Object-Oriented Programming Applied to Laboratory Automation. 1. An Icon-Based User Interface for the Analytical Director

T. ZHOU and T. L. ISENHOUR*
Department of Chemistry, Kansas State University, Manhattan, Kansas 66506

M. ZAMFIR-BLEYBERG
Computing and Information Sciences Department, Kansas State University, Manhattan, Kansas 66506

J. C. MARSHALL
Department of Chemistry, Saint Olaf College, Northfield, Minnesota 55057

Received August 6, 1991

The design and implementation of an icon-based user interface for the Analytical Director robotic expert system is described. A conceptual model is developed using an object-oriented design technique to simulate a robotic analytical laboratory. The conceptual laboratory model represents the robotic table as an abstract workbench and the laboratory activities as a set of primitive operations. Chemical properties and chemical reactions are used to construct a deep-reasoning-based inference engine, which enables the primitive operations to alter the state of the workbench. By comparing the state of the workbench with that of the real robotic table, the expert system can validate a designed procedure and update the system knowledge base. With the icon-based interface, a user can design an analytical procedure using chemical knowledge at the conceptual level, while the expert system validates the designed procedure and controls the robot execution necessary to accomplish the analysis.

INTRODUCTION

This paper presents an icon-based user interface for the Analytical Director. The goal of the Analytical Director is to build a system that can design, test, modify, and implement its own analytical procedures. An object-oriented expert system has been prototyped, and progress has been made in the areas of building a conceptual model of an analytical laboratory,[1] of optimizing a designed procedure,[2] and of developing standard robot methods.[3,4]

A robot analytical laboratory is modeled by a set of objects representing the laboratory components and the associated concepts and operations. By organizing the objects into hierarchies and enabling interactions among the objects, such an object-oriented model can be used to simulate complex activities of an analytical laboratory. Using a set of icons graphically displaying the laboratory component objects, a chemist can operate on the icons to design an analytical procedure under expert system supervision. This expert system embodies knowledge of chemicals, chemical reactions, principles of analytical methods, and legal laboratory operations.

Figure 1 shows the artchitecture of the Analytical Director. The icon-based user interface allows the chemist to simulate analytical laboratory operations by manipulating the icons and choosing actions from menus. The chemist only needs to know the chemistry to design an analytical procedure. The conceptual model of the laboratory is the core of the expert system. It enables a user to design an analytical procedure without being aware that a robot is going to execute the procedure. The expert system has full control of the robot laboratory and is capable of generating the robot procedure accurately from the user-designed procedure. The icon-based user interface allows the user to design an analytical procedure at the conceptual level while the expert system supervises the design process, validates the designed procedure, and generates the executable robot procedure.

THE USER INTERFACE

Under the current expert system the robot laboratory is capable of performing three classical analytical methods: gravimetric, spectrophotometric, and volumetric. It is developed on a SUN SPARC workstation under KEE. KEE is a Knowledge Engineering Environment developed by IntelliCorp for expert system applications. KEE supports frame-based, rule-based, procedure-oriented, and object-oriented representation methods. KEE provides a graphics package for constructing window, menu, and icon-based interfaces.

Figure 2 shows the iconic representation of the major components of the laboratory workbench, which include 30 test tubes, 14 stock containers, a chemical cabinet, a command box, a lab panel box, a balance, a water bath, a spectrophotometer, a centrifuge, a mixer, a pH meter, a YES box, a NO box, and a PROMPT window.

The *Chemical Cabinet* icon enables the user to retrieve and update the chemical information database. Selecting this icon will activate the Chemical Information Manager, an object-oriented database management module.

Fourteen stock solution icons represent 14 stock bottles on the workbench. Samples and reagents can be assigned to these icons, which can then be used to design a procedure. Thirty icons (*T-1...T-30*) represent test tubes on the workbench. The user can "transfer" samples or reagents into a *T* icon but not into stocks. All the laboratory operations such as measuring the pH and centrifuging a precipitate must have test tube icon(s) involved. The contents of the stock solutions and test tubes (i.e., the state of these icon objects) are used to determine if a designed step is a legal operation and if a designed procedure is a valid procedure. For example, a spectrophotometric method determining manganese with the target element in *T-3* in form of $Mn(OH)_2$ at the final step will not be accepted as a valid procedure because the state of *T-3* indicates the analyte is in a solid form.

Six instrument icons, balance, water bath, spectrophotometer, centrifuge, mixer, and pH meter, represent the real instruments on the workbench. A mouse click on any of these icons pops up a menu which contains a list of operations applicable to the corresponding instrument.

The *Lab Panel* icon is used for the easy input of numbers and chemical units. The PROMPT window along with the

0095-2338/92/1632-0079$03.00/0

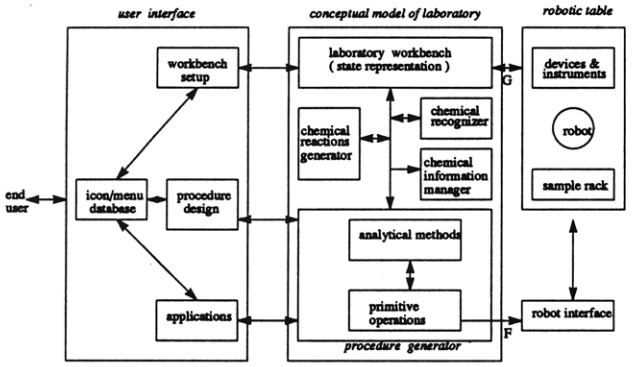

Figure 1. Architecture of the Analytical Director.

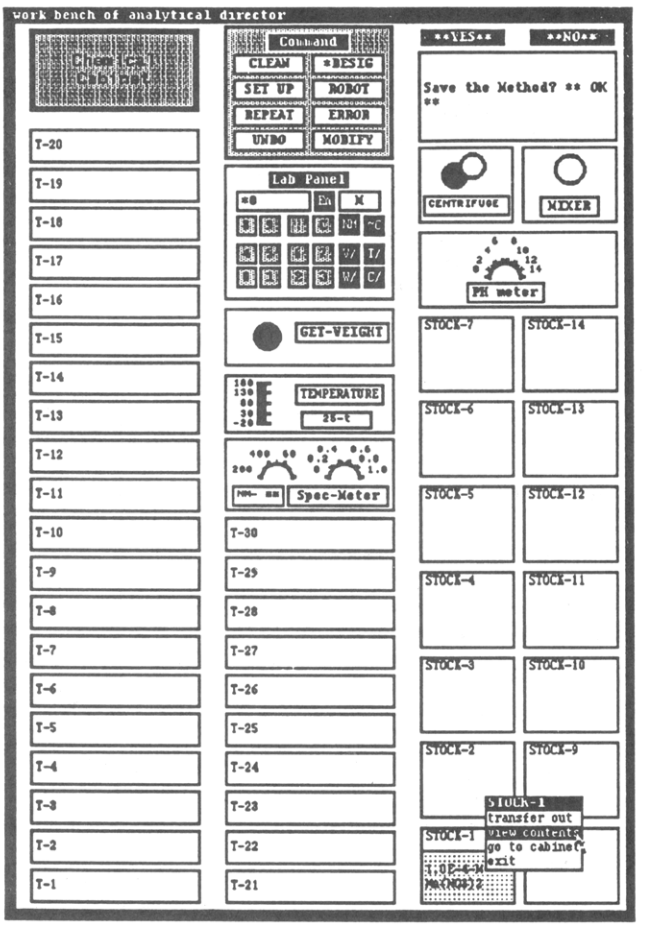

Figure 2. Icon representation of the workbench.

