J. Chem. Inf. Comput. Sci. 1990, 30, 155-159

Computer Tool Kit for Chemists. 1. Design Considerations for Interfaces

P. A. D. DE MAINE,* B. C. CARTEE,‡ M. S. WOJTYNA, and M. M. DE MAINE

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

Received November 13, 1989

Now being developed is a general expert systems package able to serve many different classes of science and engineering users. This package consists of 4 principal systems and 10 service systems, each of which can function independently or be called by other expert systems. Operational prototypes exist for 10 of the 13 systems. In such general, integrated systems the interfaces must satisfy two incompatible sets of requirements—user and system. The solution that is described consists of a hierarchical two-tier design with each tier satisfying one set of the requirements. Two examples of the implemented form of this design are given.

1. INTRODUCTION

Recently we have noted[1] that the following four problems, common to many disciplines, can be solved by applications of combinatorial mathematics, with the rules and interpretation uniquely defined by the discipline: (1) evaluation of systems of equations; (2) construction, decomposition, alteration, and combination of graphs; (3) identification of entities; (4) classification of entities. The first two problems require predictive capabilities to help design experiments.

The family of expert systems that has evolved[1] consists of 4 principal systems, for solving the 4 class problems, and 10 artificial intelligence systems support tools (AISST) that are used by the principal systems and can also be used on a stand-alone basis. The 10 AISST perform special tasks like high-speed retrieval or transporting high-level language code.

An analysis of expert systems (here defined as systems that can be easily used to discover nonobvious knowledge) has disclosed four types: conventional, automatic deductive, adaptive learning, and automatic adaptive learning. Conventional systems require examples to construct the rule (or knowledge) base or determine the classification parameters. They essentially mimic the *perceived* way humans think, and they are incapable of predicting unprecedented events (events that cannot be predicted from the state-of-the-art alone). Automatic systems are applications of combinatorial mathematics in which the rule base or classification parameters are determined by mathematical rules. Such systems can predict unprecedented events. The four principal expert systems in the proposed family[1] are classed as automatic. In particular, FRANS and SCANMAT are automatic deductive systems, and both AUTOLRN and AUTOREC are automatic adaptive learning systems. Several of the artificial intelligence system support tools are conventional expert systems.

The design of interfaces for such general systems is considered next.

2. INTERFACES

For systems that are intended to serve different classes of users and can also be invoked by other expert systems, the interfaces must satisfy two conflicting sets of requirements—user and system. The design of the interface between user and system is critically important because it ultimately determines the usefulness and flexibility of the system. Ideally, the user wants to communicate in the language of his discipline. Thus, the user interface should be highly discipline specific with limited flexibility. By contrast, the system interface should be discipline independent and very flexible. Attempts to combine these conflicting requirements into a single interface complicate the problem of maintenance and yield either specialized systems of limited usefulness or systems that must be substantially redesigned for new applications.

The separation of the user interface from the functionality of applications programs, recommended by Szekely,[2] simplifies maintenance but does not by itself assure that a general system can be combined with a specialized interactive interface. The multilevel menu-driven design proposed by Savage and Habinek[3] is useful only if all the different applications are known and the system is not invoked as a subsystem by another expert system.

Our general solution to this problem is to provide an interface with two separate parts—the interactive user-friendly interface (IUFI) and the system interface (SI). Their relation to the model base (or inference engine) is shown in Figure 1. The overall requirements for the IUFI and SI are considered in the following subsections.

2.1 Requirements for the Interactive User-Friendly Interface (IUFI). As the overriding goal of the IUFI is to provide a user-friendly interface, (1) it should be interactive, (2) it must support a language that is easily understood by the user, and (3) it must be easily changed for new applications. The objectives can be achieved with a table-driven design wherein the requisite tables usually must be produced by experts in individual disciplines.

Other functions of the IUFI include the following:

(a) The IUFI checks the validity of input and suggests corrective actions. In systems that require the use of parsers in the SI this function can be deferred to the system interface.

(b) The IUFI passes special instructions that limit the scope of the model base.

(c) The IUFI ensures that output is fully compatible with the system interface.

(d) The IUFI translates output received from the system interface for display to users. This function can also be accomplished in the system interface with information passed by the IUFI.

