High-Speed Management of the National Compound Registry†

P. A. D. de Maine

Computer Science and Engineering Department, 108 Dunstan Hall, Auburn University, Alabama 36849

Received September 21, 1997

New high-speed techniques for representing, normalizing, and manipulating graphs or topological maps such as chemical formulas are described and demonstrated. This paper focuses on the use of the new techniques for representing chemical formulas and finding occluded groups of atoms within molecules.

I. INTRODUCTION

Chemistry related literature is probably the largest technical collection in the world, and Chemical Abstracts Services, CAS, has the largest digitized scientific database. Chemical Abstracts, CA, and the National Compound Registry, NCR, are a vital part of educational and research programs. Most collections of digitized chemical related information are organized and maintained in the CAS system. There remains a need for efficient, high-speed methods for representing, retrieving, and manipulating chemical information. While the primary focus of this paper is on the NCR, the general methods that are described can also be applied to other scientific or nonscientific information.[1,2]

The two major problems associated with the NCR are as follows:

(i) Canonical and Normal Forms—Unique labels (or registry numbers) that are assigned to chemical formulas.

(ii) Fragment searches and searches for compounds with common characteristics. The two different kinds are (a) those in which the fragment(s) are specified (e.g., a keto group) and (b) those in which fragment(s) are not specified (e.g., all compounds that share any common characteristic).

Solutions for (i) have included the use of Line-Formula or Chemical Notational Systems,[3−9] atom connectivity matrixes,[10−15] and graphical/topological methods.[16−27] The early work of Morgan[28] is the basis for many National Compound Registry applications. Unfortunately none of these methods preserves the exact topological description of formulas, which means that exhaustive searches are needed to discover fragments that are not cataloged.

A key problem in data management is the retrieval of items that contain similar occluded structures or fragments that are not cataloged. Even with relatively small databases the cost of such fragment searches can be prohibitive. For chemistry the universal solution has been to create separate databases for cataloged fragments. This simplistic solution is not acceptable because of the following:

(i) Information about fragments that are not cataloged can only be obtained by exhaustive and costly searches of the original database. That inevitably means users' queries will normally be restricted to the cataloged fragments. For example, the comparatively simple query, *list all compounds with up to nine atoms or moieties that contain analogues of the skeletal chain: $(C^{14}X_2)CHY$, with X any single bonded atom (e.g., H, Cl, F etc.) and Y, O, or S,* can only be answered by an exhaustive search of the original database.

(ii) The loss of the interface between the different cataloged fragments and/or the original database needlessly complicates searches and discourages users from formulating interesting questions.

(iii) The unavoidable duplication of information can, especially in large databases such as the National Compound Registry, lead to prohibitive costs for online storage or, at best, the extensive use of slow secondary storage like magnetic tapes, archives, CD ROM, etc.

Clearly a high-speed method of maintaining and searching databases is urgently needed. This paper is devoted to describing new techniques that can be used to economically execute exhaustive searches of databases for fragments.

The basis of this work is the mathematically complete or philosophically closed[29] Transparent Query Language (TQL)[1] and its use in the High-Speed General Information Management System, HSGIMS.[2] TQL, which traces nodes in graphical representations and may serve as a data description or query language and can be used to describe virtually any information. A brief description of the Transparent Query Language follows.

I.1. Transparent Query Language, TQL. An ongoing project whose conceptual basis is as follows: *Any information (or entity) can always be represented graphically (e.g., by graphs, abstract networks or topological maps)* has resulted in the formulation of a general data structure (or language), called the Transparent Query Language,[1] that can be used to describe, retrieve, and manipulate *any* information. The information can be static (e.g., chemical formulas) or dynamic (e.g., flight paths of rockets). Because the meaning of a graphical representation is a matter of interpretation, the structure of the information (i.e., the syntax) and its meaning (i.e., the semantics) can be safely separated—a conclusion supported by Jain.[30]

There are three different kinds of graphical representation:

(i) **Simple** in which the meanings of the nodes are independent of one another. Examples are indexes that are used for inventory systems and in libraries.

(ii) **Matrix** which is used to describe and manipulate equations and their data.

† Keywords: information representations, topological maps, transparent query language, normalization, mobile canonicalization, searches for noncataloged fragments, Markush structures.

S0095-2338(97)00084-X CCC: $15.00
Published on Web 06/30/1998

Table 1. Conventional Override Codes for Specifying Uncertainty in Queries for the Values of the Partition Constant (m in M), Normalization Constant (N in M), and Kernels[a]

code	assigned meaning
1	Any nonzero value for the kernel is acceptable.
2	Any value for the kernel is acceptable.
3	GATE Override - (i) $OKV_i > 0$ means only those values that are defined by the $OKV_i/2$ pairs in OKDATA are acceptable. (ii) $OKV_i < 0$ means the $\|OKV_i\|$ values in OKDATA are acceptable.
4	NOT Override - (i) $OKV_i > 0$ means those values that are defined by the $OKV_i/2$ pairs in OKDATA are not acceptable. (ii) $OKV_i < 0$ means the $\|OKV_i\|$ values in OKDATA are not acceptable.

[a] OKV_i and OKDATA are defined in the text.

(iii) **Information Representation**, IR, which is used to describe and manipulate topological features such as those in chemical formulas and pictures.

Each of the three kinds of graph has a different set of operations that is a subset of the mathematical laws and therefore is mathematically complete (or philosophically closed).[29] Detailed specifications and uses of the Transparent Query Language, TQL, are given in ref 1. TQL, which describes paths connecting any combination of nodes in any order, is the mathematical basis of the High-Speed General Information Management System (HSGIMS).[2] The components of TQL and their natural language equivalents are TQL Queues (Book), TQL Jobs (Chapter), TQL Tasks (Paragraph), and TQL Items (Sentence). Their relationships are specified by six different kinds of logical connectors (SETA, OR, AND1, AND2, AND3, and USER). TQL Items have the linear structure

$$\mathrm{TQLI} \equiv (\mathrm{M/J}/S_0/S_1/S_2/..../S_P\ \mathrm{X})$$

M and J contain control and security information; the S_i ($=K_{1,i}\ K_{2,i}...\ K_{Q,i}$), with $0 \le i \le P$, are substrings of kernels. The $K_{j,0}$, with $1 \le j \le Q$, in S_0 describe nodes and the $K_{j,i}$, with $1 \le i \le Q$, describe the connections between nodes. Kernels can have any value other than 0−8, which are reserved for system use (subsection I.3). Within the available computer resources there are no restrictions on the size of kernels, the number of kernels in the screens S_i, the size of TQL Items, TQL Tasks, TQL Jobs, or TQL Queues. Here the primary focus is on the use of Information Representations to describe and manipulate chemical formulas.

I.2. Information Representations, IR. The IR form of TQL Item is a linear encoding of information representations. An IR of rank N can be regarded as a generalized N-dimensional topological map that is capable of being transformed in special ways. In particular, any IR form of a TQL Item can be transformed to an unique (normal) form or to another of its many different mathematically equivalent forms. They can be used to represent static or dynamic three-dimensional objects and graphs, like chemical formulas.[31−33] The IR form of TQL Item should not be confused with the many different kinds of linear notations,[3−9] atom connectivity matrix,[10−15] graphical,[16−19] and topological representations[20−27] that have been developed for entering retrieval systems or executing searches with formula or fragments of chemical compounds—TQL differs from all other kinds of representations because it alone has operations that are not found in matrix algebra. For example, the Dugundji-Ugi topological model for Organic Chemistry uses matrix algebra to describe molecules.[23−27]

Table 2. Markush Override Codes: MCHOIC, MREPET, and MLOGIC Are Defined in the Text[a]

Code	Assigned Meaning
5	This code is always followed by the code 7, which stipulates the kind of data to be used (see MCHOIC). Code 5 is associated with, FRF, RF and Action in MREPET. If FRF = 0 then this occurrence does not apply to the first node screen. |RF| is the maximum number of times the operation designated 57 will be applied. Termination is determined by the values of FRF, RF and Action. The X, Y and Z are results of an operation and B is a blank.

FRF	*RF*	*Action*	Generates
0	-7	1	X, XX, XXX, XXXX, XXXXX, XXXXXX
1	-7	1	X, XX, XXX, XXXX, XXXXX, XXXXXX, XXXXXXX
0	-7	-1	X, BX, BBX, BBBX, BBBBX, BBBBBX
1	-7	-1	X, BX, BBX, BBBX, BBBBX, BBBBBX, BBBBBBX
0	7	1	XXXXXX, XXXXXX, XXXXXX, XXXXXX, XXXXXX, XXXXXX
1	7	1	XXXXXXX, XXXXXXX, XXXXXXX, XXXXXXX, XXXXXXX, XXXXXXX, XXXXXXX

Compound Applications	Generates
2/0/-7/-1/1/7/1	YZZZZZ, XYZZZZ, XXYZZZ, XXXYZZ, XXXXYZ, XXXXXY
3/0/-7/1/1/-7/-1/0/-7/1	Y, YZ, YZZ, YZZZ, YZZZZ, YZZZZZ, XY, XYZ, XYZZ, XYZZZ, XYZZZZ, XXY, XXYZ, XXYZZ, XXYZZZ, XXXY, XXXYZ, XXXYZZ, XXXXY, XXXXYZ, XXXXXY

6	This code can only occur between two occurrences of code 7 thus: 767. It designates the kind of logical operation, OR or AND (see MLOGIC), that is to be used on the data whose kind is specified by the entry in MCHOIC for code 7 (i.e. kernels, partition screens, kernel screen, screen termination and action in the next query screen).
7	This code is associated with MCHOIC and it designates the kind of data or operation (kernels, partition screen, kernel screen, screen termination and action in the next query screen) that is to be used.

[a] MREPET contains FRF, RF, and Action.

For the purposes of definition, consider the following rank N square array in which the elements are kernels.

$$\mathrm{IR_N} \equiv \begin{Vmatrix} K_{1,1} & K_{1,2} & .. & K_{1,r} & .. & K_{1,N} \\ K_{2,1} & K_{2,2} & .. & K_{2,r} & .. & K_{2,N} \\ . & . & .. & . & .. & . \\ . & . & .. & . & .. & . \\ . & . & .. & . & .. & . \\ K_{N-1,1} & K_{N-1,2} & .. & K_{N-1,r} & .. & K_{N-1,N} \\ K_{N,1} & K_{N,2} & .. & K_{N,r} & .. & K_{N,N} \end{Vmatrix}$$

An information representation, IR_N, is a square array in which three classes of kernels are identified, CO, CI, and CII:

CO contains all kernels that lie on the principal diagonal (i.e., all $K_{i,j}$ when $i = j$).

CI contains all kernels that lie above the principal diagonal (i.e., all $K_{i,j}$ when $i < j$).

CII consists of all kernels that lie below the principal diagonal (i.e., all $K_{i,j}$ when $i > j$).

All operations that do not reclassify the kernels or alter the relative positions of CI and CII kernels with respect to CO kernels are permitted. The following five transformation operators have been identified.

a. *Identity operator* (I): $I(\mathrm{IR}_N) = \mathrm{IR}^1_N$. This operator does not affect the position or value of any kernel.

Chart 1. Data Structures Associated with Markush Override Codes

OKDATA: contains the information for NOK occurrences of the GATE and NOT override codes in the TQL Item (Subsection I.3.1).

