STANFORD ARTIFICIAL INTELLIGENCE LABORATORY
MEMO AIM-205
STAN-CS-73-370

A HEURISTIC PROGRAM TO DISCOVER SYNTHESES FOR COMPLEX ORGANIC MOLECULES

BY

N. S. SRIDHARAN
HERBERT GELERNTER
A. J. HART
W. F. FOWLER
H. J. SHUE

JUNE 1973

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Stanford University Computer Science Department Stanford, California 94305	Unclassified 2b. GROUP

3. REPORT TITLE

A HEURISTIC PROGRAM TO DISCOVER SYNTHESES FOR COMPLEX ORGANIC MOLECULES.

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
technical report, June 1973

5. AUTHOR(S) (First name, middle initial, last name)

N.S. Sridharan, Herbert Gelernter, A.J. Hart, W.F. Fowler and H.J. Shue

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1973	~~30~~ 32	9

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
SD-183 b. PROJECT NO.	STAN-CS-73-370 AIM-205
c. d.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

Releasable without limitations on dissemination.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

Organic Chemical Synthesis is found to be a suitable problem for developing machine intelligence. A previous paper described the objective and global characteristics of the project. The present article aims to describe the program organization as a heuristic search, the design of the Problem Solving Tree and the search procedures in considerable detail. Examples of syntheses discovered and the problem solving tree developed are given. The programs are written mostly in PL1(F) applicable to an IBM 360/67 and the timings (batch mode) indicate that we have fast and efficient practical systems.

STANFORD ARTIFICIAL INTELLIGENCE LABORATORY
MEMO AIM-205

JUNE 1973

COMPUTER SCIENCE DEPARTMENT
REPORT STAN-CS-73-370

A HEURISTIC PROGRAM TO DISCOVER SYNTHESES FOR COMPLEX ORGANIC MOLECULES

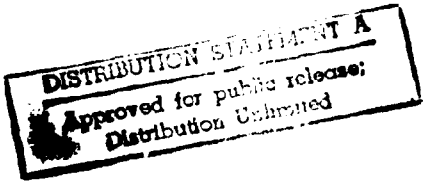
by

N.S. Sridharan
Dept. of Computer Science, Stanford University

Herbert Gelernter, A.J. Hart
Dept. of Computer Science, S.U.N.Y. at Stony Brook

W.F. Fowler, H.J. Shue
Dept. of Chemistry, S.U.N.Y. at Stony Brook

Abstract: Organic Chemical Synthesis is found to be a suitable problem for developing machine intelligence. A previous paper described the objective and global characteristics of the project. The present article aims to describe the program organization as a heuristic search, the design of the Problem Solving Tree and the search procedures in considerable detail. Examples of syntheses discovered and the problem solving tree developed are given. The programs are written mostly in PL1(F) applicable to an IBM 360/67 and the timings (batch mode) indicate that we have fast and efficient practical systems.



STRUCTURE OF THE PAPER

I. Introduction
   1.1 Problem Definition
   1.2 Specification
   1.3 Characterization

II. Overview
   2.1 The Substructure System
      2.1.1 Structure Representation
      2.1.2 Chemical Vocabulary
      2.1.3 Question Answering System
      2.1.4 Reaction Library
      2.1.5 Reaction Module
      2.1.6 Problem Solving Tree (PSTREE)
      2.1.7 List of Available Compounds
   2.2 An Overview of the Heuristic Program
      2.2.1 An Overview of the Solution Generation Process

III. Executive Program and Problem Solving Heuristics
   3.1 The Organization of the Executive Program
      3.1.1 Initialization
      3.1.2 Heuristic Search
      3.1.3 Results of Heuristic Search
      3.1.4 Details of Subgoal Generation
   3.2 Search Procedure
      3.2.1 Merit of Compound
      3.2.2 Merit of Subgoal
      3.2.3 Merit of Sequence
      3.2.4 Goal Selection
   3.3 Pruning the PROBLEM SOLVING TREE
      3.3.1 Eliminating Invalid Subgoals
      3.3.2 Invalidation by Cost Considerations
      3.3.3 Elimination of Redundant Subgoals
      3.3.4 Elimination of Unpromising Subgoals

IV. Program Performance and Critique

A HEURISTIC PROGRAM TO DISCOVER SYNTHESES FOR COMPLEX ORGANIC MOLECULES

I. INTRODUCTION

Organic chemical synthesis is found to be a suitable problem for developing machine intelligence where the resulting system promises to be of genuine utility. The challenge of the work of program design arises out of the complexity of the task of synthesis, the large base of scientific knowledge and vocabulary required, and the abstruse rules of reasoning employed by the "experts" in the field. A previous paper (9) described the problem, giving a general approach based on the techniques of heuristic search for solutions and some interesting examples of the results produced by the programs.

We wish to describe in the present communication, details of the program design.

1.1 PROBLEM DEFINITION

Synthesis in practice involves i) the choice of molecule to be synthesized; ii) the formulation and specification of a plan for synthesis (involving a valid reaction pathway leading from commercial or readily available compounds to the target compounds with consideration of feasibility with regard to the purpose of synthesis); iii) the selection of specific individual steps of reaction and their temporal ordering for execution; iv) the experimental execution of the synthesis and v) the redesign of syntheses, if necessary, depending upon the experimental results.

In contrast to the physical synthesis of the molecule, the activity ii) above can be termed the 'formal synthesis'. This development of the specification of syntheses involves no laboratory technique and is carried out mainly on paper and in the minds of chemists (and now within a computer's memory!). Formal synthesis is our only concern in this project.

1.2 SPECIFICATION

The programs and results described herein represent the completion of the first phase of an ambitious research goal to design a system which takes as input some representation (now the Wiswesser linear name - see Ref. 4) of the target compound together with a list of conditions and constraints that must govern the solution of the problem. The program (see Fig. 1) has at its disposal a list of compounds, the "shelf library", that can be assumed available with some indication of cost and availability (available on-hand or in industrial, pilot or laboratory quantities) and so on. The programs use a reaction library containing generalized procedures for the synthesis of functional groups and structural features singly or in combination. The output is to be a set of proposed synthesis procedures (noting the absence of a theoretical "best" or "only" solution). Each proposed synthesis is to be a valid reaction path- way from the available compounds annotated with estimated yields for each step of the procedure together with by-product predictions and target molecule separation procedures (with estimated separation efficiency) for each step. The syntheses are to be rated ad hoc on the estimated degree to which each procedure satisfies the conditions of the problem.

1.3 CHARACTERIZATION

The reader can readily realize that much of our effort was directed into building up a chemical structure representation and a large vocabulary of chemical terms defined upon the structure

representation. Perhaps the more fascinating challenging facet of the problem was that of constructing a rational basis for the reasoning process involved in designing syntheses. Syntheses are not brought forth in a flash of understanding but are developed one step at a time. The development of these steps are highly interdependent and occur with general techniques which can be learned. We have watched and interacted with a chemist, Dr. Frank W. Fowler, while he developed syntheses. We attempted to isolate useful components* in Dr. Fowler's problem-solving activity. Some of his techniques have been incorporated as heuristics within the synthesis search algorithm.

*see chapter 2 of thesis for a discussion of this.

II. OVERVIEW

We have approached the design of our system at two levels: concepts and implementation. The conceptual divisions are the heuristic program, overall methodology, the reaction library and its use, the information store and its utilization, and the list of available compounds. The implementation of these on a substructure made of a system for information representation, storage, retrieval and manipulation, will be reviewed here first. Readers wishing to skim, may safely go on to section 2.2.

2.1 THE SUBSTRUCTURE SYSTEM

The principal aim in the design of this system is to facilitate communication with the computer regarding chemistry. Based upon a convenient structure representation we have built up a vocabulary of chemical terms and concepts about chemical structures. There is a separately designed representation of reaction schema and manipulative transformations. The interface between the library of reactions and chemical structures is a question-answering package. Another important part of the implementation is the information structure used by the heuristic program to record partial discovery sequences. The programming support system for information representation and manipulation is the subject of a Doctoral thesis in preparation by A.J. Hart (Ref. 2).

2.1.1 STRUCTURE REPRESENTATION

A representation is designed for basic chemical entities like atoms, ions, free radicals and compounds. These notions are inseparable from their chemical structure. The structure representation therefore provides for the actual topological properties and the stereochemical details of the compound. The structure is made amenable to inspection, modification and manipulation. A linear and canonical name is indispensable for indexed lists of compounds and for compounds to be compared for equivalence. The linear canonical notation developed by Wiswesser (Ref. 4) is used, with our own modifications introduced for expedience. One noteworthy feature of the Wiswesser notation is that it preserves full structural information (although this information is not handy for manipulation). Therefore algorithmic interconversion of the linear notation and the topological notation is feasible and has been programmed to translate a large class of notation.

Each topological description is accompanied by an attribute table of interesting atoms, bonds, groups and rings, which serves in two major capacities. First, it forms the basis for answering several questions directly and for building answers to complex questions. Second, it provides handles to specific groups and atoms of the topological description and thus helps the reaction selection routines and structural transformation routines.

2.1.2 CHEMICAL VOCABULARY

Starting with basic ideas such as the number of unshared electrons in an atom, the number of hydrogens connected to an atom and the degree of linkage, we have developed a vocabulary of nearly 150 terms. Examples of some of the higher level terms are , unsaturation, acid sensitive, degree of sensitivity, strong electron-releasing groups, base sensitive groups.

