Representational issues in modelling genetic map data

Simon Parsons
Department of Electronic Engineering
Queen Mary and Westfield College
Mile End Road
London E1 4NS
S.Parsons@qmw.ac.uk

Abstract

Much of the large amount of data of inconsistent, incomplete and uncertain information produced by the Human Genome Project is information about genetic maps. The increasing use of computers to store and interpret genetic map data has stimulated research into the kinds of representation and automated reasoning that may be applied to typically imperfect genetic information. This paper discusses the representation of a small amount of genetic information using a system of argumentation and shows how this formalism may be used to handle the different defects in the data.

1 INTRODUCTION

Around the world a huge scientific effort is being expended in the Human Genome Project. The rather grandiose aim of this massive collaboration is, broadly speaking, to elicit all the information encoded in human DNA by the end of the century. This is clearly a vast undertaking, and will result in the production of a huge amount of data—estimates are around 3×10^9 bytes for the whole genome, with the necessary experimental results accounting for many times this figure. With such huge volumes of information, the use of databases for storing data, and of computer systems to process the data, is essential [4, 14]. However, the situation is complicated by the fact that much of the data is highly imperfect. Therefore, in handling this kind of data, unless one wants to ignore these imperfections completely [2], it seems sensible to consider using some of the techniques developed to handle imperfect information, especially those used in artificial intelligence since they lend themselves directly to computer-based implementation.

One particular area of interest is the representation of information about genetic maps. Genetic maps are one of the ways of dealing with DNA data, and they are especially useful since they deal with DNA at quite a high level of abstraction. The functional unit of a length of DNA is what is known as a gene, a piece of DNA that contains information about how to build a protein. A genetic map is an ordered list of loci, which are the locations of genes, so that it identifies the active pieces of a length of DNA, and specifies the order in which they lie on the length of DNA. Now, representing an ordered list of loci does not sound too difficult, but this does not take account of the laboratory data upon which the maps are based [12]. These data tend to take the form of binary relations between loci, typically stating that two loci are close to one another, or that one locus precedes another. From this one has to infer the order and relative location of the loci [13, 18], always bearing in mind the fact that the data may be inaccurate and that different laboratories may disagree on the results.

2 THE REPRESENTATIONAL PROBLEM

Considering the bio-informatics literature, it is possible to identify the major representational issues. According to Guidi and Roderick [12], it is essential to be able to represent both ambiguous data, where results from two different laboratories completely contradict each other, and uncertain data, where experimental error makes it clear that certain results cannot be guaranteed to be correct. Now, no system in use by molecular biologists can handle this kind of information, and certainly cannot support the kind of reasoning required with this kind of information, which is to make all the inferences of all the different maps that the data will support, to identify the different pieces of information that support the different maps, to identify the contradictions between the maps that may be derived, and, when clarifying information arrives, to allow the contradictions to be resolved.

Similar requirements are recorded by Harley and Bonner [13], who apply logic programming to handle genetic map data. However, rather than explicitly dealing with the problems of ambiguity and noisy data, Harley and Bonner instead concentrate on the repre-

$$\text{Ax}\,\frac{}{\Delta \vdash_{ACR} (St, l, Sg)}\;(l : St : Sg) \in \Delta \qquad \wedge\text{-I}\,\frac{\Delta \vdash_{ACR} (St, G, Sg) \quad \Delta \vdash_{ACR} (St', G', Sg')}{\Delta \vdash_{ACR} (St \wedge St', G \cup G', \mathsf{comb}(Sg, Sg'))}$$

$$\wedge\text{-E1}\,\frac{\Delta \vdash_{ACR} (St \wedge St', G, Sg)}{\Delta \vdash_{ACR} (St, G, Sg)} \qquad \leftarrow\text{-E}\,\frac{\Delta \vdash_{ACR} (St' \leftarrow St, G, Sg) \quad \Delta \vdash_{ACR} (St, G', Sg')}{\Delta \vdash_{ACR} (St', G \cup G', \mathsf{comb}(Sg, Sg'))}$$

$$\wedge\text{-E2}\,\frac{\Delta \vdash_{ACR} (St \wedge St', G, Sg)}{\Delta \vdash_{ACR} (St', G, Sg)} \qquad \leftarrow\text{-I}\,\frac{\Delta \cup (St, \emptyset, Sg) \vdash_{ACR} (St', G, Sg')}{\Delta \vdash_{ACR} (St' \leftarrow St, G, \mathsf{comb}(Sg, Sg'))}$$

