Information Compression by Multiple Alignment, Unification and Search as a Framework for Human-Like Reasoning*

Gerry Wolff,
School of Informatics, University of Wales at Bangor,
Dean Street, Bangor, LL57 1UT.
E-mail: gerry@sees.bangor.ac.uk

Abstract

This paper is an overview of the idea that several aspects of human-like reasoning may be understood in a framework of information compression by 'multiple alignment', 'unification' and 'search'. These concepts will be described in outline together with a description of the SP61 computer model which is a partial realisation of the framework. A selection of examples are presented, illustrated with output from the SP61 model.

1 Introduction

This paper is an overview of the way in which human-like reasoning may be understood as information compression (IC) by 'multiple alignment', 'unification' and 'search' (ICMAUS). This framework is described in outline, together with a brief description of the SP61 computer model which is a partial realisation of the framework. A selection of examples are presented, illustrated with output from the SP61 model.

Since an ability to reason with uncertainties is, perhaps, the most distinctive feature of everyday human reasoning, a study of human-like reasoning is, to a large extent, a study of 'probabilistic' reasoning, broadly conceived. In this paper, the phrase "human-like and probabilistic reasoning" will be abbreviated as 'HPR'.

A much fuller description of the ICMAUS ideas with a greater range of examples may be found in [8] and other sources referenced therein. An early account of how different kinds of reasoning may be modelled as ICMAUS and how reasoning may be integrated with learning may be found in [10]. This early account was written before the development of a working computer model and methods for calculating probabilities.

*Paper to be presented at the International Conference on Pure and Applied Practical Reasoning, FAPR2000, Imperial College London, September 2000.

1.1 Context

The work to be described is part of a programme of research exploring the conjecture that *all kinds of computing and formal reasoning may usefully be understood as ICMAUS*. The intention is to discover or develop a framework which can integrate and simplify a range of concepts in computing.

Apart from the modelling of different kinds of reasoning, there is evidence of the potential of the ICMAUS framework in a number of other areas: 1) Article [9] describes how the ICMAUS framework may provide an interpretation of the concept of 'computing' itself. 2) Article [7] shows how the framework may be applied to the parsing and generation of natural language texts. 3) The similarity between parsing and pattern recognition suggests that the framework may be useful in that area too. 4) Earlier work [11] shows how a process for finding good partial matches (which is now incorporated in the ICMAUS model) provides a powerful technique for best-match information retrieval. 5) The current programme of research was inspired by earlier work on unsupervised learning of grammars (see [12] and earlier papers cited there). Work is now in progress to realise the potential of the framework for unsupervised learning of grammars and other kinds of knowledge structure.

1.2 Related Research

There is now a vast literature relating to HPR: 'standard' parametric and non-parametric statistics; *ad hoc* uncertainty measures in early expert systems; Bayesian statistics; Bayesian/belief/causal networks; Markov networks; Self-Organising Feature Maps; fuzzy set theory and 'soft' computing'; the Dempster-Shaffer theory; abductive reasoning; reasoning with default values and non-monotonic reasoning; autoepistemic logic, defeasible logic, probabilistic, possibilistic and other kinds of logic designed to accommodate uncertainty; Minimum Message Length encoding (MML) or Minimum Description Length encoding (MDL); algorithmic probability and algorithmic complexity theory; truth maintenance systems; decision analysis; utility theory; and so on.

Useful reviews of a variety of issues in this area, with an emphasis on logics for reasoning with uncertainty, may be found in [1]. Currently, Bayesian networks (see, for example, [2]) are amongst the most popular systems for reasoning with uncertainties.

1.2.1 MML/MDL Encoding

As described in [8], the ICMAUS concepts are founded on concepts of MML/MDL encoding. The general idea here is that, if we are trying to induce a 'grammar' or other kind of knowledge structure from a 'corpus' of 'raw' data, we should seek to minimise $(G + E)$, where G is the size of the grammar (in bits or equivalent units of information) and E is the size of the corpus when it is encoded in terms of the grammar.

This important insight (due to Solomonoff [3, 4]) guards against the induction of grammars that are too small (e.g., the grammar which is merely a set of rules corresponding to the set of ASCII characters that can generate every possible sequence of ASCII characters) or grammars that are too large (e.g., the grammar that has a single rule which is simply a copy of the corpus). With grammars of the first type, G is small but E is large and with grammars of the second type the reverse is true. Seeking to minimise (G + E) seems to produce a happy compromise. It also yields valuable insights into the nature of 'generalisation' in grammar induction (see [12]).

In case readers are wondering about how inductive learning might relate to HPR, an 'ICMAUS' view of the relationship is described below.

Although the present research is founded on established principles, the way in which the notion of multiple alignment has been developed in this research and applied to the modelling of HPR, appears to be a novel approach to the subject with no close relation in any other research.

