**4 th Research/Expert Conference with International Participation**
**"QUALITY 2005", Fojnica, B&H, November 09 - 12, 2005**

# MATRIX CALCULATION BACKWARD CHAINING IN RULE BASED EXPERT SYSTEM

**M. sc. Marko Hell**
**University of Split, Faculty of Economics**
**Split, Croatia**

**Ph. D. Zlatan Kulenović**
**Ph. D. Radovan Antonić**
**University of Split, Maritime Faculty**
**Split, Croatia**

**Key words:** Expert system, backward chaining, matrix calculation, production rules

**SUMMARY**
*This paper is continuation research about application of matrix calculation in system based on knowledge in backward chaining process. First the initials values forming will be shown, and then suggested algorithm proved by an example. The application of matrix calculation in this process is achievable due to matrix note "IF … THEN" of rules. Including previous work titled "Matrix calculation of forward chaining in rule based expert system", matrix calculation is used in conclusion process basis strategies. In this manner base for detailed analysis of conclusion process is made, purposefully for speeding the process and possibility for synthesis with other methodologies.*

## 1. INTRODUCTION

Previous paper titled "*Matrix calculation of forward chaining in rule based expert system*"[1] proposed and showed with an example application of matrix and vector calculation in forward chaining conclusion process. By recommendation from the same paper, in this paper it will be suggested and shown possibility for use matrix calculation in backward chaining conclusion process. It is appropriate to use backward chaining conclusion if there is few possible solutions, and lots of known information. Backward chaining process starts after the goal attribute is defined. Base idea in backward chaining can be divided in three parts: *initial values forming*, *finding out attributes set used in conditional part of rules from conflict set of previous attributes* and *memorize all attribute with empty conflict set*. After the all necessary attributes with empty conflict set are found, it is necessary to enter their values. If it is possible, they defined the value of goal attribute. As in previous paper base for detecting fruit will be used, and attribute FRUIT is chosen as goal attribute. Detailed review of base is given in pervious mentioned paper [9, 2].

[1] Hell, M.; Slišković, M.; Kulenović, Z.: *Matrix calculation of forward chaining in rule based expert system;* Proceedings the 6th International Scientific-Tehnical Conference Process Control 2004 / Krejči, Stanislav (ur.). - Pardubice : University of Pardubice , 2004. 170. Kouty nad Desnou, Češka, 8-11.06.2004, Process Control 2004, ISBN 80-7194-662-1

## 2. INITIAL VALUES FORMING

Initial value forming, due to simplest review, is divided in three sections.

### 2.1. Forming conflict sets matrix of all attribute

Base on rule, matrix $X$ type $k$ x $m$ is formed, where $k$ is number of attribute, and $m$ number of rule. Matrix $X$ shows set of conflict rule of all attributes. Elements of X matrix can have values 0 or 1 depending whether rule $j$ ($j$ = 1…m) belong to conflict sets of attribute $i$ ($i$ = 1…k). In other words, provided that attribute $i$ can be generated by rule $j$ value $x_{ij}$ will be 1.

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1m} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & & \vdots \\ x_{k1} & x_{k2} & x_{k3} & \cdots & x_{km} \end{bmatrix} \tag{1}$$

First of all, it is necessary to define attributes and their values.

**“Fruit” base example:** The following table show attributes and values that every attribute can assume.

*Table 1. Attributes and their values.*

| Attributes | Attribute values |
|---|---|
| FRUIT | BANANA, PLUM |
| COLOUR | BLUE, YELOW |
| SHAPE | ELONGATE, ROUND |
| DIAMETER | LESS THAN 10 |
| TYPE OF FRUIT TREE | TREE |

Further, using attributes and their values it is necessary to define set rules. The set rules are:

**IF** shape = round **AND** diameter = less than 10 **THEN** type of fruit-tree= tree
**IF** type of fruit-tree= tree **AND** colour = blue **THEN** fruit = plum
**IF** shape = elongate **AND** colour = yellow **THEN** fruit= banana

In this example based on the set of rules, by initial values of attributes, the attempt to recognise fruits will be done[1]. In that case goal attribute is “fruit”. GOAL ATTRIBUTE: fruit

***Set of conflict rules of all attributes forming - example:*** According to the table 1 and rule set defined in previous chapter it is visibly that base is consist of five attributes and three rule. Table 2. shows conflict rules set of all attributes and it is possible to obtain $X$ matrix.

