tabase structure, irrespective of whether it lies in part or wholly within variable parts of the structure. For a substructure query Q, the number and variety of MAY screens cannot be determined. Consequently, in order to use in this case the improved algorithm described above and to ensure against loss in recall, $Q_{MUST} \cup Q_{MAY}$ would have to be assigned such that every bit was set ("bit density" = 100%). The bitstring match $S_{MUST} \subseteq Q_{MUST} \cup Q_{MAY}$, therefore, becomes valueless in the case of a substructure query.

The two degenerate screening algorithms described here have been implemented, and detailed investigation of screening performance will begin shortly when a test database of generic structures from chemical patents becomes available for experimental purposes. The results of these investigations together with details of the implementation of higher level generic structure screening and substructure search algorithms will be reported separately.

ACKNOWLEDGMENT

We are indebted to Dr. J. Silk, whose advice has contributed to the formulation of the search strategies described in this paper, and Dr. M. Elder and Dr. P. Willett for helpful discussions. We also acknowledge the British Library Research and Development Department for provision of research funding during the period October 1981 to October 1983.

REFERENCES AND NOTES

(1) Lynch, M. F.; Barnard, J. M.; Welford, S. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 1. Introduction and General Strategy". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 148-150.
(2) Barnard, J. M.; Lynch, M. F.; Welford, S. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 2. GENSAL: A Formal Language for the Description of Generic Chemical Structures". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 151-161.
(3) Welford, S. M.; Lynch, M. F.; Barnard, J. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 3. Chemical Grammars and Their Role in the Manipulation of Chemical Structures". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 161-168.
(4) Barnard, J. M.; Lynch, M. F.; Welford, S. M. "Computer Storage and Retrieval of Generic Structures in Chemical Patents. 4. An Extended Connection Table Representation for Generic Structures". *J. Chem. Inf. Comput. Sci.* **1982**, *22*, 160-164.
(5) Welford, S. M.; Lynch, M. F.; Barnard, J. M. "Towards Simplified Access to Chemical Structure Information in the Patent Literature". *J. Inf. Sci.* **1983**, *6*, 3-10.
(6) Barnard, J. M.; Lynch, M. F.; Welford, S. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 6. An Interpreter Program for the Generic Structure Description Language GENSAL". *J. Chem. Inf. Comput. Sci.*, following paper in this issue.
(7) Welford, S. M.; Lynch, M. F.; Barnard, J. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 7. TOPOGRAM: A Topological Grammar for the Representation, Generation and Recognition of Chemical Radical Classes". In preparation.
(8) Lynch, M. F.; Harrison, J. M.; Town, W. G.; Ash, J. E. "Computer Handling of Chemical Structure Information"; Macdonald: London, 1971.
(9) O'Korn, L. J. "Algorithms in the Computer Handling of Chemical Information". *ACS Symp. Ser.* **1977**, *46*, 122-148.
(10) Lynch, M. F. In "Chemical Information Systems"; Ash, J. E.; Hyde, E., Eds.; Ellis Horwood: Chichester, England, 1975; Chapter 12.
(11) Adamson, G. W.; Cowell, J.; Lynch, M. F.; McLure, A. H. W.; Town, W. G.; Yapp, A. M. "Strategic Considerations in the Design of a Screening System for Substructure Searches of Chemical Structure Files". *J. Chem. Doc.* **1973**, *13*, 153-157.
(12) Feldman, A. J.; Hodes, L. "An Efficient Design for Chemical Structure Searching. I. The Screens". *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 147-152.
(13) Von Scholley, A. submitted for publication in *J. Chem. Inf. Comput. Sci.*
(14) Farmer, N. A.; O'Hara, M. F. "CAS ONLINE—A New Source of Substance Information from Chemical Abstracts Service". *Database* **1980**, 10-25.
(15) Dittmar, P. G.; Farmer, N. A.; Fisanick, W.; Haines, R. C.; Mockus, J. "The CAS ONLINE Search System. 1. General System Design and Selection, Generation, and Use of Search Screens". *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 93-102.
(16) Graf, W.; Kaindl, H. K.; Kniess, H.; Schmidt, B.; Warszawski, R. "Substructure Retrieval by Means of the BASIC Fragment Search Dictionary Based on the Chemical Abstracts Service Chemical Registry III System". *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 51-55.
(17) Graf, W.; Kaindl, H. K.; Kniess, H.; Warszawski, R. "The Third BASIC Fragment Search Dictionary". *J. Chem. Inf. Comput. Sci.* **1982**, *22*, 177-181.
(18) Feldman, A.; Hodes, L. "Substructure Search with Queries of Varying Specificity". *J. Chem. Inf. Comput. Sci.* **1979**, *19*, 125-129.
(19) "CAS ONLINE Screen Dictionary for Substructure Search", 2nd ed.; Chemical Abstracts Service: Columbus, OH, 1981.
(20) Balent, M. Z.; Emberger, J. M. "A Unique Chemical Fragmentation System for Indexing Patent Literature". *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 100-104.
(21) Adamson, G. W.; Lynch, M. F.; Town, W. G. "Analysis of Structural Characteristics of Chemical Compounds in a Large Computer-Based File. Part 2. Atom-Centered Fragments". *J. Chem. Soc. C* **1971**, 3702-3706.