2.1.3 QUESTION-ANSWERING PACKAGE

The heuristic program requires answers to many questions about a structure before it decides on the class of reactions to attempt, and requires many more answers as individual reactions are tested for validity and side-effects. These questions are answered from the attribute table, the Wiswesser name, the structure description or from an extended attribute vector derived for the structure. The question-answer package is concerned chiefly with 'yes-no' questions.

2.1.4 REACTION LIBRARY

The reaction library is a collection of reaction schema grouped into several sets. All schema in a set synthesize the same structural feature but may use different reactions, reactants and reaction conditions. A reaction schema consists of a rule for structure transformation and several sets of tests on the structure of a compound. The reaction conditions to be used and preference ratings have initial values, and modifications are indicated based on test results.

2.1.5 REACTION MODULE

The reaction module is a set of conventions and programs that utilize the reaction library. There are routines to read in the transformation, parameters of the reaction and the associated tests. The tests are conducted through the interface with the question-answering module. The changes occurring at the reactive site are carried out in two steps. In the first step, the structure of the given compound is examined around the reactive site are identified. In the second step, the substitution fragment structures are recombined around one or more other reactive groups thus generating the transformed structures.

2.1.6 PROBLEM SOLVING TREE

The heuristic program uses a tree-like structure to store partial results and alternative courses of action. The data items in the tree can be names of compounds, functional groups, schema references and status and cross-reference information. The data structure for the problem solving tree is discussed fully in a forthcoming thesis (Ref. 2). In the terminology of heuristic search, the problem solving tree for synthesis is an AND/OR tree.

It is convenient to imagine the tree to be made of a set of compound names, one tied to another by relational pointers. However, there are five layers of structure in the tree (Fig. 2). Below a compound are a list of nodes indicating the chapters of the reaction library that were used. Each chapter is followed by a list of references to functional features of the compound that were considered relevant to the chapter. Each reaction schema applied to this feature is listed below it. The reaction schema leads to a set of subgoals each of which, in turn, is a set of compounds needed for the reaction mentioned in the schema. To allow efficient maintenance and processing of the tree, each compound occurring in the tree is entered in a Symbol Table. The tree is utilized chiefly to store relationships between compounds and to maintain problem solving

information. The program can find from the tree and the symbol table answers to such questions as:

How difficult does this compound seem to be?
How good is this sequence of steps?
Is this compound solved?
Has this compound been attempted to be synthesized?
If so, what was the result of the attempt?
What is the best sequence of steps developed till now?
What are all the instances of this compound occurring in the tree? and so on.

We have briefly described the various data structures and routines that form the tools of the heuristic synthesis program. Evidently, this same set of tools could be used to build not only a synthesis program but also an analysis program or a reaction simulation program.

2.1.7 LIST OF AVAILABLE COMPOUNDS

The ALDRICH chemical catalog (Ref. 3) of 7547 Wiswesser names of compounds was used to generate the shelf library. The initial version used a reduced set of 3653 Aldrich listed compounds augmented with names of commonly available reagents and certain gases. The list was set up for direct look-up by compound name.

2.2 AN OVERVIEW OF THE HEURISTIC PROGRAM

Before a program to automatically specify synthesis procedures could be written, the elements of synthesis had to be isolated and defined. Their mutual interaction were considered in a general sense.

The input to the program, consisting of the name of the compound to be synthesized, constraints on the result, and the list of available compounds, sets the context of the synthesis. The information store can supply reactions and can answer questions of chemistry. The program therefore is made up of a definition of synthesis together with a repertoire of techniques for syntheses.

The definition of a synthesis is implicit in the program and is as follows:

i) The starting materials are to be available compounds and the sequence of reactions should lead up to the goal compound
ii) The intermediate compounds in the syntheses may be known or unknown compounds, but each must be a valid structure and stable under the conditions of the synthesis
iii) It is important to develop and to identify syntheses which have a good yield, where the reactions will take place with high probability and with ease. Extreme demands are not to be placed on equipment and starting materials. Sequences of reactions are to be simple, involving specific transformations with minimal side-effects and correctional steps.

2.2.1 OVERVIEW OF THE SOLUTION GENERATION PROCESS

The process of finding a synthesis for a compound is to some extent analogous to the process of proving a theorem in geometry or to that of establishing proofs in formal logic. The parallels we can draw between our program and the Geometry Theorem Proving Program (Ref. 5) are:

Theorem	Target compound
Rules of deduction	Generalized reactions
Proof	A synthesis sequence
Axioms and premisses	List of available compounds

The Geometry Theorem Prover provided some directions and we patterned the present program to follow it. However, significant differences arise owing not only to the differences in the definitions

of the problems but also to the dissimilarity in the nature of the problems themselves. We will stress the differences in the problem solving procedures.

Potential reactions to generate a compound can be devised by examining the structural composition of the molecule. This forms the basis for working backward. The major method of the program is an analytic search procedure, whereby the last step of the synthesis is the first step to be discovered, the next to the last step is the second step to be discovered and so on. Thus the discovery sequence is the reverse of the synthesis sequence.

The working backward procedure has five major steps. Initially the generating goal is the target compound.

1. Analysis of the generating goal
2. Selection of reaction schema and its validation
3. Subgoal generation
4. Entering information in the problem solving tree
5. Selection of next generating goal from the problem solving tree

Analysis of the generating goal: the structural description is developed from its Wiswesser name and its attribute table is constructed. The latter task incorporates the identification of functional groups and ring structures of importance. A choice of functional group to synthesize is made by consulting the attribute table. Attention is given to diverse functionality and suitable choice is made among multiple occurrences.

Schema is chosen from among the schema that synthesize that given type of functional group. These are reactions that can lead to the generating goal in one conceptual step, although several reactions may be involved. The set of tests to be performed, some on the generating goal and some on the subgoals embody many of the chemical heuristics that guide the program. Detailed consideration is given to reactive sites in the molecule, competing reactions and side effects. If any serious conflict is detected, the associated reaction schema (or the subgoal when the test is made on a subgoal compound) is rejected. Based on the results of tests the ad hoc merit rating may be modified (raised if a conjugated activating* group is present and lowered if steric hindrance is detected) the reaction procedures may be modified (a different reagent might be specified in the presence of groups sensitive to the usual reagent) or protection reactions initiated for sensitive groups.

*see any standard textbook on organic chemistry (Ref. 6) or glossary in thesis.

Subgoal generation: The transformation rule contained in the schema is interpreted in the generating goal and the reactants needed for the reaction are defined following the two steps that were mentioned before. The Wiswesser names are computed for the reactants.

Entering information into the problem solving tree: The shelf library is consulted to identify reactants that are available. An estimate of cost and availability is obtainable at this point. An ad hoc overall merit is computed for each subgoal based on the adjusted reaction merit and the estimated simplicity of reactant compounds (available compounds get the highest simplicity values, compounds that resulted in total failure in an attempt for their synthesis get the lowest values). If any reaction has all its reactants available or synthesized in the course of the program's working, it completes a synthesis. All completed syntheses are collected on a list. If a synthesis-search termination condition has not been signaled, the "best" subgoal is selected for further development and the procedure is recursively continued. A synthesis-search tree is thus generated, the structure of which depends upon the algorithm chosen for determining

the best subgoal for further consideration. This algorithm will be discussed further in this report. A discovery sequence is complete when it leads from the goal compound to a set of available starting materials. The program does not halt after finding just any synthesis. It pursues subgoals till either all prospective syntheses are found to be of significantly lower merit or it consumes one or more available resources.

III. THE EXECUTIVE PROGRAM AND PROBLEM SOLVING HEURISTICS

3.1 THE ORGANIZATION OF THE EXECUTIVE PROGRAM

The simplified flowchart of the heuristic search process embodied in the executive program is shown in Figure 3. The following discourse explains the flowchart using Vitamin A (top of Figure 4) as the example input compound.

3.1.1 INITIALIZATION

The standard reaction library is initialized for use, unless some changes are indicated by input. The provision for designating any set of reaction schema in the library as temporarily unusable and the facility for reordering the priorities assigned to reactions help one to customize the reaction library for any particular run.

The standard catalog of available compounds is also initialized for use.

The name of compound to be synthesized is subsequently read in by the program. A problem solving tree is created with one subgoal at its root. (The tree, for us, grows downward with the root at the top!) The subgoal consists of a single compound -- the goal compound.

3.1.2 HEURISTIC SEARCH

Following initialization, the program enters the main part, repeatedly selecting a goal from the problem solving tree and sprouting subgoals for it.

3.1.2.1 GOAL SELECTION

The selection of a goal from a problem solving tree follows a simple heuristic algorithm and is further explained in section 3.2.4.

The first time, of course, the goal to be selected is the only compound in the tree - the target molecule.

3.1.2.2 SUBGOAL GENERATION

The details of subgoal generation for a chosen goal are given in Figure 5 and are followed in detail with an example in section 3.1.4. We shall presently skip over the details and note that for the goal selected all valid subgoals will be generated using all applicable reaction schema. For Vitamin A, the subgoal generation process enters 43 valid subgoal at the first level directly below Vitamin A, representing as many distinct ways of getting Vitamin A in one reaction step. Out of the 43 valid subgoals, 9 subgoals have two compounds together in each yield and a count of 52 subgoal compounds. Out of these, 3 compounds were found available through the Aldrich Chemical Catalog. The problem solving tree after the subgoal generation would appear as shown in Figure 6. The detailed diagram shows that 8 reaction schema were used and that no subgoal represents a completed synthesis for Vitamin A, after the first level of subgoal generation.

