PROJECT REPORT

PART A –

PREDICTION AND CLASSIFICATION OF ORGANIC REACTIONS USING EXTENDED UGI'S SCHEME

PART B -

CHEMOINFORMATICS USING NEURAL NETWORKS

Submitted by:

SWETA SHARMA (ID – 110509042)
SWASTIK MUKHERJEE (IS-110509044)
B.E IN COMPUTER SCIENCE AND TECHNOLOGY
8th SEMESTER(2009-2013)

Under the supervision of

Professor Somnath Pal
Department of Computer Science and Technology
Bengal Engineering and Science University , Shibpur

BONAFIDE CERTIFICATE

This is to certify that "***Chemoinformatics using Neural Networks***" is a bonafide work carried out by **Miss Sweta Sharma** and **Mr. Swastik Mukherjee** under the guidance of **Professor Somnath Pal** , Assistant Professor of Department of Computer Science and Technology , Bengal Engineering and Science University , Shibpur.

I hereby attest that the project is original and has not been submitted or published anywhere else.

Dated:

Sweta Sharma ID-110509042
Swastik Mukherjee ID-110509044
BESU, SHIBPUR
8th SEM, CST(2009-2013)

Professor Somnath Pal
Dept . Computer Science & Technology
BESU , SHIBPUR

ACKNOWLEDGEMENT

An endeavour over a long period can be successful only with the advice and guidance of all the well-wishers , some directly through their technical assistance and some through their encouragement and help.

We take this opportunity to convey our heart-felt thanks and deep sense of gratitude to all who encouraged and stood by our side in completing this project.

We are also thankful to **Professor Somnath Pal** for all his active help and constant motivation in the direction of completing the project. Further we are especially thankful to Ms. Shyantani Maiti(ME Department of Computer Science & Technology BESU , Shibpur) for her active guidance and support for completion of our project successfully.

Last but not the least, we are thankful to all our friends and our families who have directly or indirectly helped us in completing the project.

By,
Sweta Sharma
Swastik Mukherjee
BESU , Shibpur
Computer Science And Technology
8th Semester(2009-2013)

TABLE OF CONTENTS

BONAFIDE CERTIFICATE....2

ACKNOWLEDGEMENT....3

CESC LIMITED....5

INFORMATION TECHNOLOGY....8

PROJECT TITLE....9

INTRODUCTION....9

SCOPE....10

OBJECTIVES....10

APPLICATION DEVELOPED....11

DEVELOPMENT TOOLS....12

DATABASE....13

BRIEF SUMMARY OF THE APPLICATION....14

FLOWCHART....19

DATAFLOW DIAGRAM....20

CODE SNIPPETS....21

SCREENSHOTS....73

CHALLENGES DURING DEVELOPMENT....88

LIMITATIONS AND FUTURE ENHANCEMENT....89

BRIEF DESCRIPTION OF THE DEVELOPMENT TOOLS....90

CONCLUSION....102

BIBLIOGRAPHY....103

PART A

PREDICTION AND CLASSIFICATION OF ORGANIC REACTIONS USING EXTENDED UGI'S SCHEME

INTRODUCTION

Chemistry, like any Scientific discipline, relies heavily on experimental observations, and therefore on data . In chemistry, the enormous increase in the number of compounds and the data concerning them resulted in increasingly ineffective data-handling , on the side of the procedures as well as the users. One way out of this disaster is the electronic processing , by computer methods , of this huge amount of data available in chemistry. The main challenge of electronic processing of this data is to represent molecules in computer . The molecule species consist of atoms and bonds that hold them together . Moreover, compounds can be interconverted into other compounds by chemical reactions .

The 2D graphical representation of chemical structures in structure diagrams can be considered to be the universal natural language of chemists . These structure diagrams are model and are designed to make the molecules more conceivable . In such a model, the atoms are typified by their atomic symbols and the bonding electrons by lines. However, the chemical structure diagram is an incomplete and highly simplified representation of a molecule . It only explains the topology (which atoms are connected by which bond type) and not the 3D arrangement (topography) of the atoms in a molecule .

One of the major tasks of chemoinformatics is to predict the outcome of a chemical reaction . For this purpose the first step is to classify existing chemical reactions , on the basis of which unknown reactions can be predicted . In the mid 1970s , Ugi and his co-workers developed a technique to represent chemical reactions by means of Reaction matrices (R-matrices) . Later researchers had studied several reaction schemes and finally , Ugi's work , which classify chemical reactions into 30 different classes , came to be known as Ugi's scheme . In this project we have proposed an efficient algorithm to automate classification of chemical reactions based on Ugi's scheme . In the following subsections we first elaborate the representation of molecules and chemical reactions using graph theoretic techniques -

Bond-Electron matrix (BE-matrix) and Reaction matrix (R-matrix) respectively .

OBJECTIVES

We have developed an efficient algorithm based on a model-driven approach developed by Ugi and his co-workers for classification of chemical reactions . Our algorithm takes educts of a chemical reaction as input and generates its appropriate products as output . There were several reactions which could not be classsified by Ugi's reaction classes so using this model driven approach some new reaction classes have been found .This gives an extension to the existing Ugi's reaction classes .

Ugi's Reaction Classes consists of 30 reaction classes . Many reactions have been found from The Chemical Thesaurus(a chemical reaction database) that could not be classified into the reaction classes according to Ugi's Scheme, so many new reaction classes have been found from a database of reactions which have been used as an extension to Ugi's Scheme . More 26 new reaction classes have been found which will help in more efficient classification of various chemical reactions .

Ugi's Scheme deals mainly with the bond electron matrices of the educt and product from which we can easily calculate and find the reaction matrix of chemical reaction. According to the reaction matrix of a particular reaction, the reaction class of that particular reaction is determined .

We have developed an algorithm to automate the construction of educt matrices , prediction of possible products of chemical reactions and then analysis of the products to generate stable products of the reaction . The reactants are given as input and as output we get a set of probable products. An algorithm has been developed for prediction of chemical reactions. This algorithm has been developed based on 30 Ugi's reaction classes and 26 new reaction classes of extended Ugi's scheme .

PREVIOUS WORK

The previous work involves study on several naming conventions of chemical compounds such as IUPAC name , line notations , SMILES coding , etc. Chemoinformatics represent chemical structures . It transfer various types of representation of chemical structures into application programs . It transforms molecular structure to language understandable computer representation. Chemical nomenclature denotes compounds in order to reproduce and transfer them from one coding to another .

Chemical structures are usually stored in a computer as molecular graphs . A graph is an abstract structure that contains nodes connected by edges . In a molecular graph the nodes correspond to the atoms and the edges to the bonds . A graph can also be represented as a matrix . Thus, quite early on, matrix representations of molecular structures were explored . Their major advantage is that the calculation of paths and cycles can be performed easily by well-known matrix operations. The matrix of a structure with n atoms consists of an array
of n X n entries. A molecule with its different atoms and bond types can be represented in matrix form in different ways depending on what kind of entries are chosen for the atoms and bonds. Thus, a variety of matrices has been proposed: adjacency, distance, incidence, bond, and bond-electron matrices.

Reactions represent the dynamic aspect of chemistry, mainly the interconversion of chemical compounds . Careful experiments are planned to analyze the individual steps of a reaction. Objective of chemoinformatics is to assist chemists in giving access and knowledge of chemical reactions . Reaction classification serves to combine several reaction instances into one reaction type . In this way , a vast number of observed chemical reactions is reduced to man- ageable number of reaction types . There are two approaches for reaction classification : 1.Model Driven approaches and 2.Data Driven approaches. In this case only Model Driven approach has been used .

A BRIEF OVERVIEW

Matrix for representation of chemical reaction

Representing atoms as nodes and bonds as edges, a molecule can be represented as graph , which is called molecular graph [2].Such graphs can also be represented as matrices [11] .The matrix of a structure with n atoms consists of an array of n_n entries. A molecule with its di_erent atoms and bond types can be represented in matrix form in di_erent ways depending on what kind of entries are chosen for the atoms and bonds [19]. Thus, a variety of matrix representations for chemical compounds are in use [7]: adjacency, distance, incidence, bond, and bond-electron matrices. Among these representations bond-electron matrix representation provides all information given by bond matrix and additionally number of free electrons of an atom. In this work we have used BE-matrix, the representation of which is explained in the following .

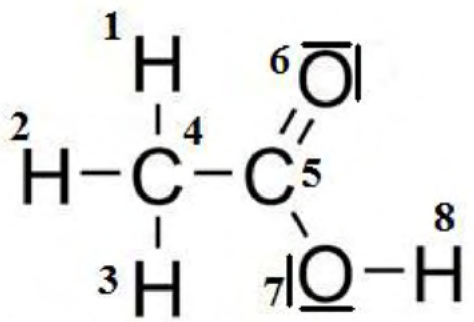


Fig 1: Acetic acid with free electrons where electron pair is indicated by a line.

	1	2	3	4	5	6	7	8
1	0	0	0	1	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	1	0	0	0	0
4	1	1	1	0	1	0	0	0
5	0	0	0	1	0	2	1	0
6	0	0	0	0	2	4	0	0
7	0	0	0	0	1	0	4	1
8	0	0	0	0	0	0	1	0

Table 1: BE MATRIX OF ACETIC ACID

Bond Electron Matrix

-

The bond-electron matrix (BE-matrix) was introduced in the Dugundji-Ugi model . The BE-matrix of a molecule with n atoms is an n x n symmetric matrix, with 0th row/column representing each atom of the molecule. It indicates bond order in the off-diagonal elements and the number of free valence electrons of the corresponding atom in the diagonal elements (e.g., O6 = 4 in Table 1). The graph in Figure 1 represents the acetic acid but additionally incorporates free electrons of two oxygen atom and corresponding bond electron matrix is shown in the Table 1. BE-matrices can be constructed not only for single molecules but also for ensembles of them, such as the starting materials of a reaction or final product(s) of the reaction. Consider the reaction of Figure 2, where Educts (E) ethylene and hydrogen-bromide produce the Product (P) ethyl-bromide. Table 2 shows the corresponding BE-matrices of educt E and product P of the reaction in Figure 2.