YES and NO boxes are used by the expert system to offer suggestions and provide explanatory questions to the user.

The *Command Box* contains a group of icons that are used by a chemist along with the expert system to design a procedure and supervise the robot's activities. Through the *CLEAN* icon, a user can clean test tubes and/or stock solutions. The *SET UP* icon is used to assign new chemical reagents or new samples to the stock containers. Three icons, *REPEAT*, *UNDO*, and *MODIFY*, are used mainly in the design process to repeat, delete, or modify the last procedure step. The *ERROR* icon can be used to invoke an explanatory message and to correct design errors. By clicking on the *ROBOT* icon and selecting the pop-up menu, a user can start the robot and execute a valid procedure or have the robot follow the icon manipulation directly. Through the **DESIGN* icon, a user can change the strategy of procedure design. One way is to design a procedure without chemical reaction simulation. In this mode, a user's icon manipulations do not have many restrictions. The expert system generates each step according to a user's icon operation even if the step is not valid. A chemist might prefer this mode when he is sure of the correctness of a procedure and wants to save the time needed for chemical reaction simulations. Another way of designing a procedure is by simulating the chemical reactions associated with the icon manipulations. The system will then keep track of the state of the workbench and only legal operations are accepted. Thus, the expert system can assist the design process and assure that the robot can execute the designed procedure.

DESIGN AND IMPLEMENTATION

The use of an object-oriented system[5,6] suggests a different perspective of the automated chemical laboratory. The traditional model of the automated chemical laboratory places the laboratory robot in a preeminent position among the laboratory resources. The robot "knows" where chemicals are stored, what kind of chemical reactions occur when chemicals are mixed together, how to weigh a chemical, how to measure a spectrum, how to centrifuge a sample, etc. In other words, the classical model of a laboratory robot is in the image of the human expert in a laboratory surrounded by passive laboratory equipment.

An alternative model of the laboratory designates the laboratory resources as a collection of objects, each of which also may be considered an intelligent instrument or a "robot". For example, we may conceptualize each of the test tubes as a robot with certain intelligence, encapsulated as an object, that knows and does all test tube operations, such as preparing solutions and performing the chemical reactions of mixed chemicals. Similarly, things like centrifuges, mixers, and spectrometers are robot-objects, each able to send and receive messages and each capable of performing the tasks for which it was designed. In this model, the laboratory robot is only one of the robot-objects in the laboratory. Like the other robot-objects, the laboratory robot knows how to perform those tasks for which it is directly suited and how to send and receive messages to and from peer robot-objects to complete other tasks.

The difference between these two models would suggest that conventional procedure-based programming could be used for the first while object-oriented techniques would be a natural choice for the second. Furthermore, implementing many conceptual robots with the object-oriented programming paradigm should be far easier and much more flexible than attempting to program the robot in the image of human expert.

The Object-Oriented Representation of the Laboratory. There are many different interpretations of the term 'object'. We use the definition by Smith and Tockey:[7] "An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain". This definition indicates that there is a computer representation of a real-world object by an "abstract object" and "abstract operations" for manipulating the objects. Objects are the fundamental building blocks of a computer system. In our conceptual model, both the real components of a laboratory (such as instruments, containers, and chemicals) and the abstract concepts (such as laboratory operations, chemical reactions, and analytical methods) are represented by objects. Each object incorporates both data structure (data attributes) and behavior (method attributes). Each object has "intelligence": it knows its own state, represented by data attributes, and can operate on and alter its state by activating method attributes. By creating a set of discrete objects and defining the relationships and interactions among the objects, we have built an object-oriented model of our robot laboratory. As shown in Figure 3, the objects are structured into five hierarchies.

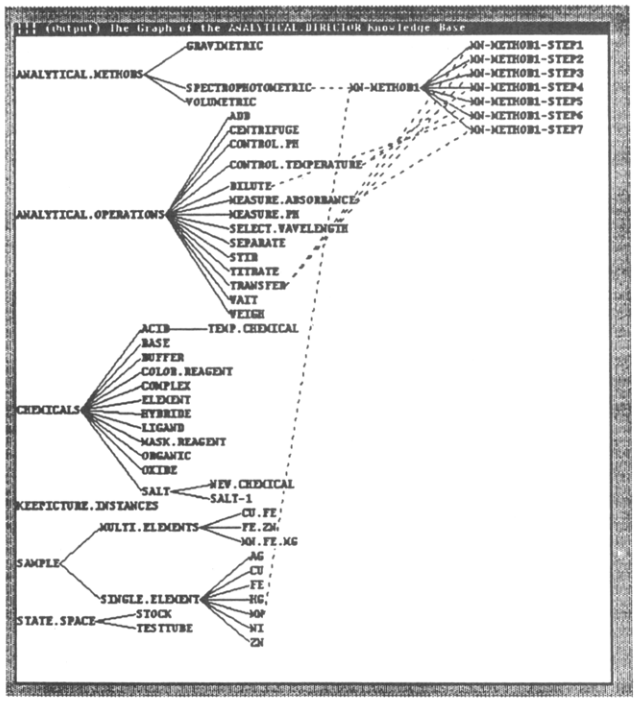

Figure 3. Object-oriented representation of an analytical laboratory.

The *analytical.methods* hierarchy, which consists of four objects, captures the knowledge of the general principles of the three analytical methods. The analytical.methods object defines the attributes common to any kind of analytical methods like the data attribute where_is_sample, where_is_standard, and target_element_state and the method attribute validate_procedure and generate_robot_procedure. Three objects, gravimetric, spectrophotometric, and volumetric, are created as subclasses of the analytical.methods object to represent the three analytical methods. As subclasses, each of these three objects inherits the data and method attributes of the parent object analytical.methods. Because each of them has its own requirements, way of validating a procedure, and way of generating the robot procedure, the behavior of validate_procedure and generate_robot_procedure is defined within each subclass object. Each subclass object also has its own data and method attributes. For example, the spectrophotometric object contains its own data attributes sample_absorbance and standard_absorbance along with the inherited attributes from the parent object analytical.methods.

