A Method For Quantifying the Importance of Facts, Rules and Hypotheses

by

T.J. Parsons
British Aerospace, Hatfield

Abstract

A labelled digraph is used as a model of a simple database, nodes representing facts (or classes of facts) and arcs the relationships between these facts. An expression for the number of microstates in which such a data structure may exist is derived and used to calculate a measure of intrinsic entropy. This measure is fundamentally related to the information content of the structure and its change, on adding or subtracting an item of information from the database, may be used to associate a value called *Importance* with the item.

An algorithm called the " Database Monitor Program (DMP) " is introduced. Its function is to guarantee that the digraph database exists in a ' least complex state' by replacing relationships between nodes by relationships pertaining to classes of nodes, but with the constraint that the information content of the structure remains unchanged. Once in this minimal condition, the *Importance* of an item is defined in terms of the change it induces on the minimum entropy state.

Like measures of entropy, the *Importance* of an item is a relative concept in that its value depends on the context of the database. It is argued that this is an intuitively appealing measure in that a system is only able to judge the importance of an item in the light of existing knowledge.

Two additional concepts termed *confidence* and *significance* are introduced and used to assess the formation of 'class concepts' within the database. The use of these three measures for conflict resolution is also discussed.

Finally, an example system developed within the Poplog environment is presented and extensions of the work are discussed.

1 Introduction

At least six different techniques exist which attempt to handle vagueness or incompleteness in the information that is to be presented to a knowledge–based system.

Bayesean Probability, Certainty theory [7], The Theory of Evidence [6] and Possibility Theory [8] all seek to assign numerical values to assertions, and arithmetic functions to logical connectives — hence providing a mechanism for calculating and associating values with rules and goals. This mechanism may be called "Probabilistic Reasoning".

The work of Bundy [1] discussed the limitations of this purely numerical approach. In the case of non-independent laws, knowledge from all combinations of evidence and hypotheses has to be considered and this is not feasible in many applications. Bundy also argues that the final value obtained for goal assertion does not represent the probability of that assertion but rather is a measure whose exact meaning is ill-defined and can only be used to rank goals in some order. Bundy proposed an Incident Calculus to overcome these difficulties.

The Plausibility Theory of Rescher [5] is often used in combination with the other techniques in an attempt to deal with unreliable knowledge.

With the limitations of probabilistic reasoning in mind, this paper seeks to develop a different approach, termed "Importance Theory".

A significant feature of this method is that it neither assigns *a-priori* values to assertions or rules nor depends on probabilistic reasoning to assess the significance of hypotheses given the current evidence. Hence the inherent weakness of Probabilistic Reasoning is avoided; rather, a value which measures how 'important' the assertion, rule or hypothesis is to the database as a whole, is derived.

It is argued that *importance*, *significance* and *confidence* provide useful criteria by which the formation of class concepts within a database may be judged. The same criteria may also be used to judge the success of a model hypothesis which we wish to impose upon a database of facts. The preferred set of hypotheses, as reflected by our measure, is simply that which explains most of the database.

AVC 1987 doi:10.5244/C.1.7

The theory is developed initially for a database containing only certain information.

2 Knowledge Representation

The data structure capable of storing relational information upon which the ideas presented in this paper are developed is termed a directed graph (digraph). Digraphs are a simple and convenient way of representing relational information.

Digraphs may be formally defined by the following primitives and axioms:
The Primitives are:

p1 A set V of elements called nodes

p2 A set X of elements called arcs

p3 A function F whose domain is X and whose range is contained in V

p4 A function S whose domain is X and whose range is contained in V

The Axioms are:

A1 The set V is finite and not empty.

A2 The set X is finite.

A digraph may contain lines, parallel lines and loops but two restrictions on the concept of a digraph are of interest. The first we term a 'restricted digraph' where the existence of parallel lines is forbidden. The second we term a 'simple digraph' where the existence of both parallel lines and loops is forbidden.