Figure 1: Argumentation Consequence Relation

sentation of heuristic rules for building maps, rules of the form:

IF *two STS probes hit a common YAC*
THEN *the two probes are close together*

which are true in general, but may lead to false conclusions which may later need to be withdrawn. Overall, then, it seems that to build genetic maps we need a system capable of representing and reasoning with uncertain, default, and inconsistent information in such a way that the support for each conclusion is identified explicitly. This kind of reasoning is exactly that provided by argumentation [16], and the way in which argumentation may be used to solve the problem is the subject of this paper. Before we discuss the system in the next section we will introduce a simple example which will be used through out the paper to illustrate the issues at stake.

We are interested in the order of four loci. Experiments from one laboratory tell us that locus a is close to locus b, locus x is close to locus y, and a precedes b in the sense that it is closer to a given end of the DNA than b. As with all experimentally derived information, this data is uncertain to some degree. There is also data to support both x preceding y and y preceding x (such contradictory information is not uncommon). Experiments in another laboratory, which on past experience is less reliable than the first, tell us that a is closer to the top of the chromosome than x, while data from a third laboratory, which is even less reliable, indicates that x precedes b. We can represent all this information using the relations *precedes(X, Y)* and *close(X, Y)*.

In addition we have some very simple rules for inferring new relations between loci, and for building ordered tuples of four loci, which are rough genetic maps. The first rule expresses the transitivity of the *precedes* relation, while the second comes from the fact that known pairs of loci are usually some distance apart so that, for instance, a and b will be some distance from x and y meaning that if a precedes b and a precedes x, b also precedes x. This, then, is a default rule for coping with incomplete information about the order of two pairs of loci. The third rule is that three *precedes* relations referring to four different loci are sufficient to build a simple genetic map. Like the transitivity rule, this is completely certain.

Thus our example data, like real genetic data, includes information that is both uncertain and contradictory, as well as including default rules.

3 A SYSTEM OF ARGUMENTATION

In classical logic, an argument is a sequence of inferences leading to a conclusion. If the argument is correct, then the conclusion is true. Consider the simple deductive database [5] Δ_1 which expresses some of the information in the example as if it were completely certain (the rule is the transitivity rule).

$$\begin{array}{ll} f1 : precedes(a, x). & \Delta_1 \\ f2 : precedes(x, b). & \\ r1 : precedes(X, Z) \leftarrow precedes(X, Y) & \\ \qquad\qquad\qquad\qquad \wedge\, precedes(Y, Z). & \end{array}$$

From this database the argument $\Delta_1 \vdash precedes(a, b)$ is correct because $precedes(a, b)$ follows from Δ_1 given the usual logical axioms and rules of inference. Thus a correct argument simply yields a conclusion which in this case could be paraphrased "$precedes(a, b)$ is true in the context of $f1$,$f2$ and $r1$". In our system of argumentation this traditional form of reasoning is extended to allow arguments to indicate support and doubt in propositions as well as proving them, by assigning labels to arguments which denote the confidence that the arguments warrant in their conclusions. This form of argumentation may be summarised by the following schema:

$$\text{database } \vdash_{ACR} (\text{Sentence}, \text{Grounds}, \text{Sign})$$

where $\vdash_{ACR}$ is a consequence relation for a logic of argumentation, which sanctions inferences made using the rules in Figure 1 (which are adapted from those in [10] to fit our deductive database context), along with the identity $\neg St \equiv \perp \leftarrow St$ ($\perp$ is logical contradiction). For a detailed exposition of these rules see [16]—there is no room here to go into the detailed justification for them.

Informally, Grounds (G) are the facts and rules used to infer Sentence (St), and Sign (Sg) is a number or a symbol drawn from a dictionary of possible numbers or symbols which indicate the confidence warranted in the conclusion. The use of a number of different dictionaries of signs is one of the marks of generality of argumentation since most formalisms for handling imperfect data are restricted to a single dictionary. The rules in Figure 1 are independent of the dictionary used—different dictionaries imply different *combination* functions comb for combining signs during the construction of arguments. Typically we will have a number of different arguments for a given sentence, and so we *flatten* these to give a single measure which may or may not be expressed using the same dictionary.

What this system of argumentation gives us is a general framework for expressing logical facts which can incorporate different models of uncertainty by varying the signs and their associated combination and flattening functions. As we shall see in the next section, the scheme also offers a means of representing default information and of handling inconsistent information.