2.2 Requirements for the System Interface (SI). The primary roles of the SI are to convert the input to the general data structures that are used by the model base and to protect the model base against errors in the input received from the IUFI. Other requirements of the SI include the following:

(a) The SI should be accessible to the model base (or inference engine).

(b) The SI should be interactive with respect to the IUFI if the IUFI requires information from the model base.

(c) The SI should be interactive with respect to both the IUFI and the model base for real-time systems.

* Author to whom correspondence should be addressed.
‡ Deceased.

0095-2338/90/1630-0155$02.50/0

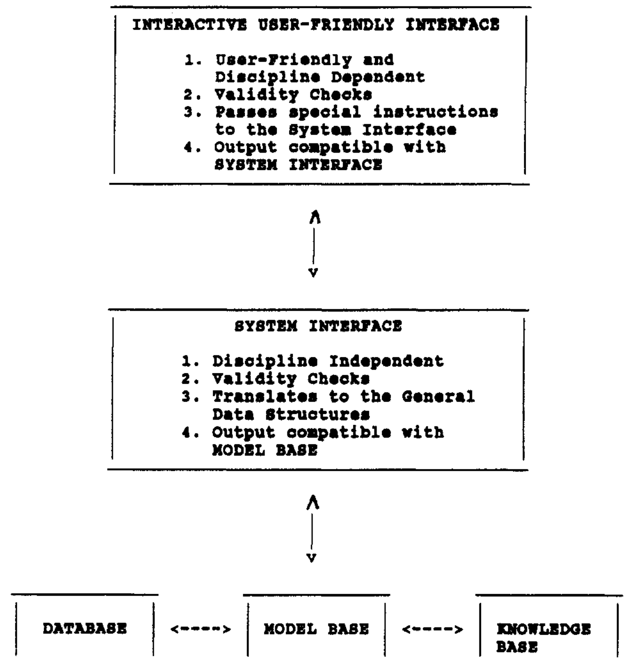

Figure 1. Relationships between the components of an expert system. In FRANS and CURFIT the interactive user-friendly interfaces are separate (i.e., preprocessors) and the system interfaces are integral parts of the systems.

(d) The SI must perform validity checks and suggest corrective actions and output answers if those functions are deferred in the IUFI.

3. CRITERIA FOR THE INTERACTIVE USER-FRIENDLY INTERFACE (IUFI)

Many computer systems that have much to offer are often not used or are used ineffectively because their interfaces with users are poor.[4] Close scrutiny of man–machine barriers has resulted in an increased effort to provide more attractive and easier to run programs, with improved user aids to achieve greater user satisfaction and acceptance.[5–8]

The design of effective man–machine interfaces includes the careful consideration of perceptual psychology, cognitive psychology, and human interaction with machines[9,10] as well as the role of man in the design process.[11–14] The success or failure of interface programs is measured as much by their ease of use as by their functional capacity.[6,7,9,15] Erhich has suggested that the dialogue components should be designed by specialists in human–machine interfaces and the computational components by computer specialists.[6]

For our systems the discipline-independent parts [that is, the skeletal structure of the user interface, the system interface, and the model base, knowledge base, and databases (see Figure 1)] are designed by computer experts. The dialogue portions of the user interface are designed by using experts in the discipline as a test bed. The design phase begins with a logical analysis, the goal of which is to realize *logical structures* that are information independent and therefore can be efficiently used for many different disciplines.

The seven factors that together critically determine the usefulness of IUFIs are considered in the following seven subsections.

3.1 Simple and Consistent Interaction. In well-written interfaces the terminology and operational procedures are uniformly available and consistently applied throughout the entire system,[11,15] and the activity required by the user is minimized by having all aspects (vocabulary, sequence, consequences) appear natural to the user.[5,13]

Consistency means that mechanisms are used in the same way whenever they occur.[15] The goal of consistency is to permit users to make predictions about one part of the system from occurrences in other parts.[8,11,16,17] Good display formats are uniform and lack exceptions and special conditions. Studies have shown that the principle of positional consistency,[9,10] which is the display of menu items in the same relative positions, reduces the time the user requires to view the screen, requires fewer instructions, and increases the accuracy of operation.[10]

If occasionally certain rules do not apply, an additional memory load is demanded of the user.[11] To ameliorate this problem, good designs have consistent formats and wording and permit users to invoke global commands, like HELP or CANCEL, at any time.[9]