Strictly, the data structure should be termed a 'hyper-digraph" in analogy with 'hyper–graph' structures [2], in that we permit nodes of the digraph to consist of a hierarchy of nested digraph structures.

3 Terms and Definitions

The concept of a *cluster* and a *class* are introduced.

1. A given subset of the nodes of the digraph database is termed a *cluster*. A 'Cluster' may thus contain just one node.

2. An Assertion, Rule or Hypothesis that is to be added or removed from the database is termed an *item*.

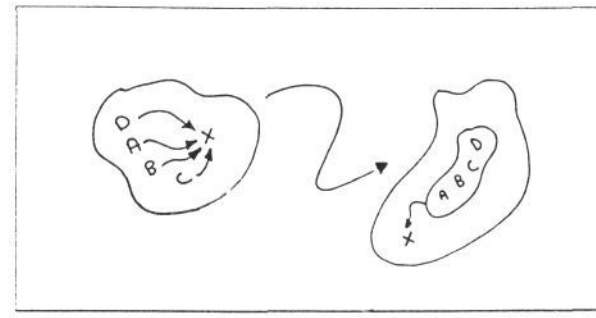

Figure 1: Complexity change on forming a class w.r.t. X

3. A 'cluster' is termed a *class* if each member of the cluster behaves identically with respect to a second cluster within the database. This second cluster may be the same, partially the same, or completely different (as regards its members) from the first cluster. The act of replacing individual relationships between members of a cluster and the database by a relationship between a class and the database constitutes the act of class formation and this process is illustrated in Fig. 1

4. Members of a 'class' may or may not behave identically with respect to each other, but those which do may be said to form a sub-class of that class.

5. The *minimal complexity state* of a digraph database is that form of the database containing the minimum number of microstates consistent with preserving the information within it.

The formation of a class α within a database may be considered to be the result of an operation on that database. If that operator is $\hat{R}$ acting on the database D then formally we write:

$$\hat{R}_\alpha D \rightarrow D_\alpha \qquad (1)$$

By the definition of a class, introduced above, it can be seen that the creation of a certain class does not affect the ability of each member of that class to be considered as members of another class. Hence the formation of a set of classes in a database, whether they overlap or not, results in the formation of a simplified database which is independent of the order in which classes are created, i.e., if $C(D)$ is the complexity of a database, then :

$$C(D_{\alpha\beta\gamma}) = C(D_{\beta\gamma\alpha}) = C(D_{\gamma\beta\alpha})...etc \qquad (2)$$

Fig. 2, presents a small example database and illustrates the reduction in complexity incurred under the transformation $\hat{R}_{\alpha\beta\gamma}$ and Fig. 3 traces the

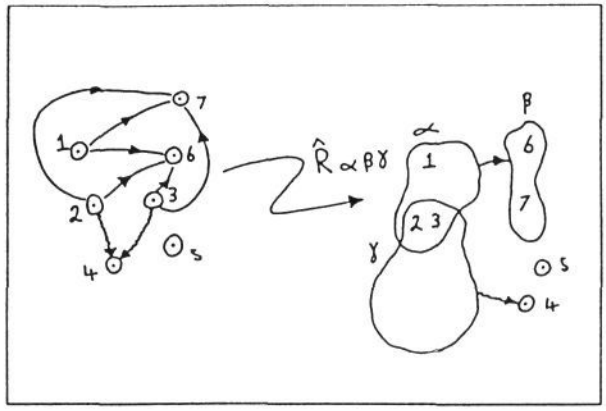

Figure 2: Unstructured database changing on creating classes α, β, γ

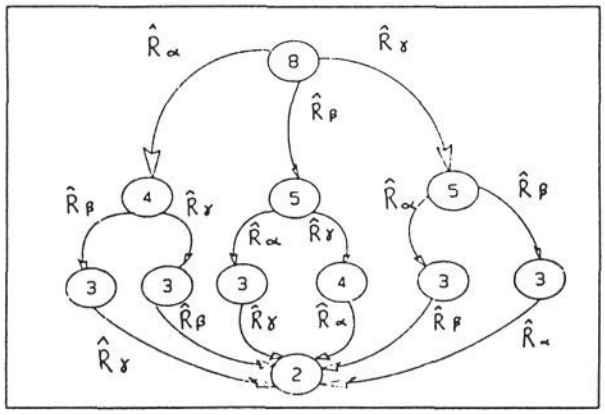

Figure 3: Tracing Arc Complexity changes on creating α, β, γ

changes in the number of arcs in the database on the formation of these classes to illustrate equation 2.