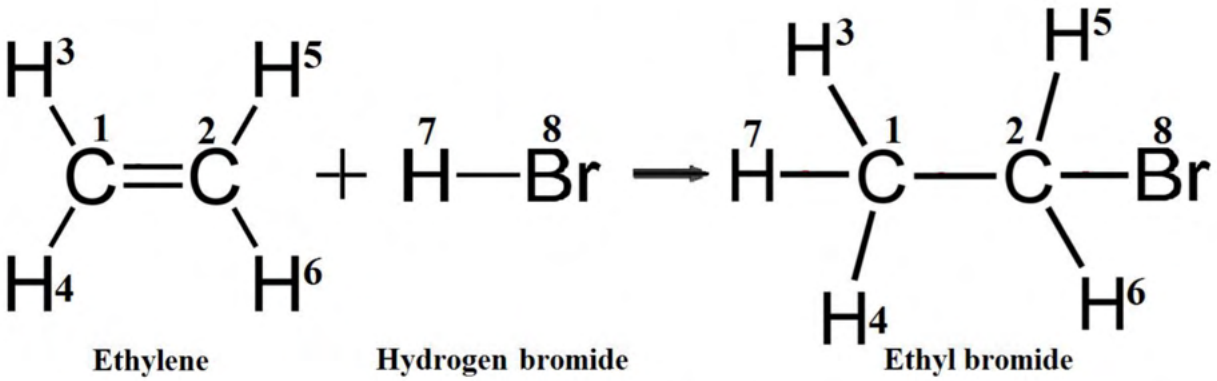


Fig 2 : The reaction of ethylene and hydrogen bromide to give ethyl bromide

E=

	C	C	H	H	H	H	H	Br
C	0	2	1	1	0	0	0	0
C	2	0	0	0	1	1	0	0
H	1	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
H	0	1	0	0	0	0	0	0
H	0	1	0	0	0	0	0	0
H	0	0	0	0	0	0	0	1
Br	0	0	0	0	0	0	1	6

P=

	C	C	H	H	H	H	H	Br
C	0	1	1	1	0	0	1	0
C	1	0	0	0	1	1	0	1
H	1	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
H	0	1	0	0	0	0	0	0
H	0	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
Br	0	1	0	0	0	0	0	6

Table 2 : Representation of chemical reaction using BE-matrix. The reaction of ethylene and hydrogen bromide acid to give ethyl bromide (Figure 2)

Reaction Matrix

Once the BE-matrices of the educt E, and the product P, of a reaction have been determined one can calculate R = P – E . The R-matrix corresponding to chemical reaction shown in Figure 2 is shown in Table 3 . As can easily be seen from the R-matrix of Table 3 , the entries r_{ij} indicate the bonds broken and made in the course of this reaction . An R-matrix expresses the bond and electron rearrangement in a reaction . The negative entries indicate the breaking of that many numbers of bonds between the corresponding atoms and positive entries similarly represent making of that many number of bonds between the corresponding atoms .

This R-matrix , which is a symmetric matrix , is called reaction matrix . The sum of all the entries in an R-matrix must be zero . The R-matrix of Table 3 reflects a reaction Scheme , the breaking and making of two bonds in the chemical reaction shown in Figure 2.

	C	C	H	H	H	H	H	Br
C	0	-1	0	0	0	0	1	0
C	-1	0	0	0	0	0	0	1
H	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	-1
Br	0	1	0	0	0	0	-1	0

Table 3: R-matrix of the reaction in (Figure 2)

Classification of Chemical Reaction

Classification of chemical reactions combine several reaction instances into one reaction class . In this way, a large number of known chemical reactions are reduced to a manageable number of reaction classes . There are two approaches for reaction classification : **1. Model-Driven Approach** and **2. Data-Driven Approach** . In this work we have considered the Model-Driven Approach . Among the several reaction classification schemes Hendrickson's scheme and Ugi's scheme are widely used in chemoinformatics . However , Hendrickson's scheme concentrated mainly on C-C bond-forming reactions and has 7 classes , whereas Ugi's scheme is more general , it is applicable to a wide variety of chemical reactions , and has 30 classes .

Ugi's scheme for reaction classification

Systematic studies on Ugi's scheme and how they are realized in organic reactions were performed . A printed compilation of 1900 reactions dealing with the introduction of one carbon atom bearing a functional group was analyzed and each reaction assigned manually to a corresponding reaction class of Ugi's scheme . The results are listed in Table 4 , which is known as the Ugi's scheme . Clearly , this choice of a reference set of organic reactions in was arbitrary , not necessarily representative of the whole set of organic reaction types described in the literature , and therefore not free from bias .

Reaction class	Reaction type
R23	A-B $\longrightarrow$ A + B:
R32	A: + :B $\longrightarrow$ A=B
R11	A-B + C $\longrightarrow$ A-C + B
R21	A-B + C: $\longrightarrow$ A-C-B
R33	A + :B-C $\longrightarrow$ A-B + C:
R12	A: + B-C $\longrightarrow$ A-C + B:
R25	A + B-C + D: $\longrightarrow$ A-B + C-D
R1	A-B + C-D $\longrightarrow$ A-C + B-D
R3	A-B + C-D $\longrightarrow$ A + B-C + D:
R5	A-B + :C-D $\longrightarrow$ A-C-B + D:
R8	A-B + C-D + E: $\longrightarrow$ A-C + D-E + B:
R34	A=B $\longrightarrow$ A: + B:
R35	A=B + C: $\longrightarrow$ A=C + B:
R36	A=B + C: + D: $\longrightarrow$ C-A-D + :B
R37	A=B + C-D + E: $\longrightarrow$ C-A-D + :B-E
R31	A=B + C-D + E-F $\longrightarrow$ C-A-E + D-B-F
R2	A-B + C-D + E-F $\longrightarrow$ A-C + D-E + B-F
R15	A-B + C-D + :E-F $\longrightarrow$ A-E-D + B-C + F:
R28	A-B + C-D + E-F + G: $\longrightarrow$ A-F + B-D + E-G + C:
R17	A-B + C-D + E-F + G-H $\longrightarrow$ A-D + B-H + C-E + F-G
R38	A-B + C-D + E-F + G-H $\longrightarrow$ A-C + B-G + D-E-H + F:
R30	A-B + C-D + E-F + G-H + I: $\longrightarrow$ A-I + B-D + C-F + E-G + H:
R22	A-B-C $\longrightarrow$ A-C + B:
R7	A-B-C + D-E $\longrightarrow$ D-B-E + A-C
R9	A-B-C + D-E $\longrightarrow$ A-D + E-C + B:
R39	A-B-C + D-E + F-G + H: $\longrightarrow$ A-F + C-H + E-B-G + D:
R40	A-B-C + D-E + F-G + H-I $\longrightarrow$ A-G + C-D + E-F + H-B-I
R41	A-B-C + D-E-F $\longrightarrow$ A-F + B=E + C-D
R19	A-B-C + D-E-F + G-H $\longrightarrow$ A-F B=E + C-H + D-G
R10	A-B-C + D=E $\longrightarrow$ A-D-C + B=E

Table 4 : Ugi's reaction classes

Extension of Ugi's scheme

The reactions that Ugi's scheme failed to classify

From The Chemical Thesaurus (a chemical reaction database) many reactions have been tested to classify them into Ugi's scheme . But at the same time it has been found that some reactions cannot still be classified under the Ugi's scheme . Some example reactions that cannot be classified under Ugi's scheme is shown in Figure 3 to Figure 5 .

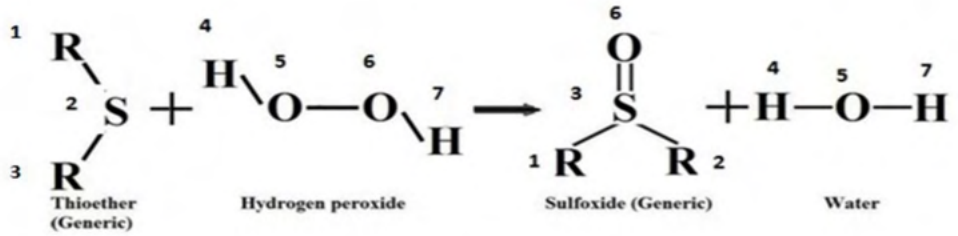


Figure 3 : The reaction of Thioether (Generic) and Hydrogen peroxide to give Sulfoxide (Generic) and water

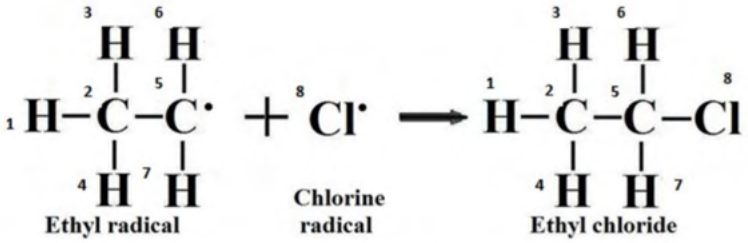


Figure 4 : The reaction of Ethyl radical and Chlorine radical to give Ethyl chloride

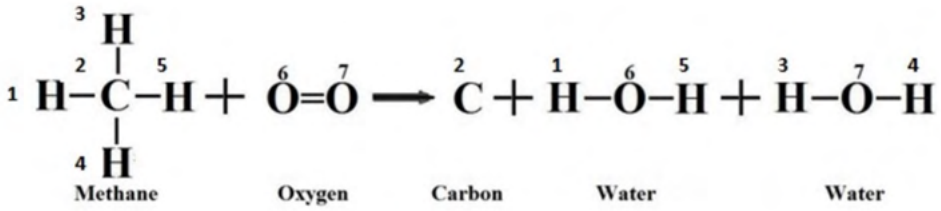


Figure 5 : The reaction of Methane and Oxygen to give Carbon and water

<u>The new reaction classes</u>

-

The reactions in Figure 3, Figure 4 and Figure 5 are not classified into Ugi's reaction class . Using these reactions new reaction classes were found . The tables on the next page lists the new reaction classes .