3.2 Mimimize the User Memory Workload. Well-designed, friendly interactive programs minimize the mental workload of the user by restricting the number of items that must be remembered and limiting the consequences of forgetting.[5,8,11,13,18] Such programs also limit the need for frequent reference to manuals as memory refreshers.[3,5]

There are several standard methods for limiting or avoiding extensive memorization in interactive interfaces. Both the input required from the user and the interface output are minimized.[8] Rules and vocabularly are kept simple, and concepts familiar to the user are used when possible.[9,19] Menus, prompting, and a good help facility are the most frequently used memory aids.[9,11,13]

Consistent and uniform menus are a major aid to users, because the choice of an item from a displayed list is much easier than remembering command words or concentrating on the exact syntax of an operation.[9,13] Improved ease of operation with menu selection is obtained if both recognition memory and passive response are also used.[3]

Good help functions are a vital part of interactive interfaces. While memory aids and prompts are sometimes sufficient, query in depth requires a help facility.[9,11] A program's response to a help request should not destroy the display of the context in which help was requested. Frequently employed techniques either repeat the screen image or use window managers.[9]

3.3 Know the Users. The first principle of designing an interactive interface is to know the users with respect to communication, special problems, experience, education, and background. The interface activities should be modeled to fit the expected users,[11,13,19] and all communications should be in a language familiar to the users.[7,9]

3.4 Adapt to the Users' Abilities. Users of interactive interfaces have diverse computer experiences and knowledge that changes with time. Well-designed interfaces are sufficiently flexible to assist the novice without slowing down or irritating the more experienced user.[5,7,9,11,19,20] The success of an interface is frequently measured by the amount of training required. A well-designed interface is easy to use, educates the user, and has a positive reinforcement for learning.[4,11,18]

Good interfaces have brief memory aids for experienced users and access to greater detail for the novice. Normally this is accomplished with the aid of a help facility that permits query in depth and returns detailed instructions for responses and suggestions.[5,9] Some large interfaces support several help levels for use by the different classes of users.[15] Clear error messages can also serve as training aids for the novice and reminders for the expert.[13] However, in some interfaces the help facility is used to supplement error messages.[11,13] Occasionally, some tradeoffs are made to achieve ease of learning for the novice versus both speed and convenience for the expert.[14]

3.5 User Feedback. It is critically important that users be constantly aware of their location in the interface, the status of the interface, and the results of user actions.[8,19] Users normally assume that entered actions are correct unless explicitly told otherwise.[21] It has been shown that good results are obtained with feedback that is immediate, unambiguous, highly visible, and carefully worded to identify the type of activity taking place and to help in making appropriate decisions.[5,8,11,22,23] Feedback about closure, which is defined as a statement or group of statements that indicates completion of a task, is psychologically important because it produces a feeling of relief and reduces the user's memory load.[9,12]

Well-designed diagnostic facilities and well-constructed error messages have gone far toward making an interface appealing to its users.[7,12,13,24] Good error messages are clear, concise, informative, nonnegative, and nonthreatening and successfully guide the user in the proper use of commands.[7,12,24]

3.6 Validity Checks and Error Recovery. Studies have shown that users experience stress if they have difficulty in recovering from errors, and the result is lower user satisfaction and overall productivity.[8] A well-designed interactive interface allows users to make mistakes with confidence that their errors will not produce dire consequences.[5,23] Effective interfaces have good error detection techniques, error messages that explain the cause of the problem and recovery procedures, and reversible actions or backup to a convenient restarting point.[11,13,25] Another technique, used in the FRANS system interface,[26] is to classify all errors as either recoverable or terminal. Recoverable errors are automatically processed by the system itself, and the user is notified of the action taken. Terminal errors generate explicit messages of the actions that the user must take before restarting the system.

