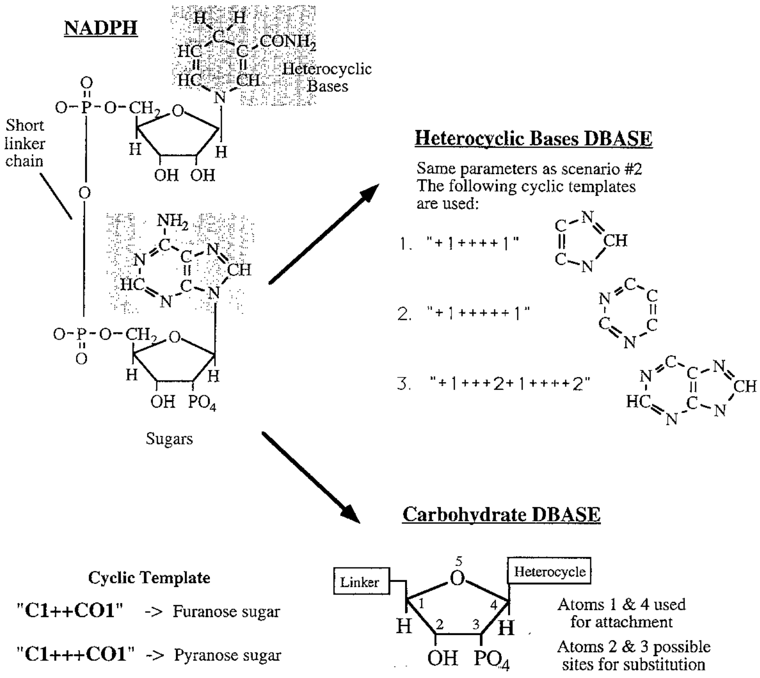
DBCROSS was then used to randomly link carbohydrate structures to heterocyclic compounds in a 1:1 ratio. In each cross, the heterocyclic compound was joined specifically to sugar carbon 4 or 5, as discussed above. Initially, 2000 SMILES 'offspring' were generated. After processing, extraction of unique elements, and 3D coordinate generation, 1900 structures remained (about 95%). The entire process required approximately 40 min of CPU time (R4000 Indigo).

A database of short linker fragments was generated using the standard four SMILES components 'C', 'N', 'O' and 'C(=O)'. No templates or side chains were used. DBMAKER was directed to generate a small database of 100 compounds containing 3–5 backbone components.

Finally, DBCROSS was again employed to randomly mate a single linker with two heterocyclic/carbohydrate structures. In each cross, the backbone atoms of the selected sugars were joined specifically to the *ends* of the chosen linker chain. Initially, 2000 SMILES 'offspring' were generated. After processing, extraction of unique elements, and 3D coordinate generation, approximately 1800 structures remained (about 90%). The entire process required approximately 65 min of CPU time (R4000 Indigo). Representative compounds are displayed at the bottom of Fig. 14. As shown, these structures meet the objectives already discussed.

Discussion

As we have demonstrated, DBMAKER-generated databases can be utilized in numerous aspects of ligand design. No matter what de novo design approach is employed, the availability of 3D structural information is paramount. The majority of ligand design systems screen 3D databases in order to find compounds that orient key functional groups to match a given pharmacophore. As this technique is limited to the extent and quality of one's databases, the ability to control the size, shape, structure and functional group content of the search structures is of great utility. It is these components that give rise to


NADPH
Heterocyclic Bases
Short linker chain
Sugars
Heterocyclic Bases DBASE
Same parameters as scenario #2
The following cyclic templates are used:
1. "+1++++1"
2. "+1+++++1"
3. "+1+++2+1++++2"
Carbohydrate DBASE
Cyclic Template
"C1++CO1" -> Furanose sugar
"C1+++CO1" -> Pyranose sugar
Linker
Heterocycle
Atoms 1 & 4 used for attachment
Atoms 2 & 3 possible sites for substitution

Component	Backbone Freq.	Sidechain Freq.	# Atoms	No. of Connections
C	1.00	0.02	1	3
O	----	0.80	1	2
CO	----	0.10	2	2
OC	----	0.04	2	2
N	----	0.02	1	2
NC(=O)	----	0.02	2	2

# Sidechains	Sidechain Lengths
2 (50%)	1 (50%)
3 (50%)	2 (40%)
	3 (10%)

Fig. 14. DBMAKER scenario #4. Attempt to generate ligands that mimic the structure of NADPH.

possible lead compounds when screened against a particular query.