2 The ICMAUS Framework and the SP61 Model

In its most general form, the ICMAUS framework is envisaged as a system for the unsupervised inductive learning of grammar-like structures which works like this:

1. Starting with little or no knowledge of the 'world', the system receives raw data from the world via its 'senses'. These data are designated 'New'.

2. As each portion of New is received, the system tries to compress it by encoding as much as possible of it in terms of patterns or structures that the system already knows. If for example, the system knows the pattern "information compression" and has given it the code "IC", then whenever this pattern appears in New, it may be abbreviated as "IC".

3. Portions of New that can be encoded in terms of stored knowledge, are stored in their encoded form. Every other portion is stored in raw form but is given a new code by the system for possible use in the encoding of New information in the future. All stored knowledge is designated 'Old'.

In these broad terms, this incremental scheme for compression is similar to the well-known and widely-used Lempel-Ziv algorithms (well described in [6]). What is different about the ICMAUS scheme is its search for 'good' partial matches between patterns and the way a concept of 'multiple alignment' has been developed. In particular, in some applications (e.g., parsing), multiple alignment can achieve the effect of encoding New information in a hierarchy of 'levels'. This is explained in [8].

The focus of interest in this paper is on stage 2 of the ICMAUS framework. As we shall see, the search for a 'good' encoding of each incoming pattern can, as a by-product, via multiple alignment, yield an interesting variety of kinds of HPR.

2.1 Multiple Alignment

Multiple alignment is a term borrowed from bio-informatics where it means the arrangement of two or more sequences of symbols in horizontal rows one above the other so that, by judicious 'stretching' of sequences where necessary, symbols that match each other from one sequence to another can be brought into alignment in vertical columns. A 'good' alignment is one in which relatively large numbers of symbols are brought into alignment. Multiple alignments like these are used in the analysis of sequences of amino acid residues in proteins or sequences of bases in DNA molecules.

The term *pattern* is used here rather than 'sequence' as a reminder that, at some stage, it intended to generalise the ICMAUS ideas to alignments of two-dimensional patterns, or even patterns in higher dimensions.

In the present research, the concept of multiple alignment has been adapted in the following ways:

- In accordance with the overall framework described above, one or more of the patterns to be aligned is designated 'New' and the rest are classified as 'Old'.
- 'Good' alignments are ones that achieve a relatively large compression of New by unification and encoding in terms of patterns in Old. Details of the method of encoding and the way in which IC may be calculated are given in [7] and [8].
- By contrast with alignments in bio-informatics, alignments in the present research can contain two or more *appearances* of any one pattern. Notice that this is *not* the same as two or more *copies* of a pattern. There is only a single pattern in two or more appearances of a pattern which means, amongst other things, that it is not permissible to match a symbol in one appearance of a pattern with the corresponding symbol in another appearance of the same pattern - because that would mean matching the symbol with itself.

Examples will be seen below showing how this concept of multiple alignment can be used in practice.

2.2 The SP61 Model

As it stands now, the SP61 model is designed for stages 1 and 2 of the ICMAUS framework and makes no attempt at stage 3. Work is currently in progress to develop a new version of the model to accommodate the framework in its full generality, including stage 3.

A relatively full description of the SP61 model, is given in [7]. A briefer description is given in [8].

The SP61 model has been developed to find 'good' alignments between a single pattern in New and one or more patterns in Old. An outline of how it works is shown in Figure 1.

```
main()
{
     1 Read one or more patterns into Old and one pattern
          into New.
     2 Assign the pattern in New as the first and only member
          of a list of 'driving' patterns.
     3 Identify the patterns in Old as the initial list of
          'target' patterns.
     4 while (new alignments are being formed) compress() ;
}

compress()
{
     1 while (the end of the current list of driving patterns
          has not been reached)
     {
          1.1 Select the first or next driving pattern from
              the current list.
          1.2 Search for one or more 'good' alignments, each one
              between the selected driving pattern and any one of
              the set of target patterns (the patterns in Old and
              the 'unified' alignments formed by the program).
          1.3 Unify each of the new alignments so that it can be
              treated as a single sequence of symbols during
              matching. The structure of the alignment is
              preserved for other purposes
          1.4 Each new unified alignment is kept in a temporary
              store until the end of this iteration.
     }
     2 Add the newly-formed unified alignments to a cumulative
          list of alignments formed by the program.
     3 Empty the list of driving patterns and then select a
          few of the best of the newly-formed unified alignments
          and add them to the list of driving patterns, ready
          for the next iteration of compress(). Although these
          patterns are identified as driving patterns,
          they also retain their status as target patterns.
}
```