3.7 Controlled Response Time. A computer system's response time is defined as the number of seconds between a user-initiated activity and the display of the results.[22] Many studies have established that the response time is important in determining user satisfaction and the frequency of user errors.[4,9,12,22]

Long system response times frequently result in user frustration and dissatisfaction, a decrease in efficiency, and an increase in error rates.[4,22] Uncertainty with regard to the duration and perceived cause of delays influences user satisfaction.[4] However, the effects of uncertain delays can be reduced with the periodic display of informative messages that both indicate actions being taken and explain the cause of the delay.[4,9,12] Many studies have shown that productivity increases as the response time decreases.[22] However, rapidly responding systems can surprise and disrupt the user[12] and can result in lower comprehension, ill-considered decisions, less learning, and more data entry errors.[19]

Another important factor that determines both user satisfaction and the accuracy of performance is the amount of variance in the response times.[12,22] Some experiments have shown that users perform better with relatively slow systems will small response time variances than they do with faster systems that have high response time variances.[9,12,22]

4. APPLICATIONS

Two interactive user-friendly interfaces (IUFI) that have been implemented are described. Examples of the use of the CURFIT and FRANS systems have appeared in ref 1 and in the references cited therein.

4.1 Curve-Fitting (CURFIT) System. Details of the CURFIT interactive user-friendly interface (CFIUFI) will be found in its system manual[27] and demonstrations of its use in the user manual.[28] CFIUFI contains a parser and performs all preprocessing and validity checks. The system interface is a simple read routine with minimal validity checks (to ensure that the input was generated by the CFIUFI) that accepts input numbers in a tabular (or fixed format) form and constructs the matrices that serve as the general data structures for the CURFIT system.

Table I. Computer-Generated Raw Data for Random Walks (or Polymers) on a Regular Tetrahedral Lattice[a]

0.10000E+01	0.10000E+01	0.97532E+00	0.93717E+00
0.89170E+00	0.84483E+00	0.79676E+00	0.74878E+00
0.70260E+00	0.65791E+00	0.61462E+00	0.57383E+00
0.53503E+00	0.49884E+00	0.46474E+00	0.43243E+00
0.40193E+00	0.37333E+00	0.34682E+00	0.32221E+00
0.29911E+00	0.27750E+00	0.25769E+00	0.23868E+00
0.22127E+00	0.20506E+00	0.18985E+00	0.17564E+00
0.16293E+00	0.15092E+00	0.13971E+00	0.12920E+00
0.11949E+00	0.11059E+00	0.10228E+00	0.94569E-01
0.87580E-01	0.80962E-01	0.74874E-01	0.69136E-01
0.63950E-01	0.59203E-01	0.54686E-01	0.50590E-01
0.46644E-01	0.42868E-01	0.39542E-01	0.36467E-01
0.33722E-01	0.30937E-01;		
2	4	6	8
10	12	14	16
18	20	22	24
26	28	30	32
34	36	38	40
42	44	46	48
50	52	54	56
58	60	62	64
66	68	70	72
74	76	78	80
82	84	86	88
90	92	94	96
98	100		

[a] The first 50 values are the fractions of successful walks, FSW_N, at the corresponding step, N, in the last 50 values.

To illustrate the use of the CFIUFI, suppose that the raw data in Table I are to be fitted to the four-choice form of the Hammersley–Morton attrition equation:

$$\ln (FSW_N/(4/3)^{N-1}) = -AC_4N + IN_4$$

FSW_N is the fraction of successful walks (or polymers) with N steps (or skeletal bonds) on a regular tetrahedral lattice. AC_4 and IN_4 are the four choice values for the attrition constant and intercept. The maximum tolerances for FSW_N and N are MT_{FSW} and MT_N, respectively. The error bounds for the abscissa, EB_y, and ordinate, EB_x, are given by

$$EB_y = \ln (1.0 + MT_{FSW}) = \ln ((1.0 + MT_{FSW})FSW_N/(4/3)^{N-1}) - \ln (FSW_N/(4/3)^{N-1})$$

$$EB_x = MT_N N$$

For this example MT_{FSW} and MT_N are set to 0.005 and 0, respectively.

The data deck produced by the CFIUFI is displayed in Table II. The pertinent output obtained from the CURFIT system for this input is shown in Table 1 of ref 1. To within 0.5% of the fraction of successful walks, FSW_N, the raw data in Table I are not described by the four-choice form of the Hammersley–Morton equation. Moreover, there is not an asymptotic solution.