4 The Database Monitor Program (DMP)

Class formation is the technique by which the digraph database is iteratively simplified by the DMP until the 'minimal complexity state' is reached. The first step of the algorithm transforms a digraph into a restricted digraph by removing duplicated information in the form of parallel lines. The second part of the algorithm successively reduces the number of arcs and nodes in the database by class formation. The information content of the database is preserved throughout this transformation.

The DMP is central to the ideas presented in this paper because, before the importance of an item can be calculated with respect to a database by noting the change in entropy of that database on introduction of that item, it is necessary to ensure that the database is in its current minimal state.

It will be shown later that running the DMP on a static database gives a measure of the initial disorder of that database. Adding an item at this point and again running the DMP enables a measure of importance for the item to be defined as a function of the difference in these minimal states before and after addition of the item.

Equation 2 is extremely important in its implications to the design of the DMP because each class formation gives a reduction in the complexity of the database (or at least it can never give an increase) and because the final measure of the complexity reduction is independent of the path taken to reach it, we can guarantee to *always* find the global minimum. Indeed local minima do not exist. The DMP may be pseudo coded as follows:

```
PROCEDURE importance(database:list,
value:real)

{ Routine to reduce a labelled digraph
  database to its minimal complexity state.
  The value returned is a measure of the
  initial database's disorder. Classes or
  Nodes are called Clanodes}

BEGIN
  FOR (each type of labelled arc) DO
    REPEAT
      FOR (each clannode) DO
      {
       merge two clannodes if they
       behave identically w.r.t.
       another clannode.
      }
      END;
    UNTIL (no further complexity reduction);
  END;
  value:= initial_complexity-final_complexity;
END;
```

5 Unlabelled Digraphs— Derivation of Measures

The following analysis yields an expression for the state of entropy of a database modelled by the digraph structure. Consider a database capable of existing in a large number of states and changing its internal structure under the addition or removal of items (assertions, rules or Hypotheses). The ques-

tion is posed 'How does the entropy of the database change under such transformations? '.

Classical Physics defines the entropy of a system in terms of the number of Microstates in which it can exist, that is:

$$S = K(\ln \Omega) \tag{3}$$

where (Ω) is the number of Microstates.

The task remains to develop a formula yielding the number of Microstates in which a digraph of N nodes and m arcs may exist.

Given N nodes there are $N(N-1)$ possible places to insert an arc and N places to insert a loop. Hence the number of 'slots' P in which an arc may be placed is:

$$P = N^2 \tag{4}$$

Some thought shows that the number of distinct ways of placing m arcs into P slots (the number of microstates) is:

$$\Omega(m) = \frac{P!}{m!(P-m)!} \tag{5}$$

and hence the expression for entropy becomes

$$S = K \ln\left[\frac{P!}{(m!(P-m)!)}\right] \tag{6}$$

When adding or removing items to the database, the number of arcs and nodes is changed and a new value of the entropy measure results:

$$S_2 = K \ln\left[\frac{P_2!}{(m_2!(P_2-m_2)!)}\right] \tag{7}$$

where m_2 and P_2 are related to the number of arcs and nodes in the digraph after the addition of an item.

If P_1 and m_1 pertain to the state of the database before the addition of these items, then the associated change in entropy is:

$$\delta S = K \ln\left[\frac{P_1!m_2!(P_2-m_2)!}{(P_2!m_1!(P_1-m_1)!)}\right] \tag{8}$$

Sterling's approximation may be used to analyse the behaviour of equation 6 for large databases.