Figure 1: Outline of the operation of the SP61 model.

```
0        G G A G C     A G G G A G G A T G G     G G A    0
         | |   | |     | | | | | | | |   | |     | | |
1 %S 0 G G G   G C C C A G G G A G G A   G G C G G G A #S 1

(a)

0      G G A     G C     A G G G A G G A T G G G G A    0
         | |     | |     | | | |   | |   | | |   | |
1 %S 1 A G A C T G C C C A G G G G G G C T G G A G A #S 1

(b)

0      G G A G C A   G G G A G G A T G G G G A    0
       | | |     |   | | | | | | |   | | | | |
1 %S 2 G G A     A A G G G A G G A A G G G G A #S 1

(c)

0      G G   A G C A G G G A G G A T G G G G A    0
       | |   |   | | | | | | | | |   | | | | |
1 %S 3 G G C A   C A G G G A G G C   G G G G A #S 1

(d)
```

Figure 2: The best alignments found by SP61 between one DNA in New and each of four different sequences in Old.

2.2.1 Finding Good Partial Matches Between Two Patterns

At the heart of SP61 (step 1.2 of the *compress()* function in Figure 1) is a method for finding good full or partial matches between two patterns [11]. This method achieves the effect of dynamic programming (see, for example, [5]) but, unlike standard methods for dynamic programming, it can match strings of arbitrary length and can vary the thoroughness of searching according to need.

To give a feel for what it can do, Figure 2 shows the best alignment found by SP61 between one DNA sequence in New and each of four different sequences in Old.

The first alignment in the figure may be unified by collapsing the two rows into one. This yields the sequence 'G G G A G C C C A G G G A G G A T G G C G G G A'. In a second iteration of the *compress()* function, this sequence may be aligned with each of the remaining three patterns in Old. The best alignment found in this case is shown in Figure 3. In this kind of way, alignments of several different patterns may be built up.

The foregoing may sound plausible enough but there is an important qualification. Figure 2 (a) can be collapsed into a single sequence but the same is

```
0        G G A     G C     A G G G A G G A T G G     G G A    0
         | | |     | |     | | | | | | | | | | |     | | |
1 %S 0 G G G |     G C C C A G G G A G G A | G G C G G G A #S 1
  |        | |     | | | | | | | |   | |   | | |       | | |
2 %S 1 A   G A C T G C C C A G G G G G G C T G G A     G A #S 2
```

Figure 3: The best alignments found by SP61 in a second iteration of the *compress()* function of Figure 1.

not true of the remaining three alignments in the figure. This is because each of these last three alignments contain 'mis-matches': parts of the alignment where one or more unmatched symbols in one sequence appears opposite one or more unmatched symbols in the other. In Figure 2 (d), for example, 'A T' in the top row appears opposite 'C' in the bottom row.

It is intended that mis-matches of that kind will be handled in stage 3 of the ICMAUS framework. Pending a solution to that problem, the SP61 model has been designed so that it discards all alignments containing mis-matches at the end of the first iteration of the *compress()* function and in all subsequent cycles.

Despite this apparent limitation of the model, it performs well in certain applications such as parsing and generation of natural language [7], best-match information retrieval [11], and in modelling different kinds of probabilistic reasoning [8].

2.2.2 Computational Complexity

The time complexity of SP61 has been estimated [8] to be $O(nm)$ in a serial processing environment, where n is the length of New in bits and m is the total size, in bits, of all the Old patterns against which New may be aligned. If SP61 were to be adapted for a high-parallel processing environment, it is estimated that the time complexity would approach $O(n)$, depending on how the parallelism was applied. In all cases, the space complexity is estimated to be $O(m)$.

These figures for computational complexity show that, although the model is not as good as one might wish, its scaling properties are well within the bounds of what is normally regarded as acceptable.

In the following sections, examples from the model are presented which are quite small. Readers should note that **small examples are presented both for the sake of clarity and to save space. They do *not* represent the limits of what the model can do.** Much larger and more complex examples may be seen in [7].

3 Best-Match Pattern Recognition and Information Retrieval

One of the most prominent features of human (and animal) perception, is the ability to recognise an object or pattern when only part of it is seen or heard. In almost any scene, there are many objects that are partly obscured by other objects and yet we normally recognise them without difficulty. Within wide limits, it does not matter which parts of an object are obscured and which are visible. A few bars of music may be enough to identify a symphony. Words containing mis-spellings or 'typos' of various kinds can often be identified correctly, even without disambiguating context.

This kind of ability, which we use constantly in everyday living, may be seen as a form of reasoning. If we see part of a chair in a typical scene and infer that the whole chair is present in the scene, we are, in effect, reasoning that the evidence of the part we can see means that the rest of the chair will be found if we look behind whatever other object or objects are obscuring it. This seems to be no different in principle from Sherlock Holmes inferring from scratches around the keyhole in the front door of a house that the owner of the house was often drunk when he (or she) came home.

