World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:1, 2009

Discovery of Quantified Hierarchical Production Rules from large set of Discovered Rules

Tamanna Siddiqui, and M. Afshar Alam

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

***Abstract*—**Automated discovery of Rule is, due to its applicability, one of the most fundamental and important method in KDD. It has been an active research area in the recent past. Hierarchical representation allows us to easily manage the complexity of knowledge, to view the knowledge at different levels of details, and to focus our attention on the interesting aspects only. One of such efficient and easy to understand systems is Hierarchical Production rule (HPRs) system. A HPR, a standard production rule augmented with generality and specificity information, is of the following form:

Decision **If** < condition>
Generality <general information>
Specificity <specific information>.

HPRs systems are capable of handling taxonomical structures inherent in the knowledge about the real world. This paper focuses on the issue of mining Quantified rules with crisp hierarchical structure using Genetic Programming (GP) approach to knowledge discovery. The post-processing scheme presented in this work uses Quantified production rules as initial individuals of GP and discovers hierarchical structure. In proposed approach rules are quantified by using Dempster Shafer theory. Suitable genetic operators are proposed for the suggested encoding. Based on the Subsumption Matrix(SM), an appropriate fitness function is suggested. Finally, Quantified Hierarchical Production Rules (HPRs) are generated from the discovered hierarchy, using Dempster Shafer theory. Experimental results are presented to demonstrate the performance of the proposed algorithm.

***Keywords*—**Knowledge Discovery in Database, Quantification, Dempster Shafer theory, Genetic Programming, Hierarchy, Subsumption Matrix.

I. Introduction

IN KDD, the most predominant representation of the discovered knowledge is the standard production rules in the form If P Then D [12]. As the number of rules becomes significant, they are not comprehensible to people as meaningful knowledge, from which they can gain insight into the basis of decision-making. Much world knowledge is best expressed in the form of hierarchies. Hierarchies give comprehensible knowledge structure that allows us to manage the complexity of knowledge, to view the knowledge at different levels of details, and to focus our attention on the interesting aspects. Several efforts have been made in the recent past towards automated discovery of hierarchical structure in large databases [3]-[9]. Uncertainty pertains to information that is not definitely fixed, not precisely determined. Uncertainty must be quantified in order to use it systematically in decision-making processes. Uncertainty Quantification is the quantitative characterization and use of uncertainty in information applications In Genetic Programming (GP) [10][11] the basic idea is the evolution of a population of "program", i.e., candidate solutions to the target problem. A program (an individual of the population) is usually represented as a tree, where the internal nodes are the functions (operators) and the leaf nodes are terminal symbols. Both the function set and terminal set must include symbols suitable for the target problem. Each individual of the population is assessed regarding its ability to solve the target problem. This evaluation is conducted by a fitness function, which is problem-dependent. Individuals undergo the action of genetic operators such as reproduction, crossover and mutation. Once genetic operators have been applied to the population based on given probabilities, a new generation of individuals is created. These newly created individuals are evaluated by the fitness function. The whole process is repeated iteratively for a fixed number of generations or until other termination criterion is met. The result of GP (the best solution found) is usually the fittest individual created along all the generations [11]. In the present work, a post-processing scheme based on GP and Dempster Shafer Theory is presented that takes quantified if then rules as input and discovers crisp hierarchical structure. Further, Quantified Hierarchical Production Rules(HPRs)[2], are generated using the discovered hierarchy. A concept of Subsumption Matrix (SM) is used to summarize the relationship between the classes. An appropriate encoding, suitable genetic operators and effective fitness function are suggested for the proposed scheme.

Tamanna Siddiqui is with the Department of Computer Science, Jamia Hamdard (Hamdard University), New Delhi 110 062, India (e-mail: ja_zu_siddiqui@hotmail.com).

M. Afshar Alam is with the Department of Computer Science, Jamia Hamdard (Hamdard University), New Delhi 110 062 India (e-mail: afshar@rediffmail.com).