4.2 Function Recognition and Numerical Solution (FRANS) System. The FRANS interactive user-friendly interface (FRIUFI) differs from that for the CURFIT system in that all validity checks are deferred to the system interface, which supports a comprehensive LL1 grammar.[26b] Details of the FRIUFI will be found in its user manual.[29]

To illustrate the use of the FRIUFI, suppose that all possible solutions are required for the following reaction system with a reversible rate equation and an equilibrium equation.

$$2.1A + B \Leftrightarrow 0.9C, \text{ RKF, RKB}$$

$$2X \leftrightarrow C, \text{ EK1}$$

A, B, C, and X are reactants. RKF and RKB are the specific

Table II. Data Deck Generated by the CURFIT Interactive User-Friendly Interface from the Raw Data in Table I with the Maximum Tolerance in the Fraction of Successful Walks Equal to 0.5%[a]

0	0	2341TOOLCF1			
0	1	4	52		
52	2	2	0	0	0
-1	0	0	0	0	0
3	0	1	0		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499	0.00499	0.00499		
0.00499	0.00499				
0.0					
-0.28766E+00	-0.86297E+00	-0.14633E+01	-0.20785E+01		
-0.27035E+01	-0.33328E+01	-0.39667E+01	-0.46042E+01		
-0.52431E+01	-0.58842E+01	-0.65275E+01	-0.71715E+01		
-0.78169E+01	-0.84622E+01	-0.91083E+01	-0.97557E+01		
-0.10404E+02	-0.11053E+02	-0.11702E+02	-0.12351E+02		
-0.13001E+02	-0.13651E+02	-0.14301E+02	-0.14953E+02		
-0.15604E+02	-0.16255E+02	-0.16907E+02	-0.17560E+02		
-0.18211E+02	-0.18863E+02	-0.19515E+02	-0.20169E+02		
-0.20822E+02	-0.21475E+02	-0.22128E+02	-0.22782E+02		
-0.23434E+02	-0.24088E+02	-0.24742E+02	-0.25397E+02		
-0.26050E+02	-0.26702E+02	-0.27357E+02	-0.28010E+02		
-0.28667E+02	-0.29326E+02	-0.29983E+02	-0.30639E+02		
-0.31292E+02	-0.31954E+02	0.0	0.0		
0.20000E+01	0.40000E+01	0.60000E+01	0.80000E+01		
0.10000E+02	0.12000E+02	0.14000E+02	0.16000E+02		
0.18000E+02	0.20000E+02	0.22000E+02	0.24000E+02		
0.26000E+02	0.28000E+02	0.30000E+02	0.32000E+02		
0.34000E+02	0.36000E+02	0.38000E+02	0.40000E+02		
0.42000E+02	0.44000E+02	0.46000E+02	0.48000E+02		
0.50000E+02	0.52000E+02	0.54000E+02	0.56000E+02		
0.58000E+02	0.60000E+02	0.62000E+02	0.64000E+02		
0.66000E+02	0.68000E+02	0.70000E+02	0.72000E+02		
0.74000E+02	0.76000E+02	0.78000E+02	0.80000E+02		
0.82000E+02	0.84000E+02	0.86000E+02	0.88000E+02		
0.90000E+02	0.92000E+02	0.94000E+02	0.96000E+02		
0.98000E+02	0.10000E+03	0.0	0.0		
1.0	0				
0.0	0.0	0.0	0.0		

[a] A detailed description will be found in the *Operation and Logic Manual for the CURFIT System*.[28]

rate constants for the forward and reverse reactions, respectively, and EK1 is the equilibrium constant.

The operation manual for the FRANS system[30] contains examples of the formats for entering information. Here we need to know the following:

(i) Comments are entered on separate lines, thus

/: THIS IS A COMMENT /:

They can be entered anywhere in a data deck.

(ii) The system command SELECT=1 designates a prediction problem.

(iii) Equations are entered as above and terminated with a semicolon.

(iv) The input status of every parameter must be designated as either GIVEN (>0), NOT MEASURABLE (<0), or MAYBE GIVEN or COMPUTED (=0). In our case all parameters will be designated MAYBE, thus A=0; B=0; ... etc.