Computer Storage and Retrieval of Generic Chemical Structures in Patents. 6. An Interpreter Program for the Generic Structure Description Language GENSAL

JOHN M. BARNARD, MICHAEL F. LYNCH,* and STEPHEN M. WELFORD
Department of Information Studies, University of Sheffield, Sheffield S10 2TN, U.K.

Received December 8, 1983

A computer program is described that carries out syntactic and semantic analysis of generic structures encoded in GENSAL, a formal language for the description of such structures, simultaneously generating an extended connection table representation of the structure. Desirable enhancements to the program in the areas of structure diagram input, nomenclature translation, and linear formula analysis are discussed.

INTRODUCTION

Part 2 of this series[1] presented a formal language, GENSAL, which is designed for the representation of generic chemical structures in a form that remains as close as possible to that used in chemical patent specifications and abstracts but that is sufficiently formalized to be amenable to automatic processing by computer. GENSAL consists of a mixture of conventional two-dimensional structure diagrams and text, the latter including various special *delimiter* words ("IF", "BEGIN", "OSB", etc.) and symbols ("/", "&", "$=", etc.), integers, chemical nomenclatural terms, and linear formulas. A comprehensive instruction manual for GENSAL has recently been prepared.[2] GENSAL has some features in common with the R_k notation used for generic query structures in the COUSIN system developed at the Upjohn Co.[3]

Part 4 of this series[4] described an extended connection table representation (ECTR) for generic structures, which is a complete and unambiguous (though nonunique) representation of a generic structure. The ECTR is based on connection tables and the use of parameters to the "chemical grammars"

0095-2338/84/1624-0066$01.50/0

described in part 3 of this series;[5] it includes facilities for the indication of fixed and variable substitution positions, multiplicity of occurrence of parts of the structure, and the Boolean relationships between parts of the structure. The recently proposed extended block-cutpoint tree (EBCT) representation for generic structures[6] is similar in many respects, although specific applications have not been described. The ECTR is used for the automatic generation of descriptive fragments for a generic structure screening system[7] and in the development of other search and screening algorithms for searching files of generic structures.[8]

As was stated in part 1 of this series,[9] generic structures expressed in GENSAL are converted automatically into the corresponding ECTR. This operation can be compared to that of compilation for a computer programming language[10] in which the *source code* (intelligible to the programmer) is converted to the *object code* (actually used by the computer for the execution of the program). This paper describes the operation of such a program, which in view of the interactive nature of GENSAL input is called the GENSAL Interpreter Program.

FUNCTION OF THE INTERPRETER

The Interpreter is written in the programming language Pascal and is currently implemented on the Sheffield University PR1ME 750 minicomputer. It performs automatic syntactic and semantic analysis of GENSAL input, reports errors it encounters, and generates the ECTR corresponding to the GENSAL. At present, only a subset of the GENSAL language is fully implemented, since the format for the representation of GENSAL's special conditions and restrictions in the ECTR has yet to be adequately defined; however, the Interpreter is upward compatible with the full language, with the result that generic structures encoded in GENSAL will not need to be recorded to take account of future work on the Interpreter.

The overall structure of the Interpreter is similar to that of a programming language compiler. The formal grammar of GENSAL has been designed to make it a member of the class of context-free grammars known as the $LL(k)$ grammars[10] and the syntax analysis is based on the method of top-down recursive descent parsing popularized by Knuth.[11] "Symbol Tables" are maintained for the variables (*substituents* and *multipliers*) used, and the ECTR is generated concurrently with the analysis.