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

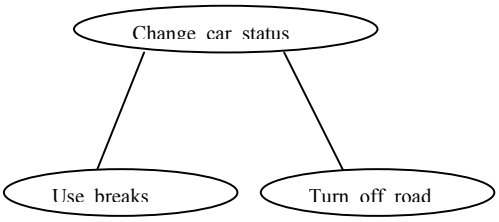
II. Background

The concept of CPR as suggested by Michalski and Winston has the following form:
If P {premises/preconditions}
Then D {actions/decision}
Unless C {censor conditions}
A censor is a low likelihood condition when hold will block the rule. So when the system is having low resources, it can skip checking the censor conditions. If the resources are available, the censor conditions are examined, increasing the certainty factor of making a high speed decision or reversing the decision itself. The above concept of CPR has been extended to HCPR [1] to incorporate both aspects of precision namely certainty and specificity. Two new operator added to CPR and we have the concept of HCPR
having the general form as follows:
D {decision/concept/action}
If P [p1,p2,p3,…,pn] {preconditions}
Unless C [c1,c2,…,cn] {censor conditions}
Generality [G%] {general information}
Specificity S [s1,s2,…,sk]
{mutually exclusive set of specific information}
As a special case, dropping the unless operator due to time constraint, HPR takes the form
D{decision/concept/action}
If P[p1,p2,…,pn] {preconditions}
Generality [G%] {general information}
Specificity S [s1,s2,…,sk]
{mutually exclusive set of specific information}

The root represents the most general concept in a HPR tree and any child in tree is more specific case of its parent. As the concept becomes more specific, the number of elements in its precondition part will increase obviously. However it is not required to list all such elements because total inheritance is an inherent feature of the HPRs tree structure; each HPR inherits the entire preconditions set of its parent HPR, and thus of all of its ancestors. So the redundancy is minimized in the listing of preconditions in the child node. HPR system collects fragmented knowledge and represents these as collective one and hence significantly reducing the knowledge base. This representation scheme reduces the complexity of the discovered knowledge substantially, makes knowledge base easy to understand and efficient for future processing.

As an example, consider the following HPRs [1]:
level 0
change_car_status If [obstacle_ahead]
Generality [ ]
Specificity [use_breaks, turn_off_road]
level 1
use_breaks If [speed_distance_ratio_high]
Generality [change_car_status]
Specificity [ ]
turn_off_road If [not_on_bridge]
Generality [change_car_status]
Specificity [ ].

The above related HPRs form a tree (called HPR-tree) giving the following hierarchical structure Fig.1.



III. Dampster Shafer Theory

It provides a method for explicitly representing the ignorance inherent in knowledge and of reasoning in the absence of certain information. According to Haddawy the belief in a proposition A is represented by a shafer interval [s(A) .. p(A)] which is a subinterval of the unit interval [0 .. 1], where s(A) is called the support and p(A) is called the plausibility of the proposition A. These values can be thought of as an upper and lower bounds on the probability of A. The uncertainty of a proposition is then defined as the width of the interval: u(A) = p(A) – s(A). The precision of the probability estimate for the proposition A is then defined as: PRE(A) = 1 – u(A) . The PRE of the decision is the measure, which will be used to characterize the certainty of an inference.

The variable precision logic (VLP) represents knowledge in terms of facts and rules. To apply Dempster Shafer theory to VLP, we must provide an interpretation of rules and facts in terms of shafer intervals. Dempster Shafer theory can be defined as:
$S(A) + s(\sim A) + I = 1$, Where I stands for Ignorance. According to Dempster Shafer theory a belief factor is associated with each flat rule like $P \rightarrow D$: γ, which can be computed as [2]

$$\gamma = |P \rightarrow D| / |P| . \quad (1)$$

This belief factor represents the support of the rule. Support of the decision can be obtained by using following formula:

$$s(D) = s(P)*s(P \rightarrow D)=s(P)* \gamma \quad (2)$$

where s(P) is probability of occurrence of proposition A.

If rules can be expressed with uncertainty quantification using Shafer interval:

IF $P1_{[s(P1), p(P1)]}$ and $P2_{[s(P2), p(P2)]}$ THEN D
[s(P> D), p(P> D)]
Dempster Shafer theory can be used for approximate reasoning in following manner:
a) prob (P_1 andP2) . [$s(P_1) * s(P_2)$, $p(P_1) * p(P_2)$]
b) prob (P_1 or P_2) . [$s(P_1) + s(P_2) – (s(P_1) * s(P_2))$, $p(P_1) + p(P_2) –(p(P_1) * p(P_2))$]
c) prob (¡« P) . [1-p(P), 1-s(P)]
d) prob (D) . [$s(P)*s(P \rightarrow D)$, $(1 - s(P)) + s(P) * p(P \rightarrow D)$]

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

IV. Subsumption Matrix

A class Di can be defined by set of properties (values of distinct attributes), class_prop(Di). Let Di and Dj be any two classes with the set of properties class_prop(Di) and class_prop(Dj), respectively. We can define the degree of subsumption (deg_sub(Di, Dj)) as follows:

$$\text{deg_sub}(Di,Dj) = \frac{|\text{class_prop}(Di) \cap \text{class_prop}(Dj)|}{|\text{class_prop}(Di)|} \quad (3)$$

where deg_sub(Di, Dj) € [0,1].