The *chemicals* hierarchy is an object-oriented approach to database management. The expert system must have knowledge about chemicals and chemical reactions in order to simulate the relevant chemical reactions in a procedure during the design process. Because of the large amount of chemical information required, it is necessary to have a database management system. Further, the system needs to have a module that uses the properties retrieved from the database to predict chemical reactions. The chemicals hierarchy object set only deals with the management of the chemical properties. The chemicals object defines several database management method attributes such as add_new, select, get_properties, modify, and delete. Presently, the database contains chemical properties for 12 basic chemical types (see Figure 3). Twelve objects are created as subclasses of the chemicals object to represent the 12 chemical types. Each of them contains some data attributes corresponding to the important properties of the type and inherits all the method attributes from the parent object chemicals.

The *laboratory.operations* hierarchy consists of a set of objects representing primitive laboratory operations necessary for gravimetric, spectrophotometric, and volumetric analyses. Each laboratory operation object contains data and method attributes needed for the conceptual model. For example, the transfer object contains the data attributes amount, source, and destinations and the method attributes simulate_the_transfer. Each laboratory operation object also contains the attributes for the robot interface so that the primitive operation can be executed by a sequence of robot operations.

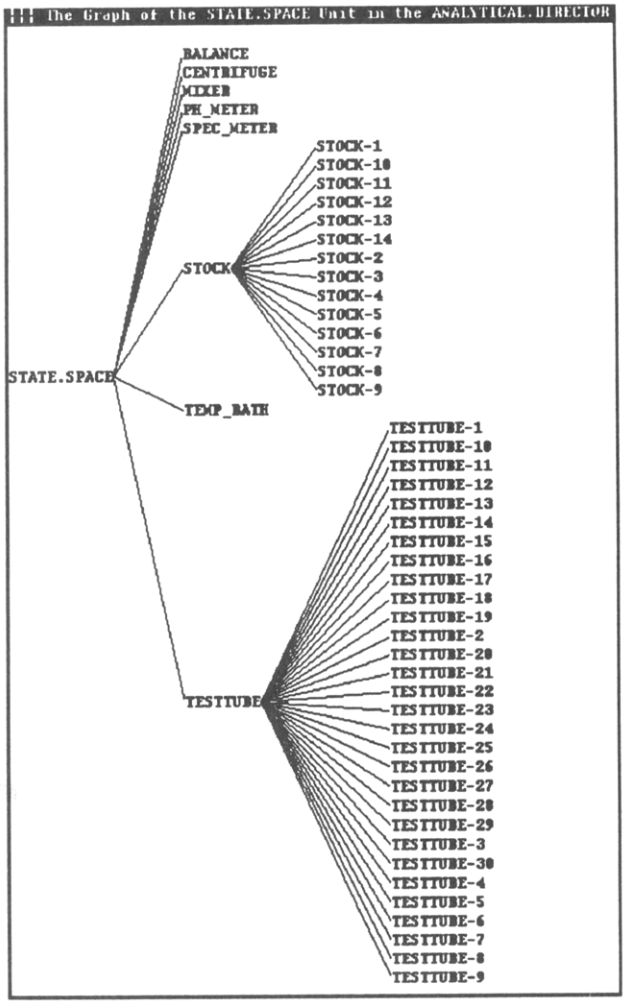

Figure 4. State.space hierarchical representation of the workbench.

The *sample* hierarchy captures more knowledge about laboratory procedures. Each procedure corresponds to a certain kind of sample: the element(s) being determined, the sample state, and the matrix. The information is important for the self-learning and automated procedure design envisioned for future implementation. Further, this hierarchy organizes the designed procedures by their target elements.

The *state.space* hierarchy organizes the objects representing the components of the workbench. Because of the limited window size, not all the components (objects) are displayed in Figure 3. Figure 4 shows the complete hierarchical structure of the state.space. Each object represents a laboratory instrument or device with its properties and "behavior". For example, consider the stock object. (Refer to Figure 5 for the data attributes and the method attributes.) The data attributes specify the content (a chemical reagent or a sample or empty) of a stock solution. The method attributes define the behavior of the stock object. (Stock object defines a class of objects, stock-1 to stock-14. The data and method attributes of stock-1 to stock-14 are automatically duplicated from the stock object.) As instances (examples) of stock object, stock-1 to stock-14 objects inherit all these data attributes and method attributes.

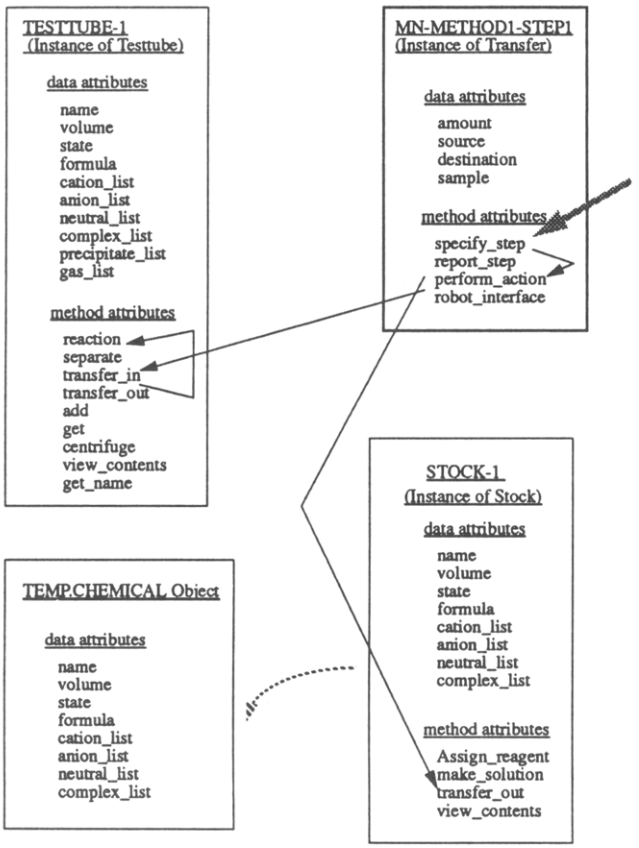

Figure 5. Passing of messages in generating a transfer step.

To initialize stock-1 to contain reagent 0.01 N $Mn(NO_3)_2$, the user can simply activate the method attribute assign_new_reagent of stock-1, and the behavior defined by *Assign new reagent* method will take over all the remaining work.