If both P and m are large then:

$$\ln(n!) = n\ln(n) - n \tag{9}$$

where n is either P or m. The expression for entropy becomes:

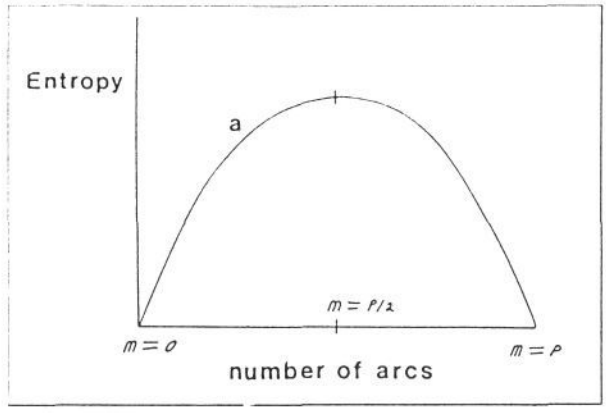

Figure 4: Diagram showing how entropy changes with the number of arcs in the database

$$S = K\Big[P\ln(P) - m\ln(m) - (P-m)\ln(P-m)\Big] \tag{10}$$

Note, by substitution in equation 10, that the function is symmetric in that if either $m \rightarrow P$ or $m \rightarrow 0$ then:

$$S \rightarrow 0 \tag{11}$$

and differentiating entropy with respect to the number of arcs yields:

$$\frac{\partial S}{\partial m} = -K\big[\ln(m)\ln(P-m)\big] \tag{12}$$

The function thus behaves as illustrated in Fig. 4. The underlying symmetry of the function reflects the essential duality of graph structures, in that the presence or absence of an arc may represent information.

The function illustrated in Fig. 4 warrants further discussion. At first inspection the symmetry about point $m = P/2$ is worrying because it implies an ambiguity in our measure of entropy.

The number of arcs in our database is related to the number of nodes. The minimum number of arcs the database can have is N whilst the maximum number it may have is N^2 (N is the number of nodes). Thus the point of symmetry $N^2/2$ lies in between this possible number of arcs for all N > 2 and one might hence indeed expect the entropy measure to be ambiguous. However, when one more arcs is added to the minimum number of arcs $m_{min} = N$, the DMP, as discussed in section 4, will recognise the existence of a class with two members and will simplify the knowledge base accordingly. The overall effect of the DMP is to ensure that the entropy

measure returned lies in that monotonically increasing region of the curve below $N^2/2$ and labelled "a" in Fig. 4 An unambiguous measure of the state of entropy for the knowledge base is thus assured.

6 Normalised Measures

A normalised measure of the Entropy change within a database may be developed by noting that the maximum possible importance that could be attributed to an item would be the one permitting the DMP to simplify the database to a 2-node, 1-arc structure.

Substituting P_2=4, m_2=1 into equation 8 gives:

$$\delta S_{max} = K \ln \left[\frac{P_1!}{(4m_1!(P_1 - m_1)!)} \right] \qquad (13)$$

δS_{max} is related to the constant K of equation 8. and the normalised measure of entropy change, called importance, becomes

$$I = \frac{\delta S}{\delta S_{max}} \qquad (14)$$

This function ranges between the values 0 and 1, 1 indicating that the maximum possible change to the database has occurred.

Note that, in practice, it is not necessary to fully evaluate all the factorials in equation because P and m are typically of the same magnitude.

In section 10, this normalised measure is used to calculate the importance of forming three concept classes alpha, beta, and gamma, within the example database of Fig. 2

7 Labelled Digraphs—Derivation of Measures

In the previous section an expression for the number of Microstates in which an unlabelled digraph could exist was developed. The analysis may be extended to labelled digraphs by noting that such a structure may be effectively decomposed into a set of singly labelled digraphs and this is illustrated in Fig. 5. If there are L labels then:

$$S = K \ln \left(\sum_{i=1}^{L} \frac{P!}{m_i!(P - m_i)!} \right) \qquad (15)$$

where m_i is the number of arcs of a given label.