Reaction Class	Reaction Type
R50	A.+ B. $\longrightarrow$ A-B
R51	A-B-C. $\longrightarrow$ A. + B=C
R52	A-B-C. $\longrightarrow$ A. + .B-C.
R53	A. + .B + :C $\longrightarrow$ A-C-B
R54	A-B-C + D: $\longrightarrow$ A.+ B=D + C.
R55	A: + B-C-D $\longrightarrow$ A=C + B-D
R56	A-B + C-D $\longrightarrow$ A-D + C. + B.
R57	A-B-C + D-E-F $\longrightarrow$ A-E-C + D-B-F
R58	.A-B + C-D-E $\longrightarrow$ A=D + B-C + E.
R59	A-B-C-D + E-F $\longrightarrow$ A-E +B=C + D-F
R60	A-B-C + D-E + F-G $\longrightarrow$ D-B-F +E-A + G-C
R61	A=B + C. $\longrightarrow$ C-A-B.
R62	A=B + C. +D. $\longrightarrow$ C-A-B-D
R63	:A=B + C-D $\longrightarrow$:A.-C +B=D
R64	A=B +C-D $\longrightarrow$ .A-B-C + D.
R65	A=B-C-D + E-F $\longrightarrow$ D-A-E + F-B=C

Table 5 : New reaction classes from R50 to R65(Part 1)

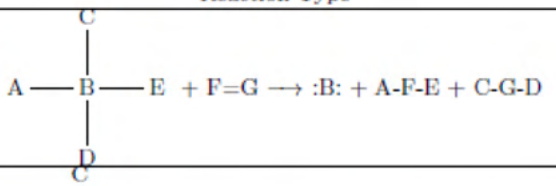
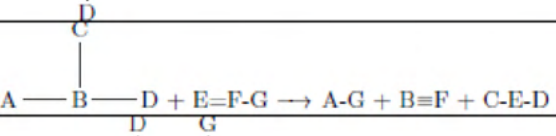
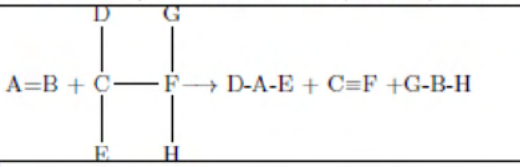
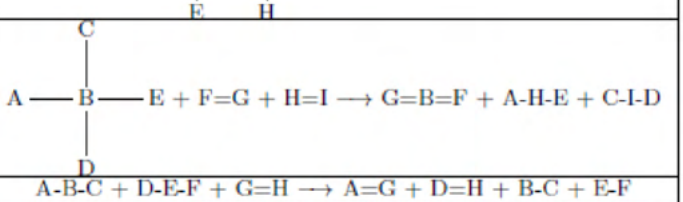
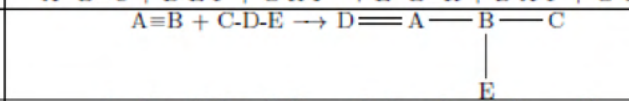
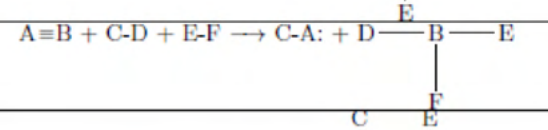
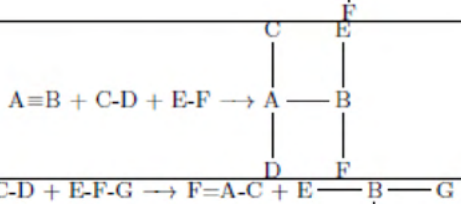
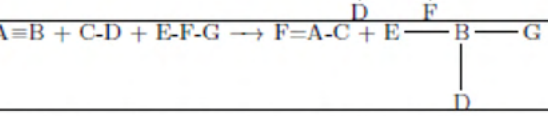
Reaction Class	Reaction Type
R66	
R67	
R68	
R69	
R70	A-B-C + D-E-F + G=H $\longrightarrow$ A=G + D=H + B-C + E-F
R71	A=B=C + D-E-F + G-H-I $\longrightarrow$ E=B=H + D-A-F + G-C-I
R72	
R73	
R74	
R75	

Table 6 : New reaction classes from R66 to R75(Part 2)

APPLICATION DEVELOPED

APPLICATION : Prediction and classification of chemical reactions .

FEATURES OF THE APPLICATION

- **EDUCT MATRIX**

  Generates the educt matrix from the input reactants .

- **PREDICTION**

  Predicts the possible products of the input reaction .

- **ANALYSIS**

  Analyzes the possible products for stability and then generates the possible stable products of the input reaction .

DEVELOPMENT TOOLS

Platform (OS) :	Linux (Fedora 16) .
Database :	Mysql.
Database Connection:	JDBC.
Software used:	1. JDK 6.0. 2. Jre 1.6. 3. Lex tool.
Programming & Scripting Languages used:	1. Java. 2. C .
Text Editor:	1. 'gedit' editor in Linux. 2. Text pad in Windows.

DATABASE

NAME OF THE DATABASE - chem

TABLES USED -

- A table is stored in the database for each organic compound . The tables have the same name as the compounds .
- A table named "*classification*" is used for classifying the reactions into one of the reaction classes of the ugi's scheme.

STRUCTURE OF THE TABLES-

- Table for organic compound "CH4" has the following structure .

Field	Type	Null	Key	Default	Extra
struc	varchar(100)	YES		NULL	

- "*classification*" table has the following structure .

Field	Type	Null	Key	Default	Extra
rule	int(11)	YES		NULL	
rclass	varchar(100)	YES		NULL	
rtype	varchar(100)	YES		NULL	

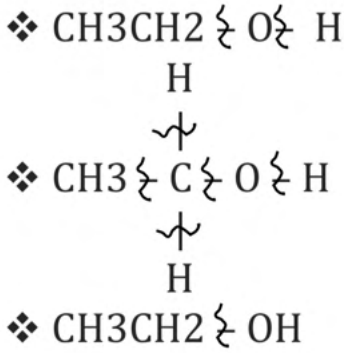
ALGORITHM

The entire prediction algorithm can be divided into three stages –

- Construction of educt matrices from input reactants .
- Prediction of products using the extended ugi's scheme .
- Stability analysis of the products to determine the list of stable products .

Construction of educt matrices from input reactants

Since presence of a functional group affects the type of reaction a compound will undergo an analysis of organic chemical reactions was carried out for each of the functional groups (such as –OH , -COOH , -Cl , -CHO etc.) as well as for compounds without functional groups (such as pure alkane , alkene and alkyne) . This analysis helped us to determine the way bonds break in any compound during a chemical reaction . For example CH3CH2OH (an alcohol) undergoes reactions in which the bonds break in any one of the following ways –



This serves our purpose of finding the components of the BE-matrices of each compound .

In order to construct the BE- matrices of the reactants the first task is to determine the structure of the reactants (i.e. the

way different atoms of the reactants are linked to each other as well as their bond order) .

Step 1 : Determining the structure of the educts

The data structure to store the structure of the organic compounds is described below :

- Struct element
  {
  Char *name;
  Struct element *upper , *lower , *left , *right;
  Int bond , ion_charge;
  };

  It is known that any organic compound consists of a chain of carbon atoms linked with either hydrogen atoms or any functional group . However , few exceptions are there such as ethers and thiols where the central atom is oxygen and sulphur respentively . Following are examples of such compounds .
- CH3 – O – CH3
- CH3 – S – H

  The data structure stores the central component (i.e. C , CH , CH2 , CH3 etc) in the variable named '*name*'. The '*left*' , '*right*' , '*upper*' and '*lower*' point to similar components linked to this central atom . If the central component is part of the main chain then the variable '*bond*' stores the bond order of the central component with the component to its left that it is linked with , otherwise , if it is a branch of such a central component then the variable '*bond*' stores the bond order of this branch with the central component . The variable ion_charge stores the charge of the component in case the component is an ion.

  For example ,let us consider the compound CH3CH2ClCOOH.