However, in view of the flexibility of GENSAL that allows it to express generic structural information in a natural way, similar to that used in patent specifications and abstracts, some of the semantic checking can be rather complex. For example, in order to ensure compatibility of bond types, the consistency of occurrences of variables, and the validity of substitution position sets, etc., it is necessary to have constant access to information supplied earlier in the GENSAL and already incorporated into the ECTR. This makes certain aspects of the Interpreter rather more complicated than compilers for modern structured programming languages such as Pascal, where there is considerable redundancy of information (declaration before use of types, variables, etc.). GENSAL's endeavor to remain as close as possible to the language of patent specifications and abstracts means that GENSAL structure descriptions contain little, if any, redundant information, with a consequent increase in the complexity of the "housekeeping" needed to detect inconsistencies and—to an extent—a lack of security. The trade-off between security and flexibility in programming language design has been discussed by Wirth.[12]

The Interpreter processes GENSAL input line by line, as the structure description is typed at a terminal. Apart from a maximum line length of 86 characters and a requirement that single words and symbols may not be split over a line boundary, the input is completely free format. Many of the errors that are detected are signaled for immediate correction, after which processing can continue. Whilst it has been designed for interactive input of GENSAL, the Interpreter will also process GENSAL structure descriptions previously stored, though in this case processing ceases as soon as the first error is encountered. It is also possible, during interactive input, to stop processing and exit from the Interpreter (for example, if it is found that a mistake has been made) by entering a blank line; the mistake can then be corrected with the GENSAL Editor module, described below.

The bulk of the information required for the ECTR is found in the *substituent values* in the definitions of *substituents*. In general, each *substituent value* in the GENSAL corresponds to a single Partial Structure in the ECTR. The next three sections consider the ways in which *substituent values* given respectively as structure diagrams, as nomenclatural terms, and as linear formulae are incorporated into the ECTR; the initial structure diagram for the invariant part of the generic structure is treated in the same way as any other structure diagrams.

STRUCTURE DIAGRAMS

Structure diagrams form an integral part of GENSAL generic structure descriptions, but the Interpreter has been designed to be independent of any particular system for their input and display. Thus, the definition of the **structure diagram** in GENSAL is implementation dependent, whereas most other aspects of GENSAL are implementation independent, being fully defined in the syntax of the language. Any suitably modified or specially written specific chemical structure graphics system could be used with the Interpreter, provided that it conforms to certain minimum requirements and has a suitable software routine to interconvert its own connection table format with that used in the ECTR. These requirements may be summarized as follows.

The system must permit the input and validation of the conventional two-dimensional structure diagrams commonly used by chemists, which may include atoms of any element, the bonds that connect them, and also GENSAL's structural variables, called *substituents*, which may have names in the range R1–R63. The *substituents* in any structure diagram may be those introduced earlier in the structure description or may be new; they may be unconnected (for example an ion) or may have one or two connections. *Substituents* with more than two connections are not permitted, and each occurrence of the same *substituent* must always have the same number of connections.

It must be possible to indicate in a structure diagram that a particular atom or *substituent* node is "connected back" to a previously defined part of the structure. The bond making this connection is called an "apical" bond. It must also be possible for a *substituent* node to be shown to have a variable-position connection to one of a number of nodes in the diagram, with the option of specifying the ring system to which the connection is made. (This corresponds to the convention of a bond drawn into the middle of a ring.)

It must be possible to apply a GENSAL *multiplier* with a name in the range M1–M63 to a *substituent* node in a diagram to indicate that it has a multiple (constant or variable) occurrence. In the case of a *multiplier* applied to a singly connected *substituent*, it indicates that the *substituent* occurs a number of times in the positions available for it; a *multiplier* applied to a doubly connected *substituent* indicates a chain of repetition of the *substituent*.

The structure diagram input and display system must allow the specification of electronic charges on the atoms and sub-

stituents and must check for illegal valencies etc., as well as calculating the number of hydrogens attached to each atom (this number indicates whether or not the atom is available for potential further substitution).

The only bond types it is necessary for the user to distinguish are *single*, *double*, *triple*, and *any*. It is intended that the distinction between ring and chain bonds, and the perception of aromaticity and tautomerism, will be automatic, partly because the ring/chain environment of bonds, and the possibilities for aromaticity and tautomer formation, may depend on the alternative values for *substituents* being considered (e.g., where two *substituents* may have independent chain values or may combine to form a ring). Some preliminary work on ring detection (which has been extensively studied in connection with computer-assisted synthesis analysis) is currently in progress.[13]

In the present version of the Interpreter, both input and display of structure diagrams are handled by a modified version of the program developed by Feldmann and others,[14] which is used in a variety of systems including the CSSR and NIH/EPA substructure search systems. The modifications made to it allow it to conform to the requirements outlined above, to carry out the necessary validation and calculation of hydrogen counts etc., and to pass a reformatted connection table back to the Interpreter. It has the advantage that it uses only line-printer characters for display of structure diagrams and does not therefore require any special graphics hardware; however, the quality of the diagrams is often poor.