Scenarios #1 and #2 show how a database can be designed to complement the steric and electrostatic characteristics of an active site. In the DHFR example, we find an active site that has both steric and electrostatic constraints. The active site is divided into two distinct regions. One has hydrogen-bond donor elements exclusively, while the other contains only hydrogen-bond acceptor sites. Each domain is relatively flat, and both are linked by a narrow channel. To attack this problem, we generated two sub-databases, each designed to complement one of the two regions. The final search database was constructed by crossing one member from each sub-database with a short hydrocarbon linker fragment. In doing so, we improve the chances that a particular structure in the database will complement the active site. As shown in the results, the database was successful in meeting the electrostatic requirements of the active site, while conforming to the steric demands.

In scenario #2, we target the P1+P2 region of the HIV-1 protease. This crescent-shaped active site is larger than the previous example, however, it is also relatively flat. Furthermore, hydrogen-bond donor and acceptor groups are interspersed throughout the region. Since the active site is relatively flat, we selected planar, cyclic structures to serve as anchors upon which functional groups could be distributed. The three SMILES templates were chosen to produce planar structures of differing sizes. Numerous other backbone SMILES templates could have been chosen, so long as the structures do not violate the steric confines of the cavity. In this example, hydrogen-bond donors and acceptors were mixed and randomly distributed throughout the cyclic templates. As shown in the results, the database was successful in meeting the requirements of the active site.

Scenarios #3 and #4 describe another facet of DB-MAKER. These examples highlight the ability of DB-MAKER to customize a database directed at a particular chemical construction. In scenario #3, we have an HIV-1 protease inhibitor lead compound which we are interested in derivatizing to improve activity. Thus, we would like to generate a 3D structural database of derivatives for further analysis. Two databases must first be produced: one containing the structures of various lead compounds, the other containing the possible substituent structures.