The Chemical Information Manager. The *Chemical Information Manager* (CIM) is an object-oriented database system that can manage the chemical information in an intelligent and dynamic way. Through the *cabinet* icon, a chemist can activate the CIM and perform data input, retrieval, query, and removal as with any other database management system. Thus, a user can input or update the chemical properties and chemical reactions about a new procedure before the procedure is designed. CIM is integrated with the other modules of the Analytical Director to enable a deep-reasoning-based knowledge inference.

CIM assists the procedure design process by supplying proper chemical properties. When a user assigns a reagent to a stock, CIM reads the chemical formula or the chemical name and then determines its type, e.g., acid, base, salt, etc. According to the chemical type, CIM searches the corresponding object class set to retrieve the relevant properties of the chemical. These properties are used to specify the contents of the stock. When a chemical is transferred into a test tube which already contains chemicals, CIM fetches the relevant data so that the *Chemical Reaction Generator* can simulate the chemical reactions.

In the current system, self-learning occurs in two ways. The system can acquire and update information about chemicals, chemical reactions, and analytical procedures. And, the system can modify the reasoning process by using such information. CIM plays an important role in acquiring and modifying chemical properties. When new chemicals and new chemical reactions are encountered in a design process, CIM compares the new data with the properties in the database and stores the information in the database automatically. When the robot finishes the execution of an analytical procedure, the results contain valuable information (how sensitive a method is, time needed for finishing the procedure, absorption peaks, etc.). By interacting with the Procedure Generator, CIM updates those which are already in the database and stores those which are not yet in the database.

The Chemical Reaction Generator. The *Chemical Reaction Generator* (CRG) predicts the chemical reactions when two or more chemicals are mixed together. It uses a chemical reaction algorithm and a rule-base system to generate all the possible reactions and then applies the constraints provided by the state.space to select the legal reactions.

In general, chemical reactions are complex thermodynamically and kinetically controlled processes. Many reactions would occur simultaneously in a test tube upon the mixing of several chemicals. One approach to computer simulation is to consider al the possible chemical reactions and "perform" them in a sequential order. Because most procedures of the three analytical methods would tend to avoid kinetically controlled reactions, such computer simulation could be successful. We define the *Reaction Possibility* (RP) as a property of a chemical reaction which determines the order in which it occurs among a set of possible reactions in the system being simulated. We have created a set of rules to approximately determine the reaction possibility. All rules have a consistent "if-then" form, such as:

IF (Reactant-1 is *cation* and reactant-2 is *anion*) and (*Cation* and *anion* forms *salt*)

THEN (*Product* is the *salt*) and (RP is the pK_{sp} of the *salt*).

Such forms can easily be used by a chemist to create or modify the rules on *reaction possibility*. When a rule is invoked, the CRG activate the CIM constantly to check the if-condition and to predict the reaction products and calculate the reaction possibility. For example, consider the combination of $AgNO_3$ and $FeCl_3$; the chemical reaction algorithm checks all the possible reactions between the pairs of Fe–NO_3, Fe–Ag, Ag–Cl, and NO_3–Cl. The only possible reaction results from the cation Ag^+ and anion Cl^-. CIM is invoked to confirm that AgCl is indeed a salt and that the pK_{sp} is 9.74, which is assigned to the reaction possibility by the rule above.

The current rule base covers four categories of inorganic reactions: acid–base reactions, precipitation reactions, coordinate reactions, and oxidation–reduction reactions. The reaction possibilities are calculated from pK_a, pK_b, pK_{sp}, $pK_{(unstable)}$, and standard reduction potentials. The complexity caused by pH effect is approximated by using formal pK_{sp}, $pK_{(unstable)}$, and formal standard potentials either in neutral, acidic, or basic solutions. For special reactions (like organic reactions) involved in some procedures, the current system stores them directly.

The Procedure Generator. The *Procedure Generator* (PG) plays a central role in the conceptual model. It interacts with the CIM and CRG and applies the constraints to the state space to help design and validate a procedure.

Shank and his colleagues[8] maintain that a small set of primitive actions will account for what must be represented in the physical world. Following this philosophy, we grouped the activity of standard laboratory robotics system into families of primitive actions. We list below the families of primitive actions necessary for volumetric, gravimetric, and spectrophotometric analyses.

ADD
CENTRIFUGE
CONTROL.PH
CONTROL.TEMPERATURE
DILUTE
MEASURE.ABSORBANCE
MEASURE.PH

SELECT.WAVELENGTH
SEPARATE
STIR
TITRATE
TRANSFER
WAIT
WEIGH

Each of these primitive operations is represented by an object. Each object has some data attributes and method attributes. The data attributes specify the parameters of an operation. The method attributes specify how the operation will change the state.space of the workbench. Each procedure step will be represented by an object which is an instance of one of these primitive operation objects.

Figure 5 shows how the step *MN-METHOD1-STEP1* (refer to Figure 3), "transfer 1 mL of 0.001 M $Mn(NO_3)_2$ from stock-1 to testtube-1", is specified by a user and accepted by the expert system. An object *MN-METHOD1-STEP1* is created as a child of both objects *MN-METHOD1* and TRANSFER. It is a child of *MN-METHOD1* because it is a step of the method. It is an instance of TRANSFER because this step is a transfer operation. As a child of the TRANSFER object, it inherits all the data attributes and method attributes of the transfer operation. The method specify_step of the *MN-METHOD1-STEP1* is first activated, which asks the user to specify the data attributes of the object, amount, source, destination, and sample. The method attribute perform_action is then activated after the user specifies these data attributes. The perform_action method first sends a message to transfer_out method of the stock-1 object, which "transfers out" 1 mL of the solution by assigning the relevant data attributes of the TEMP.CHEMICAL object. The perform_action method then sends a message to transfer_in method of the object testtube-1, which "transfers in" the chemical specified by the temp.chemical object. The reaction method of the object testtube-1 is then activated to perform all the possible chemical reactions between the old contents (specified by the data attributes of the testtube-1 object) and the transferred reagent (specified by the data attributes of temp.chemical object). Finally, the state of the destination test tube of the transfer operation, testtube-1, is updated by changing the values of the relevant data attributes of testtube-1.

As an object-oriented system, the process of designing a procedure is a process of creating step-objects as instances of certain primitive operation objects. As a result of inheritance, both the data and methods of the operation are available to simulate the chemical reaction, report a description about the step to a user, and control the robot execution.

Standard Robot Interface Protocol. Once a procedure is designed and validated, a robot procedure can be generated so that the laboratory robot can execute that procedure without human assistance or intervention. This is achieved by creating certain data and method attributes in both analytical.methods and laboratory.operations hierarchies.