The decisions about the applicability of a reaction schema and the validity of a subgoal in relation to a problem solving tree are not formal but are heuristic. Such decisions involve the utilization of judgemental knowledge culled from chemists and thus is a heuristic procedure. All subgoals judged worth retaining are entered into the problem solving tree. Compounds are checked for availability

in the compound catalog and are marked 'available' when they are. The Aldrich Catalog number is also entered at the same time.

The availability of certain compounds might have completed the syntheses of compounds higher up in the tree.
If all compounds needed for any reaction are either marked 'available' or 'solved' then the subgoal higher up in the tree which linked to the reaction is subsequently marked 'solved'.

If no valid subgoal could be generated for the goal compound then it is marked 'stuck'. For a higher level goal if all subgoals are 'stuck' then the higher level goal is also marked 'stuck'.

3.1.2.3 HEURISTIC EVALUATION OF MERIT

First, all newly created subgoals and compounds are evaluated and assigned a merit rating in the scale of 0-10. Available compounds receive the highest (best) possible rating 10. Multiple occurrences of a compound in the tree are cross referenced so that they receive equal ratings. Compounds not available through the Catalog are evaluated by a function based on their structural characteristics and location and environment in the problem solving tree. The reaction used on each subgoal is assigned a merit as well. The selection of a subgoal is based upon both these values as will be explained in section 3.2.4.

The evaluation function is ad-hoc and volatile and does not perfectly characterize the difficulty of synthesizing a compound. However, it is an excellent heuristic in that, for large classes of compounds it characterizes well the difficulty of synthesizing, and is computationally simple and inexpensive. Further, our procedure for backing up evaluations from lower levels and making continuous revision of merit ratings as the search unfolds, compensates for any imperfections of the evaluation.

Thus, the second aspect of the heuristic evaluation of subgoals is the backing up and revision of merits of all affected subgoals higher up in the problem solving tree. As mentioned above this is an essential step and permits goal selection to proceed top-down beginning at the root of the problem solving tree.

3.1.2.4 STOPPING CRITERIA

Each time through the main cycle of the program, after evaluation and before invoking subgoal selection, the program considers a decision to stop.

The program takes stock of resources used up and resources remaining. Those resources examined include time, memory space in core, and the disc space used to store the problem solving tree. The size of the problem solving tree can also be used to limit investigation.

If all subgoals are either 'available', 'solved', or 'stuck', no further work can be carried out and the program must halt.

At present, no limits are placed on the number of completed syntheses that may be discovered. The program thus has the mandate of exploring the best avenues available without exhausting any critical resources.

It is conceivable that one could compare the merit of the best unsolved subgoal with the merit of completed syntheses thus far, and decide upon the usefulness of further exploration.

3.1.3 RESULTS OF HEURISTIC SEARCH

After several repetitions of the heuristic search steps described above, one expects to have a well developed problem solving tree, which is richly informative potential and solved syntheses for the target

molecule. One must be cautioned against hastily assessing the program performance by counting completed syntheses only. Potential syntheses at any stage are of value because completed syntheses for intermediate compounds are suggestive and significant in themselves.

Furthermore, it is not the number but the quality of the proposed syntheses that need be assessed. More importantly, for those of us investigating artificial aspects of this work, it is the intelligence exhibited through the well-balanced tree development and controlled attention switching that is of value.

With that word of caution, the reader is now refered to Figure 4 for a schematic of the state of the problem solving tree for Vitamin A after six minutes of total machine time had elapsed.

The problem solving tree has subgoals that were 'solved' (e.g. 4,5) that were 'stuck' (e.g. 8, 16) and that were not explored (not numbered, indicated by horizontal bars). There were over 120 subgoals generated, the maximum depth of any subgoal in the tree being 8. The branching of the tree was 43 at the first level; this number dropped sharply to 17 and 30 at the next level and to 10 at the next further level. The circled numbers indicate the order in which subgoals were selected for development. The pattern of tracking is interesting to us, as it seems indicative of the artificial intelligence imparted by the heuristics. It is certainly far from the ideal one would want, but is superior to the plodding of a breadth or depth search

We pursue details of the heuristic search in subsequent sections. Readers wishing to avoid details of subgoal generation and the search strategy may skip to the last section, IV.

3.1.4 DETAILS OF SUBGOAL GENERATION

The subgoal generation details are given as a flowchart in Figure 5. The problem solving tree for Vitamin A (Figure 6) will be referred to in the following explanation.

The subgoal generation is entered with the Wiswesser linear string corresponding to Vitamin A as the goal (Step 1). The explicit topological structure description is devised (Step 2) before scanning it for recognizable attributes or features. (Step 3). The features relevant to aspects of synthesizing the molecule are sought. For Vitamin A, several structural features are recognized and labelled accordingly as carbocyclic ring, olefin bond and alcohol.

The latter two attributes have reaction schemata available in the reaction library. (The schemata in the library are grouped into reaction chapters and are identified with the name of the structural feature each group can synthesize).

When the 'olefin bond' chapter is chosen (step 4), a further selection of a specific site has to made from the five different occurrences of olefin bond in Vitamin A. Referring to Figure 6, the reader can see the five-fold branching from the olefin bond chapter layer, one branch for each site chosen.

Considering the first instance of olefin bond (step 5), the reaction chapter for olefin bond synthesis has 5 schemata in it. One of them fails the heuristic tests for applicability (Step 6 and 7). Thus in Figure 6, the attribute instance (=1) layer branches four-fold to the reaction schema layer.

Each schema in the reaction library has a definition of the structural transformation executed by the reaction. The transformation is written as a goal pattern (in an appropriate language designed to represent structural

patterns) and a subgoal pattern. The application of the transformation involves matching the goal pattern to the goal compound (step 8). The pattern matching program, a graphical pattern-controlled sub-structure matching algorithm, has been described in full in the thesis (ref. 1).. The matching routine can find all ways of matching the pattern with the compound. Each such successful match will lead to a set of subgoal compounds (step 9).

If a reaction transforms one compound into another each subgoal generated will always consist of a single compound. However, reactions utilizing two or more compounds in generating a product compound will yield a set of two or more compounds for each subgoal. In such cases the subgoal node is said to form and AND-node implying that this subgoal will be considered solved only if every compound in the set is considered solved. AND-nodes are marked in Figure 6 by short horizontal bars over the compounds. For the first instance of olefin bond attribute, schema 3 generates two subgoals each of which has two compounds in it.

The programs that carry out the pattern matching and subgoal definition, work with the explicit topological descriptions, and the subgoal so defined is given explicitly. These subgoals are reduced to the compact Wiswesser names (step 10). The generation of Wiswesser names provides not only savings in storage space but also has the more important benefits of a) allowing one to recognize inexpensively and efficiently occurrences of the same compound in several nodes of the problem solving tree, using the canonical nature of the Wiswesser names (step 11) and b) allowing one to query the Aldrich Chemical Catalog for the availability of compounds (step 13).

The subgoals are posted into the problem solving tree (step 12) after pruning out those that are duplicate or circular (step 11).

In step 14, a check is made to see if any of the subgoals can be marked 'solved' and if so, a further check is made for compounds higher up or elsewhere in the tree that might have been newly synthesized.

Following this expansion of the problem solving tree, the program cycles through all attribute instances and all reaction chapters. When all valid chapters have been dealt with the subgoal generation process is completed and control is returned to the executive.

3.2 SEARCH PROCEDURE

To generate multi-step syntheses the program needs a rule for selecting from the problem solving tree the next compound to develop. One can either set up a convention to specify selection leading to such well-known strategies as depth-first or breadthwise development. Some theorem provers and look-ahead tree searching programs in game-playing employ such tactics.

There are several ways of guiding a heuristic search when subgoals can be evaluated so that subgoals can be compared with a view to making a choice. Using an evaluator, in AND/OR trees such as our problem solving tree, one general principle is to select:

i) among alternative subgoals one with the highest value, and
ii) among conjoint entries in a subgoal one with the lowest value.

Our program presently utilizes a static evaluation function (involving no lookahead, no heuristic prediction of the probability of success) that is meant to be a crude model of the problemsolving complexity of a compound, a subgoal and a reaction sequence. It represents a zero-order attempt to abstract the organic chemists' judgement to provide heuristic guidance for the program. We anticipate that major changes and improvements will be forthcoming in this phase of research.

3.2.1 MERIT OF A COMPOUND

Compounds are evaluated on a scale of 0-10. A compound that is available commercially, cheaply and in bulk gets the highest rating, 10. A compound for which syntheses have been found by the program is assigned the best merit among its syntheses. A compound which has been developed in the tree (i.e., itself has subgoals) bears the best merit of its subgoals. For other compounds the merit is calculated according to a function that has been and will continue to be highly volatile.

$$\text{CPMERIT} = \left(\frac{10}{1.2}\right)\left(\exp(-0.1385*FG) + 0.2\exp(-0.0346*C)\right)$$

C = number of carbons not belonging to any ring
FG = weighted sum of entries in attribute table

thus intended to yield higher merit for compounds with fewer identifiable features and lesser size as counted by carbons. Evidently there is a higher weighting for FG than for C.