The program allows the building of a structure diagram by the use of simple commands for the creation of rings and chains, the specification of atom and bond types, etc.; it is possible to draw the current structure diagram at any stage. Even for a relatively simple structure diagram, this method of construction can be an extremely time-consuming process, and partly for this reason, an operational system would certainly require a much more sophisticated software module for structure diagram input.

As an enhancement to the system, the DARING program[15,16] for conversion of Wiswesser line notation (WLN) to Feldmann-format connection table has been incorporated, and this allows much more rapid input, with WLN, of the major part of a structure diagram. Of course, as WLN is not able to show such features as *substituents* and *multipliers*, these have to be added subsequently by the normal Feldmann commands.

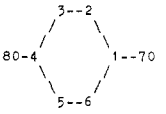
In GENSAL, structure diagrams are always preceded by the delimiter "SD", which signals to the Interpreter to switch control to the structure diagram input system, which is a separate program module. Thus, replacing the current Feldmann system would require little or no change to the Interpreter itself. Figure 1 illustrates the input of a structure diagram in a GENSAL structure description.

NOMENCLATURAL TERMS

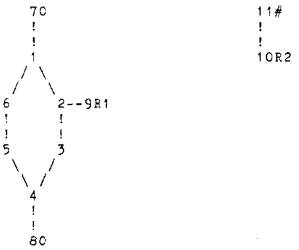
The nomenclatural terms used in GENSAL can be divided into four classes: those representing a simple fully defined specific radical (e.g., "methyl", "cyclohexyl"), those that represent a homologous series or class of radicals (e.g., "alkyl", "cycloalkyl"), those that can be regarded as compounded from either or both of the first two classes (e.g., "chloroalkyl", "halophenyl"), and those to which no structural identity can be readily assigned (e.g., "electron-withdrawing group").

In the case of the last class, it is by their very nature not possible to incorporate any structural information into the ECTR, and so they are represented as simple text strings in "Other" Partial Structures. As an approximation, for searching purposes in an operational system, it might be feasible to include some structural information for certain of these terms by, for example, holding a dictionary of common

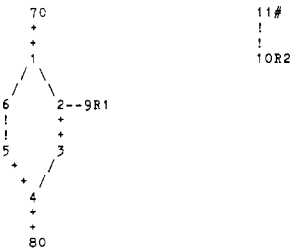
```
  2  GENSAL: SD
SD:
RING 6
SD:
ABRAN 1 1   4 1
SD:
SATOM 7 8
ATOM TYPE =
O
SD:
DRAW
```




```
SD:
ABRAN 2 1
SD:
SATOM 9
ATOM TYPE =
R1
SD:
VARPOS 1 2
SUBSTITUENT IDENTIFIER:
R2
SD:
DRAW
```




```
Positions available for variable-position label 11:  1  2  3  4  5  6
SD:
SBOND 1 7   4 8
BOND TYPE=
D
SD:
SBOND 2 3   4 5
BOND TYPE=
D
SD:
END
SD ERROR: Illegal valency on atom at node 4.
Structure Diagram rejected.
SD:
DRAW
```




```
Positions available for variable-position label 11:  1  2  3  4  5  6
SD:
SBOND 4 5
BOND TYPE=
S
SD:
SBOND 5 6
BOND TYPE=
D
SD:
END
  3  GENSAL:
```

Figure 1. Input of a structure diagram with the GENSAL Interpreter using the Feldmann program[14] as modified. The user's input is shown underlined. A six-membered ring is created, with various attachments, which are specified as "O" and "R1". A variable-position *substituent* is added by the special command VARPOS, which specifies the ring to which the attachment is to be made. The variable-position attachment is indicated in the diagram by a dummy node (node 11), which is shown as "#", and initially the program assumes that all the atoms in the ring specified are potential points of attachment. When the validation routine checks valencies however, the list of available positions may be restricted. The user adds double bonds to the oxygen atoms and within the ring. However, these are incorrectly positioned, and when the validation routine is called (END command), the error is detected, and the user is automatically returned to the structure diagram input mode to correct it. (NB: The Feldmann program shows double bonds by means of "+" signs.)

specific electron-withdrawing groups, but there are no plans for doing this at present.