Each primitive operation object has a method attribute execute_robot, which activates a robot interface function. In our current implementation, a Zenith 80286 PC is used to run the Zymate System V along with PyTechnology Software to control a Zymate II robot and other instruments. A serial communication link is established between the SUN workstation and the PC. The PC runs the System V in a remote-controlled mode so that it can both receive commands from and send results back to the SUN workstation. For each primitive operation, the robot interface function sends a sequence of robot commands to the PC which controls the robot laboratory to accomplish the operation defined in the conceptual model. In a production system, additional error checking and validation of the individual robot operations would be required. The expert system uses the results from the PC to assure the robot laboratory is in a proper working state. Because all these robot interface functions are hidden from users, such an expert system can be used by other robotics laboratories.

With the Analytical Director, it is possible to design and save a procedure in one laboratory, then execute the procedure in another robot laboratory. We envision a robot laboratory that can input an existing standard written procedure and convert it automatically for robotic execution.

There are at least four layers involved in the standardization of robot procedures. The first layer is the robotics laboratory physical resources needed for different analytical methods. Only if we have a standard basic set of laboratory resources needed for each analytical method can we have standardized robot interface commands to achieve the same work.

The second layer is the standard robot interface commands for the analytical laboratory. These include both syntax and semantics. The syntax specifies the format used to pass parameters and commands. The semantics specifies what kind of actions the robot laboratory needs to accomplish for a command. For example, if the robot control process receives a command of "measure absorbance of testtube-3 at 560 nm", the command might be "measure_absorbance" with two parameters, 3 and 560. The semantics needs to specify what would happen if the spectrophotometer is not on, if the spectrophotometer is on and wavelength is not set to 560 nm, and if the spectrophotometer is on and the wavelength is set at 560 nm.

The third layer is the idea of a conceptual laboratory for which a standard set of laboratory operations is defined. The computer simulation of a real laboratory occurs in this layer.

The fourth layer is the user interface built on top of the third layer. At this level only chemistry knowledge and concepts are required to design a procedure. Certain standards are needed for either natural language processing or pseudo chemistry language processing (like the icon-based user interface of this paper) to design procedures.

Such standardization needs a joint effort from both academic and industrial laboratories. We believe that such standardization will eventually result in the wide acceptance of robotics laboratories. The benefits of standard robot procedures are obvious. With the first and second layers standardized, a system such as ours can design and validate procedures and deliver them to other laboratories. These laboratories need not have an expert system to execute these standard robot procedures. With the first three layers standardized, different laboratories can have their own knowledge-based systems and design procedures for their own user interfaces, yet they can exchange their procedures and execute the designed procedures in other robotics laboratories. With all the four layers standardized, we will design procedures and store them in a language both chemists and robotics expert systems can understand and execute. Such standard robot procedures will ensure consistent results at different robotics laboratories.

AN EXAMPLE USING THE ICON-BASED INTERFACE

In this section, we will use the text given below to demonstrate how a procedure is designed, validated, and executed by the Analytical Director.

> *Place* 1 mL of sample of unknown Mn^{2+} with a pipet into a test tube. *Add* 2 mL of 6N HNO_3 to *acidify* the solution. *Add* 1 mL of KIO_4 and *boil* for 3–5 min. Cool to room temperature and *dilute* to 10 mL. *Measure* the absorbance of the unknown. *Prepare* a series of

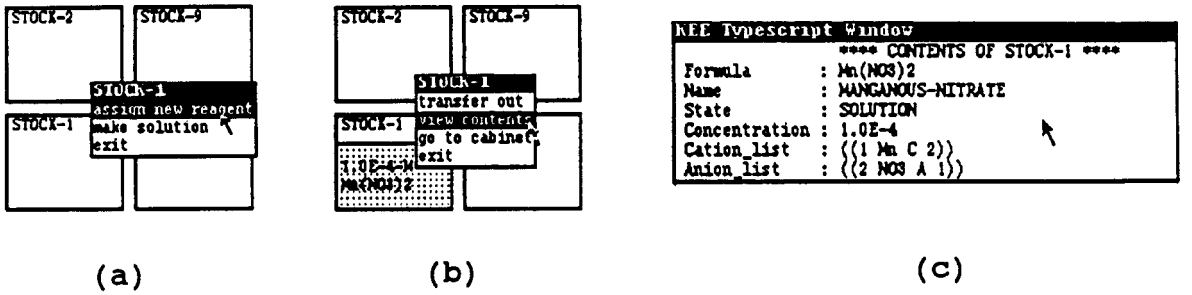

Figure 6. Icon manipulations to assign a stock and view its content.

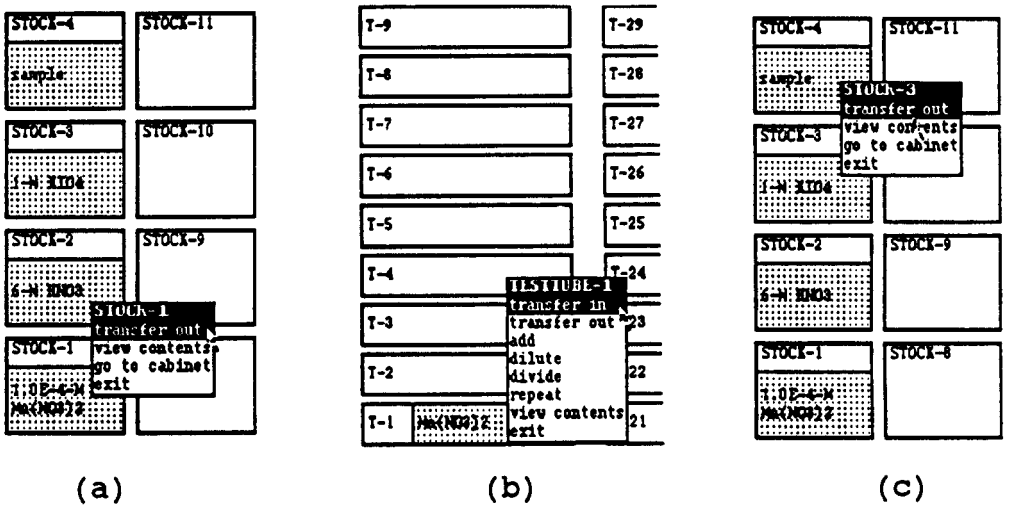

Figure 7. Icon manipulating for generating (a) step 1, (b) step 2, and (c) step 3.

standards in the same way and *measure* their absorbance.

A chemist might abstract the above procedure as follows:

(1) TRANSFER 1 mL of Mn sample to testtube-1

(2) TRANSFER 2 mL of 6N HNO_3 to testtube-1

(3) TRANSFER 1 mL of KIO_4 to testtube-1

(4) CONTROL.TEMPERATURE of Mn sample of testtube-1 to 100 °C for 5 min

(5) CONTROL.TEMPERATURE of Mn sample of testtube-1 to 25 °C

(6) DILUTE Mn sample to a volume for measuring absorbance

(7) MEASURE.ABSORBANCE of Mn sample of testtube-1

The purpose of the user interface is to assist a chemist to design such a conceptual procedure and to represent it within the expert system so that the robot can execute the procedure. After the procedure is designed, each of the steps will be represented by an object in the knowledge base (notice the seven step-objects in Figure 3).