Its structure will be :

❖
```
        Cl
         |
CH3 – CH2
         |
       COOH
```

The data structure '*element*' will look like :

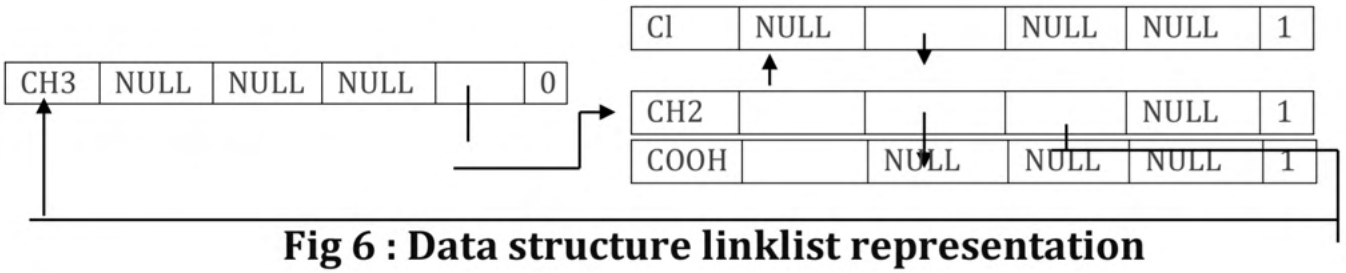


Fig 6 : Data structure linklist representation

Following are the local variables used that helped in the construction of the structure of the educts –

1. Start – It points to the first node of the linklist 'element'
2. Inter & New_node– Stores the intermediate central components of the main carbon chain . Initialised to NULL at the beginning of the algorithm .
3. Bond_left – Stores the unsaturated bond of the intermediate central component of the main carbon chain .
4. Combination_no – A five element array to store the possible number of combinations for each educt .
5. Counter – A global variable to store the number of educts scanned so far .
6. Func_group – A five element array to keep a count of functional groups in each educt .
7. Charge – A variable to store the charge of the anion or the cation .

8. Arr – A five element array to hold pointers to the starting node of each educt in the input reaction .
9. Already_formed – A five element array to store the information whether the BE-matrices of these educts have already been constructed .
10. Spcl_array – A five element array to store pointers to the BE-matrices of educts whose BE-matrices were constructed without pre-determining their structures .

Following are the list of functions used in the algorithm –

1. get_unsaturated_bond(char *str) – Used to determine the unsaturated valency of the central carbon . For example for CH2 this function will return 2 since its two valencies are still unsaturated .
2. increase_combination(int combination_no , char *str , int counter) – Used to increase the number of possible combinations (in the array combination_no) of the counter-th educt when a functional group 'str' is scanned .
3. trim_and_find_ion_charge(char * str) – Used to determine the charge of an ion and at the same time remove the charge symbols (':' and '.') from the string 'str' .

To determine the structure several rules were written in lex that are described below –

1. { C | CH | CH2 | CH3 } – This rule implies that a carbon atom belonging to the main chain has been encountered . To add it to the structure following operations are performed :
    - **Inter=New_node;**
    - **New_node = malloc(struct element *);**
    - **If (start is NULL) then start= New_node and New_node->bond=0 else Inter->right=New_node ;**
    - **New_node->name=yytext;**
    - **New_node->left=Inter;**
    - **New_node->bond=Bond_left;**
    - **Bond_left = get_unsaturated_bond(yytext)-Bond_left;**

2. { COOCO | COO | CO | O | S } – This rule is written for elements other than alkyl groups which can form the central component in the educt structure . To add it to the structure following operations are performed :
    - **Call increase_combination(Combination_no,yytext,counter);**
    - **Func_group[counter]++;**
    - **Inter=New_node;**
    - **New_node=malloc(struct element *);**
    - **If (start is NULL) then start= New_node and New_node->bond=0 else Inter->right=New_node ;**
    - **New_node->name=yytext;**
    - **New_node->left=Inter;**
    - **New_node->bond=1;**
    - **Bond_left=1;**
3. { F | Cl | Br | I | OH | COOH | CHO | CN | NH2 } – This rule scans the functional groups linked to the central component of the main chain . To add it to the structure following operations are performed :
    - **Call increase_combination(Combination_no,yytext,counter);**
    - **Func_group[counter]++;**
    - **Branch=malloc(struct element *);**
    - **Branch->bond=1;**
    - **Branch->name=yytext;**
    - **Branch->upper=Branch->lower=Branch->left=Branch->right=NULL;**
    - **Bond_left--;**
    - **If (New_node->upper=NULL) then New_node ->upper=Branch;**
    - **If (New_node->lower=NULL) then New_node ->lower=Branch;**
    - **If (New_node->left=NULL) then New_node ->left=Branch;**
    - **If (New_node->right=NULL) then New_node ->right=Branch;**
4. { [:.]*(C | CH | CH2 | CH3) [:.]* } – This rule is written for carbocations . To add carbocations to the structure following operations are performed :
    - **Inter=New_node;**
    - **New_node = malloc(struct element *);**
    - **If (start is NULL) then start= New_node and New_node->bond=0 else Inter->right=New_node ;**
    - **Charge=trim_and_find_ion_charge(yytext);**
    - **New_node->name=yytext;**
    - **New_node->left=Inter;**
    - **New_node->bond=Bond_left;**
    - **Bond_left = get_unsaturated_bond(yytext)-Bond_left-charge;**
    - **New_node->ion_charge=charge;**
5. { O. | COO. } – To add these to the structure following operations are performed :
    - **Call increase_combination(Combination_no,yytext,counter);**
    - **Charge=trim_and_find_ion_charge(yytext);**

- **Func_group[counter]++;**
- **Branch=malloc(struct element *);**
- **Branch->bond=1;**
- **Branch->name=yytext;**
- **Branch->ion_charge=charge;**
- **Branch->upper=Branch->lower=Branch->left=Branch->right=NULL;**
- **Bond_left--;**
- **If (New_node->upper=NULL) then New_node ->upper=Branch;**
- **If (New_node->lower=NULL) then New_node ->lower=Branch;**
- **If (New_node->left=NULL) then New_node ->left=Branch;**
- **If (New_node->right=NULL) then New_node ->right=Branch;**

6. { + } – This symbol indicates that the next educt has to be scanned . Following are the operations performed :
   - **Arr[counter]=start;**
   - **Start=NULL;**
   - **New_node=NULL;**
   - **Bond_left=0;**
   - **Counter++;**
7. { "\n" } – This indicates end of line that is end of the input reaction . Following are the operations performed :
   - **Arr[counter]=start;**
   - **Counter++;**
   - **If(counter<5) then Arr[counter]=NULL;**
8. For certain specific inorganic compounds and ions generally found to participate in organic reactions the BE-matrices were directly constructed and following operations were performed :
   - **Comb_covered[counter] and combination_no[counter] were appropriately updated .**
   - **Already_formed[counter]=1;**

Step 2 : Construct BE-Matrices

The following is the data structure used to represent the BE-matrix –

❖ Struct comb_array
{
Char name[20][50];
Int total , central_atom;
Int be_matrix[20][20];

};

The variable named ' *name* ' stores the name of the individual components of the educt . Variable ' *total* ' keeps a count of the total number of such components in the matrix and be_matrix is a symmetrical 2-D matrix to store the BE-matrix of the educt .

Following are the list of variables used in the algorithm –

1. pure_alkyl – A variable used to denote whether an educt contains a functional group or is it a pure alkane , alkene or alkyl .
2. alkyl – A variable used to store chain of alkyl groups . For example , if the educt is CH3CH2CH2CH2Cl then alkyl="CH3CH2CH2CH2" so that educt is reduce to the form R-Cl .

Following are the list of functions used in the algorithm –

1. allocate_be_matrix( ) – Used the dynamically allocate required space for the BE-matrices of the educts .
2. check_for_OH( ) – Checks the presence of functional group "OH" as a branch of the central component .
3. add_to_BE_matrix_OH( ) – Add all possible ways in which 'OH' undergoes bond breakage in a chemical reaction to the BE-matrices .
4. check_for_Oion( ) – Checks the presence of functional group "O." as a branch of the central component .
5. add_to_BE_matrix_Oion( ) – Add all possible ways in which 'O.' undergoes bond breakage in a chemical reaction to the BE-matrices .
6. check_for_COOH( ) – Checks the presence of functional group "COOH" as a branch of the central component .

7. add_to_BE_matrix_COOH( ) – Add all possible ways in which 'COOH' undergoes bond breakage in a chemical reaction to the BE-matrices .
8. check_for_COOion( ) – Checks the presence of functional group "COO." as a branch of the central component .
9. add_to_BE_matrix_COOion( ) – Add all possible ways in which 'COO.' undergoes bond breakage in a chemical reaction to the BE-matrices .
10. check_for_CHO( ) – Checks the presence of functional group "CHO" as a branch of the central component .
11. add_to_BE_matrix_CHO( ) – Add all possible ways in which 'CHO' undergoes bond breakage in a chemical reaction to the BE-matrices .
12. check_for_CN( ) – Checks the presence of functional group "CN" as a branch of the central component .
13. add_to_BE_matrix_CN( ) – Add all possible ways in which 'CN' undergoes bond breakage in a chemical reaction to the BE-matrices .
14. check_for_COOCO( ) – Checks the presence of functional group "COOCO" as the central component .
15. add_to_BE_matrix_COOCO( ) – Add all possible ways in which 'COOCO' undergoes bond breakage in a chemical reaction to the BE-matrices .
16. check_for_COO( ) – Checks the presence of functional group "COO" as the central component .
17. add_to_BE_matrix_COO( ) – Add all possible ways in which 'COO' undergoes bond breakage in a chemical reaction to the BE-matrices .
18. check_for_CO( ) – Checks the presence of functional group "CO" as the central component .
19. check_for_CO_halide ( ) – Checks the presence of the functional group 'CO-X' where 'X' is any halogen .