A SM that summarizes the relationship between the classes,D1, D2,……,Dn is an n × n matrix defined as under:

$$SM[Di,Dj] = \begin{cases} 1 \text{ if } \text{deg_sub}(Di, Dj) = 1 \\ \text{i.e. } Di \subseteq Dj \\ 0 \quad \text{otherwise} \end{cases} \quad [4]$$

V. Genetic Programming Approach

As a post-processing scheme, we are using GP to discover crisp hierarchical production rules from the flat rules as input. The details of encoding, genetic operators and the fitness function for the proposed scheme are discussed in the following subsection:

A. Encoding

A hierarchical structure is encoded as list representing a general tree:

Tree: (Root (sub-tree 1) (sub-tree 2)…… (sub-tree i)…… (sub-tree k)), where sub-tree i is either empty or has the same structure as Tree. For example the hierarchy in Fig. 2.

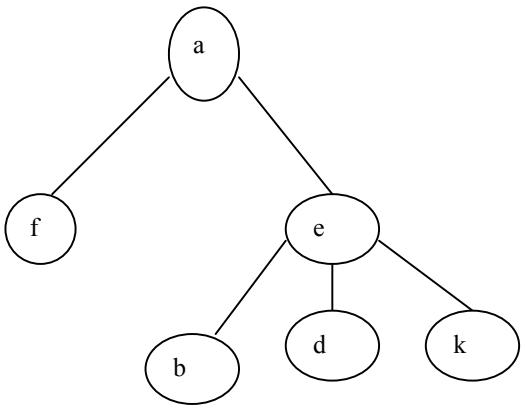


Fig 2 Hierarchy

would be encoded as (a (f (e (b) (d) (k))).

An individual, as hierarchy must satisfy the following condition: Di I Dj = Ø for any two classes Di and Dj at the same level in the hierarchy. During crossover/mutation operators, if any of the offspring or mutated individuals does not satisfy the above condition, then it will be rejected as an illegal individual.

B. Genetic Operators

The new elements in the population are generated by means of three operators: reproduction, crossover and mutation.

a) Reproduction

The reproduction operator selects one individual of the present population in proportion to its fitness value, so that the fitter an individual is the higher the probability that it will take part in the next generation of individuals. After selection, the individual is copied into the new generation without any modifications. Reproduction reflects the principle of natural selection and survival of the fittest [11].

b) Crossover

The crossover operator replaces a randomly selected sub tree of an individual with a randomly chosen sub-tree from another individual and creates new offspring by exchanging sub-trees (i.e., sub-lists) between the two parents. The crossover point was chosen at random for both parents. For example, consider the following two individuals as parents (the "crossover point" is indicated by a tilted line and the sub trees swapped by crossover are shown in bold):

Parent 1: (a(b (c (h (n)))

Parent 2: (a(f) (e (b) (d) (k)))

with corresponding hierarchical structure is given in Fig.3.