3.2.2 MERIT OF A SUBGOAL

The merits of each of the compounds in the subgoal is multiplied together, and compounded with the adjusted ease assigned to the reaction involving this subgoal, the resulting value when normalized to a 0-10 scale gives the merit of a subgoal. For example, consider a scheme that generates two subgoal compounds by splitting the goal at strategic locations. The above computation scheme for merit will prefer to split the goal at its middle, to get nearly equal sized molecules in the subgoal.

3.2.3 MERIT OF A SEQUENCE

The merit of a reaction sequence is computed by starting at the terminal elements and iteratively computing the merit of the final compound that is synthesized. The merit of the sequence is the computed merit of the final compound.

The process of devising a good computational scheme for merits is one of trial and error. When the chemist reviews the pattern of tracking the space exhibited by the program, we may freely tamper with the factors entering into merit computation and the way in which they are combined together.

3.2.4 GOAL SELECTION

After each cycle of goal selection, subgoal generation, merit evaluation, the tree is updated so that each compound correctly reflects the effects of the latest cycle. Some of the original sequences can now be found solved, or stuck or to be of a greater or lower merit than previously supposed. Such updating is done by a merit backup procedure. Following that, a simple subgoal selection algorithm is used. At each level of the problem solving tree, starting from the top level, the subgoals at the next level are examined and the best one is selected. Among the compounds in that subgoal, the one with the least merit is selected. This is carried out till a compound that has not yet been sprouted is gotten.

Thus the algorithm can switch attention from one subgoal to another quite easily. However, frequent attention switching is not desirable and a hesitation is programmed in the algorithm. This is done by forbidding switching attention ifthe merit of the newly selected goal is not 'significantly' larger than the last goal that was sprouted.

Rule for hesitation in attention switching:

If (merit of new goal chosen - merit of last leal sprouted) > hesitation

threshold
then pursue new goal
else pursue old goal.

The hesitation threshold in the above rule can be set externally and when set to zero would indicate no hesitation to attention changing.

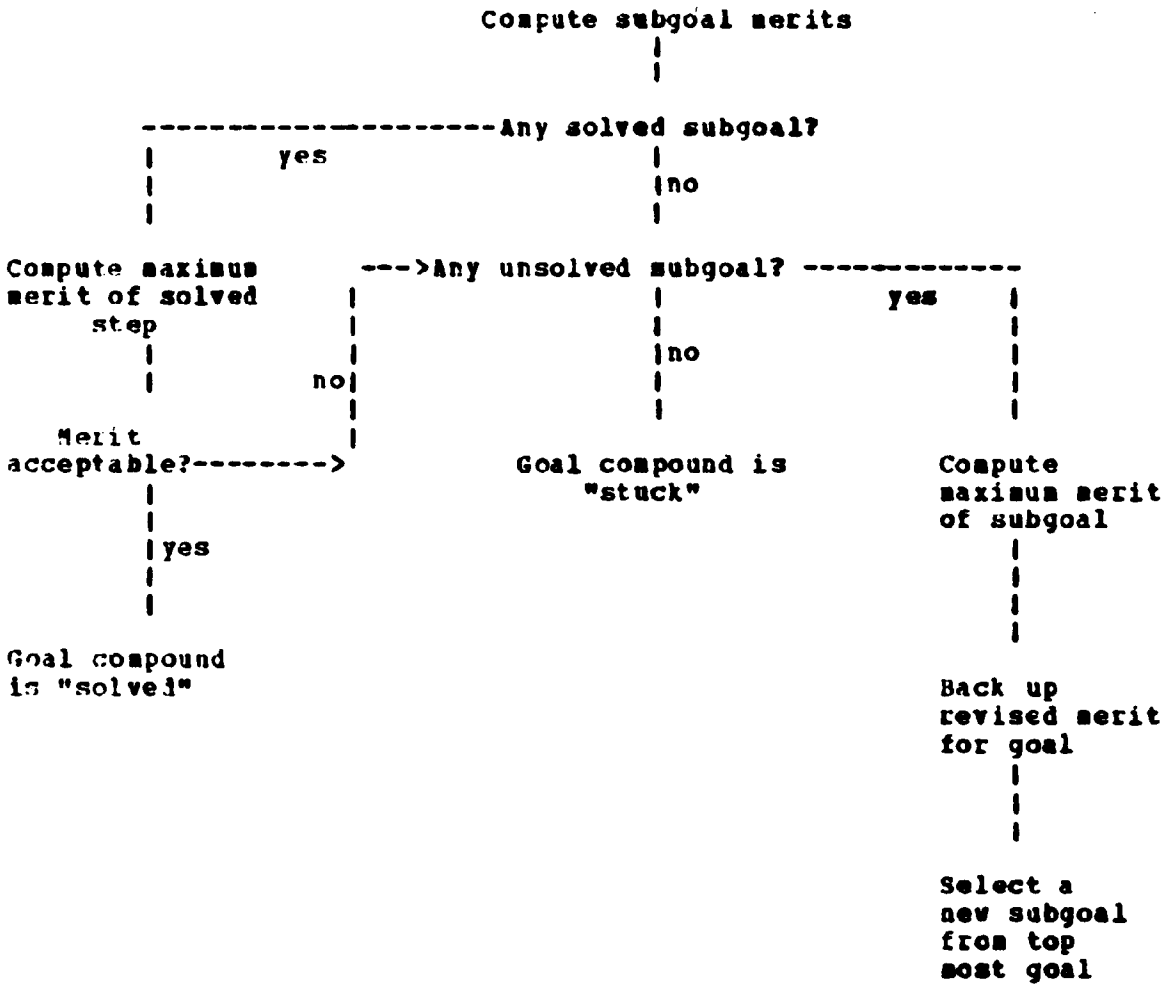
Though simple in description, such a procedure does demonstrate quite a complex tracking behavior in the tree. (See Fig 5.5) First, by repeatedly working on the most meritorious subgoal the tree generated is rich in information, with complex subgoals left relatively undeveloped and simpler subgoals pursued mostly to completion. This algorithm is expected not to give very narrow and deep, nor broad and shallow trees, but to allow development of multiple syntheses routes. The success of the algorithm surely depends on the ability to prune the tree efficiently of invalid and potentially sterile subgoals and on the ability to recognize promisingly good subgoals.

When a goal compound sprouts subgoals of good merit, one expects that the algorithm may plunge deeper into the tree. However, the possibility of working on a conjunct compound of higher complexity does help to broaden the tree.

As syntheses for intermediate compounds are completed, they are evaluated and then accepted or rejected. This is in consonance with the notion that short syntheses are not necessarily the best. When each synthesis is discovered for the target compound the program is afforded a chance to broaden the search and diversify the approach to solutions. It is an important ability for the program to judge completed syntheses and compare them against promises held out by the unsolved subgoals. Subgoal selection strategies in a problem solving tree containing several complete solutions are interesting but will not be discussed here.

The criteria by which syntheses are accepted need not be static. If a synthesis of low merit is forced upon one conjunct (because no better synthesis is possible with the given data base of knowledge) other conjuncts should have their level acceptable merit lowered accordingly in order to keep decisions consistent.

The following flowchart illustrates how the selection of subgoals is done.


Compute subgoal merits
Any solved subgoal?
yes
no
Compute maximum merit of solved step
Any unsolved subgoal?
yes
no
no
Merit acceptable?
Goal compound is "stuck"
Compute maximum merit of subgoal
yes
Goal compound is "solved"
Back up revised merit for goal
Select a new subgoal from top most goal

3.3 PRUNING THE PROBLEM SOLVING TREE

The power of the heuristic search technique depends not only upon a good subgoal selection method but also crucially on the tree pruning rules that can be formulated. Even though within the framework of solution generation there are similarities in techniques employed by the Geometry Machine and the Synthesis Program, major departures occur in the rules for tree pruning. Whereas Euclidean Geometry was genuinely formalizable and Organic Chemistry is not, our pseudo-formalization of organic chemistry is very complicated. Thus the direct cost measured in terms of computing effort of subgoal generation is substantially greater for chemistry. Investment of effort to cast off invalid and unproductive branches of the tree brings greater returns in our case than in that of the Geometry Machine. The problem of eliminating invalid, redundant and unpromising subgoals is analyzed below.

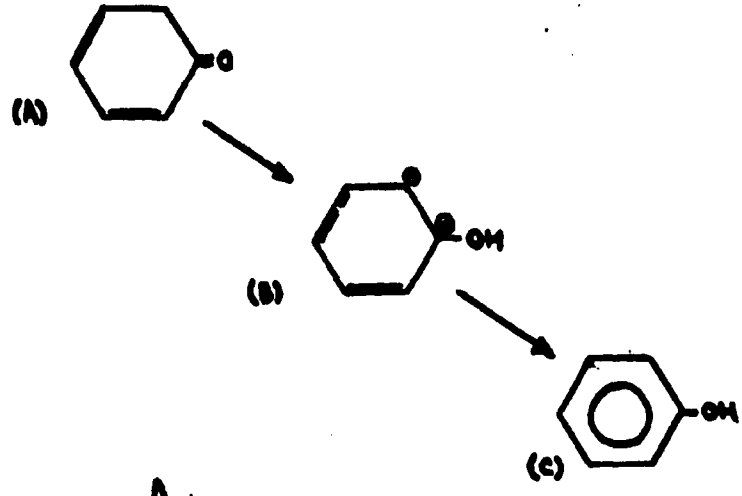
3.3.1 ELIMINATING INVALID SUBGOALS

The purely mechanical generation of subgoals can in general give rise to invalid structures. We do not foresee valence violations to occur in our system but generated structures may violate spatial constraints not adequately represented in our model of chemical structures. Certain structures can be subject to thermodynamic or kinetic instabilities (that is, may decompose when left alone, self-polymerize or adversely react with the products of the reaction). The Geometry Machine could test the validity of a generated expression by interpreting it in a model of the formal system, a diagram. There appears to be no simple model or rule that will indicate instabilities for arbitrary structures.