20. check_for_CO_NH2 ( ) – Checks the presence of the functional group 'CO-NH2' .
21. check_for_CO_alkyl ( ) – Checks the presence of the functional group 'CO-X' where 'X' is any alkyl group .
22. add_to_BE_matrix_CO_halide( ) – Add all possible ways in which 'CO-X' undergoes bond breakage in a chemical reaction to the BE-matrices , where 'X' denotes a halogen .
23. add_to_BE_matrix_CO_NH2( ) – Add all possible ways in which 'CO-NH2' undergoes bond breakage in a chemical reaction to the BE-matrices .
24. add_to_BE_matrix_CO_alkyl( ) – Add all possible ways in which 'CO-X' undergoes bond breakage in a chemical reaction to the BE-matrices , where 'X' denotes any alkyl group .
25. check_for_O( ) – Checks the presence of functional group "O" as the central component .
26. add_to_BE_matrix_O( ) – Add all possible ways in which 'O' i.e an ether undergoes bond breakage in a chemical reaction to the BE-matrices .
27. check_for_halogen( ) – Checks the presence of functional group "X" as a branch of the central component , where 'X' denotes a halogen .
28. add_to_BE_matrix_halogen( ) – Add all possible ways in which 'X' undergoes bond breakage in a chemical reaction to the BE-matrices , where 'X' denotes a halogen .
29. add_to_BE_matrix_pure_alkyl( )- This function is called when the educt does not contain any functional group . In such a case the educt matrix is split into its constituent atoms in all possible combinations and the BE-matrices are constructed accordingly .
30. add_to_BE_matrix_alkyl( ) – This function is called when after constructing BE-matrices for the functional groups present in the educt the variable 'alkyl' is not NULL .

31. copy_to_BE_matrix( ) – This function copies the BE-matrices already constructed for certain specific educts into the BE-matrices allocated for all the educts .

The following are the steps in the construction of the BE-matrices –

i. allocate_be_matrix( );
ii. for(i=0;i<5;i++) do
a. if(already_formed[i]==0 && arr[i]!=NULL) then
s=arr[i];
do
if(func_group[i]=0) then
pure_alkyl=1 break;
if(check_add_to_alkyl (s) then
concatenate (s->name,alkyl) ;
if(check_for_OH) then
count the number of 'OH' attached to 's' ;
add_to_BE_matrix_OH ;
if(check_for_Oion) then
count the number of 'O.' attached to 's' ;
add_to_BE_matrix_Oion ;
if(check_for_COOH) then
count the number of 'COOH' attached to 's' ;
add_to_BE_matrix_COOH ;
if(check_for_COOion) then
count the number of 'COOion' attached to 's' ;
add_to_BE_matrix_COOion ;
if(check_for_CHO) then
count the number of 'CHO' attached to 's' ;
add_to_BE_matrix_CHO ;

if(check_for_CN) then
count the number of 'CN' attached to 's' ;
add_to_BE_matrix_CN ;
if(check_for_COOCO) then
add_to_BE_matrix_COOCO ;
if(check_for_COO) then
add_to_BE_matrix_COO ;
if(check_for_CO) then
if(check_for_CO_halide) then
add_to_BE_matrix_CO_halide ;
if(check_for_CO_NH2) then
add_to_BE_matrix_CO_NH2 ;
if(check_for_CO_alkyl) then
add_to_BE_matrix_CO_alkyl ;
if(check_for_O) then
add_to_BE_matrix_O ;
if(check_for_halogen) then
add_to_BE_matrix_halogen ;
s=s->right;
while(s not equals NULL);
if(pure_alkyl=1) then
add_to_BE_matrix_pure_alkyl;
if(alkyl is not NULL) then
add_to_BE_matrix_alkyl;

b. copy_to_BE_matrix( ) ;

Step 3 : Construct Educt matrices

call merge_BE_matrix( );

merge_BE_matrix( ) – A function which merges the BE-matrices of educts to form a list of possible educt matrices .

Prediction of products using the extended Ugi's scheme

A hierarchical approach has been used to facilitate easy prediction of chemical reactions based on the 56 reaction classes as in Tables 4, 5 and 6 (including both 30 Ugi's reaction classes and 26 reaction classes of extended Ugi's Scheme). The educt matrices are observed and from here itself the probable outcomes of the reaction are obtained .

After the construction of the n X n educt matrices (where n is the number of parts in the input to the application or in the reactant) , the educt matrix is observed carefully. According to Ugi's scheme the reaction matrix is obtained from the difference between the product and educt matrix. Here we see that directly from the educt matrix we can predict the set of probable outcomes of reactions. Based on the following properties we can hierarchically predict chemical reactions :

1. Number of reactants taken as input to the reaction.
2. Type of bond-single, double or triple to be determined from the educt matrix.
3. Presence of free electron to be determined from the educt matrix.
4. Number of parts in the educt matrix to be determined from the educt matrix.

The above properties lead us to the 56 reaction classes. Following the rules of the reaction classes we can thus predict the outcome of chemical reactions. We have thus used a tree based approach to facilitate the easy prediction of chemical reactions based on Ugi's reaction classes (including extended Ugis scheme) .

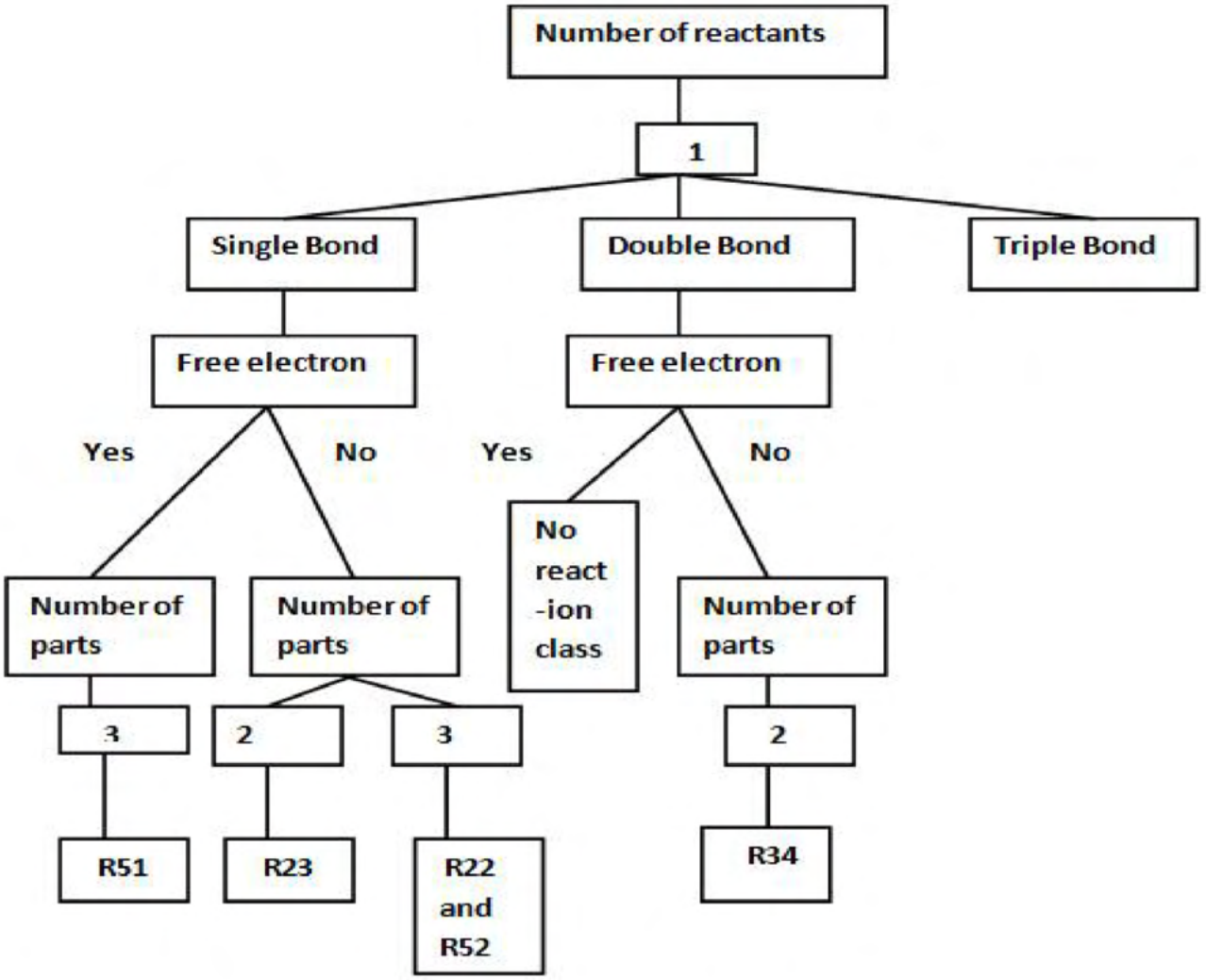


Fig 7 : Hierarchical approach to predict chemical reactions when number of reactants is one

The above tree as shown in Figure 6.1 helps us to _nd the set of probable products when number of reactants is one. The reaction classes which are in the leaf nodes when number of reactant is one are(56 reaction classes as shown in Tables 4 , 5 and 6) :