Both examples may be seen as the completion of a partial pattern and in every case the reasoning cannot be certain. In identifying the partly-obscured chair, we may find that the parts we could not see are indeed missing. And of course, if Sherlock Holmes were not a fictional character who never makes mistakes, his inference about the owner of the house could easily be wrong.

3.1 Example

To see how this kind of ability may be modelled by SP61, consider Figures 4 and 5. The first figure shows a simple database of information about types of motorised vehicles, where each type of vehicle is represented by a pattern describing the attributes of that type. Readers will notice some redundancy in this database (information is often repeated in different patterns) and we shall say more about this in Section 4. The number in brackets at the end of each pattern is a notional frequency of occurrence of that type of motor vehicle in some imaginary sampling of traffic at a particular location.

For the case where the pattern 'engine cab few-windows closed compartment' is in New (representing an incomplete description of any one type of vehicle) and the patterns from Figure 4 are in Old (representing the types of vehicle that the system knows about), Figure 5 shows the two alignments found by SP61 which encode the maximum number of symbols from New (which in this case is all the symbols in New).[1]

[1]In this and subsequent examples, readers will see that it is necessary for there to be consistency in the ordering of symbols amongst the patterns in New and Old. This is because SP61 treats all patterns as *ordered* arrays of symbols. Notwithstanding this feature of the way the program works, it is possible with SP61 to model the effect of matching unordered collections of attributes. This is described in Section 10 of [8].

```
mini-bus passenger-vehicle engine seats 10 #ss
     wheels 4 #ws windscreen saloon many-windows
     luggage-compartment #mns (500)
double-decker passenger-vehicle engine
     seats 40 #ss wheels 4 #ws windscreen
     saloon two levels #sn many-windows
     luggage-compartment #dr (2500)
coach passenger-vehicle engine seats 40 #ss
     wheels 4 #ws windscreen saloon many-windows
     luggage-compartment #ch (1000)
family-car passenger-vehicle engine seats 4 #ss
     wheels 4 #ws windscreen saloon many-windows
     luggage-compartment #fr (20000)
sports-car passenger-vehicle engine seats 2 #ss
     wheels 4 #ws windscreen saloon soft top #sn
     many-windows luggage-compartment #sr (700)
flatbed-lorry goods-vehicle engine seats 2 #ss
     wheels 4 #ws cab windscreen carries-load
     few-windows flat load platform #fy (2700)
pantechnicon goods-vehicle engine seats 2 #ss
     wheels 4 #ws windscreen cab carries-load
     few-windows large closed compartment #pn (750)
van goods-vehicle engine seats 2 #ss
     wheels 4 #ws windscreen cab carries-load
     few-windows small closed compartment #vn (2400)
```

Figure 4: A set of patterns describing different kinds of motorised vehicle. The number at the end of each pattern is a notional frequency of occurrence in an imaginary sample. *Key:* ch = coach, dr = double-decker, fr = family-car, ge = goods-vehicle, fy = flatbed-lorry, mns = mini-bus, pn = pantechnicon, pe = passenger-vehicle, sr = sports-car, sn = saloon, ss = seats, ws = wheels, ve = vehicle, vn = van.

```
0       ee                          cb     fs     cld  ct     0
        |                           |      |       |   |
1 vn ge ee ss 2 #ss ws 4 #ws wn cb cd fs sl cld ct #vn 1

(a)

0       ee                          cb     fs     cld  ct     0
        |                           |      |       |   |
1 pn ge ee ss 2 #ss ws 4 #ws wn cb cd fs le cld ct #pn 1

(b)
```

Figure 5: The two best alignments formed by SP61 with 'engine cab few-windows closed compartment' in New and the patterns shown in Figure 4 in Old. *Key:* cb = cab, cd = carries-load, cld = closed, ct = compartment, ee = engine, fs = few-windows, ge = goods-vehicle, le = large, pn = pantechnicon, ss = seats, sl = small, vn = van, ws = wheels, wn = windscreen.

For any combination of New and Old, the set of alignments found by SP61 that encodes the most symbols in New will be referred to as the 'best set of alignments' or 'best alignments'. Almost invariably, alignments selected in this way have the highest 'compression score' calculated according to the methods described in [8].

In the present case, the two best alignments have, in effect, selected the types 'van' and 'pantechnicon' as the two types of vehicle which give the best fit to the pattern in New.

Using methods which are described in [8], SP61 calculates the relative probability of 'van' to be 0.911 and the relative probability of 'pantechnicon' to be 0.089. In effect, these are probabilistic inferences that the attributes in the alignments which are **not** matched with any symbols in New will be found in association with the symbols in New.

In general, the symbols in any ICMAUS alignment that are *not* matched to any other symbols may be seen to be probabilistic inferences generated by the system.