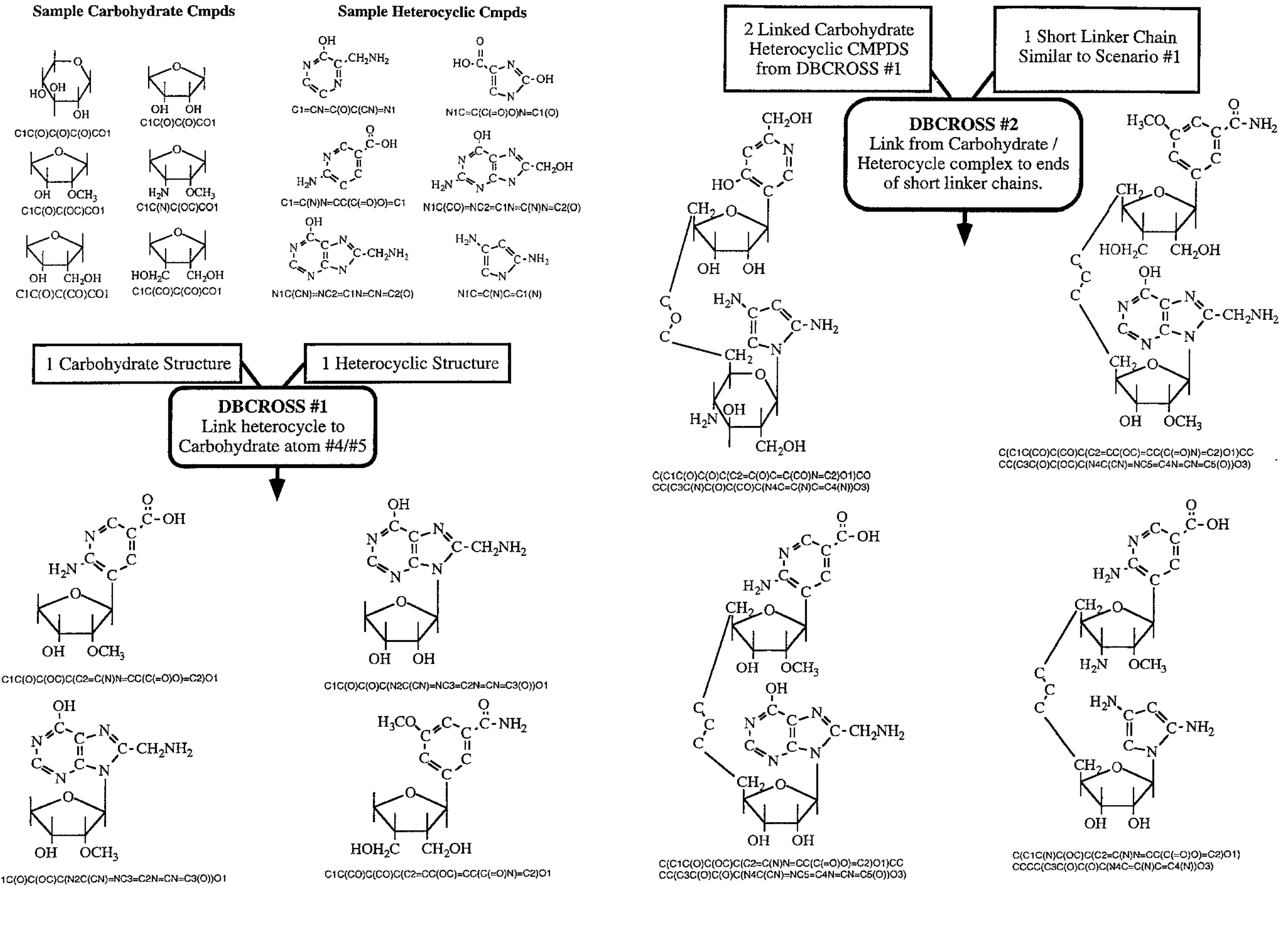

Sample Carbohydrate Cmpds
Sample Heterocyclic Cmpds
C1C(O)C(O)C(O)CO1
C1C(O)C(O)CO1
C1C(O)C(OC)CO1
C1C(N)C(OC)CO1
C1C(O)C(CO)CO1
C1C(CO)C(CO)CO1
C1=CN=C(O)C(CN)=N1
N1C=C(C(=O)O)N=C1(O)
C1=C(N)N=CC(C(=O)O)=C1
N1C(CO)=NC2=C1N=C(N)N=C2(O)
N1C(CN)=NC2=C1N=CN=C2(O)
N1C=C(N)C=C1(N)
1 Carbohydrate Structure
1 Heterocyclic Structure
DBCROSS #1
Link heterocycle to
Carbohydrate atom #4/#5
C1C(O)C(OC)C(C2=C(N)N=CC(C(=O)O)=C2)O1
C1C(O)C(O)C(N2C(CN)=NC3=C2N=CN=C3(O))O1
C1C(O)C(OC)C(N2C(CN)=NC3=C2N=CN=C3(O))O1
C1C(CO)C(CO)C(C2=CC(OC)=CC(C(=O)N)=C2)O1
2 Linked Carbohydrate
Heterocyclic CMPDS
from DBCROSS #1
1 Short Linker Chain
Similar to Scenario #1
DBCROSS #2
Link from Carbohydrate /
Heterocycle complex to ends
of short linker chains.
C(C1C(O)C(O)C(C2=C(O)C=C(CO)N=C2)O1)CO
CC(C3C(N)C(O)C(CO)C(N4C=C(N)C=C4(N))O3)
C(C1C(CO)C(CO)C(C2=CC(OC)=CC(C(=O)N)=C2)O1)CC
CC(C3C(O)C(OC)C(N4C(CN)=NC5=C4N=CN=C5(O))O3)
C(C1C(O)C(OC)C(C2=C(N)N=CC(C(=O)O)=C2)O1)CC
CC(C3C(O)C(O)C(N4C(CN)=NC5=C4N=CN=C5(O))O3)
C(C1C(N)C(OC)C(C2=C(N)N=CC(C(=O)O)=C2)O1)
CCCC(C3C(O)C(O)C(N4C=C(N)C=C4(N))O3)

Fig. 14. (continued).

These two databases are then crossed using the DBCROSS program to form the database of derivatives. Due to the combinatorial nature of this process, a large number of structures can be produced from various functional groups and a few substitution sites.

DBMAKER can also be used to generate databases that complement the synthetic strengths of a laboratory. For example, suppose that an investigator is interested in developing peptide mimetics as possible HIV-1 protease inhibitors. During the database generation, an occupancy requirement can be specified to eliminate all peptide bonds, 'C(=O)N = O'. In addition, one can specify the inclusion of a specific desired linkage and require that all database structures contain that construction.

Scenario #4 shows the construction of a database of compounds whose structure mimics that of NADPH. This example illustrates how complex structures can be assembled iteratively from a few simple components. NADPH can be broken down into two nucleosides joined by a short linker chain. Each nucleoside can be broken down again into a heterocyclic base attached to a furanose or pyranose sugar. Thus, two sub-databases are required: one database of heterocyclic bases and one of carbohydrates. The heterocyclic base structures are generated with the exact same parameters as those listed in scenario #2.