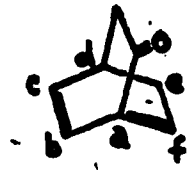
One could, of course, make use of the available chemical literature and compile tables of instabilities. Structures reported in the literature could then be cleared for instabilities by simple table look up. The practicality of this method is questionable. Even on other grounds, to restrict the class of acceptable intermediate subgoals to tabulated compounds alone would severely limit both our interest and the range of possible syntheses.

There are two other possible approaches in invoking whatever results of physical chemistry we can systematize, generalize and formalize sufficiently to be of help in pruning invalid subgoals. We have made progress in extracting simple models of instabilities. A series of tests against these models can aid in determining common invalidities.

Example: There is a model of alpha-proton migration to a carbonyl group by which a ketone can become a hydroxyl group. Another rule states that an aromatic ring is more stable than a cyclo-diene. Thus when the structure A (Hexadiene-ketone) is examined for validity, the structure C (Phenol) is found to be its more stable configuration.



Example: Consider the structure



If one follows the ring a b c d e f keeping track of bond angles, one would find that the orbitals of atom f and atom a which are to form a double bond, are perpendicular to each other and that the strain of forming the bond will be excessive.

The programming of these models is not complete. Just as a chemist does not derive all his information theoretically, the program too can suggest experiments to answer some questions, upon which certain critical choices will be made. As the program develops sophistication there are exciting prospects for using quantum mechanical considerations. Wahl (Ref. 8) has reported on BISON which uses quantum mechanical concepts of structure to calculate molecular orbitals. An extension of such a program may yield criteria for stability of arbitrary structures.

3.3.2 INVALIDATION BY COST CONSIDERATIONS

In general, reactions have yields that are far less than 100%. Thus, if a compound C is used as a starting material to produce some fixed amount of the goal compound, one would expect to require more compound C if the synthesis involves long sequences. Thus, at deeper levels in the problem solving tree one would tend to disfavor expensive starting materials more strongly than one would near the top of the tree. Compounds that are not available in bulk can be removed from consideration for those sequences where the expected yield becomes low. This kind of behavior is not to be found in Theorem Proving programs because occurrences of a theorem or an axiom are identical no matter where they appear in the tree.

3.3.3 ELIMINATION OF REDUNDANT SUBGOALS

Many 'problem independent' tree pruning rules applicable to problem solving tree were developed in earlier AI works. It is our observation that these syntactic rules for pruning from the problem solving tree those subgoals which though valid, would lead to circular, redundant and otherwise unsatisfactory syntheses are not directly applicable. Syntactic grounds alone are not sufficient to answer the question: "To prune or not to prune?" Let us consider some situations in detail.

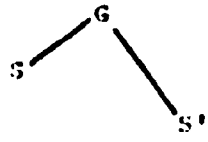
In order to eliminate redundancies, the names of the compounds alone cannot be compared, but must be treated along with the name of the reaction schema used in the generation and the merit of the partial discovery sequences.

Consider a subgoal S on the tree. Subgoal S' is newly generated and is a duplicate of S in having the same conjunct compounds. The choice which subgoal to eliminate and which to retain is not simple.

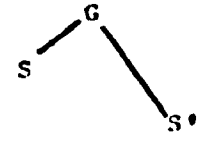
S and S' on same level

The Geometry Machine eliminates S' as it cannot lead to a better or shorter proof than can S. In our program, if the two reaction sequences are the same, any one of the two subgoals can be eliminated. With unlike reaction sequences, the program has to evaluate the sequences and retain the one that is significantly better.

When subgoal S' has all the compounds of S plus more, let us represent it as S'>S.



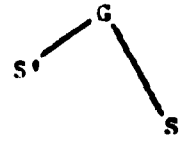
S=S'



S'>S

S' occurs lower in the tree

The Geometry Machine will clearly eliminate S' giving preference to a shorter proof. The Synthesis Program has no preference for shorter sequences unless they have greater merit evaluations. A long sequence of good and efficient reactions may be better than a short sequence of poorer reactions.

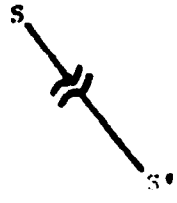


S at lower level than S'

S'>S

For the Geometry Machine the decision at this point is arbitrary. It would retain both but may prefer to try solving S before attempting S'. The decision for the Synthesis Program is again not made on syntactic grounds alone.

Our program would without question forbid circular subgoal sequences, such as



S'=S or S'>S

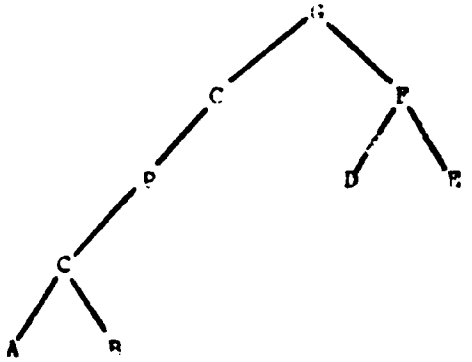
There are two special situations that are not allowed in order to permit pruning of circular subgoals.

First, there are known instances of syntheses where a small amount of rare but naturally occurring material is used in a sequence of reactions which will generate a larger quantity of the same compound as end-product. Our program does not allow such syntheses. But if there were interest in seeking recirculatory syntheses the program could be run without this rule.

Secondly, our program is concerned only with producing a chemical path for synthesis. Before the synthesis procedure is executed, it is necessary to decide upon the time sequence in which the reactions will be carried out. Here, one of the considerations to be made is about unstable compounds that cannot be kept in a test tube or in a container for any length of time.

Example: G

If C and F are both unstable then one of them, say C, has to be produced by reacting A and B and converting to a stable form P. After F is produced by reacting D and E, P is converted back to C and then allowed to react with F, assuming that conversion from P to C is quick and the F can be kept for that length of time. Thus, if the program were time sequencing the path, a graph such as the following



containing a circular sequence CPC would be a valid path.

It is clear that syntactic pruning is powerless in curbing the exponentiation of the problem solving tree. As explained before, some pruning can be achieved by introducing limited solution path evaluation even during solution generation. Domain-specific heuristics are necessary and we have freely introduced such chemical heuristics as could be simply specified. These heuristics are of help in eliminating inapplicable reactions, reactions with adverse side-effects, and reactions where the specific structure of the generating goal hinders the course of the reaction. These are powerful heuristics and we expect such domain- specific heuristics will increase as we gather experience with the program.

3.3.4 ELIMINATION OF UNPROMISING SUBGOALS

The next task is that of removing from further consideration those subgoals that hold no promise of successfully completing a solution path. This is a desirable aim which depends almost entirely on the computed measure of promise exhibited by a subgoal. If this measure is tested and proven, it is reasonable to do the pruning. If not, there is the uncertainty that even subgoals which seem promising may fail to give a synthesis and the other subgoals may in reality, be the only ones which lead to a synthesis.

The information available in the problem solving tree pertains to the complexity of the compounds, the number of conjuncts in the subgoal and the level number of the subgoal. A measure of promise does not seem computable from this available information.

For the present, we shall not attempt to eliminate unpromising subgoals, but retain all of them. We shall, however, attempt to be judicious about the manner in which subgoals are selected for development. In this way it is possible that unpromising subgoals will never be selected initially, but will be available on the problem solving

tree it needed.

IV PROGRAM PERFORMANCE AND CRITIQUE

The program has been run on several molecules successfully. We avoid mentioning the specific number of molecules as it is of no consequence in evaluating the program's performance. The compounds used range widely in their complexity, and the problem solving tree reflects to a degree, the complexity of the target molecule. The computing time can vary from a minute to a few minutes as in the case of Vitamin A.

Often the syntheses proposed by the program are claimed to be unworkable and this is attributable to the limited knowledge present in the reaction library. This situation will improve as the reaction library and the heuristic rules contained in it are expanded. On the other hand, the program has applied well-known reactions in unfamiliar contexts and has surprised the chemists by the novelty of the syntheses. The latter case arose particularly for molecules whose topological complexity made them hard to visualize. The program however, unconstrained by any need to visualize molecules, was able to apply familiar reactions to achieve syntheses that some chemists would not have considered.

The assessment of complex heuristic programs, especially those that attempt to emulate higher mental processes, has remained problematic. In attempts to make such assessments, one finds that subjective value judgements quickly overtake other considerations.

We shall conclude by saying that examination of the program runs have shown us ways to adjust search strategy including the merit evaluation functions, the back-up and revision of merit, and the effective methods of subgoal selection. The program has been benefitting consistently from each critical evaluation of a run. Performance necessarily depends on the extent of the knowledge base provided in the form of the reaction library and on the quality of heuristic judgement presented through it. Thus immediate future work will be directed mostly toward building up a better and larger reaction library.

ACKNOWLEDGEMENT

We thank Mrs. M. Fortes and Mr. S. C. Yen for their part in the success of our research effort. Financial support was largely through the Research Foundation of the State University of New York, with assistance from a NSF Computer Science Program Development Grant.