3.2 Best-Match Information Retrieval

Pattern recognition may be regarded as a form of information retrieval. The example of pattern recognition just shown may be regarded equally well as an example of how a 'query' pattern ('engine cab few-windows closed compartment') can retrieve two 'records' from a database. As before, the unmatched parts of the stored records may be regarded as probabilistic inferences which the system makes.

There is insufficient space for examples here but we may note that the ICMAUS framework can not only model the retrieval of records by direct matching, as shown in Figure 5, but it can also achieve the effect of *indirection* in information retrieval as, for example, when the query 'house' retrieves the related word 'home' and that in turn leads to other records in which that word appears.

4 Classes, Subclasses, Objects and Inheritance of Attributes

In everyday human thinking, we not only use basic classes like those illustrated in Section 3 but we recognise hierarchies of classes which increase in generality from the bottom to the top. This is a powerful means of avoiding unnecessary redundancy in the storage of information because information which is true of several different entities or classes can be stored just once in a higher-level class so that the lower-level entities or classes may store only the information which is distinctive for the entity or class. When required, the higher-level information can be *inherited* down the hierarchy of classes. Thus, for example, there is no need to store the fact that a given person ('John') has two arms and two legs because we can infer these basic facts simply by knowing that John is a person and that, in general, every person has two arms and two legs. If John has lost any arms or legs, we can store this particular fact about John in our memory of John.

As we shall see in the example illustrated in Figures 6 and 7, this idea of class hierarchies with inheritance of attributes can be modelled quite neatly in the ICMAUS framework.

The first of these figures shows a set of patterns modelled on the patterns in Figure 4 but with much of the redundancy removed by expressing repeated information only once. For example, the first pattern in the figure says, in effect, that every vehicle has an engine and a windscreen. This means that it is not necessary to repeat this information at lower levels.

In a similar way, the second and third patterns in Figure 4 may be seen to represent information which is true of the classes 'passenger vehicle' and 'goods vehicle' respectively. The remaining patterns represent individual types of vehicle.

When these patterns are placed in Old and the pattern 'engine cab few-windows closed compartment' is placed in New, the two best alignments found by SP61 are the two alignments shown in Figure 7. As before, the phrase 'the [two] best alignments' means the alignments which encode the most symbols in New.

This figure shows how the 'code' symbols like 'ss' and '#ss' have the effect of tying patterns together so that the inferences that can be drawn from the alignments are the same as before. A pair of symbols like 'ss #ss' may be regarded as a 'variable' into which information may be inserted. The relative probabilities calculated by SP61 are the same as before.

In effect, the alignments shown in Figure 7 achieve the effect of *inheritance* of attributes like 'windscreen' (from the high-level class 'vehicle') and 'carries-

```
vehicle engine seats #ss
     wheels #ws windscreen #ve (9850)
passenger-vehicle vehicle #ve saloon #sn
     luggage-compartment many-windows #pe (4000)
goods-vehicle vehicle #ve cab
     carries-load few-windows #ge (5850)
mini-bus passenger-vehicle seats 10 #ss
     wheels 4 #ws #pe #mns (500)
double-decker passenger-vehicle saloon two levels
     #sn seats 40 #ss wheels 4 #ws #pe #dr (2500)
coach passenger-vehicle seats 40 #ss
     wheels 4 #ws #pe #ch (1000)
family-car passenger-vehicle seats 4 #ss
     wheels 4 #ws #pe #cr (20000)
sports-car passenger-vehicle saloon soft top #sn
     seats 2 #ss wheels 4 #ws
     #pe #sports-car (700)
flatbed-lorry goods-vehicle seats 2 #ss
     wheels 4-12 #ws #ge
     flat load platform #fy (2700)
pantechnicon goods-vehicle seats 2 #ss
     wheels 6 #ws #ge
     large closed compartment #pn (750)
van goods-vehicle seats 2 #ss
     wheels 4 #ws #ge
     small closed compartment #vn (2400)
```

Figure 6: A set of patterns describing different kinds of motorised vehicle. The number at the end of each pattern is a notional frequency of occurrence in an imaginary sample. *Key:* ch = coach, dr = double-decker, fr = family-car, ge = goods-vehicle, fy = flatbed-lorry, mns = mini-bus, pn = pantechnicon, pe = passenger-vehicle, sr = sports-car, sn = saloon, ss = seats, ws = wheels, ve = vehicle, vn = van.

```
0          ee                           cb    fs        0
           |                            |     |
1    ge ve |                       #ve cb cd fs #ge    1
     |  |  |                        |            |
2    |  ve ee ss   #ss ws   #ws wn #ve           |     2
     |        |     |  |      |                  |
3 vn ge       ss 2 #ss ws 4 #ws                 #ge sl 3

0 cld ct     0
   |  |
1  |  |      1
   |  |
2  |  |      2
   |  |
3 cld ct #vn 3

(a)

0          ee                           cb    fs        0
           |                            |     |
1    ge ve |                       #ve cb cd fs #ge    1
     |  |  |                        |            |
2    |  ve ee ss   #ss ws   #ws wn #ve           |     2
     |        |     |  |      |                  |
3 pn ge       ss 2 #ss ws 6 #ws                 #ge le 3

0 cld ct     0
   |  |
1  |  |      1
   |  |
2  |  |      2
   |  |
3 cld ct #pn 3

(b)
```