The output generated by the FRIUFI is shown in Table III. Because there are seven parameters (A, B, C, X, RKF, RKB, and EK1) that may be given or computed, there are 126 different combinations of given parameters that may yield a value for at least one unknown parameter. The pertinent output generated by the FRANS system for the data deck in Table III is shown in Table IV. The conclusions are as follows:

Table III. Data Deck Generated by the Interactive User-Friendly Interface for the FRANS System[a]

```
/: THE DEMONSTRATION OF THE INTERACTIVE INTERFACE FOR FRANS :/
SYSTEM:
    SELECT = 1;
EQUATIONS:
    /: ENTER RATE EQUATIONS FIRST :/
    2.1*A + B <=> .9*C, RKF, RKB;
    /: ENTER EQUILIBRIUM REACTIONS LAST :/
    2*X <-> C, EK1;
CONSTANTS:
    /: SET ALL REACTANTS TO MAYBE :/
    A=0; B=0; C=0; X=0;
    /: SET ALL SPECIFIC REACTION CONSTANTS TO MAYBE :/
    RKF=0; RKB=0; EK1=0;
STOP:
```

[a] SELECT = 1 specifies a prediction question. For data set CONSTANTS all parameters have been set to zero, which means that they may be given or computed. Thus, the request is for all predictions that will yield a value for at least one parameter.

Table IV. Pertinent Part of the Output from the FRANS System with the Data Deck in Table III[a]

```
THE PREDICTIONS THAT WERE OUTPUTTED BY THE CRAMS SYSTEM ARE TO BE
TABLED NEXT. G - GIVEN; R - RECOMPUTED; BLANK - NOT COMPUTED; N -
CANNOT BE GIVEN; NC - CANNOT BE GIVEN BUT COMPUTED.
   THE MULTIPLICITIES OF GIVEN REACTANTS ARE DESIGNATED BY N THUS :-
N*G. THE PREDICTION WITH THE LOWEST MULTIPLICITY GENERALLY YIELDS
THE LOWEST MAXIMUM ERRORS FOR THE REACTION CONSTANTS WHEN THEY ARE
COMPUTED WITH THE CURFIT SYSTEM.
   THE PREDICTOR COMMAND (PRECOM) IS   0 AND THE COMPUTATIONAL
STATUS OF NO VARIABLE WAS SPECIFIED.

            12 NON-REDUNDANT PREDICTIONS ARE TABLED NEXT.

  NAME      1    2    3    4    5    6    7    8    9   10   11   12
 A        2*G  2*G  2*G  2*G    C    C    C    C    C    C    C    C
 B          C    C    C    C  2*G  2*G  2*G  2*G    C    C    C    C
 C             2*G    C    C       2*G    C    C  2*G  2*G  2*G    C
 X               C  0*G    C         C  0*G    C  0*G    C    C  0*G
 RKF             C    C    C         C    C    C    C    G    C    G
 RKB             C    C    C         C    C    C    C    G    C    G
 EK1             C    C    G         C    C    G    C    C    G    C
            1    2    3    4    5    6    7    8    9   10   11   12

             2 NON-REDUNDANT PREDICTIONS ARE TABLED NEXT.

  NAME     13   14
 A          C    C
 B          C    C
 C          C    C
 X        0*G    C
 RKF        C    G
 RKB        C    G
 EK1        G    G
           13   14

  **** OUT OF 21 ORIGINAL PREDICTIONS 14 WERE NOT REDUNDANT ****
```

[a] There are 14 different combinations of data or experiments that will yield partial (1 and 5) or complete solutions. Many combinations of inputs do not yield any result.

(1) Only 21 of the possible 126 different combinations can yield a calculated value.

(2) Seven of the 21 predictions are redundant. That is, they contain more than the required information.

(3) There are only 14 nonredundant predictions (or experiments) that can yield values for at least one unknown parameter.

(4) Nine complete predictions (i.e., predictions in which all parameters are given or computed) require values for two parameters and three (10, 12, and 14) require values for three parameters.

5. CONCLUSIONS

Described is a two-tier design for an interface that is suitable for use in general expert systems for different classes of users in different disciplines. These general systems can also be invoked by other systems. Two different applications of this design are presented.

REFERENCES

(1) (a) de Maine, P. A. D.; de Maine, M. M.; Wojtyna, M. S. Expert Systems for Science and Engineering. *Prog. Mol. Spectrosc., Proc. Analytiktreffen "Mol. Spectrosc.—Theor. Appl.", 15th; Teubner-Texte*