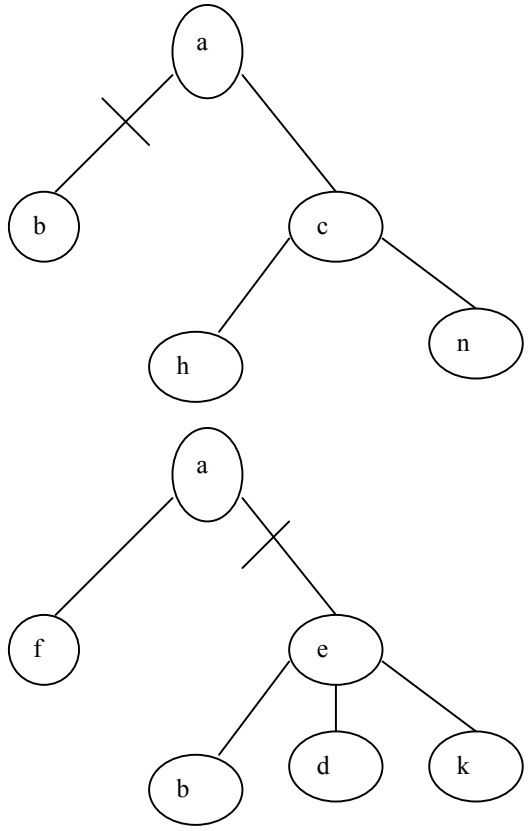


Fig. 3 Two parents before crossover

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

The two offspring resulting from crossover are:
Offspring 1: (a(e (b) (d) (k)) (c(h) (n))) and
Offspring 2: (a(f) (b)) are shown below in Fig.4.

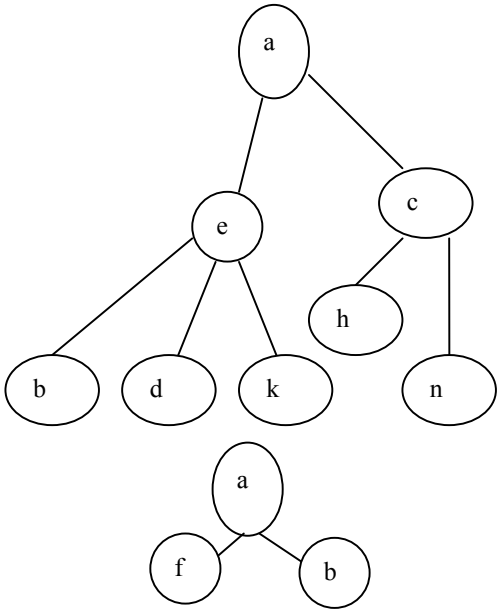


Fig. 4 Two offspring produced by crossover

c) Mutation

For the tree mutation a sub-tree/leaf is replaced by randomly chosen sub-tree/ leaf.

C. Fitness Function

The fitness function evaluates the quality of an individual in the population. For the proposed algorithm, the fitness measure of an individual is defined as:

$$\text{Fitness} = \sum SM\,[Di, Dj] \qquad (5)$$

$\forall$ Di, Dj
The wining individual has the highest fitness such that
$\forall$ i, j Di$\rightarrow$ Dj.

VI EXPERIMENTAL RESULTS

Each GP run consisted of a population of 25 individuals evolving over generations. The probability of crossover, reproduction and mutation set to 0.8, 0.1 and 0.1, respectively, and the selection method used for both parents was fitness proportionate. For the practical reason of avoiding the expenditure of large amounts of computer time on occasional oversized programs, the depth of initial programs was limited to 8 and during the run the maximum tree depth was set to 10.

Training Data Set (Employee)

Employee -lives-in	Time	Nature-of-the-Day	Decision
City	night	Sunday	Is-in-city
City	night	working	Is-at-home
City	night	working	Is-in-city
City	day	Sunday	Is-outside-home
City	night	Sunday	Is-in-city
City	day	working	Is-working-outdoor
City	night	working	Is-in-city
City	night	Sunday	Is-at-home
City	night	working	Is-at-home
City	day	Sunday	Is-entertaining-outdoor
City	day	working	Is-outside-home
City	day	working	Is-outside-home
City	day	Sunday	Is-entertaining-outdoor
City	day	working	Is-working-outdoor
City	night	Sunday	Is-at-home
City	day	Sunday	Is-outside-home
City	night	Sunday	Is-in-city
City	day	Sunday	Is-entertaining-outdoor
City	day	Sunday	Is-outside-home
City	night	working	Is-at-home
City	day	Sunday	Is-entertaining-outdoor
City	day	working	Is-working-outdoor
City	day	Sunday	Is-entertaining-outdoor
City	day	working	Is-working-outdoor
City	day	working	Is-working-outdoor

Example 1: By applying rule discovery algorithm and also using Dempster Shafer theory five flat rules along with belief factor are discovered:

1) If Emplyee_lives_in_city _y Then Emplyee _is_ in_city_y : **0.2**
2) If Emplyee _ lives_in_city_ y $\wedge$ time(night) Then Emplyee _is_at_home : **0.5**
3) If Emplyee _ lives_in_city_ y $\wedge$ time(day) Then Emplyee _is_outside_home : **0.3**
4) If Emplyee _lives_in_city_y $\wedge$ time(day) $\wedge$ day(working) Then Emplyee_is_ working_outdoor : **0.7**
5) If Emplyee_lives_in_city_y $\wedge$ time(day) $\wedge$ day(Sunday) Then Emplyee_is_ entertaining_outdoor : **0.6**

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

Support of the rules 1, 2,3,4,5 is obtained by using formula (1). Values of γ1, γ2, γ3, γ4 and γ5 are 0.2, 0.5, 0.3, 0.7, and 0.6 respectively.