Setting Up the Chemical Reagents for Designing a Procedure. A convenient way of designing a procedure is to start by assigning all the needed chemical reagents to stock solutions. Figure 6a shows the way to initiate stock-1 to contain 0.0001 M $Mn(NO_3)_2$ solution. Mouse clicking on the *stock-1* icon will activate the stock-1 object. Because stock-1 is an empty stock, stock-1 responds to the mouse clicking by popping up a menu as in Figure 6a. The menu contains the applicable operations to stock-1, assign new reagent, and make solutions. We choose assign new reagent. The user will then be prompted to input the chemical formula (or chemical name) of the reagent and the concentration. Upon receiving the chemical formula $Mn(NO_3)_2$, the CIM will analyze the chemical formula to determine the chemical type and find the relevant physical and chemical properties about the reagent from the database. According to its properties, the CIM determines its components list(s) and its state. These properties and components lists are then assigned to the data slots of the object stock-1, which implies that the stock-1 now contains a stock solution of 0.0001 M $Mn(NO_3)_2$.

After a stock reagent is successful initialized, the system graphically shows the new reagent of the stock (see Figure 6b). A user can check the properties of a newly assigned stock reagent by clicking on the stock (see Figure 6b) and selecting view contents of the pop-up menu. This menu selection would activate the method slot of view_contents of the object stock-1, which displays the properties and components of the reagent of stock-1 (see Figure 6c).

Similarly, a user can assign the other reagents. If a user inputs "sample" at the prompt of "input chemical formula or name", he or she will be allowed to specify both the target element(s) and the matrix components of a sample. Figure 7a shows the state of the workbench after all the reagents and the sample are initialized.

Designing a Procedure by Laboratory Simulation. To design the first step of the procedure, the user clicks on the *stock-1* icon. A menu (see Figure 7a) with all the possible operations for stock-1 pops up, from which the user chooses "transfer out". The user is then prompted to specify the amount being transferred and the destination. The amount, 1 mL, can be specified either from the *Lab Panel* or from the keyboard. The destination is given either from the keyboard or by clicking on the *T-1*, the icon representation of the testtube-1 object. If the destination test tube testtube-1 already contains chemicals, all the possible chemical reactions due to the newly transferred chemical will be predicted by the CRG module. The system then updates the state of the destination test tube by assigning the data attributes for the state of the content, volume, and each of the component lists along with its amount. If all the constraints (legal chemical reactions, total volume restriction, etc.) of the transfer_out operation are satisfied, step 1 is then accepted as a legal step. Figure 7b shows

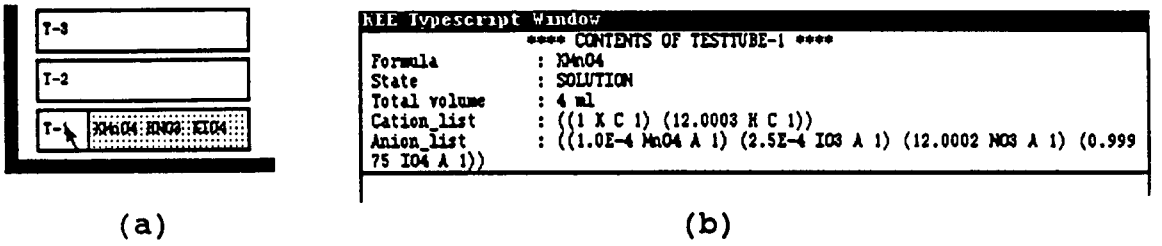

Figure 8. State of testtube-1 after step 3.

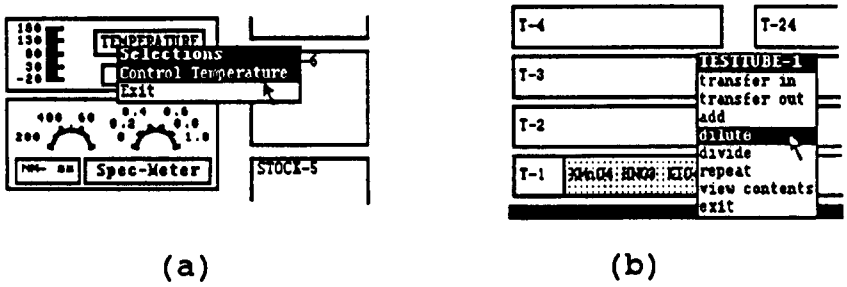

Figure 9. Icon manipulations for generating (a) step 4, step 5, and (b) step 6.

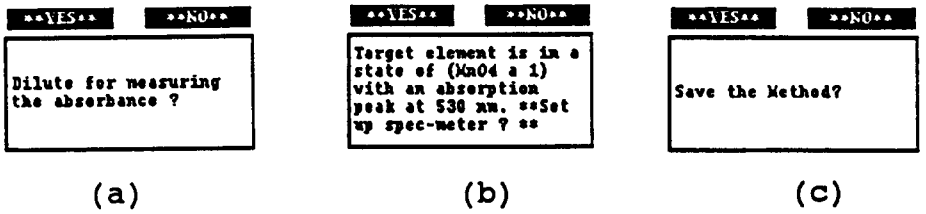

Figure 10. Explanatory mechanism for generating (a) step 6 and (b) step 7 and (c) saving the procedure.

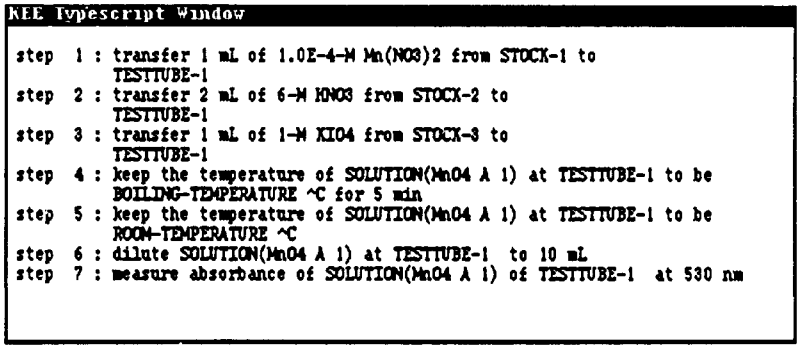

Figure 11. Literal description of the procedure.

graphically the state of the testtube-1 after the first step.

To design the second step, the user clicks on the testtube-1 and selects the option transfer in from the pop-up menu (see Figure 7b). This selection activates the transfer_in method of testtube-1 which prompts the user to specify where the solution is transferred from (stock-2) and the amount (2 mL). An alternative way of generating step 2 is by clicking on the *stock-2* icon and selecting the transfer out menu choice.