A normalised expression for the change of entropy upon a reduction or increase in the number

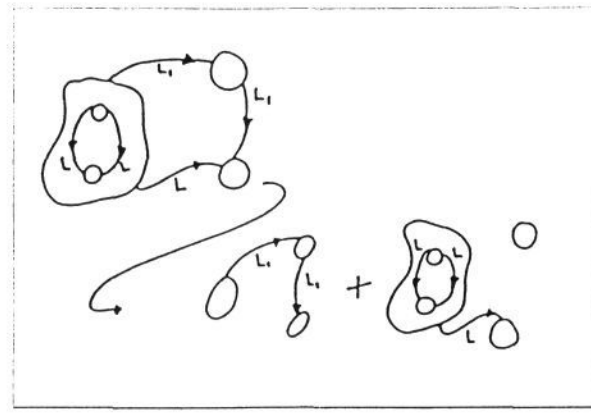

Figure 5: The decomposition of a labelled Digraph into a set of Digraphs

of arcs and nodes may again be developed by assuming that the maximum possible reduction of the labelled digraph is to a structure consisting of only two nodes and one arc.

$$I = \frac{\delta S}{\delta S_{max}} \qquad (16)$$

This equation is similar to equation 14, except that δS is derived from equation 15 rather than equation 6.

The above analysis assumes that the existence of a label L_1 between two nodes does not preclude the existence of a label L_2 between those same two nodes, i.e., the labels are assumed to be independent. In the system described here, which deals only with certain information, contradictory information already within the database is ignored and information already within the database, which is contradicted by new information, is updated.

8 Confidence and Significance measures

Fig. 6 illustrates three different ways in which the entropy of a database in its minimal state of complexity might change on adding items at point A and running the DMP at point B. Point C indicates the final state of the database.

The importance of an item may be defined as a function of the difference between the arc-node count at point A and point C.

$$I = \frac{f(m_a, n_a) - f(m_c, n_c)}{f(m_b, n_b)} \qquad (17)$$

f is the Importance function of equation 16.

Thus if $I = 0$, the information is of no importance to the database, perhaps because the item is already in the database. If $I < 0$ then the item has merely added to the complexity of the database. If $I > 0$ then the item is of importance to the database and the more important it is, the closer the value is to unity.

Two other measures are notable and they are termed Significance and Confidence. Significance is a measure of the complexity of the items to be added to the database.

$$\text{Significance} = \frac{(f(m_b, n_b) - f(m_a, n_a))}{f(m_b, n_b)} \qquad (18)$$

Thus, the nearer this measure is to unity, the greater the significance of the items added to the database.

Our measure of confidence in the importance value obtained is defined simply as

$$\text{Confidence} = \text{Importance - Significance} \qquad (19)$$

A near zero value indicates that little faith should be placed in the importance measure. A value near one indicates that the importance value is worth considering. This ability to quantify the extent of our 'confidence' in the 'Importance' value is extremely useful and no analogous measure exists in Bayesean probability theory.

The greater the significance of the items added in order to achieve a given reduction in complexity, the less confidence we have in the results. This measure is especially interesting when considering the effect of a hypothesis on a database. The least complex hypothesis to produce a reduction in the databases complexity is preferred to a more complex hypotheses (Occam's Razor?)

The complexity increase associated with the database on addition of a single fact, before the DMP is run, is easily calculated using equations 15 and 17. The number of arcs is increased by one and the number of nodes by two.

However, when several facts are to be added to the database as a group, they must themselves be regarded as a digraph and transformed by the DMP before a measure of its significance may be calculated. After this transformation, the importance of the group of facts w.r.t. the database is calculated in the established manner. The Significance and Confidence values may also be calculated.