MCHOIC: There are MCL references to Markush code 7 in the query. $kind_i$ specifies the kind of data (0 - kernels; 1 - partition screens; 2 - kernel screens ; 3 - screen termination; or 4 - action in the next query screen) that is to be used.

MCL	$kind_1$	$kind_2$	....	$kind_{MCL}$

MKERNE: There are MKL references to different Markush structures in the query that use kernels (i.e. the entry in MCHOIC is 0). NK_i is the number of times the i^{th} Markush structure references a kernel. ker_{i1}, ker_{i2}, ker_{i3}, . . . are the values for the kernels.

MKL	NK_1	$ker_{1,1}$	. . .	$ker_{1,\alpha}$	....	NK_{MKL}	$ker_{MKL,1}$	.	$ker_{MKL,\delta}$

with $\alpha = NK_1$ and $\beta = NK_{MKL}$.

MLOGIC: designates the operator (0 - OR; 1 - AND), that is to be used for each of the MLL occurrences of the Markush code 6 in the query.

MLL	$Operator_1$	$Operator_2$	....	$Operator_{MLL}$

MPARTI: MPL is the number of Markush structures in the query that reference partition screens (i.e. the entry for MCHOIC is 1). NP_i is the number of references in the i^{th} Markush structure and the PS_{ij} specify the partition screens.

MPL	NP_1	$PS_{1,1}$	. .	$PS_{1,\gamma}$	.	NP_{MPL}	$PS_{MPL,1}$	. .	$PS_{MPL,\delta}$

with $\chi = NP_1$ and $\delta = NP_{MPL}$.

MREPET: contains the values for the *FRF - RF - Action* triplets associated with each of the MRL references to the repetition factor (Markush code 5) in the query. *FRF* (0 or 1), *RF* (-N or N) and *Action* (1 or -1) are the first repetition factor, the repetition factor and the action that is to be taken (*Table 2*).

MRL	FRF_1	RF_1	$Action_1$	. . .	FRF_{MRL}	RF_{MRL}	$Action_{MRL}$

MSCREE: MSL is the number of Markush structures in the query that reference kernel screens (i.e. the entry for MCHOIC is 2). NS_i is the number of references in the i^{th} Markush structure and the KS_{ij} specify the kernel screens.

MSL	NS_1	$KS_{1,1}$	. .	$KS_{1,\varepsilon}$	.	NS_{MSL}	$KS_{MSL,1}$	. .	$KS_{MSL,\phi}$

with $\varepsilon = NS_1$ and $\phi = NS_{MSL}$.

MTERMI: MTL is the number of times that Markush code 7 with an entry in MCHOIC of 3 was invoked. The SL_i are the lengths of the screens that were constructed in MWORKA.

MTL	SL_1	SL_2	....	SL_{MTL}

NTERMI: contains the lengths of the MST screens, *NSL*, that were constructed in NWORKQ from the new TQL Item.

MST	NSL_1	NSL_2	....	NSL_{MST}

MWORKA: contains the MTL screens, *SC*, that were constructed from the Markush codes.

MTL	SC_1	SC_2	....	SC_{MTL}

MWORKQ: contains the MST screens, *NSC*, that were constructed from the new TQL Item screen with the information related to the entries in MREPET with *Action* < *0*.

MST	NSC_1	NSC_2	....	NSC_{MST}

NODEAD: contains the NOL node address(es), *NA*, obtained from the executive pointer(s) whose screens match the screens that were constructed from the override codes.

NOL	NA_1	NA_2	....	NA_{NOL}

b. *Exchange operation* ($E_{i,j}$): $E_{i,j}$ (IR_N) = $IR^E{}_N$ for $1 \le i < j \le N$. Kernels $K_{i,i}$ and $K_{j,j}$ in CO are interchanged.

c. *Cyclic shift operator* ($C_{i,j,n}$): $C_{i,j,n}(IR_N) = IR^C{}_N$ with $1 \le (i,j) \le N$

n, whose sign determines the direction of the shift, must have an absolute value between *i* and *j*. The relative values of *i* and *j* determine which CO kernels are to be cyclically shifted.

d. *Reflection operator* ($R_{i,j}$): $R_{i,j}(IR_N) = IR^R{}_N$, with $1 \le i < j \le N$. In this operation the group of kernels in CO specified by *i* and *j* is reflected through its midpoint.

e. *Rotation operator* ($\Phi_{i,j}$): $\Phi_{i,j}(IR_N) \equiv IR^{\Phi}_N$ with $1 \le i \le j \le N$. This operator affects only the value of the kernel $K_{i,j}$ and then only in a way specified by the user.

The IR form of TQL Item is a linear encoding of an information representation like IR_N. There are three different kinds (diagonal, column and row) and they are identified by values of 1, 2, and 3 for the form specification, *F* in M

Table 3. Information Collected from IR4, Which Has One Ring (i.e., TNRINGS=1)[a]

bonds		moieties			bonds		moieties		
bonded pairs	bond value	rank	value	type	bonded pairs	bond value	rank	value	type
3,8	99	1	Br	T	9,5	99	9	N	R
3,10	99	2	Cl	T	5,10	99	10	N	R
3,11	12	3	P	R	5,7	13	11	C	I
3,12	12	4	P	R	11,13	12	12	C	I
8,4	99	5	P	R	11,14	12	13	H	T
4,9	99	6	O	T	11,15	12	14	H	T
4,1	12	7	O	T	12,16	12	15	H	T
4,2	12	8	N	T	12,6	13	16	H	T

ring	RC	RBP	SR	S&ETP	ETP	RBC	LBC
1	10−5−9−4−8−3 N−P−N−P−N−P	3, 4, 5	8	5−7, 12−6	4−1, 4−2	4−1, 4−2, 5−7, 3−11−13, 3−12−6	1,1,1,2,2

[a] Types T, R, and I denote terminal, ring and interior kernels. RC, RBP, SR, S&ETP, BC, and LBC are defined in the text.

(=*k*/*FD*/*lNFT*/*m*/*AS*, see subsection II.3.1 of ref 1). Except for the value of *F*, all three kinds contain exactly the same information. The diagonal form specification, $F = 1$, is defined as follows.

1. S_0 contains all the kernels in class CO; that is, all the kernels in the principal diagonal.

2. The CI kernels, which lie above the principal diagonal in IR_N, are distributed diagonal by diagonal in screens S_1 to S_{N-1}. Thus: $S_l = K_{1,2}K_{2,3}...K_{N-1,N}$ and S_{N-1} is $K_{1,N}$.

3. The CII kernels, which lie below the principal diagonal in IR_N, are distributed diagonal by diagonal in screens S_N to S_{2N-2}. Thus $S_N = K_{2,1}K_{3,2}...K_{N,N-1}$ and S_{2N-2} is $K_{N,1}$.

The diagonal form specification is

$$\mathrm{TQLI_{IR}} \equiv (\mathrm{M/J}/S_0/S_1/S_2/S_3/..../S_{N-1}/.../S_{2N-3}/S_{2N-2}\ \mathrm{X})$$

$$\equiv (\mathrm{M/J}/K_{1,1}K_{2,2}K_{3,3}...K_{N,N}/..../K_{1,N}/K_{2,1}...K_{N,N-1}/.../K_{N,1}\ \mathrm{X})$$

The reserved system codes are discussed in subsection I.3. Normalization is described in subsection 1.4.

I.3. Special Kernel Codes. Nine integer values for kernels, 0−8, are reserved for exclusive system use. Zero indicates a null or "no information" kernel. Eight specifies the location of a dynamic security lock that will invoke an interrogation routine (section II.4 of ref 1). The other seven integer values, 1−7, are called override codes.

Override codes can be used to specify uncertainty in (a) a query TQL Item of the values for either the partition constant (*m* in M(=*k*/*FD*/*INFT*/*m*/*AS*)) or the normalization constant (*N* in M); (b) the values and/or locations of kernels in kernel screens; (c) clusters or nested information representations in kernel screens; (d) the values of kernel screens; or any combination of these. Normally override codes are used to formulate nonexplicit retrieval queries. They can also be used in storage operations to define unique explicit paths that describe complex nonexplicit queries, like the profiles for patrons that are commonly maintained in research libraries. Security locks can only be overridden with privileged operations.

Two kinds of override codes, Conventional and Markush, are considered in subsections I.3.1 and I.3.2, and the rules for using them are described in section I.3.3.

I.3.1. Conventional Override Codes (1, 2, 3, 4). The four conventional codes (Table 1) are used to specify uncertainty in values for kernels in kernel screens, the S_i in TQL Items, and *N* and *m* in M(=*k*/*FD*/*INFT*/*m*/*AS*). The so-called GATE, 3, and the NOT, 4, override codes are associated with data-structure OKDATA that contains

NOK	OKV_1	KV_{11}	KV_{12}	.	$KV_{1\gamma}$	.	OKV_{NOK}	$KV_{NOK,1}$	.	$KV_{NOK,\eta}$

with $\gamma = OKV_1$ and $\eta = OKV_{NOK}$. NOK is the number of occurrences of the GATE and NOT overrides in the TQL Item. The *KV*s can be either explicit values or conventional override codes. For the GATE override: OKV_i is either the number of pairs of kernels ($OKV_i > 0$) or the number of kernels ($OKV_i < 0$) that together define the acceptable values. If $OKV_1 > 0$ then the first pair is KV_{11}-KV_{12}, and it means that all values from KV_{11} through KV_{12} are acceptable. For the NOT override, OKV_i is either the number of pairs of kernels ($OKV_i > 0$) or the number of kernels ($OKV_i > 0$) that together define the unacceptable values (see Table 1).

I.3.2. Markush Override Codes (5, 6, 7). Markush structures are occluded substructures. The simplest example is obtain all English words that begin with "th" and end with an "e" (the, thee, there, these, therefore, ...). The three Markush override codes are used to describe a Markush structure, e.g., 577. These codes, found only in Transparent Query Language Items, can be used to specify uncertainty in the location of (a) kernels in kernel screens, (b) partitions or nested information representations in kernel screens, and (c) kernel screens themselves. In combination with conventional override codes they can be used to specify uncertainty in both the values and positions of groups of kernels.

A preprocessor eliminates all Markush override codes in a Transparent Query Language Item, TQL Item, by replacing them with screens (the *M*, *J*, and S_i) that contain no Markush override codes. In effect a Transparent Query Language Item query that contains Markush structures normally generates many Transparent Query Language Items that do not contain Markush codes. The TQL Items generated by the preprocessor are used to trace/propagate information paths in *RFILE* by matching the appropriate query screen to the node screens in the substations in *RFILE* (subsection IX.5.8.1 of ref 1). A summary of the Markush override codes is given in Table 2.

Two flags, node (NODEFL) and screen (SCREFL), and the twelve data-structures (listed in Chart 1) are associated

with the three Markush override codes. SCREFL is used to indicate operations that are to be executed in the next TQL Item screen.

I.3.3. Rules for Using Override Codes. The conventional override codes, which specify uncertainty in the values for kernels, are easily used in both retrieval and storage operations by entering them directly into the TQL Item screens. They will not be considered further here.

The Markush override codes are also entered in the query screens. However, they must be converted to screens that contain only kernels and conventional codes. The rules for using Markush codes are as follows.

(1) Code 5 must always be followed by code 7. *Action* specifies the current (>0) or the next (<0) TQL Item screen.

(2) Code 6 can only appear between two occurrences of code 7 thus: 767 or 5767.

(3) There is no restriction on the use of code 7.