Using (1) and (2) the SM is constructed for the five classes
D1= Emplyee _is_in_city_y ; D2= Emplyee _is_at_home ;
D3= Emplyee _is_outside_home ; D4= Emplyee _is_working_outdoor ;
D5= Emplyee _is_entertaining_outdoor ,

Support for these decision classes can be obtained by using formula (2)
S(D1) = s(P) * γ1 = 1.0 * 0.2 = 0.2
S(D2) = s(P) * γ2 =.0.4 * 0.5 = 0.2
S(D3) = s(P) * γ3 = 0.6* 0.3= 0.18
S(D4) = s(P) * γ4 = 0.28 * 0.7= 0.19
S(D5) = s(P) * γ5 = 0.32 * 0.6 = 0.19

Above five decision classes are mentioned in table

TABLE I
SUBSUMPTION MATRIX (5 × 5)

	D1	D2	D3	D4	D5
D1	1	1	1	1	1
D2	0	1	0	0	0
D3	0	0	1	1	1
D4	0	0	0	1	0
D5	0	0	0	0	1

The proposed algorithm produced the following individual with the highest fitness = 4 :
(D1(D2)(D3(D4)(D5))).

The corresponding hierarchy is shown in Fig.5.

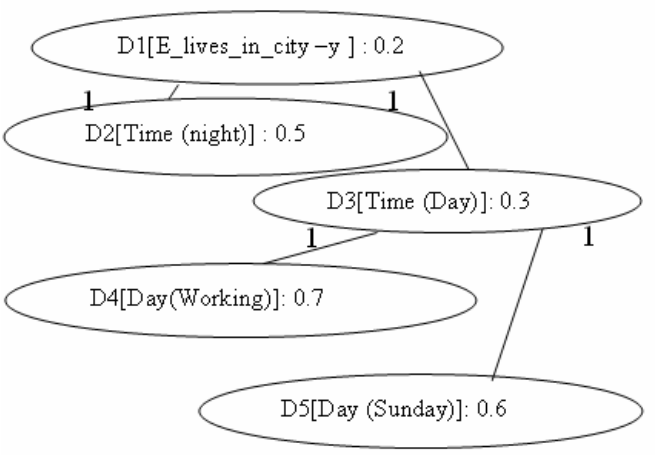


Fig. 5 Hierarchy-the individual with the highest fitness=4

From the discovered hierarchy shown in Fig. 5, the following Quantified HPRs are generated:
level 0
(D1 If [Emplyee _lives_in_city_y] Generality [ ]
Specificity [D2,D3]) [0.2 . . 1]
level 1
(D2 If [time (night)]
Generality [D1]
Specificity [ ]) [0.5 . . 1]
(D3 If [time (day)]
Generality [D1]
Specificity [D4 ,D5]) [0.3 . . 1]
level 2
(D4 If [day (working)]
Generality [D3]
Specificity [ ]) [0.7 . 1]
(D5 If [day (Sunday)]
Generality [D3]
Specificity [ ]) [0.6. . 1]

Dempster shafer intervals in the form [s(P→D), p(P→D)] are obtained for each HPR. Further uncertainty of each rule can be obtained by using formula
u(A) = p(A) – s(A).

On the above discovered Quantified HCPR rules, approximate reasoning can also be applied according to Dempster Shafer theory explained in previous section.