An operational system for storage and retrieval of generic structures in patents will undoubtedly require some sort of nomenclature translation software providing algorithmic conversion of nomenclature to structural representations for the first three classes. A considerable amount of work has already been done in this area,[17-21] and it is hoped that some of this might be adapted to the needs of the GENSAL Interpreter. The principal difference in the approach required is that whereas most previous work has concentrated on the translation of the full names for complete chemical substances, the Interpreter will require translation primarily of radical names; it will also need provision for the handling of trivial, semisystematic, and obsolete systematic nomenclature and some extension to enable it to handle generic, rather than fully specific, radical nomenclature.

The development of programs for nomenclature translation does not form part of the work being done at Sheffield, and so the present version of the Interpreter uses a simple dictionary of nomenclatural terms. However, the variety and lack of standardization of nomenclatural terms used in patents would be likely to make such a dictionary unmanageably large in an operational system. When a term is encountered in the GENSAL, it is looked up in a dictionary of standard partial structures (SPSDICT). If found, then an appropriate structural record is taken from an associated file (SPSFILE). This record will differ according to the class of term in question; if the term is not found, then it is treated as a "nonstructural" term and incorporated into the ECTR as text, though the user is given the opportunity to enter synonyms, which might then be found in the dictionary.

For simple specific radicals, SPSFILE yields a partial connection table that is incorporated directly into the ECTR. For homologous series terms, SPSFILE yields a default set of parameters to the chemical grammars, which are descriptive of the class of radicals in question. These parameters may be modified by any parameter list that follows the term in the GENSAL (for example, limiting the number of atoms or branch points etc.). The modified set of parameters is then incorporated into the ECTR.

Compound terms may themselves be subdivided into those that represent a single specific radical (e.g., 2-chlorophenyl), those that involve a homologous series term (e.g., chloroalkyl), and those that though composed only of specific radical morphemes, represent a class of radicals (e.g., chlorophenyl, which covers the ortho, meta, and para isomers). All these terms are essentially "right rooted"—that is, it is the right-hand end through which the connection is made, and for this reason, many nomenclature translation algorithms parse from right to left.[21]

A convenient, if somewhat restricted, method of handling compound nomenclatural terms within the framework of the Interpreter is to convert this "right-rooted" tree into a "left-rooted" tree, typical of GENSAL expressions, and this is the approach used at present, SPSFILE yielding a GENSAL expression corresponding to the term. Thus, for example, the SPSFILE entry for "chloroalkyl" is "alkyl SB chloro", and this expression is then processed exactly as if it had been used in the original GENSAL, the terms "alkyl" and "chloro" being looked up in the dictionary as before. In some cases, of course, the subsidiary terms obtained from SPSFILE may themselves be "compound" and subject to expansion into GENSAL expressions; this recursion can continue indefinitely.

Where the compound term represents a single specific radical, a nomenclature translation routine should be able to generate a single partial connection table for incorporation into the ECTR. For terms that represent a class of radicals (whether including a homologous series or not), more than one partial structure will be required in the ECTR, and a nomenclature translation routine could be used simply to automate the conversion (at present carried out with the dictionary) of a right-rooted term to a left-rooted GENSAL expression.

LINEAR FORMULAS

Though linear formulas occur only infrequently in patent specifications themselves,[22] they are widely used in abstracts, and are permitted in GENSAL. At present, the Interpreter treats them exactly as nomenclatural terms, looking them up in SPSDICT and obtaining structural records from SPSFILE. Thus, "C2H5" is regarded simply as a synonym for "ethyl", along with "Et", all three giving access to the same structural record.

Again, some sort of automatic processing is clearly called for, and it is hoped shortly to implement with the Interpreter the program recently described by Figueras.[23] Since the syntax of GENSAL does not distinguish between linear formulas and nomenclatural terms (whereas it does distinguish nonstructural expressions by enclosing them in quotes), the distinction will be made by attempting first to parse the term or formula as a linear formula and, if this fails, by treating it as a nomenclatural term. In the current version of the Interpreter, the dictionary look-up procedures for handling nomenclatural terms are essentially a separate program module. Very little change would be needed to the rest of the program in order to incorporate nomenclature translation and/or linear formula analysis modules.

SYNTACTIC AND SEMANTIC ERRORS

When the Interpreter detects an error of syntax (i.e., an expression that does not conform to the formal grammar of GENSAL, as defined in the syntax diagrams),[1] it prints a line of arrows under the offending word or symbol, followed by an appropriate error message. The remainder of the input line is ignored, and the user can continue input on a new line, beginning with a replacement for the erroneous token.