I.4. Manipulation of Transparent Query Language Items. There are 15 operations that can be used with Transparent Query Language (TQL) Items and many of them have three modes: Logical, Variable Precision, and Variable Integer (subsection II.3.2 and especially Table 3 in ref 1). They include all the conventional matrix operations such as transpose and multiply; operations that combine two or more TQL Items or expand/contract a single TQL Item; and some special operations needed to change, insert, or extract values in the control information (M and J) and kernels.

There are also two operations that are unique to the Transparent Query Language. They are as follows:

1. Normalization produces an uniquely defined TQL Item called the normal form.

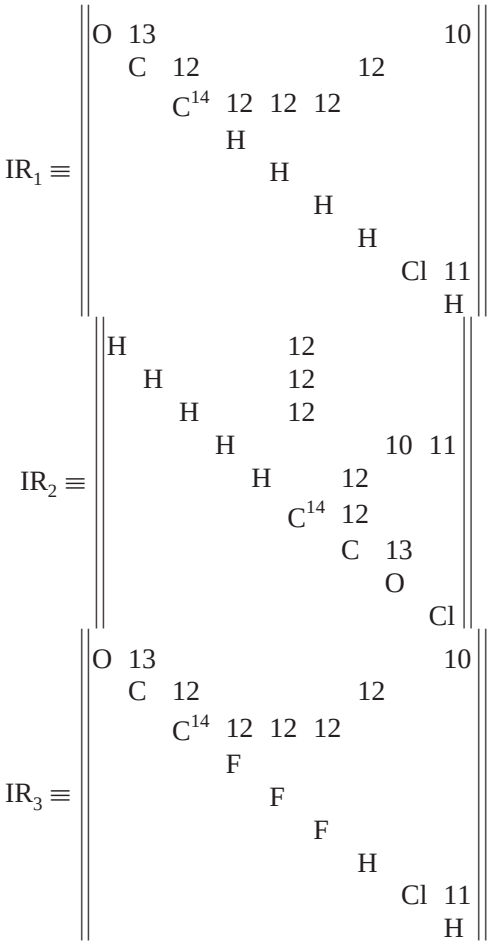
2. Mobile Canonicalization converts a normalized TQL Item to one or more of its mathematically equivalent forms.

Normalization and Mobile Canonicalization are discussed in turn in the next two subsections.

I.4.1. Normal Form of Transparent Query Language Items. An Information Representation of rank N can be rearranged in $N^2!$ different ways. There are $N^2!/N!$ different classes, each with $N!$ mathematically equivalent members that can be transformed into one another by using the interchange, cyclic shift, and reflection operations (subsection I.2 and details in ref 1). Because the three forms of information representation, specified with $F = 1$, 2, and 3 in M, are easily converted into one another, only the diagonal form, specified by $F = 1$, will be considered here.

There are 11 different forms of normalization, specified by the Normalization constant N in the control information in M (section II.2 and especially Table 2 in ref 1), and the form is determined by both the values for the control information and the values for kernels. Here only the fully normalized form, with $N = 10$ and $m = 1$, will be considered. An alternative definition of a fully normalized TQL item, TQLI, is that the logical value of the expression: M/J/S_1/S_0/S_2/S_3/..../S_{N-1}/.../S_{2N-3}/S_{2N-2}, with the kernel screens in that order, must be a maximum. While normalization has been defined in general terms for use with all three kinds of graphical representations (simple, information representation, and matrix), only information representations will be considered in this paper.

For illustrative purposes consider the following three rank 9 topological maps for aldehydes that are not partitioned. IR_1 and IR_2 are different forms of $C^{14}H_3CHO.HCl$. IR_3 is a form of $C^{14}F_3CHO.HCl$. The numbers 13, 12, 11, and 10



designate double covalent, single covalent, ionic, and weak (van der Waal forces) bonds, respectively, and C^{14} designates a radioactive isotope of carbon.

The linear diagonal form (specification $F = 1$ in M) for IR_1, IR_2, and IR_3 with terminal zero digits omitted are

$TQLI_1 \equiv$ (M/J/O.C.C.^{14}H.H.H.H.Cl.H/
13.12.12.0.0.0.0.11/0.0.12/0.0.12/0.12/0/0/0/10 X)

$TQLI_2 \equiv$ (M/J/H.H.H.H.H.C.^{14}C.O.Cl/0.0.0.0.0.12.13/
0.0.0.0.0.12/0.0.12/0.12.0.10/12.0.0.11 X)

$TQLI_3 \equiv$ (M/J/O.C.C.^{14}F.F.F.H.Cl.H/
13.12.12.0.0.0.0.11/0.0.12/0.0.12/0.12/0/0/0/10 X)

$TQLI_1$, $TQLI_2$, and $TQLI_3$ each have 362 880 equivalent forms. The unique normal forms are

$NTQL_1 \equiv$ (M/J/Cl.H.O.C.C.^{14}H.H.H.H/11.10.13.12.12/
0.0.0.0.12.12/0.0.0.0.0.12 X)

$NTQL_2 \equiv$ (M/J/Cl.H.O.C.C.^{14}H.H.H.H/11.10.13.12.12/
0.0.0.0.12.12/0.0.0.0.0.12 X)

$NTQL_3 \equiv$ (M/J/Cl.H.O.C.C.^{14}F.F.F.H/
11.10.13.12.12/0.0.0.0.12.12/0.0.0.0.0.12 X)

Clearly $NTQL_2$ is an equivalent form of $NTQL_1$. $NTQL_3$ is an analogue of $NTQL_1$.

I.4.2. Mobile Canonical Forms of Transparent Query Language Items. The Mobile Canonical Algorithm is transparently invoked when a normalized TQL Item contains conventional override codes (1, 2, 3, and 4) to generate all possible acceptable forms of the TQL Item. For example, the normal form for acetylene that has one hydrogen atom replaced by deuterium (CDCH) is TQL ≡ (M/J/D.C.C.H/12.14.12 X), with 12 and 14 specifying single and triple covalent bonds, respectively.

The query: "*List all compounds of acetylene that contain deuterium and any other covalent bonded moiety in place of the hydrogen atom*" generates

$$TQL_{REQ1} \equiv (M/J/D.C.C.1/12.14.12\ X)$$

$$TQL_{REQ2} \equiv (M/J/1.C.C.D/12.14.12\ X)$$

with 1 specifying any moiety. Other examples of its use are shown in subsections IX.4A.3 and IX.4B in ref 1.

II. PROCESSING PROCEDURE

In the preparatory steps of this procedure each TQL Item, when encountered in a queue, is processed as follows.

(a) The TQL Item is copied into a work area TQLWORK.

(b) Markush codes are expanded. This results in several TQL Items that contain no Markush override codes. Suppose that N_{TQLI} TQL Items were generated.

(c) Each TQL Item in TQLWORK is converted to its normal form(s), as defined in section III of ref 32 and especially subsection II.2.2 of ref 1, with the improved Chain Tracing Algorithm that is described in section III. The role of the Mobile Canonical Algorithm[32] has been changed. It is now transparently invoked for both retrieval and storage queries by the improved chain tracing algorithm if there are any conventional override codes in the TQL Item. That means that a TQL Item with conventional override codes can have more than one normal form.

(d) If a storage operation, i.e., MODE=1, is to be performed, then all the TQL Items in TQLWORK that were generated in step (b) are combined to form a single TQL Item.

At this point TQLWORK will contain one normalized TQL Item for MODE=1 and for MODE=0 if there are no override codes. Suppose there are TN_{TQLI} TQL Items in TQLWORK.

If there is only one TQL Item in TQLIWORK, it is used to trace/propagate an information path in the simulated communications network, *RFILE*, in the normal way. The procedure to trace paths in retrieval operations when override codes are present is described next. Each TQL Item is processed in turn. The procedure begins with the first screen in the first TQL Item. The goal is to determine the address-(es) of the nodes that are to be examined with the next screen in the TQL Item. The address(es) of the nodes that are to be examined are maintained in the pop-up stack NODEAD. This procedure is repeated until either all the screens in the TQL Item are successfully processed or one of its screens yields no node addresses, and then the next TQL Item in TQLIWORK is processed. When all TQL Items have been processed, the retrieved registry numbers are displayed.

(e) Set the TQL Item counter, NTQLC, to zero.

(f) Set NTQLIC=NTQLIC+1. If NTQLIC>TN_{TQLI} all the TQL Items have been processed; the retrieved registry numbers are displayed, and the search procedure is terminated.

(g) Store the address of the node associated with the first screen of the NTQLCth TQL Item in NODEAD; set NOL=1 and NNOL=0.

(h) If NOL=NNOL=0 all nodes have been processed. Go to (f). If NOL=0 set NOL=NNOL and NNOL=0. Pop stack NODEAD and set NOL=NOL-1.

(i) Copy the TQL Item screen that is to be examined from TQLIWORK to NWORKQ and set MST=0.

(j) Do a binary search of the node being examined for an executive pointer whose screen matches the TQL Item screen in NWORKQ. If such an executive pointer is found, add the address of the new node (obtained from the executive pointer) to stack NODEAD, set NNOL=NNOL+1 and then go to (h).

(k) For storage operations construct a new executive pointer from the screen in NWORKQ, insert it in the node, create a new node, add its address in NODEAD, and set NNOL=NNOL+1, then go to (h).

(l) For retrieval operations determine if any of the four conventional override codes are present in the TQL Item screen. If there are none go to (h).

(m) The node is sequentially searched for all executive pointers whose screens match all possible values for the screen in NWORKQ. Each node address in a successful match, obtained from the executive pointer, is added to NODEAD and NNOL=NNOL+1. Next execute (h).

It should be noted that the conventional override codes can be entered into a TQL Item screen in two different ways: (1) directly into the screen thus: $S_i = K_{1i}K_{2i}\ 2\ K_{4i}...\ K_{\gamma i}$, which means that any value is acceptable for the third kernel or (2) by using the Markush code 7 thus: $S_i = K_{1i}K_{2i}\ 7\ K_{4i}...\ K_{\gamma i}$, with the entries for MCHOIC and MKERNE equal to 0 and 2, respectively.

The Markush codes are converted to screens that contain only kernels and conventional override codes. That means, because conventional override codes are examined only when a mismatch occurs, they can be used in storage operations to propagate a single explicit path to describe a complex query that is normally answered by an exhaustive search, which involves tracing many paths. A disadvantage is that new paths (propagated after the single explicit path is created) that also satisfy the request will not be found. However, this problem can be completely circumvented by periodically executing the second option, (2) above.

A better method would be to propagate all such single explicit paths with the file designator, *FD* in M, set to a reserved value, e.g., 9999. Periodic updating with option (2) would be performed by the Database Manager. Retrieval requests generated from Markush codes would be converted to their unique explicit form and then used to access the file with $FD = 9999$.

III. NORMALIZATION PROCEDURE

A single command, NORMAL (TQLI,MODE), is invoked to normalize every Transparent Query Language Item, TQLI. MODE specifies the kind of operation (retrieval (0), storage (1), or delete (2)) that is to be performed with the normalized

TQL Item. NORMAL extracts the normalization constant (N in M), the partitioning factor (m in M), and the sizes of the partitions (r_1, r_2, ..., r_m in J) from the TQL Item and then selectively uses the Chain Tracing Algorithm to obtain the normal form specified by the value of N (Table 2 in ref 1). Consideration of CII connectors has been omitted because they qualify the meaning of CI connectors (e.g., in chemistry they can be used to specify stereoisomers). The Mobile Canonicalization Algorithm[1] is transparently invoked by the chain tracing algorithm. The only restriction is that, for all $1 \leq i \leq N-1$ for a TQL Item of rank N, kernels in screen S_{N+i-1} can be only nonnull if the corresponding kernels in screen S_i are not null.