Example 2: suppose we have 10 flat rules with 10 different classes as follows:
If P1 ∧ P2 ∧ P5 Then D1 : γ1
If P1 ∧ P2 ∧ P4 ∧ P7 Then D2 : γ2
If P1 ∧ P2 ∧ P4 Then D3 : γ3
If P1 ∧ P2 Then D4 : γ4
If P1 ∧ P2 ∧ P5 ∧ P6 Then D5 : γ5
If P1 ∧ P2 ∧ P3 Then D6 : γ6
If P1 ∧ P2 ∧ P4 ∧ P8 ∧ P11 ∧ P12 Then D7 : γ7
If P1 ∧ P2 ∧ P4 ∧ P8 ∧ P9 ∧ P10 Then D8 : γ8
If P1 ∧ P2 ∧ P4 ∧ P8 Then D9 : γ9
If P1 ∧ P2 ∧ P4 ∧ P8 ∧ P13 ∧ P14 ∧ P15 Then D10 : γ10

Finally, the proposed algorithm produced the following individual with the highest fitness = 9:
(D4(D6)(D3(D2)(D9(D7)(D8)(D10)))(D1(D5))).

The corresponding hierarchy is shown in Fig. 6.

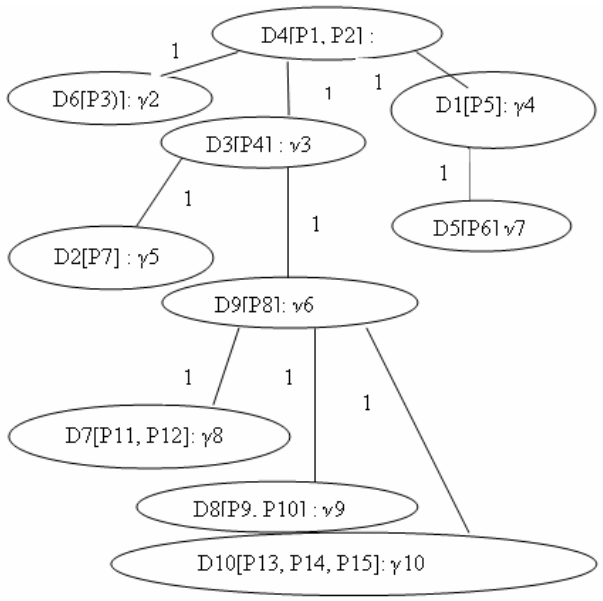


Fig.6 Hierarchy-the individual with the highest fitness=9

From the discovered hierarchy shown in Fig. 6, the following HPRs are generated:

level 0
(D4 If [P1, P2]
Generality [ ]
Specificity[D6,D3, D1]) : [γ1 .. 1]
level 1
(D6 If [P3]
Generality [D4]
(D6 If [P3]
Generality [D4]
Specificity [ ]) : [γ2 ..1]
(D3 If [P4]
Generality [D4]
Specificity [D2, D9]) :[γ3 ..1]
(D_1 [P_5]
Generality [D4]
Specificity [D5] :[γ4 ..1]
Specificity [ ]) : [γ2 .. 1]

(D3 If [P4]
Generality [D4]
Specificity [D2, D9]) [γ3 .. 1]
(D1 If [P5]
Generality [D4]
Specificity [D5]) :[γ4 .. 1]

level 2
(D2 If [P7]
Generality [D3]
Specificity [ ]): [γ5 .. 1]
D9 If [P8]
Generality [D3]
Specificity [D7,D8,D10]) [γ6 .. 1]
(D5 If [P6]
Generality [D1]
Specificity [ ]) [γ7 .. 1]

level 3
(D7 If [P11, P12]
Generality [D9]
Specificity [ ]) [γ8 .. 1]
(D8 If [P9, P10]
Generality [D9]
Specificity [ ]) [γ9 .. 1]
(D10 If [P13, P14, P15]
Generality [D9]
Specificity [ ]) [γ10 .. 1]

VI. Conclusion

As an attempt towards automated generation of hierarchies, a GP approach along with Dempster Shafer Theory is proposed to organize, summarize and present the discovered rules in the form of Quantified HPRs. Suitable genetic operators are proposed for the suggested encoding. Based on the SM, an appropriate *fitness* function is suggested. Performance of the proposed algorithm is demonstrated through experimental results, which are quite encouraging. Discovered Quantified HPR would facilitate quantitative reasoning and learning. One of the most important future research directions would be the discovery of Quantified HPRs with exceptions i.e. Quantified HCPR from large databases.