Semantic errors, where the program detects an error in the meaning of the GENSAL, are more complicated. Some of them, such as an attempt to define a *substituent* that has not yet been introduced, are detected immediately and are handled exactly as for syntax errors. Others may not be detected until considerable further processing has taken place and part of the ECTR has been generated. An example of this type of error would be an incompatibility of bond types, where a *substituent* might be introduced by appearing in a structure diagram with a double bond yet defined as "chloro", which carries only a single bond. In this case, as it would be difficult for the program to recover from the error, the user is ejected from the Interpreter and must either begin again or use the GENSAL Editor module (described below) to edit the structure description. Enhancements to the program might improve the error recovery and allow certain of the second type of error (called "failures") to be converted to the first type, thus allowing processing to continue. Figure 2 illustrates part of the input of a GENSAL structure description in which error messages are given by the Interpreter.

A choice has had to be made in deciding exactly which semantic errors to check for and which to leave undetected. This choice has been to a certain extent arbitrary and has been motivated largely by considerations such as ease of programming and the potential consequences of errors in the ECTR. The main items that are checked for are the introduction of *substituents* and *multipliers* before their definitions, the availability for substitution by a bond of the appropriate order of positions specified in a *position set*, the validity of values

```
  3  GENSAL: R1 = phenyl SB [1] ( F / Cl );
                            ^
**** ERROR: Substitution is not possible in position 1.
Remainder of input line ignored
  4  GENSAL:                  2] ( F / Cl )
  5  GENSAL: R2 = R3 SB Cl;
                  ^
**** ERROR: Unexpected symbol.
Remainder of input line ignored
  6  GENSAL: ;
  7  GENSAL: R2 = R3 SB Cl;
  8  GENSAL: R3 = F;
**** FAILURE: No available positions for further substitution on R3.
Edit existing GENSAL or start again!
```

Figure 2. Part of the input of a GENSAL structure description illustrating some of the error messages given by the Interpreter. In line 3, the user has attempted to specify further substitution on position 1 of a phenyl ring; the program realizes that position 1 is already required for the bond back to the constant part of the structure, where R1 was introduced. The error is corrected in line 4, but the user now omits the semicolon needed to separate this statement from the next. The computer gives an error message for the missing semicolon, which the user inserts on line 6. In line 7, R2 is defined in terms of (as yet undefined) R3 and is shown to be further substituted by Cl. When R3 is defined in line 8, the program realizes that it will be unable to accommodate the Cl specified in line 7 on the F, which is given as the value for R3. The program is unable to recover easily from this error, and so the user is ejected from the Interpreter with a "FAILURE" message.

given in a parameter list (e.g., attempting to specify unsaturations for "alkyl"), the compatibility of number of connections and bond orders in the introduction and definition of *substituents*, and the eventual definition of all *substituents* and *multipliers* introduced. The main item not checked for is the availability of sufficient positions for multiple substitutions, with the result that the following expressions would not be identified as erroneous:

R1 = phenyl SB [2] ⟨5⟩ methyl;

(it is clearly not possible to accommodate five methyl groups at position 2 on the phenyl ring, as has been specified)

R2 = phenyl SB [2] (F & Cl & Br & I);

(it is clearly not possible to incorporate all the specified halogens on position 2).

CONDITIONS AND RESTRICTIONS

Since the representation in the ECTR of the special conditions and restrictions often found in generic structure descriptions in patents has not yet been adequately defined, the information in the IF and RESTRICT statements of GENSAL is not used by the Interpreter. However, it does carry out syntactical analysis on these statements (though for *simple conditions* this is only rudimentary) and will process normally any *assignment statements* found in the THEN and ELSE parts of IF statements. Furthermore, no account is taken of the *assignment operator* used in *assignment statements*: all such statements are treated as if they used the "=" operator for independent selection of alternatives in the definition.

There are two main consequences of this. In the first place, the Interpreter will accept GENSAL structure descriptions containing nonindependent assignment operators, IF statements, and RESTRICT statements and will detect many types of syntax error in them. In the second place, the ECTR generated will describe a rather wider range of structures than is actually warranted by the GENSAL.

It is intended eventually to extend the definition of the ECTR and the capabilities of the Interpreter to enable the full GENSAL language to be implemented. When this has been done, structures in GENSAL that have already been stored on the computer can be reprocessed by the new version of the Interpreter, generating the enhanced ECTR in which all the special conditions and restrictions are represented.