[1] On account of simplicity the number of rules, the attributes and their values is minimal.

*Table 2. Attribute conflict rules set.*

| attribute that can be generated by rule | 1 | 2 | 3 |
|---|---|---|---|
| Shape | 0 | 0 | 0 |
| colour | 0 | 0 | 0 |
| diameter | 0 | 0 | 0 |
| type fruit-tree | **1** | 0 | 0 |
| fruit | 0 | **1** | **1** |

### 2.2. Forming conflict set matrix of goal attribute

Supposed that goal attribute is set, and that it corresponds to the attribute $i$ in matrix $X$. It is necessary to form $X$ matrix of $m \times m$ type, so all matrix elements except diagonal elements will have value zero. Diagonal elements will have value congruently row $i$ of matrix $X$. In other words, matrix Z represent conflict rule set of goal attribute arranged by matrix diagonal.

$$Z=\begin{bmatrix} z_{11} & 0 & 0 & \cdots & 0 \\ 0 & z_{22} & 0 & \cdots & 0 \\ 0 & 0 & z_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & z_{mm} \end{bmatrix} \qquad (2)$$

If the value of element $z_{ii}$ on diagonal is 1, rule $i$ belong to the conflict set of goal attribute. ***Forming conflict set matrix of goal attribute – example:*** Observing Table 2, i.e. $X$ matrix from the previous example it is visible that conflict set of goal attribute “FRUIT” is made of rules[1] 2nd and 3rd. Based on this it is possible to form $Z$ matrix.

$$Z=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

### 2.3. Forming the matrix of conditional part of rule and recursive initial matrix

Supposed that $Y$ matrix $m$ x $k$ type presents attributes set which are used in rule conditional part. Similar as in earlier matrix, $Y$ matrix elements can have value 0 or 1, whether rule $j$ ($j$ = 1…$m$) in his conditional part uses attribute $i$ ($i$ = 1…$k$) determine its value.

---

[1] **IF** type of fruit-tree= tree **AND** colour = blue **THEN** fruit = plum
**IF** shape = elongate **AND** colour = yellow **THEN** fruit = banana

$$Y=\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1k} \\ y_{21} & y_{22} & \cdots & y_{2k} \\ y_{31} & y_{32} & \cdots & y_{3k} \\ \vdots & \vdots & & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mk} \end{bmatrix} \tag{3}$$

Multiplying matrix *Z* (8) and *Y* (9) *Y1* matrix *m*×*k* type is obtained. This matrix represents initial information needed for matrix calculation use.

$$Y1 = Z \cdot Y = \begin{bmatrix} z_{11} & 0 & 0 & \cdots & 0 \\ 0 & z_{22} & 0 & \cdots & 0 \\ 0 & 0 & z_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & z_{mm} \end{bmatrix} \cdot \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1k} \\ y_{21} & y_{22} & \cdots & y_{2k} \\ y_{31} & y_{32} & \cdots & y_{3k} \\ \vdots & \vdots & & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mk} \end{bmatrix} = \begin{bmatrix} y1_{11} & y1_{12} & \cdots & y1_{1k} \\ y1_{21} & y1_{22} & \cdots & y1_{2k} \\ y1_{31} & y1_{32} & \cdots & y1_{3k} \\ \vdots & \vdots & & \vdots \\ y1_{m1} & y1_{m2} & \cdots & y1_{mk} \end{bmatrix} \tag{4}$$

Matrix elements $y1_{ij}$ with value 1 shows that rule *i* from conflict set of goal attribute comprehend in his conditional part attribute *j*. Other matrix elements have value 0.

***Forming the matrix of conditional part of rule and recursive initial matrix – example:*** Table 3 shows tabular review of every rules conditional part, i.e. attribute set which is used in rule *i*.

*Table 3. Tabular review of rule conditional part.*

| **rule uses attributes** | shape | colour | diameter | type of fruit tree | fruit |
|---|---|---|---|---|---|
| 1 | **1** | 0 | **1** | 0 | 0 |
| 2 | 0 | **1** | 0 | **1** | 0 |
| 3 | **1** | **1** | 0 | 0 | 0 |

Further, from earlier table it is possible to create *Y* matrix.

$$Y = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Multiplying matrix *Z* and *Y* matrix *Y1* matrix is obtained. This matrix represents initial matrix for recursion's calculations.

$$Y1 = Z \cdot Y = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

It can be read from *Y1* matrix that rules 2[nd] and 3[rd] are used for generation of goal attribute. The 2nd rule in his conditional part comprehends attributes COLOUR and TYPE OF FRUIT TREE[1], and 3[rd] rule in his conditional part comprehends attributes SHAPE and COLOUR[2].