4 USING ARGUMENTATION TO REPRESENT IMPERFECT DATA

In this section we work through the representation of the example introduced above, showing how argumentation can be used to express the various types of imperfection that are present in the genetic information.

4.1 HANDLING UNCERTAINTY

The most straightforward problem from a representational point of view is that of handling the uncertainty associated with the information because it is experimental data of varying quality. This may simply be handled by giving each clause in the database a numerical sign between 0 and 1, inclusive, which indicates the degree of certainty that one has in the fact represented by the clause. Thus, provided that we believe that even the third, and least trustworthy, of the laboratories, produces reasonably accurate results, a suitable distribution of values would be those in Δ_2.

$$\begin{array}{ll}
f1 : close(a,b) : (0.9). & \Delta_2 \\
f2 : close(x,y) : (0.9). & \\
f3 : precedes(a,b) : (0.9). & \\
f4 : precedes(x,y) : (0.9). & \\
f5 : precedes(y,x) : (0.9). & \\
f6 : precedes(a,x) : (0.8). & \\
f7 : precedes(x,b) : (0.7). & \\
r1 : precedes(X,Z) \leftarrow precedes(X,Y) & \\
\qquad\qquad \wedge\, precedes(Y,Z) : (1). &
\end{array}$$

What may be derived depends upon the interpretation of the numbers, since this affects the choice of combination and flattening function. For instance, if we want to make use of a probabilistic method to quantify uncertainty we have several different ways in which we can compute the probability of a proposition from the probabilities of the steps in the argument. We may use a local scheme such as that proposed by Ng and Subrahmanian [19] which requires the use of an interval probability because of the possible dependencies between the steps. Alternatively, Krause *et al.* [16] provide a global approach which takes the structure of the arguments into account. In this approach the probability of an argument for a proposition is the product of the probabilities of all the steps in the argument, while flattening arguments consists of summing the probabilities of the different arguments and then subtracting the product of the probabilities of the steps in all the arguments.

Alternatively, following Dubois *et al.* [6] we may also use possibility theory. Here the signs in a database are the lower bounds on the necessity measures of the propositions, the combination operation is minimum, and the flattening operation is maximum. This, of course, effectively ignores the interdependence of the steps in the arguments, but has significant advantages in efficiency of computation. If we take the signs in Δ_2 to be lower bounds on necessity measures we can derive, for instance:

$$\Delta_2 \vdash_{ACR} (precedes(a,y), (f4, f6, r1), (0.8)).$$

Since there is only one argument that relates to $precedes(a,y)$, there is no need to invoke the flattening function.

4.2 HANDLING DEFAULT RULES

Using signs that are lower bounded necessity measures also gives us a simple way of handling default information. As several people have pointed out, for example [1, 9], one can assign a necessity measure of 1 to all facts and rules that are known to be true, and then attach a lesser measure to default rules. Because of the minimum operator used to combine signs when rules are applied, facts that are deduced from true rules will always have a larger necessity measure than those deduced from default rules so that when comparing two hypotheses, the one deduced from true information will be preferred to that deduced from default information. Using this method, we could obtain database Δ_3 where r2 is the default rule discussed in the example, r1 is as above, and f1–f3 are some new facts about other loci that are known to be true:

$$\begin{array}{ll}
f1 : close(d,c) : (1). & \Delta_3 \\
f2 : precedes(d,e) : (1). & \\
f3 : precedes(f,c) : (1). & \\
r1 : precedes(X,Z) \leftarrow precedes(X,Y) & \\
\qquad\qquad \wedge\, precedes(Y,Z) : (1). & \\
r2 : precedes(Y,Z) \leftarrow close(X,Y) & \\
\qquad\qquad \wedge\, precedes(X,Z) : (0.85). &
\end{array}$$

This allows us to obtain results such as:

$$\Delta_3 \vdash_{ACR} (precedes(c,e), (f2, f1, r2), (0.85)).$$

which is a reasonable conclusion from the data that we have. However, were we to learn that e certainly precedes f, so that the fact "$f4 : precedes(e, f) : (1)$" was added to the database, we would be able to infer:

$$\Delta_3 \vdash_{ACR} (precedes(e, c), (f3, f4, r1), (1)).$$

thus overturning the original conclusion and demonstrating that the database and the default rule behave in the desired non-monotonic fashion, with certain information outweighing default information[1]. The system can be further complicated by allowing for different necessity measures to be assigned to different default rules, allowing their differing certainties to be expressed.