The chain tracing algorithm that is described below correctly normalizes *any* information representation (or topological map) that contains any number of different formulas, each with any combination of branches and/or rings which do not have internal rings. Rings can be heterocyclic. Internal rings are contained within rings, i.e., the anthracene structure, which has 14 carbon atoms, contains a ring with 10 carbon atoms that contains a ring with six carbon atoms. This version of the chain tracing algorithm replaces that described in ref 1. The problem of internal rings will be discussed in part III of the series "*High-Speed Manipulation of Information Representations*".[34]

For illustrative purposes suppose that a TQL Item is not partitioned − that is m (in M) = 1 and r_m (in J) = the rank of the TQL Item. This means that the new chain tracing algorithm will be invoked only once.

III.1. Chain Tracing Algorithm. The unique form for any information representation kind of TQL Item can always be determined by applying the kernel Interchange, Shift, and Reflection rules to obtain all possible mathematically equivalent forms and then selecting the one that has a maximum logical value. However, this approach is not practical because even comparatively small formulas have many equivalent forms. The role of the new chain tracing algorithm is to determine the normal form(s) in an economically viable fashion.

The bases of the new chain tracing algorithm are the following observations about the C0 kernels in an information representation. In chemical formulas C0 kernels designate atoms, moieties, ions, or free radicals; CI kernels designate bonds, and CII kernels can be used to designate stereoisomers.

(1) All C0 kernels can be classified as bonded or null. Bonded kernels are bonded to one or more kernels. Null kernels are not bonded.

(2) Bonded kernels can be classified by their number of bonds. Terminal and Interior Kernels have one and more than one bond, respectively. Branch points are kernels that are bonded to more that one interior kernel. Rings contain only interior kernels.

(3) The total number of rings in a formula (TNRINGS) is determined by its number of bonded moieties (NBMOIE) and bonds (NBONDS) thus: TNRINGS = MBONDS− NBMOIE +1.

(4) Rings are identified by the repeated occurrence of a ring kernel in a chain.

(5) For formulas with no rings:

(a) A primary chain has one terminal kernel at each end and can contain any number of interior kernels between them.

(b) Branch chains begin at a Branch Point, which is an interior kernel in a primary chain, and end with a terminal kernel.

(6) For formulas with a ring:

(a) Membership in a ring and the ring's character (e.g., heterocyclic) can be determined by tracing.

(b) Ring branch points are recognized as ring members that are attached to kernels that are not in the ring. Ring branch chains start at a branch point and end with a terminal kernel. The lengths of ring branch chains can be determined by tracing a terminal kernel to the nearest ring branch point.

(c) The starting point in the ring can be determined from the lengths of the ring branch chains and the locations of their ring branch points.

(d) A Start Terminal Pair is a terminal kernel attached to an interior kernel or to a ring branch point that has only one terminal kernel.

(e) End Terminal Pairs are recognized when a ring branch point has more than one terminal kernel.

(f) Primary chains can begin or end in a ring or a terminal pair.

(7) A composite chain describes the normal form of an information representation, which can contain more than one formula. It is constructed from one primary chain for each formula by adding unused kernels (see below).

The new chain tracing algorithm uses information about the following.

(i) the ranks of all kernels with the highest first. Kernels are numbered in the order that they appear in the IR. Equal kernels are ranked in their order of appearance (e.g., 10, 99, 77, 88, 77, 77 will be ranked as 6, 1, 3, 2, 4, 5, with 3−5 designating the occurrences of 77 in their order of appearance).

(ii) every null kernel, every bonded pair of kernels and the values for every kernel and bond.

(iii) classification of every kernel as null, terminal, or internal.

(iv) branch points.

(v) rings.

(i) and (ii) are collected by scanning the bond values for each C0 kernel in turn in the order that it appears in the Information Representation. (iii), (iv), and (v) are computed from the information for bonded pairs.

To illustrate the processing of formulas with single rings consider the following information representation for a derivative of trimeric phosphazene, IR_4.

For the trimeric phosphazene compound {P(CH$_3$, CHO)NP(Cl, Br)NP(O)N}:

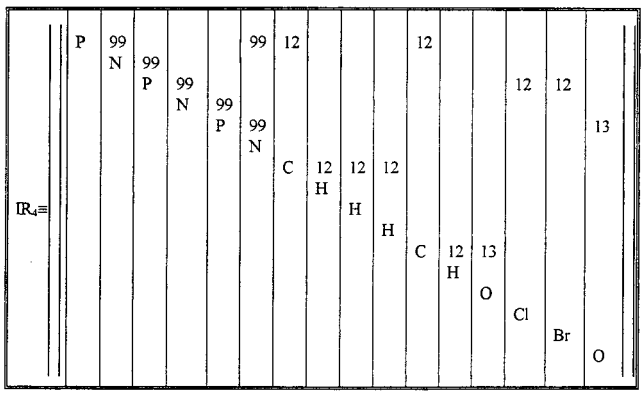


Table 4. Information Collected from IR_5, and IR_6[a]

IR_5					IR_6				
bonds		moieties			bonds		moieties		
bonded pairs	bond value	rank	value	type	bonded pairs	bond value	rank	value	type
1,5	13	1	O	I	4,7	13	1	Br	T
1,13	10	2	O	I	4,15	10	2	Cl	T
5,8	12	3	N	T	7,10	12	3	F	T
5,4	12	4	C^{14}	I	7,6	12	4	O	I
4,9	12	5	C	I	6,11	12	5	O	I
4,6	12	6	C	I	6,8	12	6	C^{14}	I
4,7	12	7	C	I	6,9	12	7	C	I
6,10	12	8	H	T	8,12	12	8	C	I
6,11	12	9	H	T	8,13	12	9	C	I
6,12	12	10	H	T	8,14	12	10	H	T
7,3	14	11	H	T	9,1	12	11	H	T
13,2	12	12	H	T	9,2	12	12	H	T
14,2	12	13	H	I	9,3	12	13	H	T
		14	H	T	15,5	12	14	H	T
					16,5	12	15	H	I
							16	H	T
rings	BP	S&ETP			rings	BP	S&ETP		
0	4	14,2; 6,10; 7,3			0	6	16,5; 9,1; 8,12		

[a] There are no rings (TNRINGS=0). Bonded pairs are designated by the ranks of the moieties. I and T mean interior and terminal kernels, respectively. Branch points, BP, are recognized as interior kernels that are bonded to at least two other interior kernels. Start and End terminal pairs (S&ETP), which are used to begin tracing and to end a primary chain, contain one terminal kernel. In interior kernels that have more than one terminal, the terminal kernels are ordered left to right by their bond values and moiety values (in that order).

Table 3 contains the information for IR_4, which contains one ring (TNRINGS=1). The symbols mean:

(a) The Ring Configuration (RC) is determined by locating the CI connector that joins the last atom (N with rank 10) to the first atom (P with rank 3) and then tracing the ring.

(b) Ring Branch Points, RBP, are members of the ring that are bonded to nonring moieties.

(c) The Start in Ring, SR, is the location in the ring that ensures a primary chain can end with the longest branch chain. It is used to trace primary chains clockwise and anticlockwise round the ring.

(d) Start and End Terminal Pairs (S&ETP) are terminal pairs that can be used to start and end primary chains. They contain either a single terminal kernel attached to a ring branch point or one of several terminal kernels that are attached to an interior kernel that is not part of the ring. For branch chains that have more than one terminal kernel, the one with either the highest bond value or lowest rank (in that order) is used.

(e) End Terminal Pairs, ETP, contain one of several terminal kernels that are attached to a ring branch point. They can only be used to terminate primary chains.

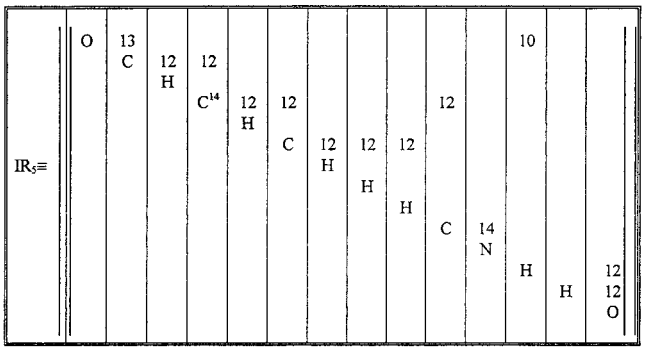
(f) A Ring Branch Chain, RBC, is a chain that begins at a ring branch point and ends with a terminal kernel.

(g) The Length of a Branch Chain, LBC, is the number of kernels in a ring branch chain that is not part of the ring.

Ring branch chains (RBC) and their Lengths (LBC) are determined by tracing each ring branch chain from its terminal kernel to its ring branch point. Primary chains are generated in the information representation for every different formula that has rings, TNRINGS>0. They begin with the start in ring (SR) or start & end terminal pairs (S&ETP), tracing both clockwise and anticlockwise. The end is an end terminal pair (ETP) or an unused SR or S&ETP.

To illustrate the application of the new chain tracing algorithm to formulas without rings consider the information representations for two different aldehydes in water, IR_5 and IR_6, below.

For the acetylene compound {CH_3C^{14}(H, CN) CHO.H_2O}:



For the acetylene compound {C^{14}(CH_3, H, C(F, Cl, Br)) CHO.H_2O}:

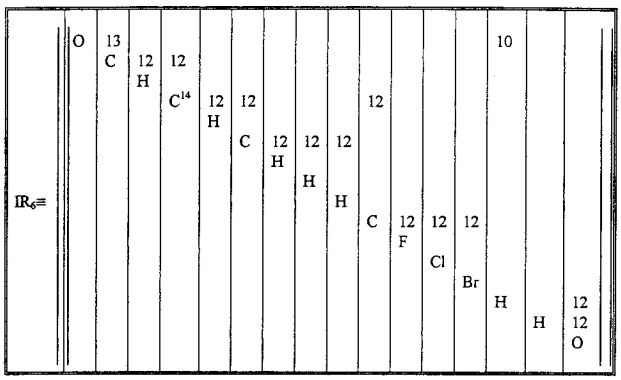


Table 4 contains the information for IR_5 and IR_6. There

are no rings because TNRINGS=0. Each formula has one branch point and three candidate starting terminal pairs.

All possible primary chains are generated for every formula in the information representation that has no rings by beginning and ending with a start & end terminal pair.

III.2. Implementation of the Chain Tracing Algorithm. The step-by-step application of the chain tracing algorithm uses the information in Tables 3 and 4, generated by scanning the information representation, to obtain the normal form thus:

Step (i) Generate all possible primary chains for every formula in the information representation.

Step (ii) Select the longest primary chain that has the highest bond values or the lowest ranked moieties (in that order) for each different formula.

Step (iii) The selected primary chains are ordered with their lengths decreasing. Chains with the same length are ordered with the values of the bonds decreasing or the ranks of their moieties increasing (in that order).

Step (iv) Each selected primary chain is expanded by appending its unused interior, ring, and terminal kernels (with their ranks increasing) in their reverse order of occurrence in the primary chain (see Example 1 below).