## 3. CONCLUSION BY BACKWARD CHAINIG PROCESS

The next example will show separation of complex and simple attributes. Complex attributes are those attributes that can be generate with a rule, while simple attributes can not be generate with a rule.

### 3.1. Complex and simple attribute determination

Reading $X(7)$ matrix it is possible to determine complexity of attribute. If the sum in matrix row $i$ is equal 0, it is simple attribute, i.e. ***it can not be generate by neither rule***. In contrary, if the sum of elements in row $i$ higher than 0, this is complex attribute, i.e. ***it can be generate by a rule***. In concordance with above mentioned, it is necessary to form $L$ matrix of $k \times k$ type. If the attribute $i$ is simple, elements on diagonal have value $l_{ii} = 1$. Other matrix elements have value 0.

$$L = \begin{bmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ 0 & l_{22} & 0 & \cdots & 0 \\ 0 & 0 & l_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & l_{kk} \end{bmatrix} \tag{5}$$

It is possible to calculate matrix of complex attributes as an inverse matrix in respect to matrix $I - L$ addition operation, where $I$ stands for unit matrix of $k$ order. Simple attributes of matrix $Y1$ (10)[3] are calculated by multiplying it with $L$ matrix. Memory of complex attribute is done by sum up the products of matrix $Yi$ and $L$ after each step of recursion.

$$S = \sum_{step}^{iterationstepsnumber} \begin{bmatrix} yi_{11} & yi_{12} & \cdots & yi_{1k} \\ yi_{21} & yi_{22} & \cdots & yi_{2k} \\ yi_{31} & yi_{32} & \cdots & yi_{3k} \\ \vdots & \vdots & & \vdots \\ yi_{m1} & yi_{m2} & \cdots & yi_{mk} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ 0 & l_{22} & 0 & \cdots & 0 \\ 0 & 0 & l_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & l_{kk} \end{bmatrix} \tag{6}$$

At the beginning $S$ matrix is null-matrix of $m \times k$ type. At the end of iteration $S$ matrix shows all simple attributes which are necessary to enter so that conflict rule $i$ of goal attribute can be applied. If the element value in matrix is higher than 0, the attribute is used in rule which is in concordance with his indexes. Multiplying $(I - L)$ matrix and $Yi$ matrix complex attributes set is determined. These complex attribute are needed to be take apart again. The iteration process is explained in following part after the example.

[1] **IF** type of fruit-tree= tree **AND** colour = blue **THEN** fruit = plum
[2] **IF** shape = elongate **AND** colour = yellow **THEN** fruit = banana
[3] This part is applied in every steps of iteration (described in further text) after the matrix Yi is obtained.

***Complex and simple attributes determination – example:*** Based on matrix $X$ and earlier stated procedure it is possible to form simple attributes matrix $L$ and complex attributes matrix $I-L$, which will be use in further procedure.

$$X = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad I-L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

From this two matrix it is visible that attribute SHAPE, COLOUR and DIAMETER are simple attributes, while TYPE OF FRUIT TREE and FRUIT are complex attributes. Complex attributes form *Yi* matrix will be write, while its simple attribute will be noted in *S* matrix.

$$S = S + Y1 \cdot L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Furthermore, observing *S* matrix it is visible that SHAPE and COLOUR are simple attributes, i.e. not generated by some rule. As it is visible that TYPE OF FRUIT TREE (in second rule, i.e. value $l_{24} = 0$) complex rule.