Step 3 can be generated by clicking on the *stock-3* icon and selecting the transfer_out option from the pop-up menu (see Figure 7c). After the destination (testtube-1) and amount (1 mL) of step 3 are specified, the CRG is activated. The CRG simulates the chemical reactions between the contents of testtube-1 and the 1 mL of KIO_4 solution being transferred. Only one possible reaction is generated, which is the reaction between anion IO_4^- and cation Mn^{2+} under acidic conditions. As the result of this chemical reaction, Figure 8a shows the graphic display of the contents of testtube-1 after step 3. Figure 8b shows a detailed description of the contents of testtube-1 after step 3.

Step 4 can be generated by clicking on the *water bath* icon and selecting the menu choice of control temperature (see Figure 9a). The system will then prompt the user to specify the temperature (boiling), the time (5 min), and the test tube to be controlled (testtube-1). Step 5 can be generated by the same icon manipulation as step 4, except no value for time needs to be assigned.

To design step 6, the user clicks on the *testtube-1* icon and selects the "dilute" action from the pop-up menu (see Figure 9b). This menu selection activates the dilute method slot of the testtube-1 object, which detects the state of the target element Mn to be permanganate MnO_4^-. (Notice the system representation of MnO_4^- as ("MnO_4" A 1) where A stands for *anion* and 1 stands for the negative charge of the anion.) The CIM is then activated to find out that permanganate has an absorption peak at 530 nm. The expert system concludes that the user is designing a procedure using a spectrophotometric method. The system then asks the user if his intention in step 6 is to dilute the test tube for measuring the absorbance (see Figure 10a). If the user clicks on the YES box, then step 6 is generated. Figure 10b shows how the system goes on to prompt the user if the measurement is to be made at 530 nm. Clicking on the YES box generates the last step, step 7 of the procedure.

As each step of a procedure is designed, the system reports a brief description of the operation. Figure 11 shows the

```
KEE Typescript Window
          *****    MN-METHOD1    *****
step  1 : transfer 1 mL of Sample (Mn(NO3)2) from STOCK-4 to
          TESTTUBE-3
step  2 : transfer (1 2 3) mL of Standard(Mn(NO3)2) from STOCK-1 to
          (TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)
step  3 : transfer 2 mL of 6-M HNO3 from STOCK-2 to
          (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)
step  4 : transfer 1 mL of 1-M KIO4 from STOCK-3 to
          (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)
step  5 : keep the temperature of SOLUTION(MnO4 A 1) at (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6) to be
          BOILING-TEMPERATURE ^C for 5 min
step  6 : keep the temperature of SOLUTION(MnO4 A 1) at (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6) to be
          ROOM-TEMPERATURE ^C
step  7 : dilute SOLUTION(MnO4 A 1) at (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)  to 10 mL
step  8 : measure absorbance of SOLUTION(MnO4 A 1) of (TESTTUBE-3 TESTTUBE-4 TESTTUBE-5 TESTTUBE-6)  at 530 nm
```

Figure 12. Complete robot procedure for Mn analysis.

English description of the designed procedure.

Validating a Procedure. Once a new procedure is designed, several requirements will be checked before a procedure is approved by the system. First, it checks to see if the target element is in the correct state at the end of the procedure. Second, it checks to see if the operation of each step is applicable. For example, the system would reject a step of transferring 2 mL of solution from a test tube with contents being a mixed precipitate. Third, operations that are applicable but not logical are rejected. An example is centrifuging a test tube with no further operation on the same test tube.

After the system validates the procedure, the user is prompted to save the new procedure (see Figure 10c).

Generating and Executing the Robot Procedure. After the user issues the command to start the robot analysis, the *Procedure Generator* (PG) develops and refines the detailed robot procedure from the designed procedure. The following steps are applied sequentially to produce and execute the robot procedure.

(*1*) *The proper concentration range for the standard is determined.* For the example above, this procedure is same as in Figure 11. That is because we have used stock-1 (the standard) in designing the procedure. Then the procedure is executed through the robot interface, and the result of step 7 is returned from the robot measurement. If the absorbance of the standard is too large, say 1.305, the PG would generate the robot procedure to dilute the standard solution at stock-1 then use the diluted standard to repeat steps 1–7 of the designed procedure.

(*2*) *The proper concentration range for the sample is determined.* After the standard is determined (or prepared by the robot) in its proper concentration range, the PG will then generate the robot procedure to determine (or prepare if needed) the sample(s) in the proper concentration range. The robot procedure generated is almost same as the one in Figure 11 with the exception of the source of step 1 being a sample (stock-4). The absorbance of the sample from step 7 is then returned. Sample dilution might be needed depending on the value of the absorbance. The preliminary result of the sample concentration is then reported.

(*3*) *The complete robot procedure is optimized.* Figure 12 shows the complete robot procedure generated by the expert system. Samples and standards with proper concentrations are used to generate this procedure. Optimization of the robot execution time and best experimental conditions, such as using the same time for color development in samples and standards, are taken into consideration. The user can change the number of standards if desired. In this example, three standards are used to assure a good linearity of the absorbance.

After the robot finishes its execution, it generates the following information: the kind of sample the current procedure is capable of determining; how long the procedure takes to analyze a sample; special properties of the procedure, such as the detection limit and linearity; and, the absorption peak and the molar absorptivity of the complex species if they are involved in the procedure. This robot-generated information is used to evaluate a newly designed procedure and to update the chemical information database. The result of the robot execution is used to produce the report of the analysis. Such analysis reports can be customized to meet different requirements.

CONCLUSIONS

An icon-based user interface has been developed for the Analytical Director to enhance its capability to design, test, modify, and implement an analytical procedure. The developed system employs object-oriented modeling techniques to simulate an analytical laboratory. The conceptual laboratory includes knowledge of chemicals, chemical reactions, laboratory operations and analytical procedures as a set of hierarchical structured objects. With the icon-based user interface, a user can interact with the conceptual laboratory and design a procedure using high level chemistry language (or notation), while the expert system validates the procedure by simulating the robot execution and the chemical reactions. Once a procedure is approved as valid, the expert system can convert the conceptual procedure into a sequence of robot commands and execute the robot procedure without human intervention. By comparing the workbench of the simulated laboratory and the real robotic table, the system can learn from the robot execution and update the chemical information database and improve its decision making processes. With a multilayer structured standard robot interface protocol, the designed procedures are stored both in conceptual description format and as sequences of robot commands. A PC-based implementation is being developed in our laboratory to demonstrate that such standard robot procedures can be executed by a microcomputer-controlled robotic laboratory to accomplish the same analysis purposes.