Figure 7: The two best alignments formed by SP61 with 'engine cab few-windows closed compartment' in New and the patterns shown in Figure 6 in Old. *Key:* cb = cab, cd = carries-load, cld = closed, ct = compartment, ee = engine, fs = few-windows, ge = goods-vehicle, le = large, pn = pantechnicon, ss = seats, sl = small, vn = van, ve vehicle, ws = wheels, wn = windscreen.

load' (from the subclass 'goods-vehicle') down to the lowest-level classes 'van' and 'pantechnicon'.[2]

5 'Deductive' and Abductive Reasoning

Here is an example of *modus ponens* syllogistic reasoning:

1. All swans are white.

2. This creature with ring number A46 is a swan.

3. Therefore, A46 is white.

The first statement is a generalisation about swans which we may represent with a pattern something like this: 'swan id #id white'. The 'variable' represented by the symbols 'id #id' is empty. We can interpret this as meaning that the statement is not about any specific swan but is about swans in general. The second statement is about a specific swan which we may represent with a pattern something like this: 'swan id A46 #id'. In this case, the 'id #id' slot is filled with an identifier for a specific swan.

The first of our two patterns is not unlike the patterns describing different kinds of motorised vehicle shown in Figures 4 and 6. Why not extend the pattern in a similar way so that it gives a fuller description of the class swans, something like this: 'swan id #id long-neck white ... bird #bird #sw'?

Figure 8 shows a small database of such patterns describing different kinds of animal, including the class swans.[3] Figure 9 shows the best alignment formed by SP61 when Old contains the patterns from Figure 8 and New is the pattern 'swan id A46 #id'.

In accordance with the idea that the unmatched symbols in an alignment may be interpreted as inferences from New and Old, this alignment produces the required inference that A46 is white. At the same time, it shows that A46 should have a long neck and all the other attributes of swans, including the fact that it is a bird. The probability assigned to these inferences by SP61 is 1.0 because there is no other alignment which matches so many of the symbols in New.

This example shows how the ICMAUS framework can achieve a kind of 'deduction' which is similar to *modus ponens* syllogistic reasoning. The word 'deduction' has been put in quotes because it is not precisely the same as syllogistic deduction in classical logic. The latter allows only TRUE or FALSE as the result of an inference, whereas 'deduction' in ICMAUS is essentially probabilistic - although it may sometimes deliver extreme values for probabilities, like

[2] Notice that 'levels' in the class hierarchy are not relelated to the positions of rows in the alignments. Apart from the convention that New is always put in the top row, the positions of rows in any ICMAUS alignment are entirely arbitrary.

[3] In a more fully-developed database for classes of animal there would also be patterns describing higher-level classes like 'mammal', 'bird' and 'insect', in much the same style as the patterns in Figure 6.

```
horse id #id four-legs runs-fast ... mammal #mml #hs (75)
giraffe id #id eats-leaves long-neck ... mammal
     #mml #gf (80)
bat id #id wings echo-sounding ... mammal #mml #bt (150)
whale id #id aquatic fish-like ... mammal #mml #whl (50)
swan id #id long-neck white ... bird #brd #sw (100)
robin id #id small red-breast ... bird #brd #rbn (150)
arctic-tern id #id pointed-beak white ... bird
     #brd #at (90)
honey-bee id #id makes-honey communal ... insect
     #isct #hb (10000)
dragon-fly id #id colourful embryo-in-water ... insect
     #isct #df (200)
```

Figure 8: A set of patterns describing different kinds of animal.

```
0 swan id A46 #id                                  0
   |    |       |
1 swan id     #id long-neck white ... bird #brd #sw 1
```

Figure 9: The single best alignment formed with 'swan id A46 #id' in New and the patterns from Figure 8 in Old.

1.0 (as in this example) or 0.0. Also, classical deduction has other properties not shared with 'deduction' in ICMAUS.

5.1 Chains of Reasoning

In addition to modelling 'single-step' reasoning like the example just shown, the ICMAUS framework can model chains of reasoning like those which are prominent in most kinds of reasoning system.