The carbohydrate database is another example of the generation of compounds which reproduce a desired chemical construction. For five-membered sugars, the basic structure is the same, as shown in Fig. 14. The oxygen completes the ring by joining carbons 1 and 4, which are used to connect with the linker and a heterocyclic base, respectively. Thus, that portion of the molecule is constant, and is reflected in the cyclic template. However, substitution is allowed at positions 2 and 3. Thus, DBMAKER is instructed to place possible substituents at these loci only. The structure of the sugars is enforced through careful selection of templates and occupancy requirements.

The remaining tasks are simple. Using DBCROSS, nucleosides are created by joining a single, randomly chosen carbohydrate with a single heterocyclic base. Then, in a cross resembling that employed in scenario #1, two nucleosides are spliced onto the ends of a short linker fragment. Complex molecules result, resembling NADPH, as shown with the SMILES representations generated by DBMAKER and DBCROSS.

The user must consider both the configuration of pharmacophoric elements as well as the size and shape of the active site in determining the appropriate database structures to generate. The extreme polarity of the DHFR active site allowed the generation of two separate databases, each designed to complement a specific portion of the active site. However, most active sites contain hydrogen-bond donors and acceptors scattered throughout. Thus, the parameters invoked should reflect the situation.

Once a database is generated, one usually receives immediate feedback as to its performance in solving a particular ligand design problem. This information can then be used to modify the parameters to remedy problems.

Usually, the first difficulty involves steric interference between the ligands and the receptor site. This results in no 'hits' being recovered from a 3D search program. Often, one must simply reduce the number of backbone components, or at least skew the utilization frequencies to that effect. In addition, the number and length of side chains specified may need to be adjusted. The backbone may be fine; however, the side chain content precludes a good fit. If templates are used, one must ensure that a 'bare and unadorned' template structure is sterically allowed. If not, adjustments must be made to the template itself before modifying the number and content of substituents.

Once steric issues have been resolved, electrostatic complementarity can be optimized by modifying the content and utilization frequencies of the various SMILES components. Obviously, a lack or excess of hydrogen-bond donors or acceptors should be remedied with the addition or reduction of complementary groups. Again, the utilization frequencies can be skewed to favor one or the other.

When placing substituents about a template, one must ensure the proper 'spacing' of side chains. Often structures result where side chains are dropped onto consecutive backbone atoms. This causes two problems. First, functional groups are crowded into a small region of the active site, limiting the interaction between ligand and receptor. Second, side chains interfere sterically with one another, limiting the conformational space they can access. This can be remedied by appropriate occupancy requirements. If side chains are being placed about an aromatic ring, the requirement ')c(< 2' limits the placement of side chains in consecutive positions.

In the examples shown above, most ligands were generated using cyclic five- and six-membered rings as 'foundations' upon which functional groups could be anchored. Thus, simple five- and six-membered cyclic templates were used. However, with active sites that are less sterically hindered, DBCYCLE can be used to generate numerous cyclic templates containing a user-specified number of rings (and ring overlap). Furthermore, crossing multicyclic SMILES representations and short linear fragments can produce more complex templates. Such templates code for large structures that have several rotatable bonds but which are, for the most part, entropically constrained. Thus, one can span a larger active site, placing functional groups throughout the region, and yet retain confidence that resulting conformations are attainable in solution.

For a given set of DBMAKER parameters, there is an upper limit to the number of structures that can be generated. In the initial generation phase, nearly all the randomly created structures will be novel. However, as the number of compounds generated increases, the number of duplications will increase. As such, the number of unique structures will asymptotically approach the upper limit. This limit, however, is entirely dependent upon the scope of the parameters. In a very constrained system, such as the carbohydrate database of scenario #4, the upper limit will be fairly small. Conversely, in a system with numerous crosses between several databases, such as the NADPH database, the upper limit may be combinatorially enormous.

Conclusions

In any approach involving 3D searching, the quality and availability of databases is always a concern. Since the actual coordinates of each DBMAKER structure are generated using CONCORD, high-quality structures result. Furthermore, since DBMAKER assembles structures using SMILES notation, a large variety of compounds can be produced. By providing users with mechanisms to tailor the database to their specific requirements as to chemical classes considered, the candidate structures can be both diverse (not reflecting any biases in experimental databases) and/or focused (limited to chemistry expertise found in an individual laboratory). There is conceivably no limit to the variety of structures which can be generated with DBMAKER. In designing this software, we felt that a flexible alternative was needed to the commercially available products. DBMAKER was developed to allow investigators, previously hindered by a lack of 3D information, to employ molecular design techniques.

Acknowledgements

This work was supported by the National Institutes of Health (Grants GM24483 and AI27302) and the Medical Scientist Training Program (CMWH) in the Division of Biology and Biomedical Sciences at Washington University (NIH Training Grant GM07200).