LIST OF FIGURES

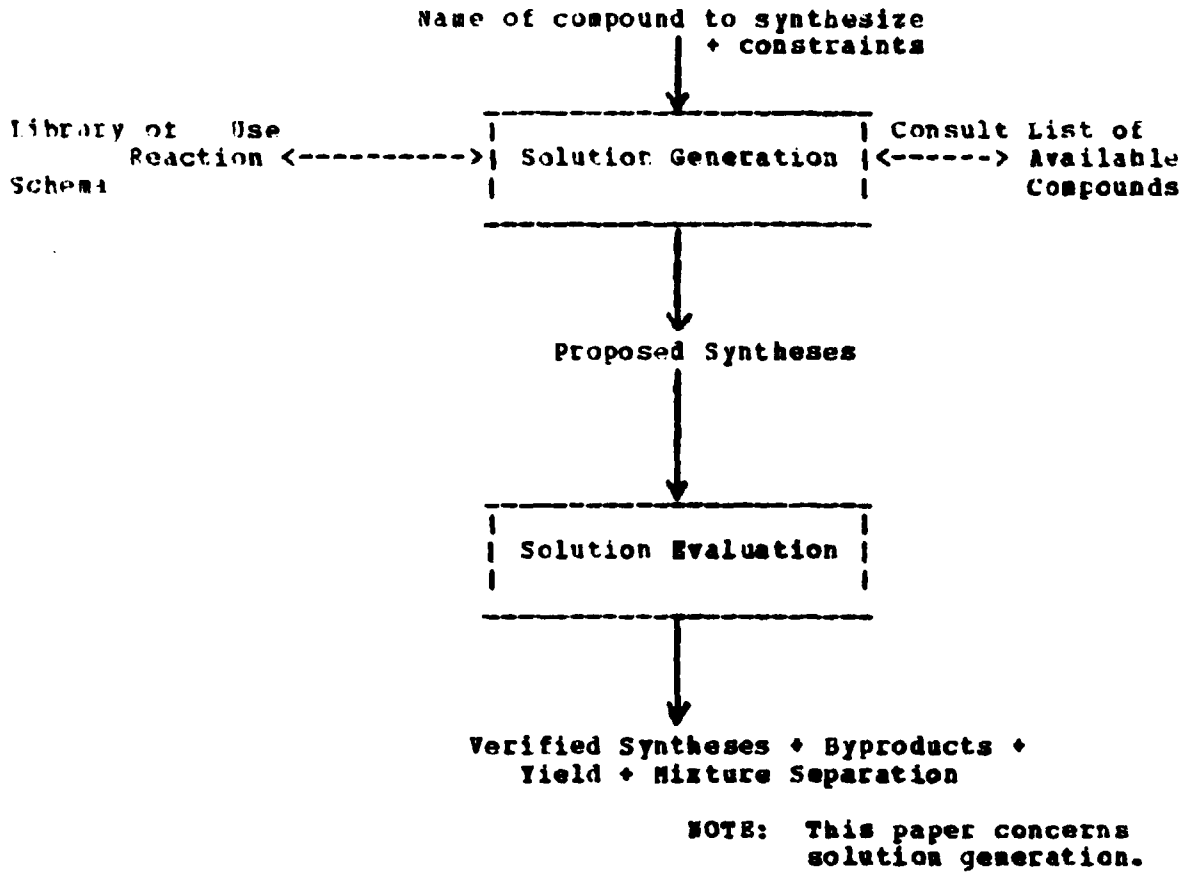
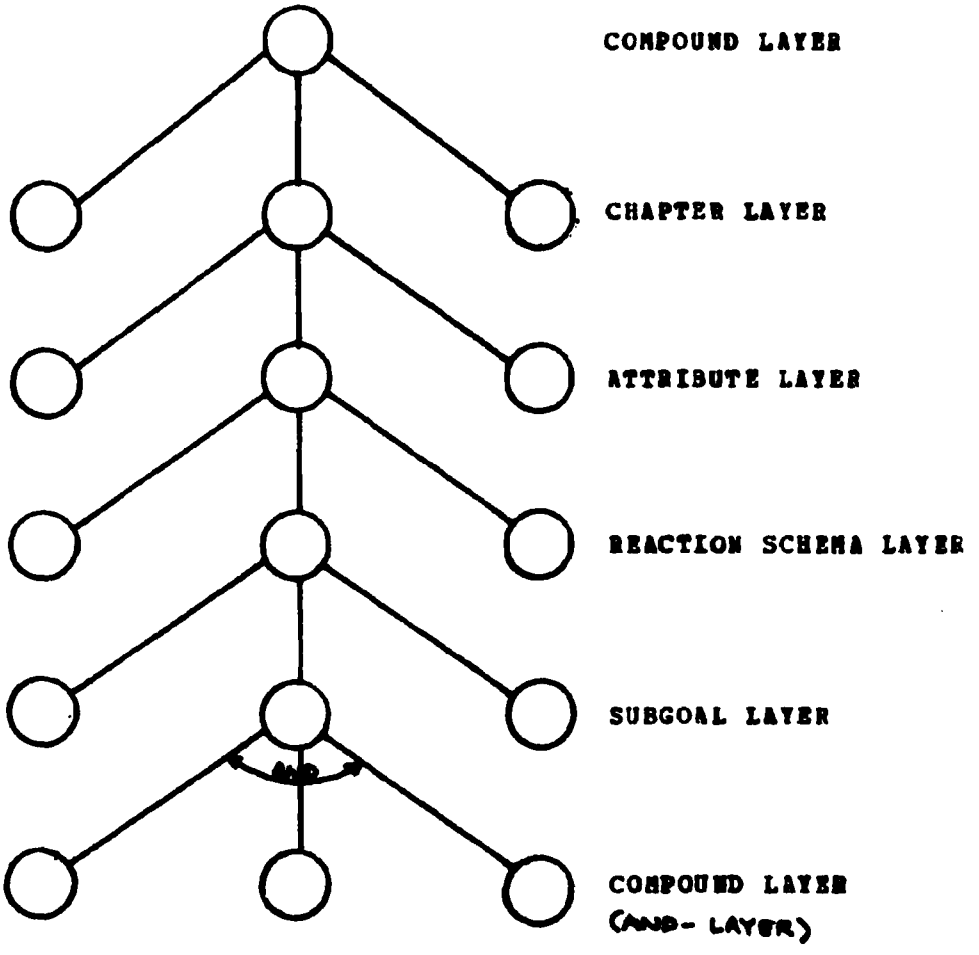

Name of compound to synthesize
+ constraints
Library of Use
Reaction Schemi
Solution Generation
Consult List of
Available
Compounds
Proposed Syntheses
Solution Evaluation
Verified Syntheses + Byproducts +
Yield + Mixture Separation
NOTE: This paper concerns
solution generation.

Figure 1.



FIVE-LAYER STRUCTURE OF THE PROBLEM SOLVING TREE

Figure 2

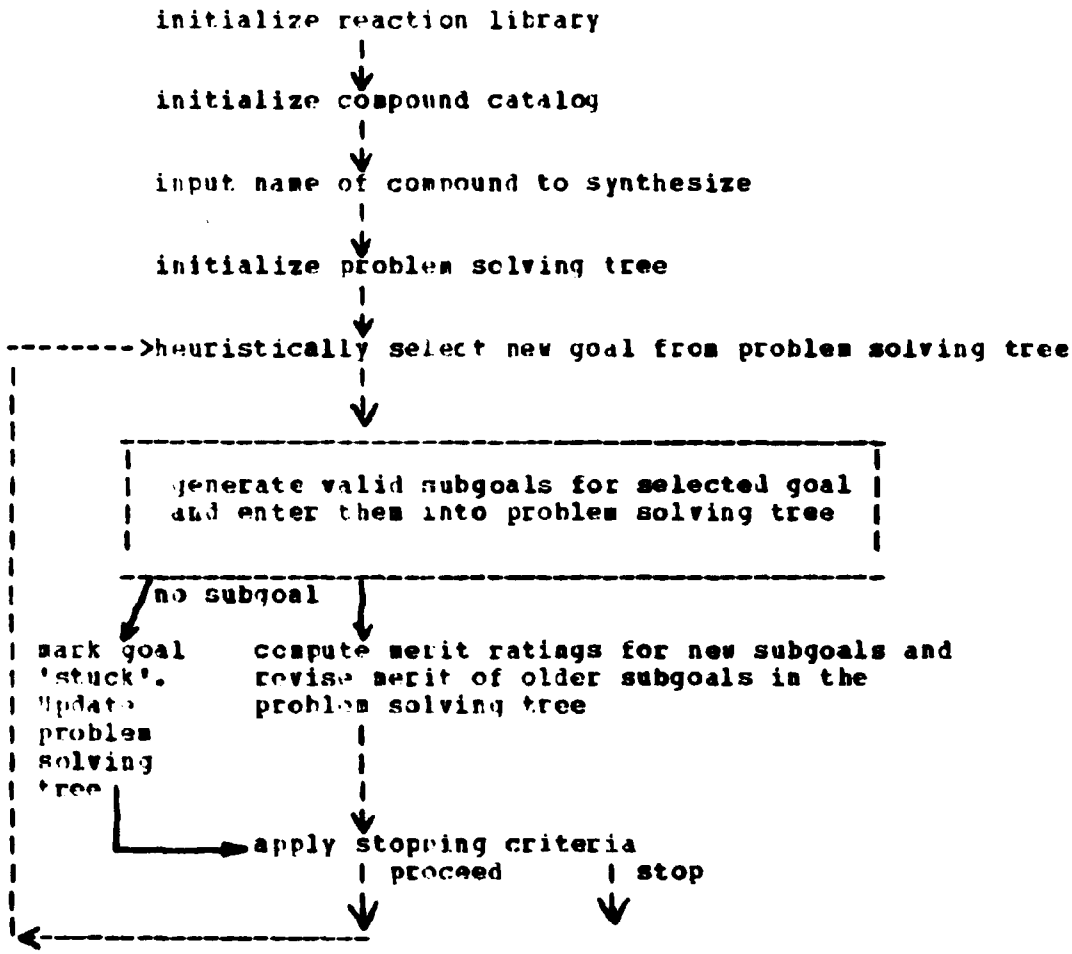


Figure 3. Simplified Flow-Chart of Heuristic Search Process

Note on Figure 4.