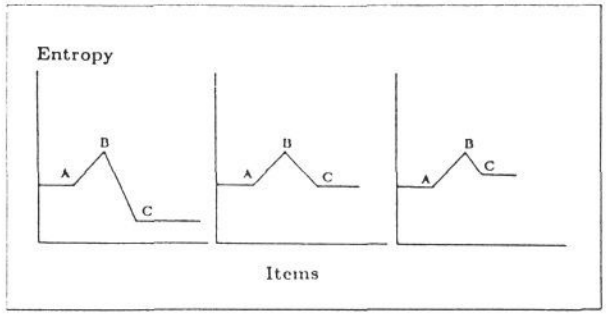

Figure 6: Diagram showing possible database Entropy changes

9 The Minimal Complexity Database – Further Notes

When adding items to the database, their importance is judged by calculating the change in complexity that occurs compared with the total change that could be possible (that is, reduction to a two node, single arc relation).

However, once the item is in the minimal complexity database, its importance is calculated by considering the displacement from this minimal state that its removal would induce on that database, the difference being that the importance value now returned is modified by the context of any items since added to the database.

The measure is now expressed as:

$$\text{Importance} = 1 - \frac{\ln(\Omega_{min})}{\ln \Omega} \qquad (20)$$

As each element is added to the database it becomes necessary to reform its minimal state in order to calculate measures for the item added and for any future items to be added.

This housekeeping may be performed efficiently by tagging both the number of input arcs and output arcs associated with any class or node. To see how this is possible, remember that the DMP reduces a digraph to a restricted digraph (i.e. removes any parallel lines), and eventually reduces the graph to such a form that neither the number of inputs or outputs to a node can exceed one. Thus the most complex structure that can exist in the minimal complexity database is a cycle with N members. These reductions are summarised in Fig. 7

On destruction of a class concept it is known, *a priori*, that only one arc related that class to the rest of the database. An exhaustive search is avoided. We can hence calculate immediately the increase in

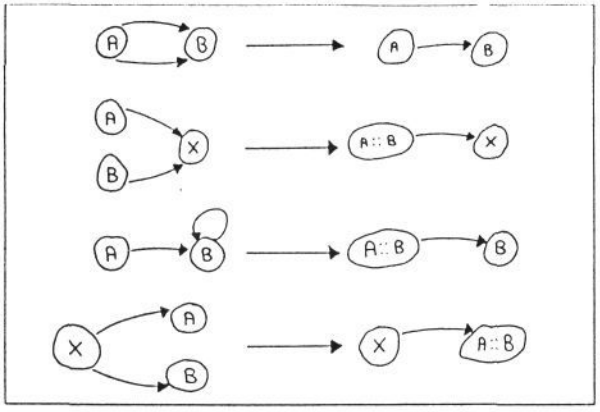

Figure 7: Database transformations under action of the DMP

the number of arcs and nodes that would be needed if that class information was to be destroyed. Also, on adding information to the minimal complexity database, nodes having more than one input or output may be tagged and only these need be addressed by the DMP Exhaustive searching of the database is again unnecessary.

The importance values associated with class concepts within the database are explored in the next section.

10 An Example System

The following system was developed within the Poplog environment [2], [4] particular use being made of 'Super Database', a Prolog type library package permitting depth-first searching and 'Sets', a package permitting the use of set operations.

The software was developed to enable a user to investigate the Importance, Significance and Complexity measures of classes formed automatically within a database by the DMP

The system was constructed so that, on the addition of facts or rules to the database, the associated values were both reported to the user and used to place any rules in a 'rule priority list'.

Relationships between nodes a,b were represented as an entry into the database of the following list.

```
[ fact ^ relationship [a] [b] ]
```

Relationships between nodes and classes, or classes and classes are of the same form.

```
[ fact ^ relationship [a b c d] [e f g h]]
```

Backtracking through the database is initiated by commands such as

```
Present([ and [ fact ^ relationship ?x ?y]
    [fact ^ relationship ?z ?w]]);
```

where x,y,z and w are variables which become instantiated on patterns within the database.