### 3.2. Computing necessary attributes for activation of the conflict rules

Previous text explained how to obtain initial Y1 matrix, and from it choose simple and complex attributes. Because complex attributes can be generating by some rules from rule base, it is necessary to find out which attributes are used in this rules. This kind of thinking suggest to recursion process. Recursion process is possible to represent with matrix note

$$Y_{i+1} = Y_i \, (I-L)XY \qquad (7)$$

where $(I\text{-}L)$ matrix, together with $X$ (7) and $Y$ (8) matrix are explained in earlier text. It is possible to calculate product of multiplication $(I-L)XY$ matrix at the beginning of the algorithm. In this manner implementation of algorithm is speed up. If the product is marked as W, recursion process by matrix note is

$$Y_{i+1} = Y_i \cdot W \qquad (8)$$

Recursion process is stopped when $Y_{i+1}$ matrix is null-matrix. Meaning, if $Y_{i+1}$ matrix are null-matrix no one attributes in $Y_i i$ matrix is not complex, i.e. every attributes of $Y_i i$ matrix are simple attributes[1]. In order to determine goal attribute value it is necessary to enter attribute values in concordance with *S* matrix (this is explained in previous chapter).

[1] Attribute complexity investigation in $Y_i$ matrix could be done by gradual multiplying $Y_i$ $(I-L)$, this will result in

***Computing necessary attributes for activation of the conflict rules – example:*** In this example first W matrix will be calculate

$$W = (I–L)XY = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 \end{bmatrix}.$$

Now it is possible to apply recursive equation (14). *First step of Iteration*: Iteration first step is taking apart complex attribute which are placed in conflict set of goal attribute.

$$Y2 = Y1 \cdot W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

It is noticeably from *Y*2 matrix that for application of $2^{nd}$ rule it is necessary to know attribute value SHAPE and DIAMETER[1]. Further, the calculation is done, i.e. simple attributes set noted in shape of *S* matrix is enlarged.

$$S = S + Y1 \cdot L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

*Second step of iteration:* Iteration second step is take apart complex attribute which are placed in attribute conflict set of conditional part of rule from conflict set of goal attribute.

$$Y3 = Y2 \cdot W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Since, *Y*3 is null-matrix based on earlier explained algorithm it is possible to conclude that there are no complex attributes in *Y2* matrix. Cause of this recursion is stop. In matrix

---

much higher number of matrix operations. Therefore, proposed procedures are kept.

[1] Comparing with *Y* matrix it is possible to found that this is $1^{st}$ rule, but it is not necessary in this work.

$$S = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix},$$

It is possible to read simple attribute which are necessary to enter in order to determinate goal attribute value.

## 4. CONCLUSION

Reducing production rules to 0 and 1, it is possible to apply matrix calculation in conclusion process of knowledge based systems. Application of matrix calculation should speed up conclusion process, which will in return contribute to quality usage of production systems together with other knowledge based systems. Presented algorithms, together with paper [9] presents base of conclusion strategies. In further research of presented problems it is necessary to make measurement of past and proposed algorithms work speed, and to affirm exact progress in speeding of the conclusion process. In respect to the fact that conclusion is made in parallel, i.e. in one iteration all necessary rules are used in once, it is possible to form criteria for attributes enter. Setting the proper criteria the number of iteration will be smaller, and speed of conclusion process will be higher. Speeding up of conclusion process can be made with different combination of basis conclusion strategies. Introducing the third dimension in S matrix, the conclusion process by level will be simplified, but due to larger number of operation this will slow down the process. All proposed research regarding to possibility for algorithm application, and necessary conclusion process addition will be studied through further work. For this purpose construction of production system on mentioned technology has been started. Further development of algorithm will depend on development speed of computer equipment which even nowadays achieve until recently unimaginable speed.

## 5. REFERENCES


[1] Luger G. F.: Artificial Intelligence-Structure and strategy for complex problem solving, Addison Wesley, 2002, Harlow, England

[2] http://www.attar.com/pages/downloads.htm

[3] http://www.zemris.fer.hr/education/is/

[4] Horvatić K.: Linearna algebra; Sveučilište u Zagrebu; 1991; Zagreb

[5] Kuratovski K., Mostowski A.: Set Theory, PWN-Polish Scientific Publishers, Warszawa and North-Holland Publishing Company, Amsterdam, 1968.

[6] Martić Lj.: Matematiče metode za ekonomske analize II, Zagreb,1972;

[7] Hell M.; Slišković M., Kulenović Z.: Matrix calculation of forward chaining in rule based expert system; Proceedings the 6th International Scientific-Tehnical Conference Process Control 2004 / Krejči, Stanislav (ur.). - Pardubice : University of Pardubice , 2004. 170. Kouty nad Desnou, Češka, 8-11.06.2004, Process Control 2004, ISBN 80-7194-662-1