Synthesis-search tree (schematic) for Vitamin A. Filled-in circles represent reactants of subgoals selected for further development. Order of development is indicated by the circled numerals. Compound nodes connected by a horizontal line segment (as in subgoal 3) are both required for a given reaction. All generated subgoals on the tree that were not selected for exploration are represented by a horizontal bar, with the number of subgoals in the unexplored group indicated under the bar. Subgoals that were selected for exploration that have no progeny on the tree (as in subgoal 8) failed to generate any subgoals that could pass the heuristic tests for admission to the search-tree.

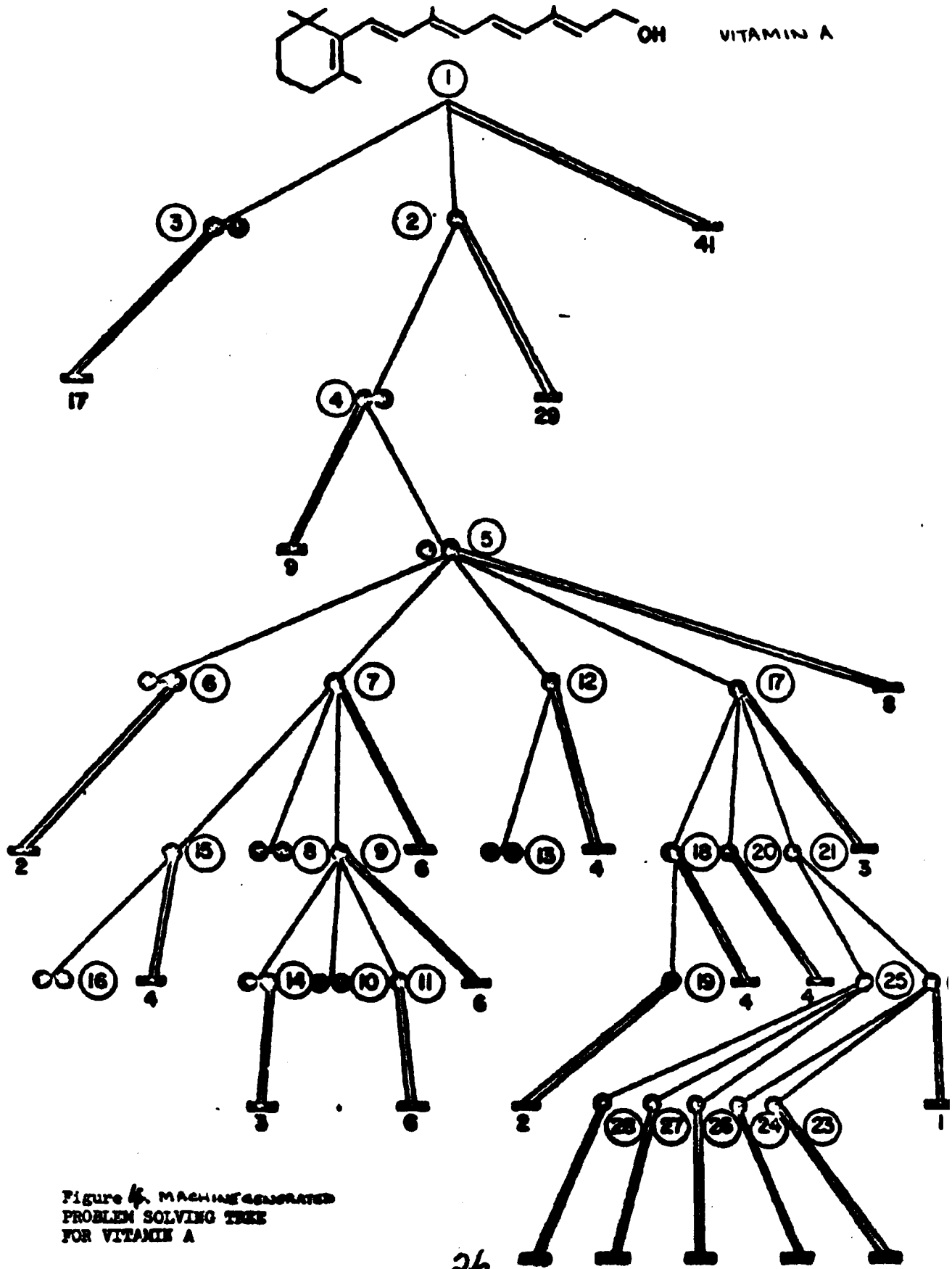


Figure 4. MACHINE-GENERATED
PROBLEM SOLVING TREE
FOR VITAMIN A

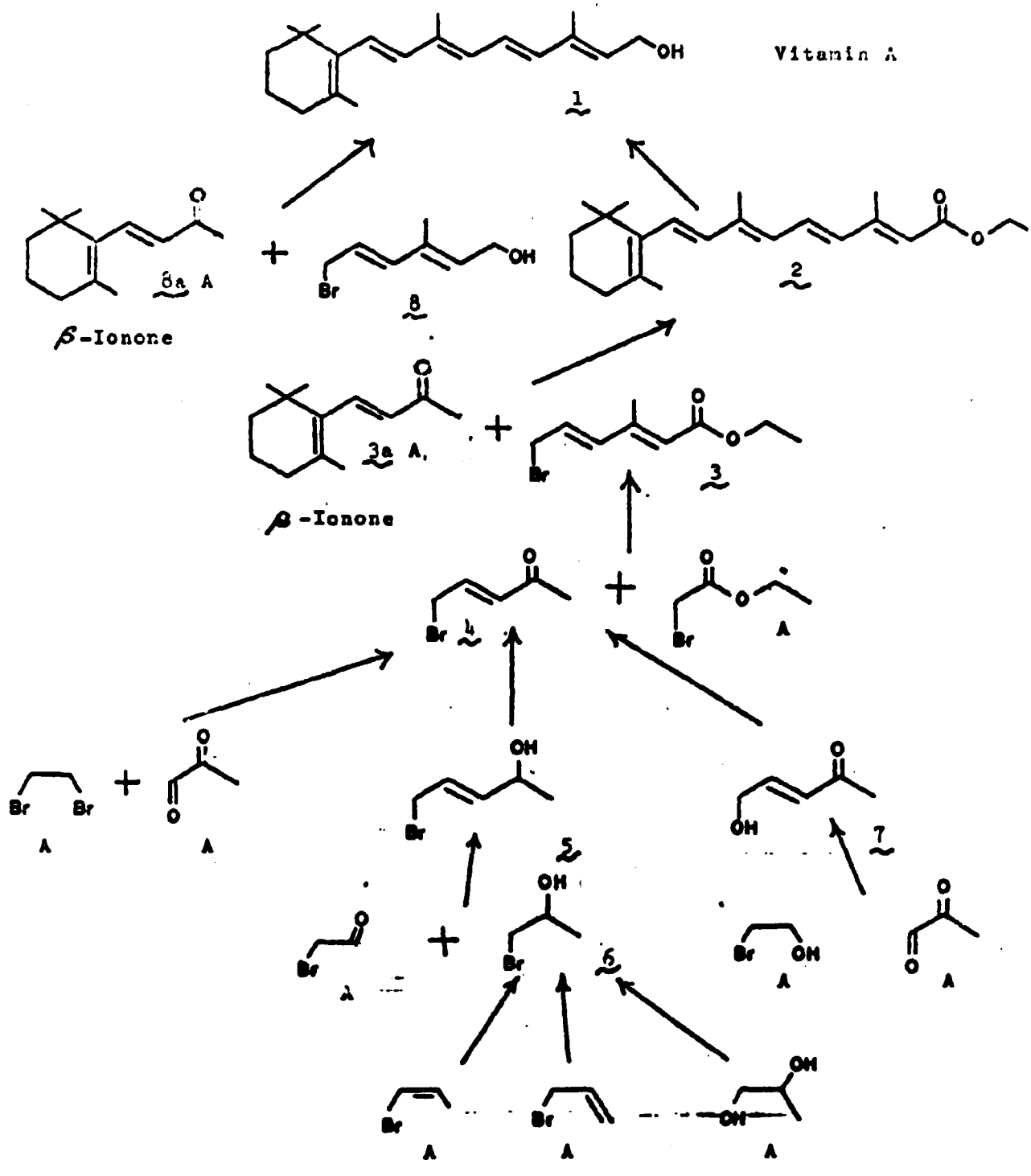


Figure 4. Syntheses found for Vitamin A.