Step (v) The composite chain for the information representation is obtained by combining the expanded primary chains in their ranked order and then appending any null kernels with their ranks increasing.

Step (vi) The normal form for the information representation is obtained by applying the kernel interchange rule to convert input order of the kernels to that specified by the composite chain.

Throughout this section a, b, and c designate a primary chain, its bond and moiety values, respectively. Examples are as follows:

Example 1 (Derivative of Trimeric Phophazene in IR_4):

Step (i). The 17 primary chains generated for IR_4 are shown below:

```
41a. 8 - 4 - 9 - 5 - 10 - 3 - 12 - 6
  b.   99  99  99  99  99  12  13
  c.    N   P   N   P   N   P   C   O
42a. 6 - 12 - 3 - 10 - 5 - 9 - 4 - 1
  b.   13  12  99  99  99  99  12
  c.  O   C   P   N   P   N   P   Cl
43a. 6 - 12 - 3 - 8 - 4 - 9 - 5 - 10
  b.   13  12  99  99  99  99  99
  c.  O   C   P   N   P   N   P   N
44a. 6 - 12 - 3 - 8 - 4 - 9 - 5 - 7
  b.   13  12  99  99  99  99  13
  c.    O   C   P   N   P   N   P   O
45a. 7 - 5 - 9 - 4 - 8 - 3 - 12 - 6
  b.     13  99  99  99  99  12  13
  c    O   P   N   P   N   P   C   O
```

```
46a. 6 - 12 - 3 - 8 - 4 - 9 - 5 - 10
  b.     13  12  99  99  99  99  99
  c.    O   C   P   N   P   N   P   N
47a. 8 - 4 - 9 - 5 - 7
48a. 8 - 4 - 1
49a. 8 - 3 - 10 - 5 - 9 - 4 - 1
410a. 8 - 3 - 10 - 5 - 7
411a. 8 - 3 - 12 - 6
412a. 6 - 12 - 3 - 10 - 5 - 7
413a. 6 - 12 - 3 - 8 - 4 - 1
414a. 7 - 5 - 10 - 3 - 8 - 4 - 1
415a. 7 - 5 - 10 - 3 - 12 - 6
416a. 7 - 5 - 9 - 4 - 8 - 3 - 10
417a. 7 - 5 - 9 - 4 - 1
```

Step (ii). Primary chain 41 is selected because it is the longest chain with the highest values for its bond configuration (99, 99, 99, 99, 99, 12, 13).

Step (iii) does not apply because there is only one formula in the information representation.

Step (iv). The expanded form for primary chain 41 is

8−4−9−5−10−3−12−6−16−11−13−14−15−1−2−7

A terminal hydrogen atom, rank 16, is bonded to the carbon atom ranked 12. The carbon atom ranked 11 is bonded to the ring branch point ranked 3. 13, 14, and 15 are the ranks of the hydrogen atoms bonded to the carbon atom ranked 11. 1−2 specify that bromine and chlorine are both bonded to the ring branch point ranked 4. The last entry, 7, specifies that an oxygen atom, ranked 7, is connected to the ring branch point ranked 5.

Step (v) does not apply because there are no null kernels (nonbonded moieties).

Step (vi). The linear normal form of the derivative of trimeric phosphazene shown in IR_4 is

$$\begin{aligned}\text{NTQLI}_4 \equiv (&\text{M/J/N.P.N.P.N.P.C.O.H.C.H.H.H.Br.Cl.O/}\\ &\text{99.99.99.99.99.12.13.0.0.12/0.0.0.0.0.0.12.0.0.12/}\\ &\text{0.0.0.0.0.0.0.0.0.12/0.0.0.0.0.12/99/0/0/0/0/}\\ &\text{0.0.0.12/0.0.0.12/0/0/13 X)}\end{aligned}$$

The original linear form of IR_4 was

$$\begin{aligned}\text{TQLI}_4 \equiv (&\text{M/J/P.N.P.N.P.N.C.H.H.H.C.H.O.Cl.Br.O/}\\ &\text{99.99.99.99.99.0.12.0.0.0.12/0/0/99/12/0/0/0/12/}\\ &\text{0.0.12.0.12/0.0.12 X)}\end{aligned}$$

Example 2 (CH_3C^{14}(H, CN) CHO·H_2O in IR_5):

The six primary chains generated for IR_5 are shown below:

```
51a. 14 - 2 - 13 - 1 - 5 - 4 - 6 - 10;
  b.    12  12  10  13  12  12  12;
  c. H   O   H   O   C  C14  C   H;
52a. 14 - 2 - 13 - 1 - 5 - 4 - 7 - 3;
  b.    12  12  10  13  12  12  14;
  c. H   O   H   O   C  C14  C   N
53a. 10 - 6 - 4 - 5 - 1 - 13 - 2 - 14
  b.    12  12  12  13  10  12  12
  c. H   C  C14  C   O   H   O   H
```

```
54a. 3 - 7 - 4 - 5 - 1 - 13 - 2 - 14
  b.    14  12  12  13  10  12  12
  c. N   C  C14  C   O   H   O   H
55a. 3 - 7 - 4 - 6 - 10
56a. 10 - 6 - 4 - 7 - 3
```

Primary chain 54 was selected because it has the highest values for its bonds. The composite chain for IR_5, which is the expanded form of that primary chain, is

3−7−4−5−1−13−2−14−8−6−10−11−12−9

and the linear normal form of the information representation is

$$\begin{aligned}\text{NTQLI}_5 \equiv (&\text{M/J/N.C.C.}^{14}\text{C.O.H.O.H.H.C.H.H.H.H/}\\ &\text{14.12.12.13.10.12.12.0.0.12/0.0.0.0.0.0.0.0.0.12/}\\ &\text{0.0.0.0.0.0.0.0.0.12/0/0.0.0.0.12/0/0.0.12/0/0/0/0.0.12 X)}\end{aligned}$$

The original linear form of IR_5 was

$$\begin{aligned}\text{TQLI}_5 \equiv (&\text{M/J/O.C.H.C.}^{14}\text{H.C.H.H.H.C.N.H.H.O/}\\ &\text{13.12.0.12.0.12.0.0.0.0.0.0.12/}\\ &\text{0.12.0.12.0.12.0.0.0.0.0.12/0.0.0.0.0.0.12/}\\ &\text{0/0/0.0.0.12/0/0/0/10 X)}\end{aligned}$$

Example 3 (C^{14}(CH_3, H, C(F, Cl, Br)) CHO·H_2O in IR_6):

The six primary chain generated for IR_6 are shown below:

```
61a. 16 - 5 - 15 - 4 - 7 - 6 - 8 - 12
  b.    12  12  10  13  12  12  12
  c. H   O   H   O   C  C14  C   H
62a. 16 - 5 - 15 - 4 - 7 - 6 - 9 - 1
  b.    12  12  10  13  12  12  12
  c. H   O   H   O   C  C14  C   Br
63a. 1 - 9 - 6 - 7 - 4 - 15 - 5 - 16
  b.   12  12  12  13  10  12  12
  c. Br  C  C14  C   O   H   O   H
```

```
64a. 12 - 8 - 6 - 7 - 4 - 15 - 5 - 16
  b.     12  12  12  13  10  12  12
  c.    H   C  C14  C   O   H   O   H
65a. 1 - 9 - 6 - 8 - 12
66a. 12 - 8 - 6 - 9 - 1
```

Primary chain 63 is the basis for the composite chain because the ranked values for its moieties are lower than those for chain 61, 62, or 64. The composite chain for IR_6 is

1−9−6−7−4−15−5−16−10−8−12−13−14−11−2−3

The linear normal form for IR_6 is

$$NTQLI_6 \equiv$$
(M/J/Br.C.C.^{14}C.O.H.O.H.H.C.H.H.H.H.Cl.F/
12.12.12.13.10.12.0.0.12/0.0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.0.12/0/0.0.0.12/0/0/0/0.0.12/0/0.12/0.12 X)

The original linear form of IR_6 was

$$TQLI_6 \equiv$$
(M/J/O.C.H.C.^{14}H.C.H.H.H.C.F.Cl.Br.H.H.O/
13.12.0.12.0.12.0.0.0.12.0.0.0.0.12/
0.12.0.12.0.12.0.0.0.12.0.0.0.0.12/
0.0.0.0.0.12.0.0.0.12/0/0/0.0.0.12/0/0/0/0/0/0/10 X)

IV. ILLUSTRATIONS

At this point it should be noted that the characteristics of the High-Speed General Information Management System, HSGIMS, relevant to this paper are as follows:

(1) It is capable of supporting any query language (Section IV of ref 1).

(2) It is Information (or data) independent and Question type (or logical data) independent.[35]

(3) Within the available computer resources, there is no restriction on the number or type or size(s) of databases that can be simultaneously managed (subsection II.3.1 of ref 1).

(4) Both retrieval and update operations have very small and bounded times. In fact the observed in-core maximum search time for queries with up to 45 kernels screen and three degrees of specificity is near 0.0001 s (subsections VIII.B and C of ref 2.

(5) It has a fool proof system that can be easily used to provide for the security of information and protection against all intruders (section II.4 of ref 1).

(6) It economically and efficiently uses primary and secondary memory and data communications systems (subsection IVB of ref 2).

To illustrate the use of the Markush override codes two different applications, an English language spelling-checker and chemical formulas, are considered in turn next.

IV.1. English Language Spelling-Checker. For illustrative purposes consider an English language spelling-checker with a database that was created with TQLI with one character per kernel screen thus: $TQLI_{SC} \equiv (M_{SC}/J_{SC}$/t/h/e/r/e D). The terminal D, which signifies that there is no referenced data in MFILE (Figure 1 in ref 1), is a registry number that could contain grammatical information, illustrations of its use, and/or the meaning of the word "there".

Two examples of the use of Markush codes are given next. Because the TQLI are in the simple form they are not normalized.