References

[1] Bharadwaj, K.K., Neerja and Goel, G.C. 1994, 'Hierarchical Censored Production Rules system employing Dampster-Shafer Uncertainty Calculus', Information and Software Technology, Vol 36, pp 155-174.
[2] Tamanna Siddiqui, K. K. Bharadwaj, "Discovery of Quantified Censored Production Rules from the Large set of Discovered rules", Proceedings of International Conference: Conference on Information Science, Technology and Management (CISTM 2006), Chandigarh, India. July 16-18, 2006.
[3] S. Levachkine and A. Guzman-Arenas, "Hierarchies measuring qualitative variables," Springer-Verlag Berlin Heidelberg 2004, A. Gelbukh (Ed.):CICLing 2004,2004,pp.262-274.
[4] B. Liu., M. Hu. And W. Hsu., " Inductive Representation of decision Trees using general rules and exceptions", AAAI-2000.
[5] Cios, K. J. Sztandera, L. M., "Continuous ID3 Algorithm with Fuzzy entropy measures. Proc. IEEE Int. Conf. Fuzzy Systems, San Diego, 469-476, 1996.
[6] U. Fayyad, G. P. Shapiro and P Smyth, "The KDD process for extracting useful knowledge from volumes of data", Communications of the ACM, vol.39, pp. 27-34, 1996.
[7] K. Sentz and S. Ferson (2002), 'Combination of Evidence in Dempster-Shafer Theory', Sandia National Laboratories report SAND2002-0835.
[8] Farhad Hussain, Huan Liu, Einoshin Suzuki, Hongjun Lu: Exception Rule Mining with a Relative Interestingness Measure. PAKDD 2000: 86-97.
[9] M. Suan, "Semi-Automatic taxonomy for efficient information searching," Proceeding of the 2nd International Conference on Information Technology for Application (ICITA-2004), 2004.
[10] J. R. Koza, "Genetic programming: on the programming of computers by means of natural selection," MIT Press, 1994.
[11] C. C. Bojarczuk, H. S. Lopes, and A. A. Freitas, " Genetic programming for knowledge discovery in chest pain diagnosis," IEEE Engineering in Medical and Biology magazine-special issue on data mining and knowledge discovery, 19(4), July/Aug 2000,pp.38-44.

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf

[12] Tamanna Siddiqui, "A KDD Tool for automated Discovery of knowledge", Proceedings of the 2nd national Conference INDIA Com – 2008, Computing for nation development, February 08 – 09, 2008.
[13] 223012 (M01) Statistical quantification of the sources of variance in uncertainty analysis: Robinson R.B., Hurst B.T., Risk Analysis, Volume 17, Nr. 4, pp 447–454
[14] K. K. Bharadwaj and R. Varshneya, "Parallelization of hierarchical censored production rules," Information and Software Technology, 37, 1995, pp.453-460.
[15] K. K. Bharadwaj and N. K. Jain, "Hierarchical censored production rules (HCPRs) systems," Data and Knowledge Engineering, North Holland, vol. 8, 1992, pp.19-34.
[16] J. Han, and Y. FU, "Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases," AAAI'94 Workshop Knowledge in Databases (KDD'94), Seattle, WA, July 1994, pp. 157-168.
[17] H. Surynato and P. Compton, "Learning classification taxonomies from a classification knowledge based system," Proceedings the First Workshop on Ontology Learning in Conjunction with ECAI-2000, Berlin, pp.1-6.
[18] B. Liu, M. Hu, and W. Hsu, "Multi-level organization and summarization of the discovered rules," Boston, USA, SIGKDD-2000, Aug 20-23, 2000.
[19] D. Richards and U. Malik, "Multi-level rule discovery from propositional knowledge bases," International Workshop on Knowledge Discovery in Multimedia and Complex Data (KDMCD'02), Taipei, Taiwan, May 2002, pp.11-19.
[20] A. A. Freitas, "A survey of evolutionary algorithms for data mining and knowledge discovery," In: A. Ghosh, and S. Tsutsui (Eds.) Advances in Evolutionary Computation, Springer-Verlag, 2002.
[21] I. De Falco, A. Della Cioppa, and E. Tarantiono, "Discovering interesting classification rules with genetic programming," Applied Soft Computing, 1, 2002, pp.257-269.
[22] M. V. Fidelis, H. S. Lopes, and A. A. Freitas, "Discovering comprehensible classification rules with a genetic algorithm," Proc. Congress on Evolutionary Computation-2000 (CEC'2000), La Jolla, CA, USA,IEEE, July 2000, pp.805-810.

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/11102/pdf