The database depicted in Fig. 2 is thus encoded as follows:

```
database==>
[fact ^ relationship  [1] [7] ]
[fact ^ relationship  [1] [6] ]
[fact ^ relationship  [2] [6] ]
[fact ^ relationship  [2] [7] ]
[fact ^ relationship  [3] [6] ]
[fact ^ relationship  [3] [7] ]
[fact ^ relationship  [2] [4] ]
[fact ^ relationship  [3] [4] ]
```

Central to the system are the following routines.

1. Status("relationship") → arcs → nodes;

   This routine counts the number of arcs and nodes within the database.

2. DMP();

   This routine, as described earlier, transforms a database with the above format into its minimal state of complexity, without loss of information.

3. Importance(m_1,n_1,m_2,n_2) → measure;

   This routine returns a measure of importance for an item added to the database (fact, rule or hypothesis) based on both the initial and final count of the number of arcs and nodes.

4. Weight(class) → measure;

   This routine reports the measure of importance to be associated with any fact that already exists within the database. Its definition will be given later.

Thus a measure of the initial disorder of a static database would be provided by the following sequence of calls.

1. status("relationship") → m_1 → n_1;
2. dmp();
3. status("relationship") → m_2 → n_2;

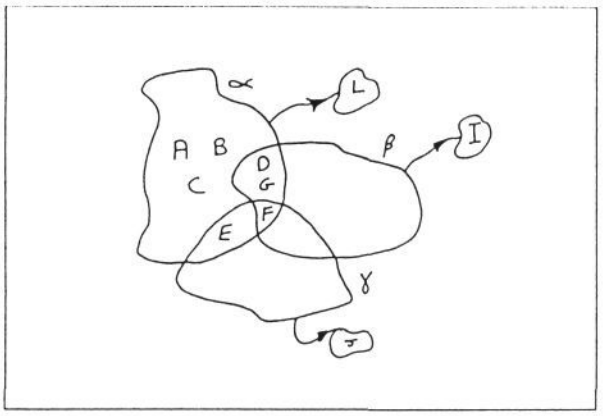

Figure 8: Class concepts and associated Importance values

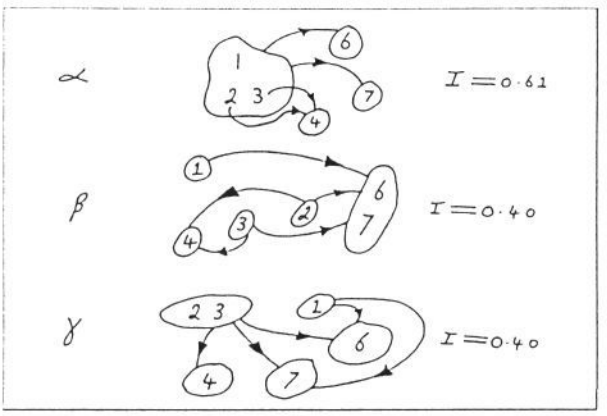

Figure 9: Illustrating class formations within the Example Database

4. importance(m_1,n_1,m_2,n_2)→ measure;

After running DMP the database becomes:

```
database ==>
[fact ^ relationship [a b c] [ f g] ]
[fact ^ relationship [b c ] [d ] ]
```

with a measure of disorder of 0.78

The complexity changes on the formation of the three classes α, β and γ in the example database of Fig. 2 may be traced. Fig. 9 illustrates the database changes incurred on forming these classes separately and lists the importance value associated with each one.

Let I($\alpha|O$) be the importance to the database when forming a class α. Let I($\alpha|\beta$) be the importance to the database when forming a class α given that a class β already exists. Using equation 16 it can be calculated that:

I($\alpha|O$) = 0.61

I($\beta|O$) = 0.4

I($\gamma|O$) = 0.4

I($\alpha|\beta$) = 0.68

I($\gamma|\beta$) = 0.64

As might be expected, the formation of class α in the database is considered to be more important than the formation of β or γ. After class β is formed, the importance of forming class α is still more important than the formation of class γ.