Example 1. *Retrieval Query (MODE=0)*: *All words with up to nine characters that begin with "t"* and "h", have an "r" or "s" and terminate with an "e".

$$TQLI_{REQ0} \equiv (M_{SC}/J_{SC}/t/h/577577577e\ D) \text{ with}$$
$$MCHOIC \equiv 6/0/3/0/3/0/3;$$
$$MKERNE \equiv 3/1/3/1 \quad OKDATA \equiv 2/r/s;$$
$$\text{and } MREPET \equiv 3/0/-7/1/1/-7/-1/0/-7/1$$

This means that for the first (0/−7/1) and third (0/−7/1) references to code 5 the applications will not occur (in case there are words that begin with "thr" or "ths" or end with "re" or "se"). The first reference will stop when "r" or "s" is inserted. The second reference to code 5 stops when there are no further screens to generate. This application of the Markush codes generates 21 retrieval requests:

$$TQLI_{REQ01} \equiv (M_{SC}/J_{SC}/t/h/3/e\ D)$$
$$TQLI_{REQ02} \equiv (M_{SC}/J_{SC}/t/h/3/1/e\ D)$$
$$TQLI_{REQ03} \equiv (M_{SC}/J_{SC}/t/h/3/1/1/e\ D)$$
$$TQLI_{REQ04} \equiv (M_{SC}/J_{SC}/t/h/3/1/1/1/e\ D)$$
$$TQLI_{REQ05} \equiv (M_{SC}/J_{SC}/t/h/3/1/1/1/1/e\ D)$$
$$TQLI_{REQ06} \equiv (M_{SC}/J_{SC}/t/h/3/1/1/1/1/1/e\ D)$$
$$TQLI_{REQ07} \equiv (M_{SC}/J_{SC}/t/h/1/3/e\ D)$$
$$TQLI_{REQ08} \equiv (M_{SC}/J_{SC}/t/h/1/3/1/e\ D)$$
$$TQLI_{REQ09} \equiv (M_{SC}/J_{SC}/t/h/1/3/1/1/e\ D)$$
$$TQLI_{REQ010} \equiv (M_{SC}/J_{SC}/t/h/1/3/1/1/1/e\ D)$$
$$TQLI_{REQ011} \equiv (M_{SC}/J_{SC}/t/h/1/3/1/1/1/1/e\ D)$$
$$TQLI_{REQ012} \equiv (M_{SC}/J_{SC}/t/h/1/1/3/e\ D)$$
$$TQLI_{REQ013} \equiv (M_{SC}/J_{SC}/t/h/1/1/3/1/e\ D)$$
$$TQLI_{REQ014} \equiv (M_{SC}/J_{SC}/t/h/1/1/3/1/1/e\ D)$$
$$TQLI_{REQ015} \equiv (M_{SC}/J_{SC}/t/h/1/1/3/1/1/1/e\ D)$$
$$TQLI_{REQ016} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/3/e\ D)$$
$$TQLI_{REQ017} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/3/1/e\ D)$$
$$TQLI_{REQ018} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/3/1/1/e\ D)$$
$$TQLI_{REQ019} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/1/3/e\ D)$$
$$TQLI_{REQ020} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/1/3/1/e\ D)$$
$$TQLI_{REQ021} \equiv (M_{SC}/J_{SC}/t/h/1/1/1/1/1/3/e\ D)$$
$$\text{with } OKDATA \equiv (21/-2/r/s))$$

Example 2. *Storage Query (MODE=1)*: *Create an unique single path that can be used to retrieve all the information obtained with the query in Example 1.*

$$TQLI_{STQ0} \equiv (M_{SC}/J_{SC}/t/h/577577577e\ D) \text{ with}$$
$$MCHOIC \equiv 6/0/3/0/3/0/3;$$
$$MKERNE \equiv 3/1/3/1; \quad OKDATA \equiv 2/r/s;$$
$$\text{and } MREPET \equiv 3/0/-7/1/1/-7/-1/0/-7/1$$

This application of the Markush codes yields one path description:

$$TQLI_{STQ0} \equiv$$
(M$_{SC}$/J$_{SC}$/t/h/3/e/t/h/3/1/e/t/h/3/1/1/e/
t/h/3/1/1/1/e/t/h/3/1/1/1/1/e/t/h/3/1/1/1/1/1/e/t/h/
1/3/e/t/h/1/3/1/e/t/h/1/3/1/1/e/t/h/1/3/1/1/1/e/
t/h/1/3/1/1/1/1/e/t/h/1/1/3/e/t/h/1/1/3/1/e/t/
h/1/1/3/1/1/e/t/h/1/1/3/1/1/1/e/t/h/1/1/1/
3/e/t/h/1/1/1/3/1/e/t/h/1/1/1/3/1/1/e/t/h/
1/1/1/1/3/e/t/h/1/1/1/1/3/1/e/t/h/1/1/1/1/1/3/e M)

The terminal M is a multiple registry number that contains the 21 D type registry numbers retrieved with the query in Example 1.

IV.2. Chemical Formulas. The Information Representation (IR) kind of Transparent Query Language Item, TQLI, is ideal for representing and manipulating chemical formulas or any other static or pictorial data.[1,2,32]

The three different kinds of formulas (with no rings, with single rings, and with multiple rings) are considered in turn in the following three subsections. A ring has a minimum of three entities—i.e., atoms or moieties. Throughout this subsection (a) bonds are designated as 99 − delocalized (aromatic) bond, 14 − triple covalent bond, 13 − double covalent bond, 12 − single covalent bond, 11 − ionic bond, and 10 − hydrogen bond and (b) $TQLI_i$ is the linear form of IR_i and $NTQLI_i$ is the normal form of $TQLI_i$. The effort of normalizing chemical formulas is summarized in subsection IV.3.

IV.2.1. Formulas without Rings. The linear normal forms for the information representations of derivatives of aldehydes in Hydrochloric Acid solutions (IR_1, IR_2, and IR_3) have been given in subsection 1.4.1.

Example 3. *Retrieval − Markush Override Codes (MODE=0)*: *List all compounds with up to nine atoms or moieties that contain the skeletal chain $C^{14}W_2$.CHY.HZ attached to a hydrogen atom or an alkyl chain with up to four carbon atoms. W is D, F, Cl, or Br; Y is O or S; and Z is any atom like Cl or moiety like ClO.*

The Markush override codes generate the following normalized information path descriptions.

$NTQLI_{REQ71} \equiv$ (M/J/1.H.3.C.C.^{14}H.3.3.H/
11.10.13.12.12/0.0.0.0.12/0.0.0.0.12/0/0.0.0.12 X)

$NTQLI_{REQ72} \equiv$ (M/J/1.H.3.C.C.^{14}C.H.H.H.3.3.H/
11.10.13.12.12.12/0.0.0.0.0.12/0.0.0.0.0.12/
0/0.0.0.0.12/0.0.0.0.12/0/0.0.12 D)

$NTQLI_{REQ73} \equiv$ (M/J/1.H.3.C.C.^{14}C.C.H.H.H.H.H.3.3.H/
11.10.13.12.12.12.12/0.0.0.0.0.0.12/
0.0.0.0.0.0.12/0/0.0.0.0.0.0.12/0.0.0.0.0.0.12/0/0.0.0.0.12/
0.0.0.0.12/0/0.0.0.12 D)

$NTQLI_{REQ74} \equiv$
(M/J/1.H.3.C.C.^{14}C.C.C.H.H.H.H.H.H.H.3.3.H/
11.10.13.12.12.12.12.12/ 0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.12/
0/0.0.0.0.0.12/ 0.0.0.0.0.12/0/0.0.0.0.0.12/
0.0.0.0.12/0/0.0.0.12 D)

$NTQLI_{REQ75} \equiv$
(M/J/1.H.3.C.C.^{14}C.C.C.C.H.H.H.H.H.H.H.H.H.3.3.H/
11.10.13.12.12.12.12.12.12/0.0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.12/
0/0.0.0.0.0.12/0.0.0.0.0.12/0/0.0.0.0.0.12/
0.0.0.0.12/0/0.0.0.0.12/ 0.0.0.0.12/0/0.0.0.12 D)

all with OKDATA ≡ (5((−2/O/S),2(−4/D/F/Cl/Br))).

Example 4. *Storage − Markush Override Codes (MODE=1)*: *Create an unique path for all compounds with up to nine atoms or moieties that contain the skeletal chain $C^{14}W_2$.CHY.HZ attached to a hydrogen atom or an alkyl chain with up to three carbon atoms. W is D, F, Cl, or Br; Y is O or S; and Z is any atom like Cl or moiety like ClO.*

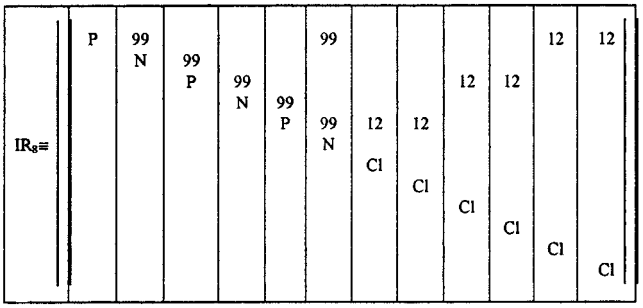
The path description generated is

$TQLI_{STQ7} \equiv$
(M/J/1.H.3.C.C.^{14}H.3.3.H/11.10.13.12.12/0.0.0.0.12/
0.0.0.0.12/0/0.0.0.12/1.H.3.C.C.^{14}C.H.H.H.3.3.H/
11.10.13.12.12.12/0.0.0.0.0.12/0.0.0.0.0.12/0/
0.0.0.0.12/0.0.0.0.12/0/0.0.12/1.H.3.C.
C.^{14}C.C.H.H.H.H.H.3.3.H/11.10.13.12.12.12.12/
0.0.0.0.0.0.12/0.0.0.0.0.0.12/0/0.0.0.0.0.12/
0.0.0.0.0.12/0/0.0.0.0.12/0.0.0.0.12/0/0.0.0.12/
1.H.3.C.C.^{14}C.C.C.H.H.H.H.H.H.H.3.3.H/
11.10.13.12.12.12.12.12/0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.12/0/0.0.0.0.0.12/0.0.0.0.0.12/
0/0.0.0.0.12/0.0.0.0.12/0/0.0.0.12/
1.H.3.C.C.^{14}C.C.C.C.H.H.H.H.H.H.H.H.H.3.3.H/
11.10.13.12.12.12.12.12.12/0.0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.0.12/
0.0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.12/
0/0.0.0.0.0.12/0.0.0.0.0.12/0/0.0.0.0.12/
0.0.0.0.12/0/0.0.0.12 M)

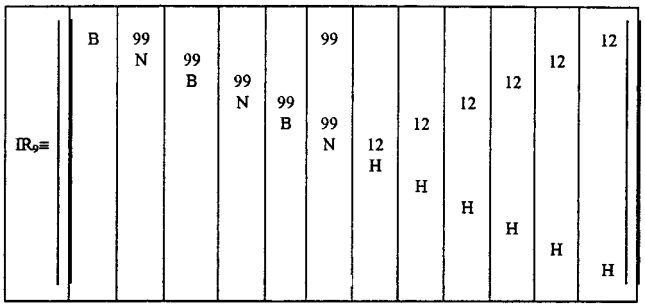
The terminal M is a multiple registry number that contains the five D type registry numbers retrieved with the query in Example 3.

IV.2.2. Formulas with Single Rings. Trimeric phosphazene and borazine, both taken from ref 36, and phenol are used to illustrate the manipulation of formulas that contain a single ring. Information representations for the three compounds are shown next.

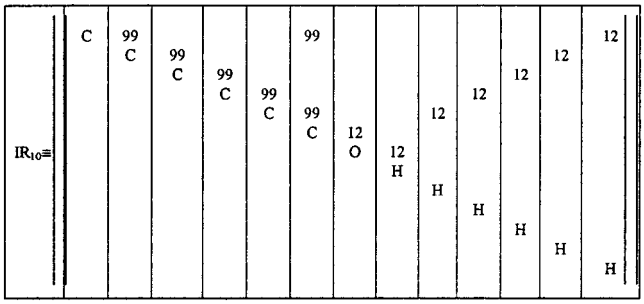
Trimeric Phosphazene ($P_3N_3Cl_6$)



Borazine ($B_3N_3H_6$)



Phenol (C_6H_5OH)



The normal forms for the above three formulas are

$$NTQLI_8 \equiv (M_2/J_2/N.P.N.P.N.P.Cl.Cl.Cl.Cl.Cl.Cl/ 99.99.99.99.99.12/0.0.0.0.0.12/ 0/0/99.0.0.12/0.0.0.12/0/0/0.12/0.12 X)$$

$$NTQLI_9 \equiv (M_3/J_3/H.N.B.N.B.N.H.H.H.H.H/ 12.99.99.99.99.99.12/0/0.0.0.0.0.12/ 0/ 0.99.0.0.12/0/0.0.0.12/0/0.0.12/0/0.12 X)$$

$$NTQLI_{10} \equiv (M_4/J_4/H.C.C.C.C.C.C.C.O.H.H.H.H.H/ 12.99.99.99.99.99.12.12/0/0/0.0.0.0.0.12/ 0/0.99/0.0.0.0.12/0/0.0.0.12/0/0/0.12 X)$$

Another form of phenol that does not contain any ring is

$$NTQLI_{101} \equiv (M_3/J_3Ph.O.H/12.12 X)$$

Example 5. *Retrieval (MODE=0)*: *List all derivatives of trimeric phosphazene, $P_3N_3Cl_6$, in which one or more Cl are replaced by F, phenyl group, etc*.