Figure 10 shows an alignment modelling a chain of reasoning that a car mechanic might pursue in trying to work out why there is loss of power in a car engine. In this case, power loss may be due to a leaking head gasket and this may be caused by insufficient torque on the head bolts - caused by a wrongly set torque wrench. Evidence in support of this line of reasoning might come from knowing that a rather inexperienced apprentice had been given the job of replacing the head and that he had indeed used the wrong torque settings.

An attractive feature of the ICMAUS scheme is that, in addition to modelling one-step reasoning and simple chains of reasoning like the examples just described, it has the flexibility to model more complex kinds of reasoning which do not fit neatly into the procrustean format of systems like Prolog and other computerised reasoning systems. Examples are described in [8].

```
0                                                   power_loss 0
                                                         |
1                                 leaking_h_gasket power_loss 1
                                         |
2                h_bolts_not_tight leaking_h_gasket           2
                        |
3 wrong_torque h_bolts_not_tight                              3
```

Figure 10: A chain of reasoning as an ICMAUS alignment.

5.2 Abductive Reasoning

A further example of the flexibility of the ICMAUS framework is that, without any special adaptation or modification, it can deliver the kind of 'backwards' reasoning that we know as 'abduction'. Figure 11 shows the two best alignments formed by SP61 with 'id A46 #id white' in New and the patterns from Figure 8 in Old. In this example, A46 might be a swan (with the characteristics of swans) or, from the information given in New and Old, it might be an arctic tern. SP61 calculates the relative probabilities in this example as 0.552 and 0.448, respectively.

```
0      id A46 #id           white                  0
       |       |             |
1 swan id     #id long-neck white ... bird #brd #sw 1

(a)

0             id A46 #id                white                   0
              |       |                  |
1 arctic-tern id     #id pointed-beak white ... bird #brd #at 1

(b)
```

Figure 11: The two best alignments formed with 'id A46 #id white' in New and the patterns from Figure 8 in Old.

6 Reasoning with Default Values and Nonmonotonic Reasoning

In Figure 8, the class 'swan' was described as 'white' without any possibility that swans could be any other colour. But we know that there is a kind of swan, originally from Australia, which is black! Thus if we make the inference that A46 is white on the strength of the information that A46 is a swan, it may turn out that we are wrong. How can this kind of 'nonmonotonic' or 'defeasible' reasoning be modelled in the ICMAUS scheme?

Let us assume that most swans are indeed white and let us make the simplifying assumption, contrary to fact, that *all* Australian swans are black. We can represent this situation with the patterns shown in Figure 8 but with the 'swan' pattern replaced by the first pattern shown in Figure 12 and with addition of the other two patterns shown in that figure.

The first of the patterns shown in Figure 12 describes the attributes of swans but is non-committal about their colour. However, the second pattern records the fact that 'common' swans are white and because its frequency is nearly as great as the frequency of the main pattern for swans, we may conclude that most swans are white. This is, in effect, a 'default value' for the colour of swans. The third pattern records our assumed fact that Australian swans are invariably black.

Now, if we put the pattern 'swan id A46 #id' in New and, in Old, the patterns from Figure 8, modified as shown in Figure 12, SP61 finds three 'best' alignments that encode three out of the four symbols from New. These are shown in Figure 13. In order from top to bottom of the figure, their *absolute* probabilities, as calculated by SP61, are: 5.392e-10, 1.668e-11 and 6.657e-12.

```
...
sw swan id #id long-neck colour #cl ... bird #brd #sw (100)
cs common sw colour white #cl #sw #cs (90)
as Australian sw colour black #cl #sw !#as (10)
...
```

Figure 12: Three patterns which are added to the patterns shown in Figure 8. The original version of the pattern for 'swan' is removed.

How should we interpret these results? The first alignment in Figure 12 suggests that A46 is a 'common' swan and, for that reason, its colour is white. The second alignment in the figure says that A46 is a swan without specifying what kind of swan it is or its colour. The third alignment suggests that A46 is an Australian swan and, for that reason, it is black.

The second alignment is also contained within the other two alignments. In effect, all three alignments confirm what is, in any case, given as a 'fact' in New: that A46 is a swan. As one might expect, the associated probability is 1.0.

What about colour? The first alignment in Figure 13 predicts that A46 is white and the third one predicts that it is black. To obtain relative probabilities we simply take the ratio of their absolute probabilities and normalise. From the figures for absolute probability given above, the relative probability of the first alignment in Figure 13 is 0.988 and the relative probability of the third alignment is 0.012. In other words, from knowing only that A46 is a swan, the alignments predict that it is much more probable that A46 is white than that it is black.