Phys. **1989**, *20*, 250-271. (b) de Maine, P. A. D.; de Maine, M. M. Computer Aids for Chemists. *Anal. Chim. Acta* (in press).
(2) Szekely, P. *Separating the User Interface from the Functionality of Application Programs*; Report CMU-CS-88-101; Computer Science Department, Carnegie Mellon University: Pittsburgh, PA, 1988; 208 pp.
(3) Savage, R. E.; Habinek, J. K. A Multilevel Menu-Driven User Interface: Design and Evaluation Through Simulation. In *Human Factors in Computer Systems*; Thomas, J. C., and Schneider, M. L., Eds.; ABLEX: Norwood, NJ, 1984; pp 165-186.
(4) Nickerson, R. S. Why Interactive Computer Systems are Sometimes Not Used by People Who Might Benefit From Them. *Int. J. Man-Mach. Stud.* **1981**, *15*, 469-483.
(5) Gaines, B. R.; Shaw, M. L. G. Dialog Engineering. In *Designing for Human-Computer Communications*; Sime, M. E., and Coombs, M. J., Eds.; Academic Pres: London, 1983; pp 1-20.
(6) Ehrich, R. W. DMS—A System for Defining and Managing Human-Computer Dialogues. In *Analysis, Design and Evaluation of Man-Machine Systems*; Johannsen, G., and Rijnsdorp, J. E., Eds.; Pergamon Press: Oxford, U.K., 1983; pp 327-334.
(7) Sime, M. E.; Coombs, M. J. In *Designing for Human-Computer Communications*; Sime, M. E., and Coombs, M. J., Eds.; Academic Press: London, 1983; pp 1-20.
(8) Williges, R. C.; Williges, B. H. Human-Computer Dialogue Design Considerations. In *Analysis, Design and Evaluation of Man-Machine Systems*; Johannsen, G., and Rijnsdorp, J. E., Eds.; Pergamon Press: Oxford, U.K., 1983; pp 239-246.
(9) Foley, J. D.; Van Dam, A. *Fundamentals of Interactive Computer Graphics*; Addison-Wesley: Reading, MA, 1982; pp 55-56, 217-243.
(10) Barnard, P. J.; Hammond, N. V.; Morton, J.; Long, J. B. Consistency and Compatibility in Human-Computer Dialogue. *Int. J. Man-Mach. Stud.* **1981**, *15*, 87-134.
(11) Gaines, B. R. The Technology of Interaction—Dialogue Programming Rules. *Int. J. Man-Mach. Stud.* **1981**, *14*, 133-140.
(12) Shneiderman, B. Human Factors Experiments in Designing Interactive Systems. *Computer* **1979**, *12*(12), 9-19.
(13) Hansen, W. J. User Engineering Principles for Interactive Systems. In *Interactive Programming Environments*; Barstow, D. R., Shrobe, H. E., and Sandewall, E., Eds.; McGraw-Hill: New York, 1984; pp 217-231.
(14) Draper, S. W.; Norman, D. A. Software Engineering for User Interfaces. *IEEE Trans. Software Eng.* **1985**, *SE-11*, 252-258.
(15) Smith, D. C.; Irby, C.; et al. Designing the Star User Interface. In *Integrated Interactive Computing Systems*; Proceedings of the European Conference on Integrated Interactive Computing Systems, Stresa, Italy, Sept 1-3, 1982; Degano, P., and Sandewall, E., Eds.; Amsterdam, North-Holland Publishing: 1982; pp 297-313.
(16) Thimbleby, H. Dialogue Determination. *Int. J. Man-Mach. Stud.* **1980**, *13*, 295-304.
(17) Marcus, A. Corporate Identity for Iconic Interface Design: The Graphic Design Perspective. *IEEE Comput. Graphics Appl.* **1984**, *4*(12), 24-32.
(18) Hatvany, J. H.; Guedj, R. A. Man-Machine Interactions in Computer-Aided Design Systems. In *Analysis, Design and Evaluation of Man-Machine Systems*; Johannsen, G., and Rijnsdorp, J. E., Eds.; Pergamon Press: Oxford, U.K., 1983, pp 231-237.
(19) Smith, H. Human Computer Communication. In *Human Interactions with Computers*; Smith, H. T., and Green, T. R. G., Eds.; Academic Press: London, 1980, pp 5-38.
(20) Mozeico, H. A Human/Computer Interface to Accommodate User Learning Stages. *Commun. ACM* **1982**, *25*, 100-104.
(21) Hayes, P.; Ball, E.; Reddy, R. Breaking the Man-Machine Communication Barrier. *Computer* **1981**, *14*(3), 19-30.
(22) Shneiderman, B. Response Time and Display Rates in Human Performance with Computers. *Comput. Surv.* **1984**, *16*, 265-285.
(23) Lieberman, H. Designing Interactive Systems from the User's Viewpoint. In *Integrated Interactive Computing Systems*; Proceedings of the European Conference on Integrated Interactive Computing Systems, Stresa, Italy, Sept 1-3, 1982; Degano, P., and Sandewall, E., Eds.; North-Holland Publishing: Amsterdam, 1982; pp 45-59.
(24) Ledgard, H.; Singer, A.; Whiteside, J. Directions in Human Factors for Interactive Systems. In *Lecture Notes in Computer Science*; Goos, G., and Hartmanis, J., Eds.; Springer-Verlag: New York, 1981; pp 146-162.
(25) Miller, L. A.; Thomas, J. C., Jr. Behavioral Issues in the Use of Interactive Systems. In *Lecture Notes in Computer Science*; Springer-Verlag: New York, 1976; Vol. 49, pp 193-215.
(26) (a) de Maine, P. A. D. Automatic Deductive Systems for Chemistry. *Anal. Chim. Acta/CTO* **1981**, *133*, 685-698. (b) de Maine, P. A. D. *Systems Manual for the CRAMS System*; Automatic Systems for the Physical Sciences, Report 6; Computer Science and Engineering Department, Auburn University: Auburn, AL, 1980; 125 pp.
(27) Cartee, B. C.; Head, D. M.; de Maine, P. A. D.; De Maine, M. M. *CURFIT Interface System Manual*; Automatic Systems for the Physical Sciences, Report 9; Computer Science and Engineering Department, Auburn University: Auburn, AL, 1986; 243 pp.
(28) de Maine, P. A. D. *Operation and Logic Manual for the CURFIT System*; Automatic Systems for the Physical Sciences, Report 2; Computer Science and Engineering Department, Auburn University: Auburn, AL, 1985; 241 pp.
(29) Wojtyna, M. S.; de Maine, P. A. D. *User's Manual for the FRANS Interactive User-Friendly Interface*; Automatic Systems for the Physical Sciences, Report 15; Computer Science and Engineering Department, Auburn University: Auburn, AL, 1988; 9 pp.
(30) de Maine, P. A. D. *Operation Manual for the CRAMS System*; Automatic Systems for the Physical Sciences, Report 5; Computer Science Department, The Pennsylvania State University: University Park, PA, 1980; 194 pp.