References

1 Goodman, A.G. and Gilman, L.S., The Pharmacological Basis of Therapeutics, Macmillan, New York, NY, 1985.

2 Fersht, A., Enzyme Structure and Mechanism, Freeman, New York, NY, 1977.

3 Beddell, C.R., The Design of Drugs to Macromolecular Targets, Wiley, New York, NY, 1992.

4 Martin, Y.C., Bures, M.G. and Willett, P., In Lipkowitz, K.B. and Boyd, D.B. (Eds.) Reviews in Computational Chemistry, VCH Publishers Inc., New York, NY, 1990, pp. 213–256.

5 Martin, Y.C., J. Med. Chem., 35 (1992) 2145.

6 Borman, S., Chem. Eng. News, 70 (1992) 18.

7 Sheridan, R.P., Rusinko III, A., Nilakantan, R. and Venkataraghavan, R., Proc. Natl. Acad. Sci. USA, 86 (1989) 8165.

8 Gund, P., Prog. Mol. Subcell. Biol., 5 (1977) 117.

9 Gund, P., Annu. Rep. Med. Chem., 14 (1979) 299.

10 Allen, F.H., Kennard, O. and Taylor, R., Acc. Chem. Res., 16 (1983) 146.

11 Abola, E.E., Bernstein, F.C. and Koetzle, T.F., In Glaeser, P.S. (Ed.) The Role of Data in Scientific Progress, Elsevier, New York, NY, 1985.

12 Rusinko III, A., Skell, J.M., Balducci, R. and Pearlman, R.S., CONCORD, University of Texas at Austin, distributed by Tripos Associates Inc., St. Louis, MO.

13 Weininger, D., J. Chem. Inf. Comput. Sci., 28 (1988) 31.

14 Chemical Abstracts Services, Columbus, OH.

15 Molecular Design Ltd, San Leandro, CA.

16 Chemical Design Ltd, Oxford.

17 Office of Technology Licensing, Berkeley, CA.

18 Bartlett, P.A., Shea, G.T., Telfer, S.J. and Waterman, S., In Roberts, S.M. (Ed.) Molecular Recognition: Chemical and Biological Problems, Royal Society of London, London, 1989, pp. 182–196.

19 Tripos Associates Inc., St. Louis, MO.

20 Sprague, J.C., Yuh, T.Y. and Allinger, N.L., J. Comput. Chem., 8 (1987) 581.

21 Stewart, J.J.P., MOPAC Manual: A general purpose molecular orbital package, USAF Academy, Colorado Springs, CO, 1985.

22 Frisch, M.J., GAUSSIAN82 User's Manual, Carnegie-Mellon University, Pittsburgh, PA, 1986.

23 Pearlman, R.S., In Dunn, W.J., Block, J.H. and Pearlman, R.S. (Eds.) Partition Coefficient: Determination and Estimation, Pergamon Press, New York, NY, 1986, p. 165.

24 Johnson, C.K., Nucl. Sci. Abstr., 19 (1965) 4153.

25 Weininger, D., Weininger, A. and Weininger, J.L., J. Chem. Inf. Comput. Sci., 29 (1989) 97.

26 Bolin, J.T., Filman, D.J., Matthews, D.A., Hamlin, R.C. and Kraut, J., J. Biol. Chem., 257 (1982) 13650.

27 Ho, C.M.W. and Marshall, G.R., J. Comput.-Aided Mol. Design, 4 (1990) 337.

28 Ho, C.M.W. and Marshall, G.R., J. Comput.-Aided Mol. Design, 7 (1993) 3.

29 Dammkoehler, R.A., Karasek, S.F., Shands, E.F.B. and Marshall, G.R., J. Comput.-Aided Mol. Design, 3 (1989) 3.

30 Martin, J.A., Antiviral Res., 17 (1992) 265.

31 Barrish, J.C., Gordon, E., Alam, M., Lin, P.F., Bisacchi, G.S., Chen, P., Cheng, P.T.W., Fritz, A.W., Greytok, J.A., Hermsmeier, M.A., Humphreys, W.G., Lis, K.A., Marella, M.A., Merchant, Z., Mitt, T., Morrison, R.A., Obermeier, M.T., Pluscec, J., Skoog, M., Slusarchyk, W.A., Spergel, S.H., Stevenson, J.M., Sun, C., Sundeen, J.E., Taunk, P., Tino, J.A., Warrack, B.M., Colonno, R.J. and Zahler, R., J. Med. Chem., 37 (1994) 1758.