There is, however, a problem when using the approach suggested by Benferhat [1] and Froidevaux and Mengin [9] in combination with that outlined above for handling uncertain information. Once the facts in the database start to be quantified with necessity measures other than 1, the comparative order expressed by the default rules and the uncertain information becomes important. Thus if we enter an uncertain fact into Δ_3 with a measure greater than 0.85, as the data from the first laboratory was in Δ_2, it means we consider this to be more certain than inferences drawn using the default rule, showing that a good deal of care must be taken when assigning necessity measures. To provide a general solution to the problem of ensuring the right interaction between defaults and uncertain information it can be argued that default rules, by definition, should only be outweighed by information that is known to be true, so they should be quantified by a value that is only outweighed by a necessity measure of 1. This can be achieved by the use of the symbolic value *dt*, inspired by Ginsberg's [11] "true by default" value, which is infinitesimally close to 1. Thus:

$$\min(value, dt) = \begin{cases} dt \text{ if } value = 1 \\ value \text{ otherwise} \end{cases}$$

$$\max(value, dt) = \begin{cases} 1 \text{ if } value = 1 \\ dt \text{ otherwise} \end{cases}$$

This approach could clearly be extended to cover preferences over sets of default rules by using an ordered set of symbolic values [8, 11]. Using *dt* we can extend Δ_3 to be Δ_4 as below, from which we can infer:

$$\Delta_4 \vdash_{ACR} (precedes(b, x), (f1, f6, r2), (0.8)).$$
$$\Delta_4 \vdash_{ACR} (precedes(y, b), (f2, f7, r2), (0.7)).$$

$$\begin{array}{ll} f1 : close(a, b) : (0.9). & \Delta_4 \\ f2 : close(x, y) : (0.9). & \\ f3 : precedes(a, b) : (0.9). & \\ f4 : precedes(x, y) : (0.9). & \\ f5 : precedes(y, x) : (0.9). & \\ f6 : precedes(a, x) : (0.8). & \\ f7 : precedes(x, b) : (0.7). & \\ r1 : precedes(X, Z) \leftarrow precedes(X, Y) & \\ \qquad \wedge\, precedes(Y, Z) : (1). & \\ r2 : precedes(Y, Z) \leftarrow close(X, Y) & \\ \qquad \wedge\, precedes(X, Z) : (dt). & \end{array}$$

indicating that *before(b, x)* is slightly more certain to be true than *before(y, b)*. Now, if we add some certain information such as *precedes(a, x)* (f8) and *close(a, b)* (f9) to Δ_4 we can establish:

$$\Delta_4 \vdash_{ACR} (precedes(b, x), (f8, f9, r2), (dt)).$$

Flattening the two arguments for $precedes(b, x)$ we can establish that it is true by default, illustrating the interaction between certain, uncertain and default information.

4.3 HANDLING INCONSISTENCY

Database Δ_3 demonstrates one way of handling inconsistent information by viewing the argument for one of a pair of inconsistent conclusions as more credible than the other[2]. A more sophisticated approach is to use the intuitive idea that arguments may be "defeated" by other arguments, examining the structure of the arguments to find conflicting alternatives, and to use the conflicts to determine which conclusions are the most acceptable.

The examination of argument structure is based upon the notions of *rebuttal* and *undercutting*. Argument Arg_1 for proposition p rebuts argument Arg_2, which supports proposition q, if p directly contradicts q (in other words p is logically equivalent to $\neg q$). Similarly, Arg_1 undercuts Arg_2 if p directly contradicts r which is one of the steps in Arg_2. We also distinguish consistent arguments, which draw facts from consistent subsets of the whole database, and tautological arguments which are based on the axioms of the logic that underlies the system of argumentation rather than the information in the database. Having made these distinctions we can identify the following classes of arguments for a database Δ, which are listed in increasing order of acceptability [7]:

A1 The class of all arguments that may be made from Δ.

A2 The class of all logically consistent arguments that may be made from Δ (so that $\perp$ cannot be derived from the steps in the argument).

[1] It should be noted that the non-monotonicity is due to the flattening function. When new information is added, the same arguments as before can always be made, however, when these arguments are flattened, the overall conclusion may change. In this case, there is still an argument for the previous conclusion $precedes(c, e)$, it is just weaker then the argument for $precedes(e, c)$. Thus argumentation is monotonic in arguments but non-monotonic in conclusions.