The normalized path description that describes this query is

$$NTQLI_{REQ11} \equiv (M/J/N.P.N.P.N.P.4.4.4.4.4.4/ 99.99.99.99.99.12/0.0.0.0.0.12/0/0/99.0.0.1.2/ 0.0.0.12/0/0/0.12/0.12 D)$$

with OKDATA ≡ (6(−1/Cl)).

Example 6. *Retrieval (MODE=0)*: *List all compounds that have a borazine ring or its analogue with the nitrogen atoms replaced by phosphorus in which any type of atom may be linked by a single covalent bond to a ring atom*.

In this one normalized query is produced:

$$NTQLI_{REQ12} \equiv (M/J/1.3.B.3.B.3.B.1.1.1.1.1/ 12.99.99.99.99.99.12/0/0.0.0.0.0.12/0/0.99.0.0.12/ 0/0.0.0.12/0/0.0.12/0/0.12 D)$$

with OKDATA ≡ (3(−2/P/N)).

Example 7. *Retrieval (MODE=0)*: *List all the esters of phenol with the form C_6H_5OR where R is an alkyl group that contains up three carbon atoms*.

The following four normalized queries are produced:

$$NTQLI_{REQ131} \equiv (M/J/H.C.C.C.C.C.C.C.O.C.H.H.H.H.H.H.H/ 12.99.99.99.99.99.12.12.12/0.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.12/0/0.99/0/0.0.0.0.0.12/ 0/0.0.0.0.12/0/0.0.0.12/0/0.0.12 D)$$

$$NTQLI_{REQ132} \equiv (M/J/H.C.C.C.C.C.C.C.O.C.C.H.H.H.H.H.H.H.H.H/ 12.99.99.99.99.99.12.12.12.12/0.0.0.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.0.12/0/0.99.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.0.12/0/0/0/0.0.0.0.0.12/ 0/0.0.0.0.12/0/0.0.0.12/0/0.0.12 D)$$

$$NTQLI_{REQ133} \equiv (M/J/H.C.C.C.C.C.C.C.O.C.C.C. H.H.H.H.H.H.H.H.H.H/12.99.99.99.99.99. 12.12.12.12.12/0.0.0.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.0.0.12/0/0.99.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.0.0.12/0/0.0.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.0.12/0/0/0/ 0.0.0.0.0.12/0/0.0.0.0.12/0/0.0.0.12/0/0.0.12 D)$$

$$NTQLI_{REQ134} \equiv (M/J/H.C.C.C.C.C.C.C.O.C.C.H.H.H.C.H.H.H.H.H.H.H/ 12.99.99.99.99.99.12.12.12.12.0.0.0.12/ 0.0.0.0.0.0.0.0.0.12.0.0.0.12/ 0.0.0.0.0.0.0.0.0.12.0.0.0.12/ 0/0.99.0.0.0.0.0.0.12/0/0/0/ 0.0.0.0.0.0.0.0.0.12/0/0/0/0.0.0.0.0.12/0/ 0.0.0.0.12/0/0.0.0.12/0/0.0.12 D)$$

Example 8. *Storage (MODE=1)*: *Create an unique single path description for the query in Example 7*.

The unique single path description that is generated is

$$NTQLI_{STQ13} \equiv (M/J/H.C.C.C.C.C.C.C.O.C.H.H.H.H.H.H.H/ 12.99.99.99.99.99.12.12.12/0.0.0.0.0.0.0.0.12/ 0.0.0.0.0.0.0.0.12/0/0.99/0/0.0.0.0.0.12/0/ 0.0.0.0.12/0/0.0.0.12/0/0.0.12/ H.C.C.C.C.C.C.C.O.C.C.H.H.H.H.H.H.H.H.H/ 12.99.99.99.99.99.12.12.12.12/ 0.0.0.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.0.0.12/ 0/0.99.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.0.12/ 0/0/0/0.0.0.0.0.0.12/0/0.0.0.0.0.12/0/0.0.0.0.12/ 0/0.0.12/H.C.C.C.C.C.C.C.O.C.C.C.H.H.H.H.H.H. H.H.H.H.H/12.99.99.99.99.99.12.12.12.12.12/ 0.0.0.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.0.0.12/0/ 0.99.0.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.0.0.12/ 0/0.0.0.0.0.0.0.0.0.12/0.0.0.0.0.0.0.0.0.0.12/ 0/0/0/0.0.0.0.0.0.12/0/0.0.0.0.0.12/0/0.0.0.0.12/ 0/0.0.12/H.C.C.C.C.C.C.C.O.C.C.C.H.H. H.C.H.H.H.H.H.H.H/12.99.99.99.99.99. 12.12.12.12.0.0.0.12/0.0.0.0.0.0.0.0.0.0.12.0.0.0.12/ 0.0.0.0.0.0.0.0.0.0.12.0.0.0.12/0/0.99.0.0.0.0.0.0.12/ 0/0/0/0.0.0.0.0.0.0.0.12/0/0/0/0.0.0.0.0.0.12/0/0.0.0.0.12/ 0/0.0.0.12/0/0.0.12 M)$$

Example 9. *Retrieval (MODE=0)*: *List all esters of phenol with the form PhOR where R is an alkyl group that contains up to three carbon atoms*.

The following four normalized queries are produced:

Table 5. The Actual, Ac-TE, and Theoretical, Th-TE, Total Efforts for Determining Full Normal Forms[a]

no.	NR	normalized item	N_{Mo}	N_{TMo}	N_{Bo}	N_{IO}	N_{CO}	N_{IN}	Ac-TE	Th-TE
1	0	$NTQLI_1$	9	5	8	72	14	7	93	8 × 9!
2	0	$NTQLI_2$	9	5	8	72	14	7	93	8 × 9!
3	0	$NTQLI_3$	9	5	8	72	14	7	93	8 × 9!
4	1	$NTQLI_4$	16	8	16	240	230	15	475	15 × 16!
5	0	$NTQLI_5$	14	7	13	182	42	13	237	13 × 14!
6	0	$NTQLI_6$	16	9	15	240	37	15	292	15 × 16!
7	0	$NTQLI_{REQ71}$	9	6	8	72	14	7	93	8 × 9!
8	0	$NTQLI_{REQ72}$	12	6	11	132	18	11	161	11 × 12!
9	0	$NTQLI_{REQ73}$	15	8	14	210	26	13	249	14 × 15!
10	0	$NTQLI_{REQ74}$	18	10	17	306	32	17	355	17 × 18!
11	0	$NTQLI_{REQ75}$	21	12	20	399	38	19	456	20 × 21!
12	1	$NTQLI_8$	12	6	12	132	110	11	253	11 × 12!
13	1	$NTQLI_9$	12	6	12	132	110	10	252	11 × 12!
14	1	$NTQLI_{10}$	13	6	13	156	144	12	312	12 × 13!
15	0	$NTQLI_{101}$	3	1	2	6	1	0	7	12
16	1	$NTQLI_{REQ11}$	12	6	12	132	110	11	253	11 × 12!
17	1	$NTQLI_{REQ12}$	12	6	12	132	110	10	253	11 × 12!
18	1	$NTQLI_{REQ131}$	16	8	16	240	230	15	475	15 × 16!
19	1	$NTQLI_{REQ132}$	19	10	19	342	386	18	718	18 × 19!
20	1	$NTQLI_{REQ133}$	22	12	22	462	474	21	957	21 × 22!
21	1	$NTQLI_{REQ134}$	22	12	22	462	474	21	957	21 × 22!
22	1	$NTQLI_{REQ141}$	6	4	5	30	8	0	36	5 × 6!
23	1	$NTQLI_{REQ142}$	9	6	8	72	14	0	86	8 × 9!
24	1	$NTQLI_{REQ143}$	12	8	11	132	20	0	152	11 × 12!
25	1	$NTQLI_{REQ144}$	12	8	11	132	20	0	152	11 × 12!
26	2	$NTQLI_{15}$	19	8	20	342	404	18	764	18 × 19!
27	2	$NTQLI_{REQ16}$	19	8	20	342	404	18	764	18 × 19!

[a] Ac-TE (= $N_{IO} + N_{CO} + N_{IN}$) and Th-TE (= number of interchange operations × the number of mathematical equivalent forms) are the numbers of operations that must be executed with and without the improved chain tracing algorithm that is described in subsection III.1. NR is the number of rings in the formula. N_{Mo}, N_{TMo}, N_{Bo}, N_{IO}, N_{CO}, and N_{IN} are defined in the text (subsection IV.3). 8 × 9! = 2 903 040.

$NTQLI_{REQ141} \equiv$ (M/J/Ph.O.C.H.H.H/
12.12.12/0.0.12/0.0.12 D)

$NTQLI_{REQ142} \equiv$ (M/J/Ph.O.C.C.H.H.H.H.H/
12.12.12.12/0.0.0.12/0.0.0.12/0/0.0.12/0.0.12 D)

$NTQLI_{REQ133} \equiv$ (M/J/Ph.O.C.C.C.H.H.H.H.H.H.H/
12.12.12.12.12/0.0.0.0.12/0.0.0.0.12/
0/0.0.0.12/0.0.0.12/0/0.0.12/0.0.12 D)

$NTQLI_{REQ134} \equiv$ (M/J/Ph.O.C.C.H.H.H.C.H.H.H.H/
12.12.12.12.0.0.0.12/0.0.0.12.0.0.0.12/
0.0.0.12.0.0.0.12/0/0.0.12/0/0/0/0.0.12 D)

IV.2.3. Formulas with Multiple Rings. To illustrate the manipulation of formulas with multiple rings consider the following information representation for an alcohol of naphthalene. This example is included for completeness. However, the description of the complex algorithm that is used is deferred to a future paper.[34]

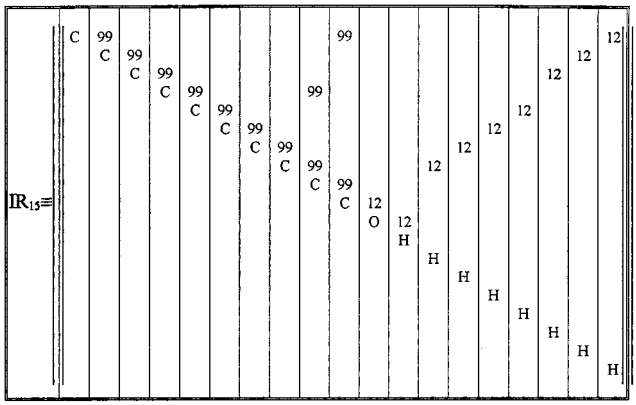
α-Naphthol ($C_{10}H_7OH$):

The linear normal form for IR_{15} is

$NTQLI_{15} \equiv$ (M_{15}/J_{15}/H.C.C.C.C.C.C.C.C.C.C.O.
H.H.H.H.H.H.H/12.99.99.99.99.99.99.99.99.99.
12.12/0/0/0/0.0.0.0.99.0.0.0.12/0/0.0.0.0.0.0.0.12/
0.99.0.0.0.0.0.12/0/0.0.0.0.0.12/0/0/0.0.0.12/0/0.0.12 X)

Example 10. *Retrieval (MODE=0)*: *List all the derivatives of the naphthalene alcohol represented in IR15 and its sulfur analogue in which the hydrogen atoms are replaced by moieties with single covalent bonds*.