1.R51
2.R23
3.R22
4.R52

5.R34

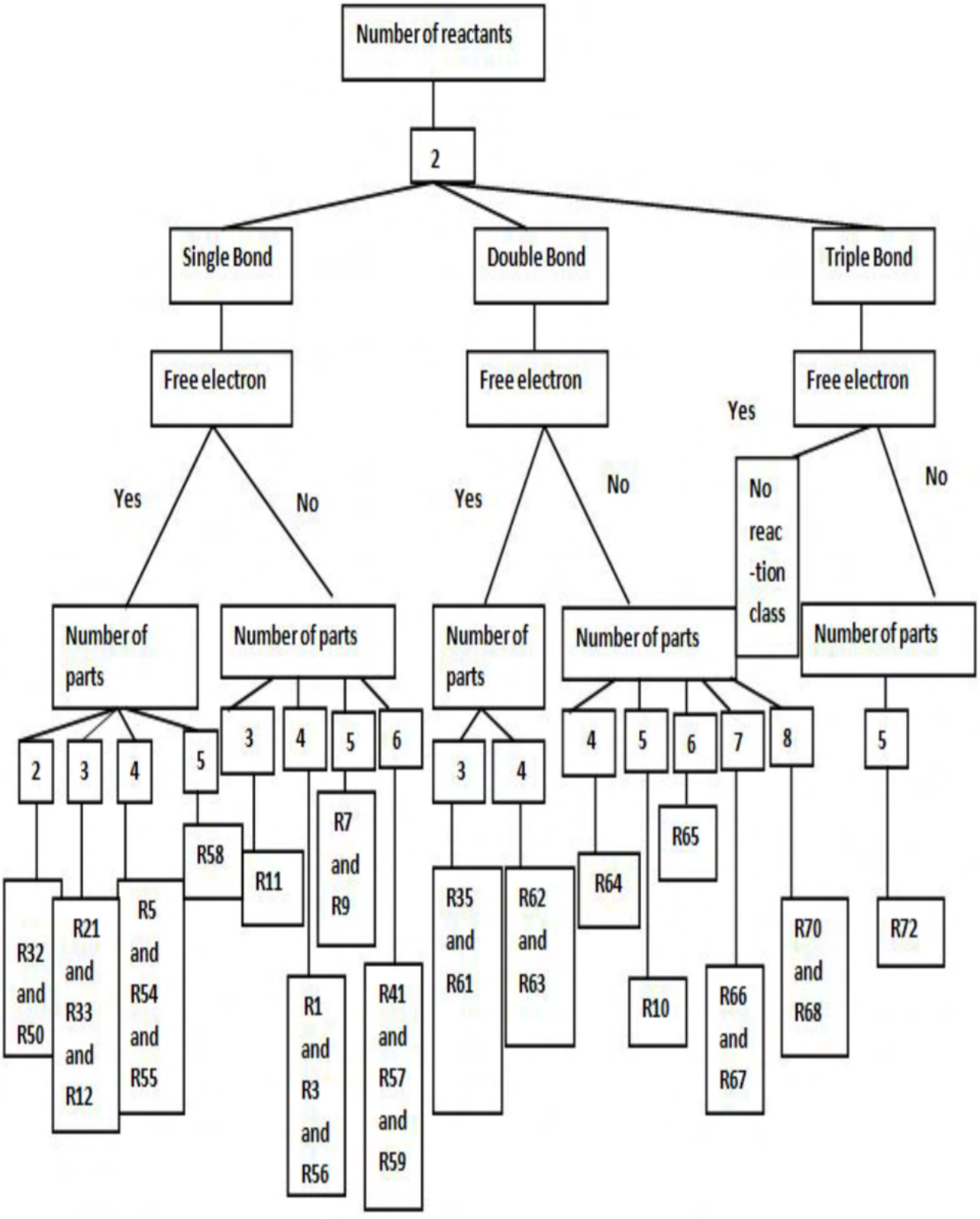


Fig 8 : Hierarchical approach to predict chemical reactions when number of reactants is two

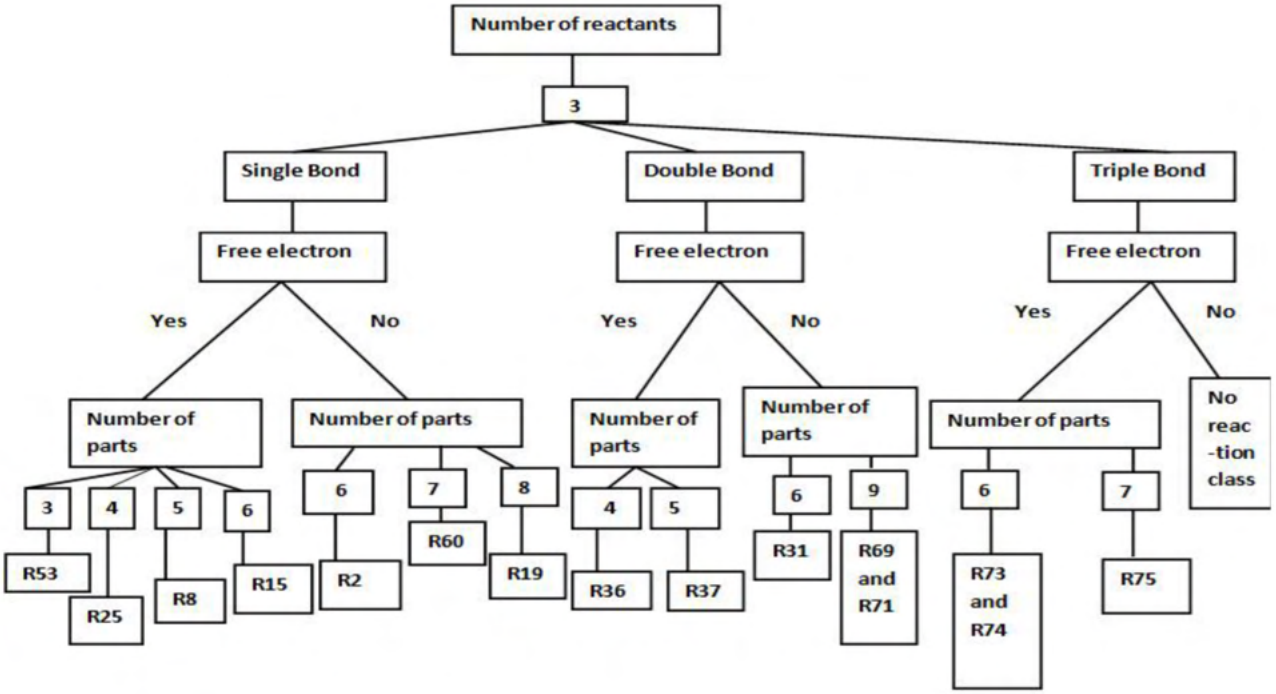


Fig 9 : Hierarchical approach to predict chemical reactions when number of reactants is three

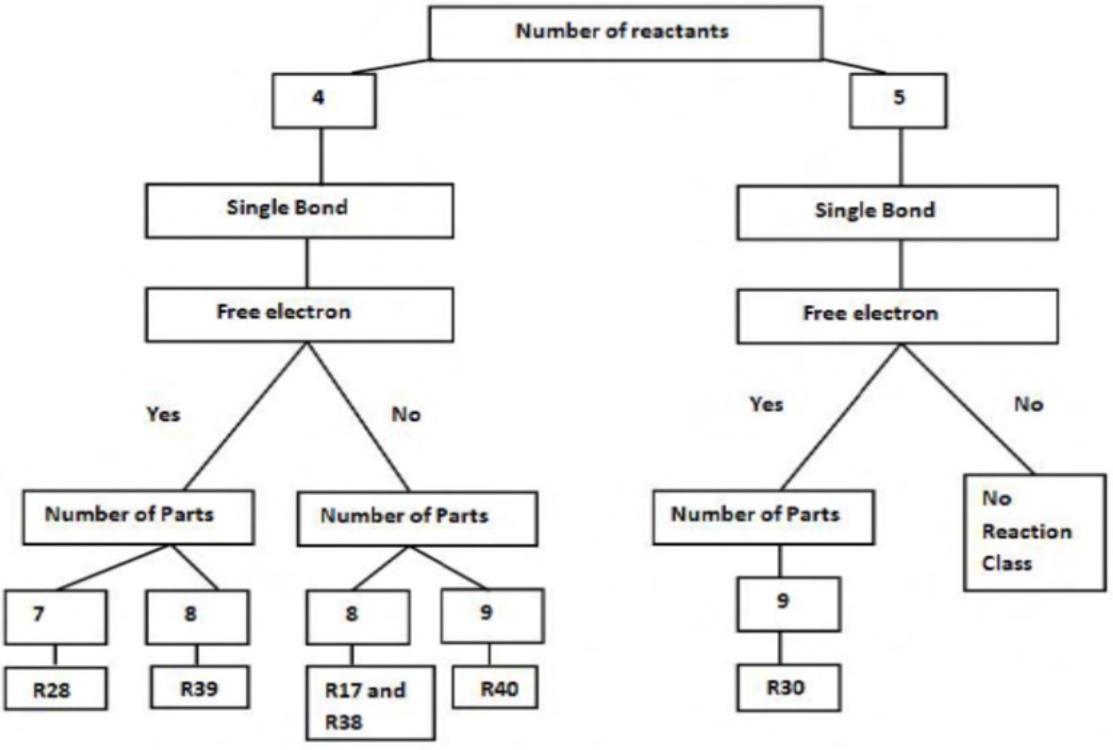


Fig 10 : Hierarchical approach to predict chemical reactions when number of reactants is four and five

Description of the tree based approach

The tree has been shown in Figures 7, 8, 9 and 10 . The parent or root node of the tree consists of number of reactants in the reaction. In this case the number of reactants may be one , two , three , four or five . The next level contains type of bond present in the educt matrix of the reactant . Bond may be single , double or triple . The next level contains the presence of free electrons . The presence of non-zero elements in the diagonal of the educt matrix indicates the presence of free electron . The next level contains the number of parts in the input reactant . The leaf nodes contain the 56 reaction classes as in Tables 4, 5 and 6 .