What happens if we introduce the additional information that A46 is Australian? Figure 14 shows the single best alignment formed by SP61 with 'Aus-

```
0              swan id A46 #id                            0
               |    |       |
1           sw swan id     #id long-neck colour       #cl 1
            |                               |           |
2 cs common sw                          colour white #cl 2

0                       0

1 ... bird #brd #sw     1
                |
2              #sw #cs 2

(a)

0    swan id A46 #id                                        0
      |    |       |
1 sw swan id     #id long-neck colour #cl ... bird #brd #sw 1

(b)

0                  swan id A46 #id                        0
                   |    |       |
1               sw swan id     #id long-neck colour       1
                |                               |
2 as Australian sw                          colour black 2

0                           0

1 #cl ... bird #brd #sw     1
   |                |
2 #cl              #sw #as 2

(c)
```

Figure 13: The three best alignments formed by SP61 with 'swan id #id' in New and, in Old, the patterns from Figure 8, modified as shown in Figure 12.

tralian swan id A46 #id' in New and the augmented/modified patterns from Figure 8 in Old. Since there is now only one best alignment, the relative probabilities of inferences that we can draw from the best set of alignments are all 1.0. Suddenly, the relative probability that A46 is black has jumped from a very low value to a value of 1.0. This switch in the inference from 'probably white' to 'certainly black' conforms with the way, in nonmonotonic defeasible reasoning, the addition of new information can change inferences drawn earlier from less information.

```
0    Australian    swan id A46 #id                       0
          |           |   |       |
1         |      sw swan id     #id long-neck colour       1
          |      |                              |
2 as Australian sw                            colour black 2

0                           0

1 #cl ... bird #brd #sw     1
   |                 |
2 #cl               #sw #as 2
```

Figure 14: The best alignment formed by SP61 with 'Australian swan id #id' in New and, in Old, the patterns from Figure 8, modified as shown in Figure 12.

7 Conclusion

This presentation of the ICMAUS framework and how it can model human-like and probabilistic reasoning has necessarily been brief. As noted earlier, more detail and a larger range of examples may be found in [8] and other sources referenced there.

I hope that readers will feel that there is sufficient potential in the system to warrant further study and investigation.

References

[1] Gabbay, D. M., Hogger, C. H. and Robinson, J. A. (Eds.): *Handbook of Logic in Artificial Intelligence and Logic Programming*, Vol. 3, *Nonmonotonic Reasoning and Uncertain Reasoning.* Oxford: Clarendon Press, 1994.

[2] Pearl J.: *Probabilistic Reasoning in Intelligent Systems.* San Francisco: Morgan Kaufmann, 1988.

[3] Solomonoff, R. A.: Formal theory of inductive inference, Parts I and II. *Journal of Information and Control* **7**, 1-22 and 224-254, 1964.

[4] Solomonoff, R. A.: The discovery of Algorithmic Probability. *Journal of Computer and System Science*, **55** (1), 73-88, 1997.

[5] Sankoff, D. and Kruskall, J. B.: *Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparisons*. Reading, MA: Addison-Wesley, 1983.

[6] Storer, J. A.: *Data Compression, Methods and Theory*, Rockville, Maryland: Computer Science Press, 1988.

[7] Wolff, J. G.: Syntax, parsing and production of natural language in a framework of information compression by multiple alignment, unification and search. Forthcoming in the *Journal of Universal Computer Science* **6** (6), 2000. A copy may be obtained from http://www.jucs.org/jucs_6_6. Three articles which are the basis of this article may be obtained from http://www.iicm.edu/wolff/1998d1, d2, d3.

[8] Wolff, J. G.: Probabilistic reasoning as information compression by multiple alignment, unification and search: an introduction and overview. *Journal of Universal Computer Science* **5** (7), 418-462, 1999. A copy may be obtained from http://www.jucs.org/jucs_5_7. The three articles on which this article is based may be obtained from http://www.iicm.edu/wolff/1998a, b, c.

[9] Wolff, J. G.: 'Computing' as information compression by multiple alignment, unification and search. *Journal of Universal Computer Science* **5** (11), 777-815, 1999. A copy may be obtained from http://www.jucs.org/jucs_5_11.

[10] Wolff, J. G.: Learning and reasoning as information compression by multiple alignment, unification and search. In: A. Gammerman (ed.), *Computational Learning and Probabilistic Reasoning*, Chichester: Wiley, 67-83, 1996. An earlier version was presented at Applied Decision Technologies '95, Brunel University, April 1995 (Proceedings of Stream 1, *Computational Learning and Probabilistic Reasoning*, 223-236, 1995).

[11] Wolff, J. G.: A scaleable technique for best-match retrieval of sequential information using metrics-guided search. *Journal of Information Science* **20** (1), 16-28, 1994.

[12] Wolff, J. G.: Learning syntax and meanings through optimization and distributional analysis. In Y. Levy, I. M. Schlesinger and M. D. S. Braine (eds.), *Categories and Processes in Language Acquisition*. Hillsdale, NJ: Lawrence Erlbaum, 179-215, 1988.