The developed system has been used to convert textbook analytical procedures and to design new procedures. We find designing procedures using icon operations at a conceptual level is convenient and efficient. Spectrophotometric procedures for analyzing some of the common metallic elements such as Mn, Cu, Fe, Ni, Co, Mg, and Ag have been designed and executed by the robot laboratory successfully. The system is accumulating more designed procedures and assimilating new chemicals and chemical reactions into its knowledge base. If a sample can be analyzed by an already existing procedure, a user can simply retrieve the procedure and execute the robot procedure with the respecified standard and sample locations on the workbench.

REFERENCES AND NOTES

(1) Bleyberg, M. Z.; Zhou, T.; Isenhour, T. L.; Marshall, J. C. The Design and Implementation of an Analytical Expert System. *Proceedings of*

the Third International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems; July 1990; ACM Press: Charleston, SC, 1990; pp 1073–1079.
(2) Bleyberg, M. Z.; Zhou, T.; Isenhour, T. L.; Marshall, J. C. An Expert System Model for Analytical Chemistry. *Int. J. Appl. Artif. Intell.*, accepted for publication.
(3) The Development of a Robotic Standard Addition Method. Eckert-Tilotta, S. E.; Isenhour, T. L.; Marshall, J. C. *Anal. Chim. Acta*, accepted for publication.
(4) The Application of Standard Robotic Methods for Water Analysis. Lee, J. R.; Isenhour, T. L.; Marshall, J. C. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 000–000.
(5) Wirfs-Brock, R.; Wilkerson, B.; Wiener, L. Designing Object-Oriented Software; Prentice Hall: New York, 1990.
(6) Tello, E. Object-Oriented Programming for Artificial Intelligence; A Guide to Tools and System Design; Addison-Wesley Publishing Co.: Reading, MA, 1989.
(7) Smith, M.; Tockey, S. An Integrated Approach to Software Requirements Definition Using Objects; Boeing Commercial Airplane Support Division: Seattle, WA, 1988; p 132.
(8) Shank, R.; Riesback, C. R. Inside Computer Understanding; Lawrence Erlbaum Associates: Hillsdale, NJ, 1981.

Wiener and Randić Topological Indices for Graphs

BERTHOLD J. MAIER

IDC Internationale Dokumentationsgesellschaft für Chemie, Otto-Volger-Strasse 19, W-6231 Sulzbach, Germany

Received June 27, 1991

We present counterexamples to two conjectures on the monotony of the Wiener and Randić indices for graphs raised in a recent paper,[1] and we prove that the conjecture on the Randić index holds for graphs with vertex degrees bounded by 8 and a strengthening holds for the chemically relevant graphs with vertex degrees bounded by 4.

INTRODUCTION

For an unoriented connected graph G with vertex set V, the Wiener index[2] χ_W is defined by

$$\chi_W(G) = \sum d(v,v')/2$$

where the sum is taken over all vertex pairs $v,v' \in V$, and $d(v,v')$ denotes the distance, i.e., the length of the shortest path, between v and v' in G.

The Randić index[3] χ_R is defined for an unoriented graph G by

$$\chi_R(G) = \sum 1/\sqrt{\deg(v)\,\deg(v')}$$

where the sum is taken over all edges vv' of G, and $\deg(v)$ denotes the vertex degree of v in G.

These are two of the many indices[4,5] which have been proposed in order to correlate macroscopic properties with graph theoretical invariants of the structural formula of the organic molecule. In a recent paper[1] the following conjectures were raised:

$$\chi_W(G') < \chi_W(G) \quad (1)$$

$$\chi_R(G') < \chi_R(G) \quad (2)$$

where G is a connected graph with $n \geq 2$ vertices and G′ is the subgraph obtained from G by deleting the last vertex in a canonical labeling of G. In ref 1 a labeling of G is called canonical if (a) it is a cooperative labeling,[1,6] i.e., starting with a vertex v_0 we next label the neighbors of v_0 in a given order, then the neighbors of the neighbors in the chosen order and so on and (b) the lower triangular submatrix of the adjacency matrix is lexicographically maximal when its rows are concatenated to a string. The conjectures 1 and 2 have been checked for all graphs with up to 9 vertices.[1]

In this paper we give counterexamples to both conjectures, but we show that conjecture 2 holds for graphs with vertex degrees bounded by 8 and a strengthening of conjecture 2 is true for graphs with vertex degrees bounded by 4. Our examples show that these bounds are optimal. The counterexample for conjecture 1 is a cycle with vertex degrees equal to 2.

THE WIENER INDEX

Let P_n denote a straight chain with n vertices and C_n a simple cycle with n vertices. Clearly, $C_n' = P_{n-1}$ in the notation of the conjecture. Now, using $1 + 2 + \ldots + i = \binom{i+1}{2}$ we obtain

$$\chi_W(P_n) = \sum_{i=2}^{n}\binom{i}{2} = \binom{n+1}{3} \quad (3)$$

$$\chi_W(C_{2n}) = n[\binom{n}{2} + \binom{n+1}{2}] = n^3 \quad (4)$$

$$\chi_W(C_{2n+1}) = (2n+1)\binom{n+1}{2} \quad (5)$$

The following gives the values of χ_W for some of these graphs which disprove conjecture 1:

G	P_9	P_{10}	P_{11}	C_{10}	C_{11}	C_{12}
χ_W(G)	120	165	220	125	165	216

THE RANDIĆ INDEX

We start with an example that was also considered independently by S. Tratch.[7] Let G_n denote the graph consisting of two bridge-heads A,B joined by n chains with two vertices each. Thus, G_n has $2n + 2$ vertices, the vertices A,B have degree n, and the other vertices have degree 2. In a canonical labeling one bridge-head, say A, has label 1, and the other bridge-head, B, has the last label. This holds, because after A we first label all the neighbors of A, the first elements in the chains, then the second elements in the chains, and finally we label B as the last element. Therefore, $G_n' = G_n - \{B\}$.

Now B is connected in G_n to n elements of degree 2, which each are connected to the second element of degree 2 in the respective chains. In G_n' the elements to which B is connected in G_n all have degree 1. Therefore, we have the contribution $n/\sqrt{2}$ from the chain bonds to $\chi_R(G_n')$ in G_n'. In G_n we have the contribution $n/2$ to $\chi_R(G_n)$ from the chain bonds, and $n/\sqrt{2n}$ from the bonds incident with B. The bonds incident with A give the same contributions to $\chi_R(G_n')$ and $\chi_R(G_n)$. Hence conjecture 2 holds for G_n if and only if

$$n/\sqrt{2} < n/2 + n/\sqrt{2n} \quad (6)$$

But this is false for $n \geq 6 + 4\sqrt{2}$.

0095-2338/92/1632-0087$03.00/0