Algorithm for hierarchical prediction of chemical reactions

Input : Educt matrices of the input reactants .
Output : Probable set of products .
Step 1 : Observe the number of input reactants in the reaction .
Step 2 : Observe the educt matrix to check the highest bond order present. If all types of bond present then consider triple(3) bond which is the highest among 0,1,2 and 3.So,in other words consider the highest value present in the educt matrix of the reaction .
Step 3 : Check to see if there is any non zero element in the diagonal of the educt matrix . If yes then free electron present, else not present .
Step 4 : Observe the number of parts of the reactants in the reaction ,i.e, determine the value of n in n X n educt matrix where n is the number of parts of the reactant in the reaction(n is the number of atoms in the reactants of the reaction) .
Step 5 : The 56 reaction classes (30 Ugi's reaction classes as shown in Table 4and 26 new reaction classes of extension of Ugi's scheme as shown in Tables 5 and 6) determine the set of products of the input reactants to the reaction. Based on the rules of the 56 classes , the output product is determined.
Step 6 : For a particular input, we can get many set of outputs or products corresponding to some of the classes among 56 reaction classes as shown in Tables 4 , 5 and 6 .
Step 7 : The rules of 56 reaction classes govern the set of products to be obtained from the given input reactants.
Step 8 : Probable set of products obtained based on the rules of 56 reaction classes .

Stability analysis of products to determine the list of stable products

After the possible products of a chemical reaction have been determined , the products have to be analyzed for valency stability . This analysis can be performed by using the following algorithm .

Algorithm for Stability analysis

Input : Educt matrices of the reactants and Product matrices of the products .

Output : List of stable products .

Step 1 : Rename the components in the Educt matrix as A , B , C , ... and reflect this renaming accordingly in the Product matrix .

Step 2 : For each component A , B , C .... in the Educt matrix calculate the summation of the elements in the Educt matrix row-wise or column-wise . Similarly calculate the summation of each component A , B , C .... in the Product matrix row-wise or column-wise .

Step 3 : If for all the components the calculated summation in the Product matrix is same as that in the Educt matrix then the Product is stable otherwise not .

Example 1

- CH3COOCOCH3 + H2O -> CH3COOH + CH3COOH

Educt matrix

	CH3COO	CH3CO	H	OH	∑
CH3COO	0	1	0	0	1
CH3CO	1	0	0	0	1
H	0	0	0	1	1

OH	0	0	1	0	1

Product matrix

	CH3COO	CH3CO	H	OH	∑
CH3COO	0	0	1	0	1
CH3CO	0	0	0	1	1
H	1	0	0	0	1
OH	0	1	0	0	1

So we see that the product of the given reaction is stable .

Example 2

- CH4 + H20 -> H-H-OH + CH2-H

Educt matrix

	CH2	H	H	H	OH	∑
CH2	0	1	1	0	0	2
H	1	0	0	0	0	1
H	1	0	0	0	0	1
H	0	0	0	0	1	1
OH	0	0	0	1	0	1

Product matrix

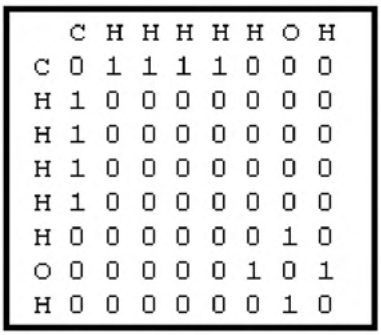
	CH2	H	H	H	OH	∑
CH2	0	0	1	0	0	1
H	0	0	0	1	1	2
H	1	0	0	0	0	1
H	0	1	0	0	0	1
OH	0	1	0	0	0	1

Since there is a mismatch in the summation of the components in the Educt and Product matrix the product is unstable .

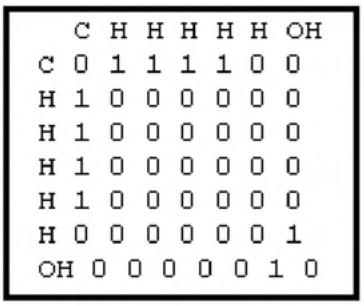
SCREEN SHOTS

❖ Example Input : CH4 + H2O

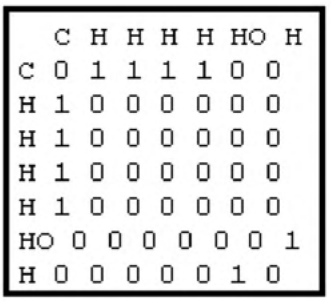
Educt matrices constructed



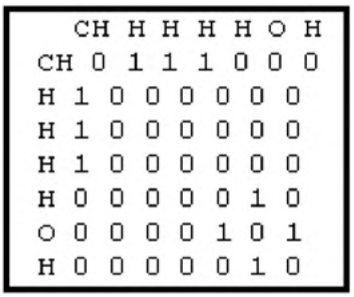
	C	H	H	H	H	H	O	H
C	0	1	1	1	1	0	0	0
H	1	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
H	1	0	0	0	0	0	0	0
H	0	0	0	0	0	0	1	0
O	0	0	0	0	0	1	0	1
H	0	0	0	0	0	0	1	0



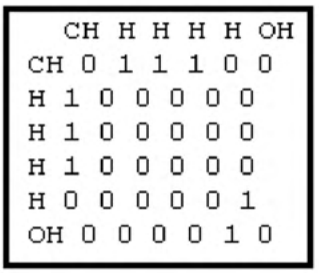
	C	H	H	H	H	H	OH
C	0	1	1	1	1	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	0	0	0	0	0	0	1
OH	0	0	0	0	0	1	0



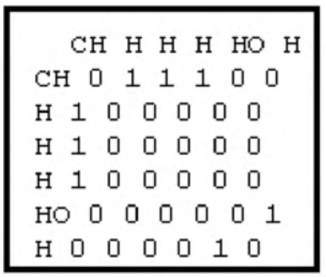
	C	H	H	H	H	HO	H
C	0	1	1	1	1	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
HO	0	0	0	0	0	0	1
H	0	0	0	0	0	1	0



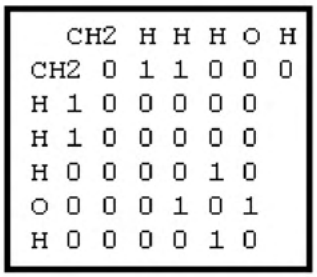
	CH	H	H	H	H	O	H
CH	0	1	1	1	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	1	0	0	0	0	0	0
H	0	0	0	0	0	1	0
O	0	0	0	0	0	1	0	1
H	0	0	0	0	0	1	0



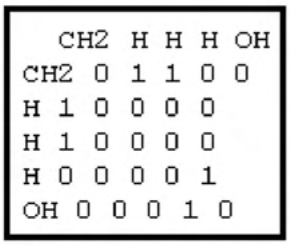
	CH	H	H	H	H	OH
CH	0	1	1	1	0	0
H	1	0	0	0	0	0
H	1	0	0	0	0	0
H	1	0	0	0	0	0
H	0	0	0	0	0	1
OH	0	0	0	0	1	0



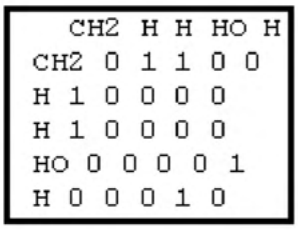
	CH	H	H	H	HO	H
CH	0	1	1	1	0	0
H	1	0	0	0	0	0
H	1	0	0	0	0	0
H	1	0	0	0	0	0
HO	0	0	0	0	0	1
H	0	0	0	0	1	0



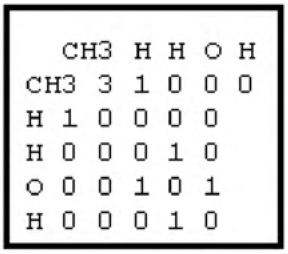
	CH2	H	H	H	O	H
CH2	0	1	1	0	0	0
H	1	0	0	0	0	
H	1	0	0	0	0	
H	0	0	0	0	1	0
O	0	0	0	0	1	0	1
H	0	0	0	0	0	1	0



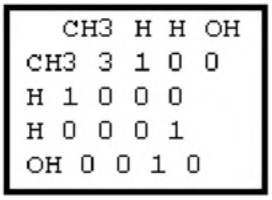
	CH2	H	H	H	OH
CH2	0	1	1	0	0
H	1	0	0	0	0
H	1	0	0	0	0
H	0	0	0	0	1
OH	0	0	0	1	0



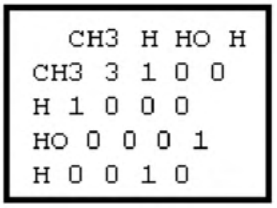
	CH2	H	H	HO	H
CH2	0	1	1	0	0
H	1	0	0	0	0
H	1	0	0	0	0
HO	0	0	0	0	1
H	0	0	0	1	0