The single normalized query that is produced is



$NTQLI_{REQ16} \equiv$
(M/J_1/4.C.C.C.C.C.C.C.C.C.C.C.3.4.4.4.4.4.4.4.4/
12.99.99.99.99.99.99.99.99.99.99.12.12/0/0/0/
0.0.0.0.99.0.0.0.12/0/0.0.0.0.0.0.0.12/
0.99.0.0.0.0.0.12/0/0.0.0.0.0.12/0/0/
0.0.0.12/0/0.0.12 D)

with OKDATA ((−1/Cl), (−2/O/S), 7(−1/Cl)).

IV.3. Summary of Effort. The effort that must be expended to obtain the full normal forms, designated thus $NTQLI_i$, are given in Table 5. The following seven contributing factors apply to each separate formula that contains bonds in the information representation: N_{Mo}, the number of moieties (atoms or named fragments) in the formula; N_{TMo}, the number of terminal moieties in the formula; N_{Bo}, the number of bonds in the formula; N_{IO}, $N_{Mo}^2 - N_{Mo}$ the number of kernels in the information representation that must be examined to determine all connections (or bonds) between moieties; N_{CO}, the number of compari-

sons that are made by the chain tracing algorithm to determine the composite chain that describes the topology of the normalized representation. This includes an estimate of the number of preliminary operations that are performed (with N_{Bo}, N_{TMo}, the values for bonds and moieties) by the improved chain tracing algorithm (subsection III.1). N_{IN} is the number of interchange operations that must be performed to convert the entered information representation to its normal form.

A measure of the actual total effort to obtain the normal form is Ac-TE ($=N_{IO} + N_{CO} + N_{IN}$). Note that the value of Ac-TE is (i) independent of the size of the database. (ii) significantly less if the entered formulas do not contain rings, NR=0, or they are in their normal forms, i.e., $N_{IN} = 0$. (iii) significantly less if rings are replaced by moieties such as Ph in Phenol (see 15 in Table 5). However the use of such names to specify groups such as aldehyde (CHO) or ketone (CO) does not significantly improve the value of Ac-TE.

Ac-TE should be compared with the maximum possible value, Th-TE, which is $(N-1)\times N!$, where N is the rank of the information representation. The information in Table 5 clearly establishes that even without replacing rings with moieties, partitioning of databases is not necessary and wasteful of resources.

V. CONCLUSIONS

Results obtained with recently developed algorithms have confirmed that searches for fragments in large databases that are not cataloged can be economically performed at high speeds in real time. Thus the common practice of cataloging selected functional groups, such as keto group and rings, is wasteful of computer resources and does not offer any advantages.

ACKNOWLEDGMENT

The author thanks his wife for the many hours spent correcting this manuscript and for many of the examples used. He is also indebted to his many students who have contributed to the development of the Transparent Query Language and to the Alexander von Humboldt Foundation for Senior Fellowships that enabled him to spend 15 months with Prof. Ivar Ugi at TU-München. From the research came the realization that the Dujundji-Ugi Mathematical Model for Organic Chemistry is applicable to all scientific disciplines.

REFERENCES AND NOTES

(1) de Maine, P. A. D.; Bradley, K. D.; Jodis, S. M.; de Maine, M. M. High-Speed Tools for Global Information Management II. Specifications and Uses of the Transparent Query Language (TQL). *J. Systems Integration* about 150 pages (In press).

(2) de Maine, P. A. D.; Bradley, K. D. High-Speed Tools for Global Information Management I. Information Processing and Retrieval. *J. Systems Integration* **1996**, *6*, 217−240.

(3) Wiswesser, W. J. *A Line-Formula Notation*; T Y. Crowell Co.: New York, 1954 and others by W. J. Wiswesser.

(4) Hayward, H. W. Patent Office Research and Development Reports, No. 21, U.S. Department of Commerce, Washington, D.C., 1961.

(5) Dyson, G. M. *ICSU Rev.* **1964**, *4*, 110 and others by G. M. Dyson.

(6) Hunsberger, I. M. et al. *Chemical Notation Systems*; National Academy of Sciences − National Research Council: Publication 1150, Washington, D.C., 1964.

(7) Hiz, H. A Linearization of Chemical Graphs. *J. Chem. Docum.* **1964**, *4*, 173−180.

(8) Agarwal, K. K.; Gelernter, H. L. A Computer-Oriented Linear Canonical Notational System for the Representation of Organic Structures with Stereochemistry. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 463−479.

(9) Dietz, A. Yet Another Representation of Molecular Structure. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 787−802.

(10) Ihlenfeldt, W.-D.; Gasteiger, J. Augmenting Connectivity Information by Compound Name Parsing: Automatic Assignment of Stereochemistry and Isotope Labeling. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 661−674.

(11) Ruiz, I. L.; Soto, J. L. C.; Gomez-Nieto, M. A. Computer Translation of Inorganic Chemical Nomenclature to a Dynamic Abstract Data Structure. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 526−533.

(12) Rucker, C.; Rucker, G. Mathematical Relation between Extended Connectivity and Eigenvector Coefficients" *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 534−538.

(13) Hu, C.-Y.; Xu, L. A New Scheme for Assignment of a Canonical Connectivity Table. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 840−844.

(14) Dyson, G. M.; Cossum, W. E.; Lynch, M. F.; Morgan, H. L. Mechanical Manipulation of Chemical Structure: Molform Computation and Substructure Searching of Organic Structures by the Use of Cipher Directed, Extendered and Random Matrices. *Inform. Stor. Retr.* **1963**, *1*, 69−99 and others by G. M. Dyson quoted in Chemical Abstracts: *56* − 1977c, 12286g; *57* − 2818b, 9186d; *58* − 13099d; *59* − 14546d; *60* − 7424a; *61* − 15318g; and *62* − 9745g.

(15) Spialter, L. The Atom Connectivity Matrix and its Characteristic Polynomial. *J. Chem. Docum.* **1964**, *4*, 261−269; *J. Am. Chem. Soc.* **1963**, *85*, 2012−2013.

(16) Fujiwara, Y.; Nakayama, T. A Graph Theory Data Base For Storage Of Chemical Structures Organized By The Block-Cutpoint Tree Technique. *Anal. Chim. Acta* **1981**, *133*, 647−656.

(17) Pogliani, L. On a Graph Theoretical Characterization of Cis/Trans Isomers. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 801−804.

(18) Balaban, A. T.; Liu, X.; Klein, D. J.; Babic, D.; Schmalz, T. G.; Seitz, W. A.; Randic, M. Graph Invariants for Fullerenes. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 396−404.

(19) Balaban, A. T. Chemical Graphs: Looking Back and Glimpsing Ahead. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 339−350.

(20) Meyer, E. A Topological Representation of Chemical Formulae for Computers. *Information Storage and Retrieval* **1965**, *2*, 205−215.

(21) Fugmann, R.; Ploss, G.; Winter, J. H. Supply of Information on Chemical Reactions. An Advanced Topology-Based Method. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 47−53.

(22) Dubois, J.-E. *Structural Organic Chemistry Revisited: Reasoning with Topology and Informatics*; Dr. Alfred Huthig Verlag GmbH: Heidelberg, 1989; pp 211−227.

(23) Dugundji, J.; Ugi, I. An Algebraic Model of Constitutional Chemistry as a Basis for Chemical Computer Programs. *Topics Curr. Chem.* **1973**, *39*, 21.

(24) Ugi, I.; Bauer, J.; Blomberger, C.; Brandt, J.; Dietz, A.; Fontain, E.; Gruber, B.; Scholley-Pjab, A. V.; Senff, A.; Stein, N, Models, Concepts, Theories & Formal Languages In Chemistry And Their Use as a Basis for Computer Assistance in Chemistry. *J. Chem. Inf. Comput. Sci.* **1993**, *34*, 3−16.

(25) Ugi, I. K. *A Qualitative Global Mathematical View of Chemistry—James Dugundji's Contribution to Computer Assistance in Chemistry*; Dr. Alfred Huthig Verlag GmbH: Heidelberg, 1989; pp 345−366.

(26) Ugi, I.; Gruber, B.; Stein, N.; Demharter, A. Set-Valued Maps As A Mathematical Basis Of Computer Assistance In Stereochemistry. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 485−489.

(27) Bauer, J.; Fontain, E.; Ugi, I. IGOR and RAIN − The First Mathematically Based Multipurpose Problem-Solving Computer Programs for Chemistry and their Use.... *Comm. Math. Chem. (MATCH)* **1992**, *27*, 31−48.

(28) Morgan, H. L. Generation of a unique machine description for chemical structures. *J. Chem. Docum.* **1965**, *5*, 107−113.

(29) A *Mathematically Complete* or *Philosophically Closed* system or language is defined as one that can be used to describe any problem in its domain. Neither the efficiency of a particular formulation or a unique way of describing a problem is implied. Examples of such systems are natural languages and matrix algebra. The efficiency of a particular formulation is determined by economic factors such as the amount of computer time. While matrix algebra can be used to solve a particular problem in a great many different ways, the most efficient solutions are those which require the minimum number of applications of the rules of matrix algebra and take the least amount of computer time. A variant of this concept—mathematical equations are not merely empirical descriptions of the physical sciences—has resulted in the discovery of both the Special and General Theories of Relativity, by Albert Einstein, and both Positrons and Antimatter, by P. A. M. Dirac.

(30) Jain Ramesh in a keynote address: *Content-based Interactivity*, at the 10th International Conference on Tools with Artificial Intelligence (ICTAI'97) in Newport Beach, CA, 3−8/November/1997. Also see: Ramesh, J., Kasturi, R., Schunck, B. *Machine Vision*; McGraw-Hill Inc.: New York, 1995.
(31) de Maine, P. A. D. High-Speed Searches for Fragments in the National Compound Registry. In *MultiComponent Reactions & Combinatorial Chemistry, Proceedings of the German-Polish Workshop*; Hippe, Z. S., Ugi, I. K., Eds.; University of Technology, Rzeszow, Poland, Sept. 28−30 1997, 1998, pp 61−83.
(32) de Maine, P. A. D.; Xia, P. High-Speed Manipulation Of Information Representations. I. Normalization And Mobile Canonicalization. *Comput. Chem.* **1998**, *22*, No. 4, 321−330.
(33) de Maine, P. A. D. High-Speed Manipulation Of Information Representations. II. Markush (Fragment) Searches. *Comput. Chem.* In press.
(34) de Maine, P. A. D. High-Speed Manipulation Of Information Representations. III. Formulae With More Than One Ring. Manuscript in preparation.
(35) Date, C. J. *An Introduction to Database Systems*, 4th ed.; Addison-Wesley Publishing Company, Inc.: New York, 1986; Vol. I.
(36) Huheey, J. E.; Keiter, E. A.; Keiter, R. L. *Inorganic Chemistry: Principles of Structure and Reactivity*, 4th ed.; Harper Collins College Publishers: New York, 1993.

CI9700844