Use of Small Computers for Large Computations: Enumeration of Polyhex Hydrocarbons

J. V. KNOP, W. R. MÜLLER, K. SZYMANSKI, and N. TRINAJSTIĆ*,‡

Computer Centre, The Heinrich Heine University, 4000 Düsseldorf, Federal Republic of Germany

Received October 27, 1989

The enumeration of polyhex hydrocarbons with up to 16 hexagons is reported. These results are now for the first time available.

Recently in this journal Tošić and Kovačević[1] reported generation and enumeration of unbranched catacondensed benzenoids with up to $h = 20$ (h is the number of hexagons in the benzenoid hydrocarbon). There have also been other papers in this journal on enumeration of benzenoid hydrocarbons.[2-4] However, all these papers (and others in the literature[5]) are incomplete in the sense that they did not give the total numbers of benzenoid hydrocarbons for $h \geq 13$. The reason is that the enumeration of benzenoid hydrocarbons is a difficult combinatorial problem.[6] It is also a computationally involved problem.[5,7]

‡Permanent address: The Rugjer Bošković Institute, P.O. Box 1016, 41001 Zagreb, Croatia, Yugoslavia.

Since we have successfully developed a very powerful algorithm for enumeration of hexagonal structures,[8] we wish to supplement the above studies with counts of polyhex hydrocarbons with up to $h = 16$. Polyhex hydrocarbons are graph theoretically represented by polyhexes.[9] Polyhexes are hexagonal systems that may be obtained by any combination of regular hexagons such that two hexagons have exactly one common edge or are disjoint. Benzenoid hydrocarbons represent a subset of polyhex hydrocarbons[10] and are graph theoretically depicted by benzenoid graphs.[9] Benzenoid graphs are those polyhexes that are 1-factorable structures.[11] A 1-factorable polyhex corresponds to a polyhex hydrocarbon with Kekulé structure(s).[12] Most of the papers reporting the enumeration of hexagonal structures did not distinguish be-