PROGRAMS ASSOCIATED WITH THE INTERPRETER

The Interpreter program is implemented as a single module of a prototype generic structure storage and retrieval system, which contains a number of other modules. Many of these are concerned with housekeeping operations for handling external files, but the other main modules include a fragment generator,[7] which generates descriptive fragment screens from the ECTR produced by the Interpreter, and an Editor module,[24] which allows interactive editing of a GENSAL structure description.

This latter is used for the correction of GENSAL notations that have been rejected by the Interpreter (failures) or where the user has deliberately exited from it by entering a blank line. It is a simple line editor, which also interfaces with the structure diagram input and display module for the editing of structure diagrams. When an editing session is complete, the corrected GENSAL can be reprocessed by the Interpreter, which switches back to interactive mode to allow completion of the structure description when the point at which the original error was discovered is reached. Modules for searching files of generic structures are currently under development; these will be the subject of future papers in this series.

ACKNOWLEDGMENT

This work has been carried out with financial support from the British Library Research and Development Department. We thank Drs. J. Figueras (formerly Kodak), T. Devon (Pfizer, U.K.), and M. Elder (SERC) and Fraser Williams (Scientific Systems) Ltd. for provision of software. Thanks are also due to Sandra Hill, Janet Kinsella, Lynne Carruthers, and Witold Szczyglowski, students in this department who have contributed to the work described here, to the staff of Sheffield University Computing Services (especially Richard Gilbert) for programming support, and to Drs. P. Willett and A. von Scholley and the attendees at the GENSAL training course held in April 1983 for helpful comments and discussions.

REFERENCES AND NOTES

(1) Barnard, J. M.; Lynch, M. F.; Welford, S. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 2. GENSAL, a Formal Language for the Description of Generic Chemical Structures". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 151–161.
(2) Hill, S. J.; Barnard, J. M.; Welford, S. M.; Lynch, M. F. "GENSAL Reference Manual"; Department of Information Studies, University of Sheffield: Sheffield, U.K., 1983.
(3) Howe, W. J.; Hagadone, T. R. "Molecular Substructure Searching: Computer Graphics and Query Entry Methodology". *J. Chem. Inf. Comput. Sci.* **1982**, *22*, 8–15.
(4) Barnard, J. M.; Lynch, M. F.; Welford, S. M. "Computer Storage and Retrieval of Generic Structures in Chemical Patents. 4. An Extended Connection Table Representation for Generic Structures". *J. Chem. Inf. Comput. Sci.* **1982**, *22*, 160–164.
(5) Welford, S. M.; Lynch, M. F.; Barnard, J. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 3. Chemical Grammars and Their Role in the Manipulation of Chemical Structures". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 161–168.
(6) Nakayama, T.; Fujiwara, Y. "Computer Representation of Generic Chemical Structures by an Extended Block-Cutpoint Tree". *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 80–87.
(7) Welford, S. M.; Lynch, M. F.; Barnard, J. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 5. Algorithmic Generation of Fragment Descriptors for Generic Structure Screening". *J. Chem. Inf. Comput. Sci.*, in press.
(8) von Scholley, A. "A Relaxation Algorithm for Generic Chemical Structure Screening". Submitted for publication in *J. Chem. Inf. Comput. Sci.*

(9) Lynch, M. F.; Barnard, J. M.; Welford, S. M. "Computer Storage and Retrieval of Generic Chemical Structures in Patents. 1. Introduction and General Stragegy". *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 148-150.
(10) Aho, A. V.; Ullman, J. D. "The Theory of Parsing Translating and Compiling"; Prentice-Hall: Englewood Cliffs, N.J., 1972; 2 Vols.
(11) Knuth, D. E. "Top-Down Syntax Analysis". *Acta Inf.* **1971**, *1*, 79-110.
(12) Wirth, N. "An Assessment of the Programming Language Pascal". *IEEE Trans. Software Eng.* **1975**, *SE*-1, 192-198.
(13) Carruthers, L. M.Sc. Dissertation, University of Sheffield, 1983.
(14) Feldmann, R. J.; Milne, G. W. A.; Heller, S. R.; Fein, A.; Miller, J. A.; Koch, B. "An Interactive Substructure Search System". *J. Chem. Inf. Comput. Sci.* **1977**, *17*, 157-163.
(15) Elder, M. "The Conversion from Wiswesser Line Notation to CIS Connection Table", Proceedings of CNA (UK) Seminar on Interconversion, Loughborough, March 1982; Chemical Structure Association: London, 1983.
(16) The DARING program is marketed by Fraser Williams (Scientific Systems) Ltd., Glendower House, Poynton, Cheshire, U.K.
(17) Garfield, E. "An Algorithm for Translating Chemical Names to Molecular Formulas". *J. Chem. Doc.* **1962**, *2*, 177-179.
(18) Dyson, G. M. "A Cluster of Algorithms Relating the Nomenclature of Organic Compounds to Their Structure Matrices and Ciphers". *Inf. Storage Retr.* **1964**, *2*, 159-199.
(19) Vander Stouw, G. G.; Elliott, P. M.; Isenberg, A. C. "Automated Conversion of Chemical Substance Names to Atom-Bond Connection Tables". *J. Chem. Doc.* **1974**, *14*, 185-193.
(20) Vander Stouw, G. G. "Computer Programs for Editing and Validation of Chemical Names". *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 232-236.
(21) Rayner, J. D. Ph.D. Thesis, University of Hull, 1983.
(22) Szczyglowski, W. L. M.Sc. Dissertation, University of Sheffield, 1983.
(23) Figueras, J. "Chemical Symbol String Parser". *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 48-52.
(24) Kinsella, J. E. M.A. Dissertation, University of Sheffield, 1982.