Fig. 8 illustrates another database of interest. Three classes α, β, γ have again been formed by the DMP and class α might be considered very important with respect to the database because of its size. The importance values, as calculated using equation 16, are 0.77, 0.31 and 0.19 for classes α, β and γ respectively. Thus class α is indeed more important than classes β and γ. However, once the classes are all formed within the database, equation 20 can be used to calculate the new Importance values 0.56, 0.38 and 0.19 for α, β and γ respectively. The importance associated with class α has been reduced considerably because of the large degree of overlap with the class concept β.

Early experiments indicated that a 50 node graph could be reduced to its minimal complexity state within 40 seconds *real time* on a VAX 11/785 under normal load conditions. Whilst this was not particularly fast, it was considered usable for developing the type of system presented in this paper. In addition it was shown that this minimal state was indeed independent of any initial ordering of that database.

11 Rules and Hypotheses

The importance of a rule with respect to our database is evaluated by using the rule to create facts which are not already explicitly present.

Some rules lead to a profusion of new facts and relationships within the database and a dramatic increase in the number of arcs and nodes. It is these rules which are considered to be the most relevant and, in the current system, the importance value is used to placed them in a 'rule priority list'. Hence the production rules fired first, upon modifications to the database, are the ones which seem most relevant.

Hypotheses may also have a dramatic effect on the database and, for clarity, a particular exam-

ple is given. Consider the situation of a relational database derived from a complex scene containing a car. The identification of a large sub-section of the relational graph containing the structural description of a car may be be considered as a class concept named "car". The consideration of the single class "car" as opposed to the complexity it may hide induces a large simplification in our database and hence the hypothesis would be given a large importance value. Expressed in a slightly different way, the class concept "car" explains a large portion of our relational database and hence has a large value of Importance associated with it. If, on the other hand, only partial evidence existed for the presence of a car, then the size of the digraph required to complete the class concept *car* and quantified as *significance* together with the *importance* of the class concept, enables a *confidence* value for the observed partial evidence to be obtained.

12 Future Work

The work presented in this paper is limited in that the theory deals only with a database containing certain information. However this forms the basis of current efforts which attempt to deal with databases represented by weighted labelled digraphs. This is regarded as an adequate representation for uncertain relational information.

Digraphs are easily represented in terms of connectivity matrices and the DMP operation may also be represented in terms of a set of matrix operations. Thus the possibility of dramatically increasing the current execution speed of algorithms presented in this paper on both serial computers and, to a much greater extent, on parallel architectures is real. This work will be presented in a future paper.

13 Acknowledgements

The work was conducted within the MMI 007 Alvey consortium. I would like to thank my ex-colleague D. F. Nuttall for many valuable conversations and my colleague Dr Clark for his help with LaTeX layout and typography.

References

[1] Bundy A. Incidence calculus: a mechanism for probabilistic reasoning. *Department of Artificial Intelligence, Edinburgh University*, Research paper no. 216 Edinburgh:, 1984.

[2] Claude Berge. *Graphs and Hyper-Graphs.* North-Holland Mathematical Library, University of Paris, 1979.

[3] J. Gibson. *An A.I. Programming Language. Cognitive Studies Program.* Technical Report, University of Sussex, Brighton, 1980.

[4] Rescher N. *Plausible Reasoning.* Amsterdam: Van Gorcum, 1976.

[5] G. Shafer. *A Mathematical Theory of Evidence.* Princeton University Press, Princeton, New Jersey, 1976.

[6] E.H. Shortliffe and B.G. Buchanan. A model of inexact reasoning in medicine. *Mathematical Biosciences 23, 351-379*, 1975.

[7] A.S. Sloman and J. Gibson. *Poplog: A Multi-language Program Development Environment in Image Technology.* Technical Report, University of Sussex, 1983.

[8] Zadeh. *Fuzzy Sets as a basis for a theory of possibility. Fuzzy sets and systems*, Amsterdam: North Holland, 1978.