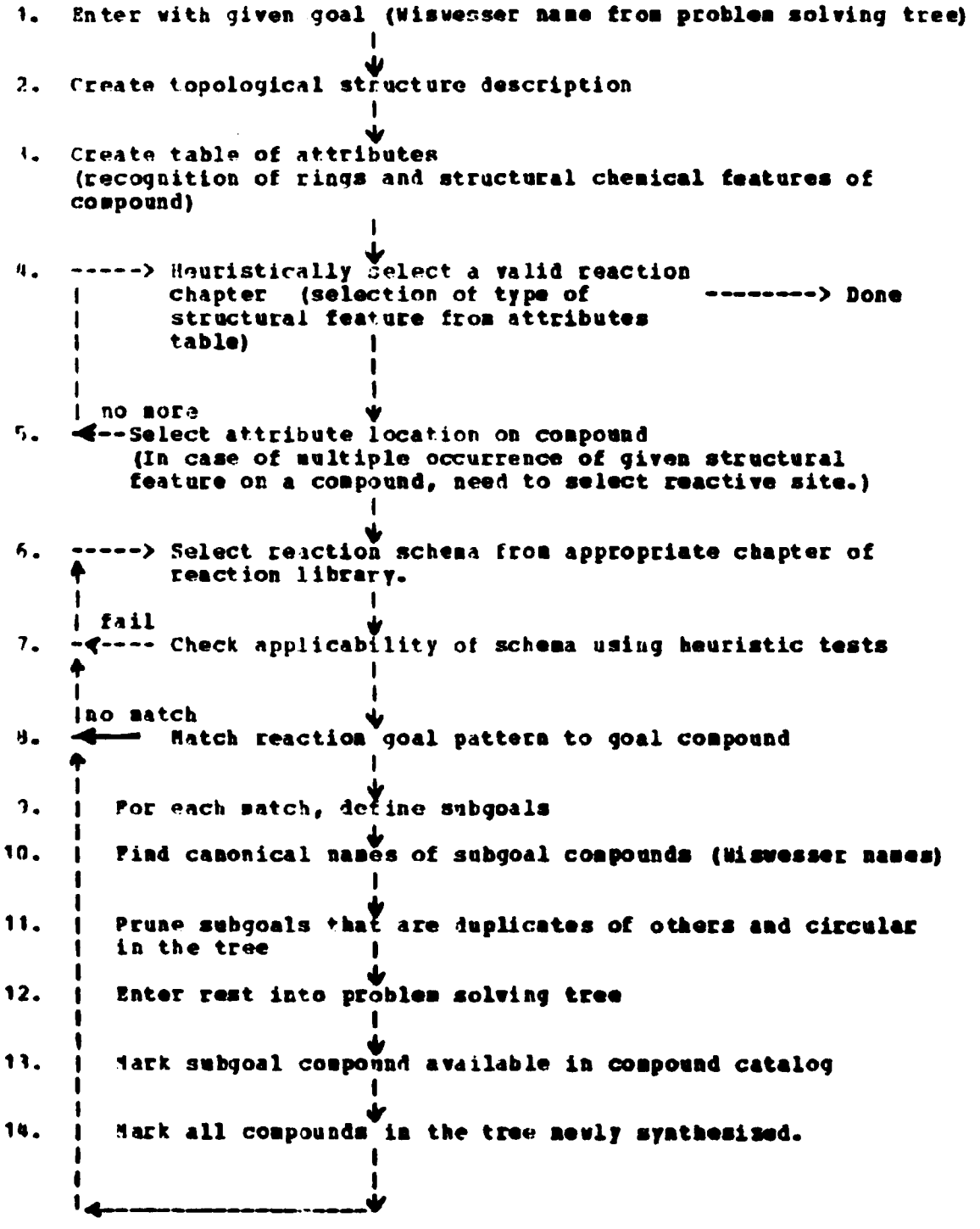


Figure 5. Details of Subgoal Generation

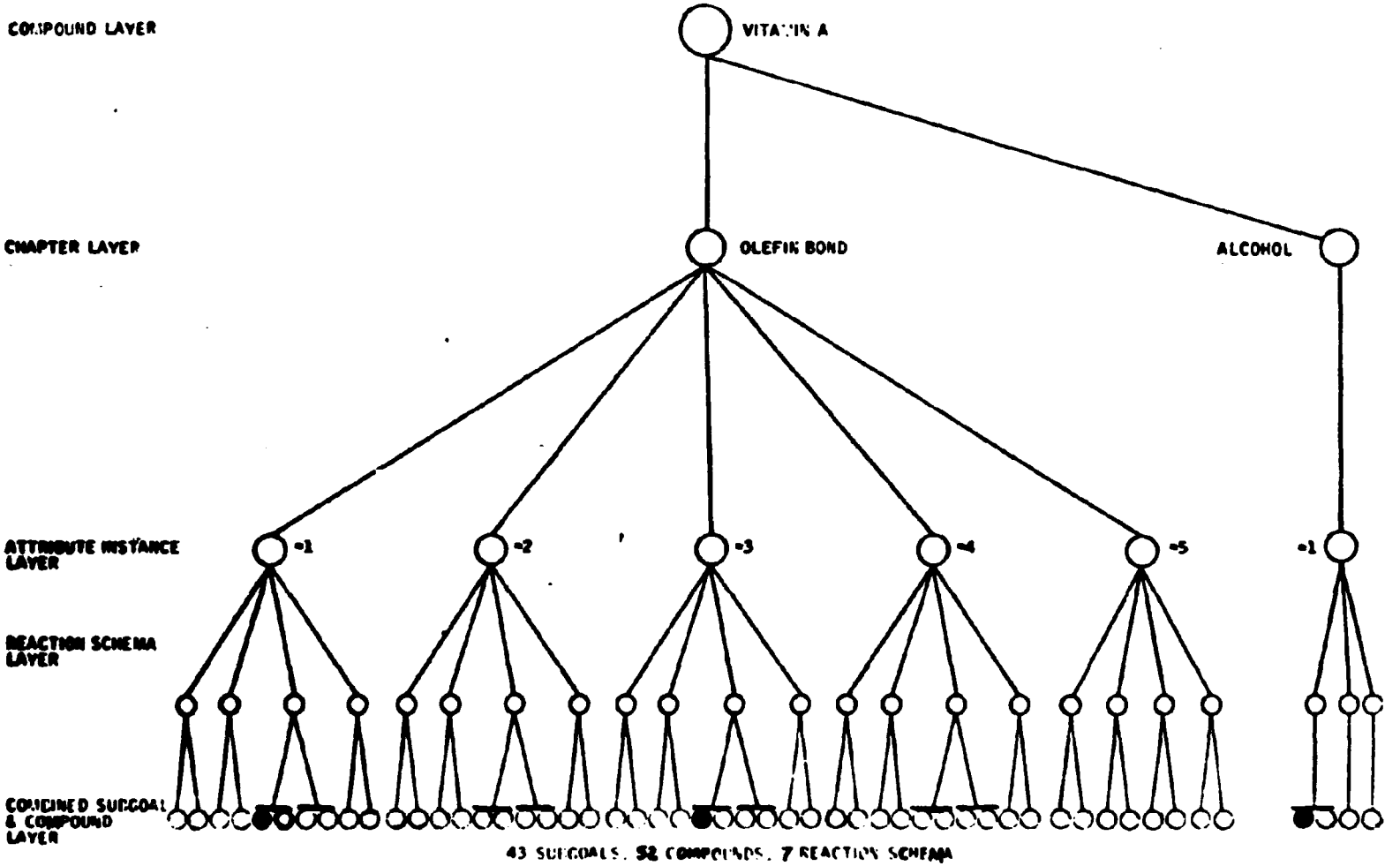


FIGURE 6 ONE LEVEL OF PROBLEM SOLVING TREE GENERATED BY THE PROGRAM

Legend: Compounds found in the Aldrich Chemical Catalog are shown as filled-in circles.

REFERENCES

1. Sridharan, N.S., An Application of Artificial Intelligence to Organic Chemical Synthesis, Doctoral Thesis, Department of Computer Science, State University of New York at Stony Brook, New York, July, 1971. (available through University Microfilms.)

2. Hart, A.J., Doctoral Thesis in preparation, State University of New York at Stony Brook, New York, 1972.

3. Aldrich catalogs, Aldrich Chemical Company.

4. Smith, E.G., The Wiswesser Line-formula Chemical Notation, McGraw-Hill: New York, 1968.

5. Gelernter, H., "Realization of a Geometry Theorem Proving Machine", in Information Processing, UNESCO, 1960.

   _____, "Machine Generatod Problem Solving Graphs", in the Proceedings of Polytechnic Institute of Brooklyn, Symposium on the Mathematical Theory of Automata, New York, April, 1962.

6. Fieser, L.F. and Fieser, M., Advanced Organic Chemistry, Reinhold: New York, 1961.

7. Fortes, M., Unpublished Master's Thesis, SUNY at Stony Brook, New York, 1971.

8. Wahl, A.C., et.al., "BISON: A new instrument for the Experimentalist", in the International Journal of Quantum Chemistry, Volume IIIS, 1970.

9. Gelernter, H., Sridharan, N.S., et.al., "Computer Methods in Organic Synthesis", Topics in Current Chemistry (to appear), Springer-Verlag, 1973.