Artificial Intelligence in Organic Synthesis. SST: Starting Material Selection Strategies. An Application of Superstructure Search†

W. TODD WIPKE* and DAVID ROGERS
Department of Chemistry, University of California, Santa Cruz, California 95064

Received August 8, 1983

A program for the interactive selection of potential starting materials given a desired target molecule is described. The program uses hierarchical search to rapidly select candidates from a large starting material library and contains a function to evaluate the appropriateness of the functionality of the starting material. The user can specify restrictions on features such as the number of atoms in the starting material, price limitations, chirality, and whether to use superstructure or substructure searching. Several examples of the results of the program are presented.

INTRODUCTION

The importance of selecting good starting materials for an organic synthesis has been known for a long time; in fact, the selection of a starting material can be the major discovery in a synthesis, with the process of converting the starting material to the target minor by comparison. The previous work in computer-assisted design of organic synthesis has focused on reaction-driven analysis backward from the target molecule in an open-ended search.[1-5] Selection of starting materials has been ignored except as an end point for the backward analysis. For example, the SYNCHEM program developed by Gelernter[1] used a catalog of available starting materials to recognize when a precursor on a synthetic sequence was available and thus a termination point.

Empirically, one observes that chemists do not always work systematically backward but sometimes make an "intuitive leap" to a specific starting material from a target without consideration of reactions needed for interconversion. This intuitive leap probably involves a *Gestalt* pattern recognition based on the chemist's knowledge of available starting materials and similarity between the starting material structure and the target structure. By selection of one or more starting materials, the search for synthetic pathways becomes focused on those connecting the starting materials to the target. The starting material acts as a "planning island",[6] an anchor point in the plan to help reduce the planning space and allow forward planning from the material as well as backward planning from the target. Our interest in synthetic strategic planning led us to pursue these starting material oriented strategies as a planning aid.

Our objective was to develop algorithmic techniques to assist the chemist in selecting potential starting materials by "heuristic leap" directly from the target.[7] We chose to treat this heuristic leap as a pure strategy, independent of reaction knowledge.[8] We wished to avoid the costly tedium of reasoning backward from the target with reactions just to hypothesize a starting material. We wished to avoid the biases and limitations that reasoning in reaction space imposes. For example, many of the more creative suggestions for starting materials cannot be discovered by reasoning with reactions either because the reaction library is incomplete or because the crucial reaction in the synthetic sequence has not yet been discovered. Thus, we seek algorithms for this heuristic leap that are free from reaction knowledge.

Although there has been no systematic study of how a chemist makes an intuitive jump to a starting material, clearly the chemist has knowledge of known starting materials and associated information such as approximate price, ease and safety of handling, and chirality. The chemist also has knowledge of synthetic reactions, which is not applied directly but rather used in abstract form to suggest possible substructures in the target and candidate starting material that would be either useful or unwanted. Finally, the chemist has knowledge of other successful syntheses and the starting materials used in them. The approach we are about to describe will use only knowledge of available starting materials.

Because there are so many starting materials available, the computer with its large, fast memory and reliable recall seems like a natural assistant to the chemist in picking starting materials. Assisting in selection of starting materials can be valuable to the chemist even if the chemist fills in the synthesis plan manually, and it can certainly be valuable to computer-assisted planning.

RESULTS AND DISCUSSION

The *Gestalt* pattern recognition that chemists perform is graphically oriented, with probable dominance given to the

† Presented at the 182nd National Meeting of the American Chemical Society, New York, Aug 1981.

0095-2338/84/1624-0071$01.50/0