	CH3	H	H	O	H
CH3	3	1	0	0	0
H	1	0	0	0	0
H	0	0	0	1	0
O	0	0	1	0	1
H	0	0	0	1	0



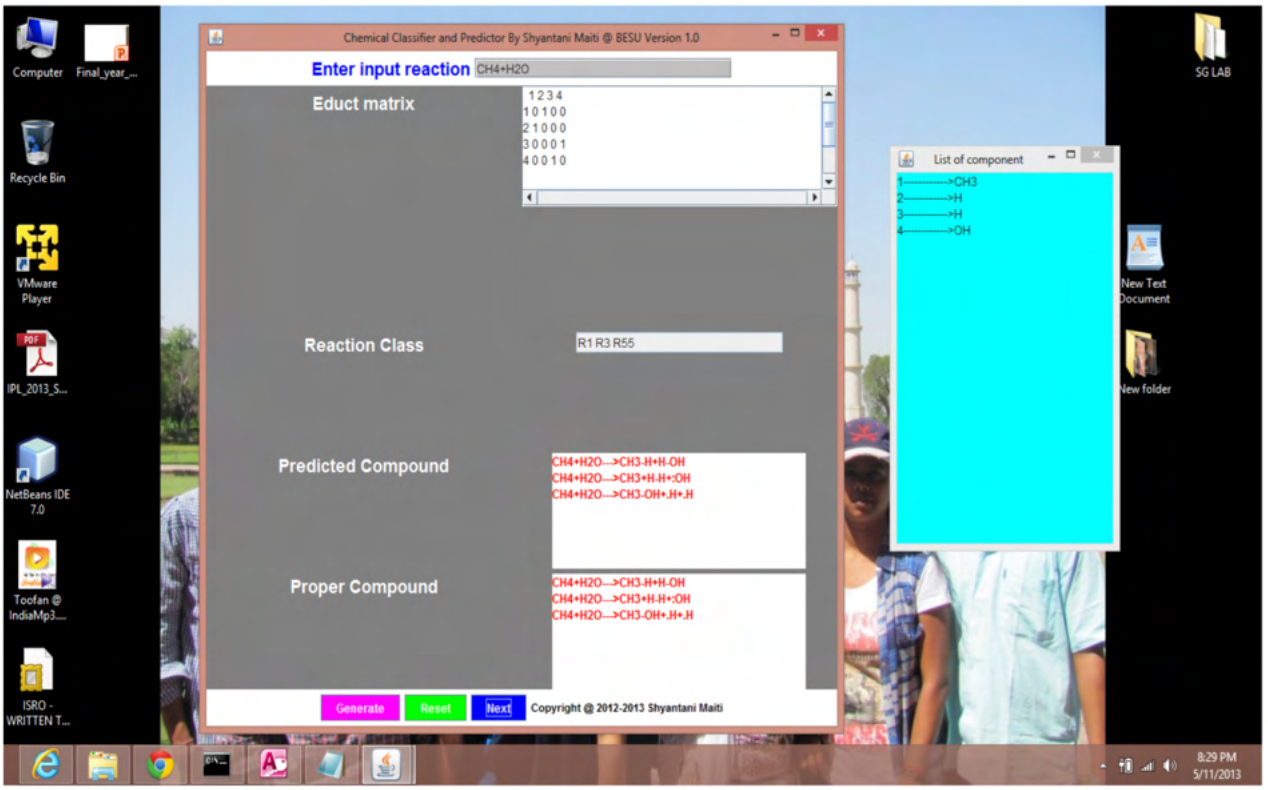
	CH3	H	H	OH
CH3	3	1	0	0
H	1	0	0	0
H	0	0	0	1
OH	0	0	1	0



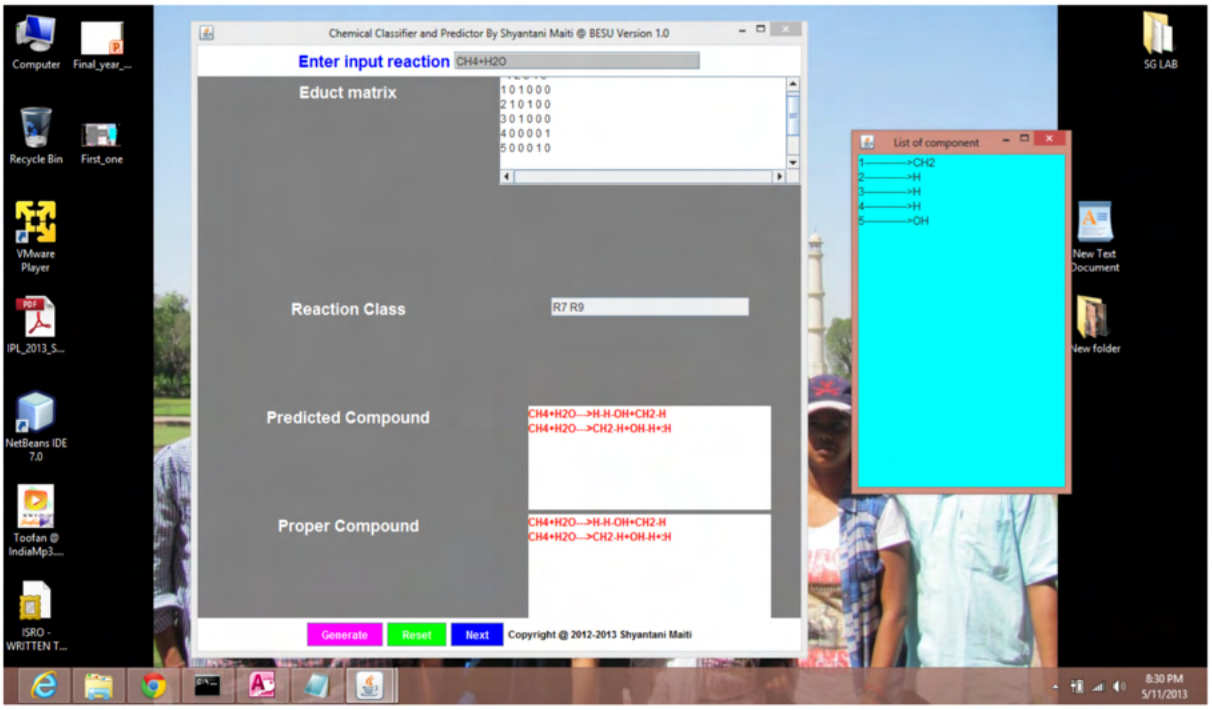
	CH3	H	HO	H
CH3	3	1	0	0
H	1	0	0	0
HO	0	0	0	1
H	0	0	1	0

Prediction of probable outputs

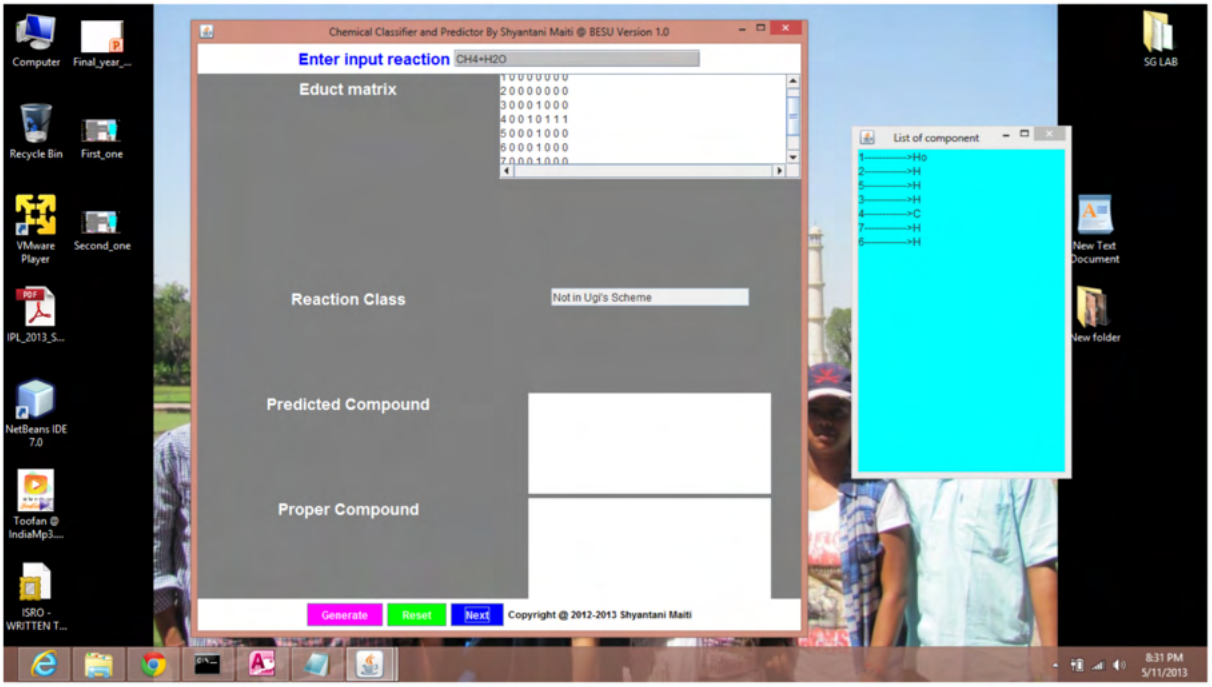
❖ Screen shot 1



❖ Screen shot 2



❖ Screen shot 3



CONCLUSION

In this project we have worked on the automatic construction of educt matrices of compounds . In this project we have also worked on automatic classification of chemical reactions based on a model-driven approach. Among several established models we have selected Ugi's scheme because it provides maximum number of classes and is used widely in chemoinformatics .

We have also worked on the prediction of the output of chemical reactions using a hierarchical approach . The input to the application are the chemical formula of the reactants and the output of the application is a set of products predicted according to 56 reaction classes (including extended Ugi's scheme). 30 reaction classes of Ugi's scheme as shown in Table 4 along with the 26 new reaction classes as in Tables 5 and 6 helps us to predict the future of chemical reactions. The construction of educt matrix of the reactants is really very important . Hence , more time was given to carefully construct the educt matrices . The educt matrix has to be carefully observed. The important points to be observed are :

1. Highest non zero element in the matrix which indicates the highest bond type present in the reactants .
2. Presence of non-zero diagonal elements in the matrix which indicates the presence of free electron .
3. Size of the educt matrix which indicates the number of parts in the reactant .

Thus, we have implemented an efficient approach to predict chemical reactions . For particular reactants in the reaction, we get probable set of products as output . This approach is very user-friendly and time saving also .