[2] In an informal semantic sense $precedes(c, e)$ and $precedes(e, c)$ are clearly inconsistent. To ensure formal syntactic inconsistency we have to add the rule $\neg precedes(Y, X) \leftarrow precedes(X, Y)$.

A3 The class of all arguments that may be made from Δ for propositions for which there are no rebutting arguments.

A4 The class of all arguments that may be made from Δ for propositions for which there are no undercutting arguments.

A5 The class of all tautological arguments that may be made from Δ.

Since each class of arguments includes all classes of arguments which are preferred to it, this idea of acceptability provides a simple way to determine an order between arguments. Each argument is ranked on the basis of the smallest acceptability class into which it falls, and it is more acceptable than every argument in any bigger class and as acceptable as any argument in the same class. Propositions may then be ranked using the order over the arguments that support them, invoked by the acceptability classes that the arguments fall into. To see how we may use acceptability classes in our example, consider adding the final rule to our database of genetic data:

$$\begin{aligned}
&f1 : close(a,b) : (0.9). \qquad\qquad \Delta_5\\
&f2 : close(x,y) : (0.9).\\
&f3 : precedes(a,b) : (0.9).\\
&f4 : precedes(x,y) : (0.9).\\
&f5 : precedes(y,x) : (0.9).\\
&f6 : precedes(a,x) : (0.8).\\
&f7 : precedes(x,b) : (0.7).\\
&r1 : precedes(X,Z) \leftarrow precedes(X,Y)\\
&\qquad\qquad\qquad \wedge\, precedes(Y,Z) : (1).\\
&r2 : precedes(Y,Z) \leftarrow close(X,Y)\\
&\qquad\qquad\qquad \wedge\, precedes(X,Z) : (dt).\\
&r3 : map(W,X,Y,Z) \leftarrow precedes(W,X)\\
&\qquad \wedge\, precedes(X,Y) \wedge precedes(Y,Z) : (1).\\
&r4 : \neg precedes(Y,X) \leftarrow precedes(X,Y) : (1).
\end{aligned}$$

From this database, we can establish:

$$\begin{aligned}
\Delta_5 \quad &\vdash_{ACR} \quad (map(a,x,y,b),(f2,f4,f6,f7,r2,r3),(0.7)).\\
\Delta_5 \quad &\vdash_{ACR} \quad (map(a,y,b,x),(f1,f2,f3,f4,f6,\\
&\qquad\qquad f7,r1,r2,r3),(0.7)).
\end{aligned}$$

Now, these solutions clearly conflict[3], but on the basis of their degrees of certainty no choice can be made between them. However, if we look at the acceptability classes into which the arguments for them fall, it is a different story. The argument for *map(a,x,y,b)* falls into A2 since it may be rebutted by the argument for *map(a,y,b,x)*, while the argument for *map(a,y,b,x)* is in A1 since it is inconsistent by virtue of containing both *precedes(x, b)* (f7), and *precedes(b, x)* (inferred from f1, f6 and r2). On this basis, the argument for *map(a,x,y,b)* is more acceptable than that for *map(a,y,b,x)*, and so *map(a,x,y,b)* is preferred to *map(a,y,b,x)*.

It is also worth noting two additional points. Firstly, from the same database, we can infer:

$$\Delta_5 \vdash_{ACR} (map(a,b,x,y),(f1,f3,f4,f6),(0.8)).$$

This argument is in A2, and has a higher necessity measure than either of the other arguments. The question is, given that we now have two methods of ranking solutions, one on the basis of acceptability of arguments and the other on the basis of the necessity measure, how do we combine their effects? Two obvious methods spring to mind: to rank the solutions on the basis of their acceptability and then use the necessity measures to order the solutions within an acceptability class, or to use acceptability to order the solutions with the same necessity measure. In this case whichever method we use we get $map(a,b,x,y) > map(a,x,y,b) > map(a,y,b,x)$ where $>$ indicates the overall order, but the order of application may be important in other cases, for instance, when the solution with the highest necessity measure is in the lowest acceptability class, or vice versa. The second point is illustrated by the fact that we can infer:

$$\Delta_5 \vdash_{ACR} (map(a,y,x,b),(f1,f4,f5,f6,f7,r1,r3),(0.7)).$$

This solution is supported by an argument in A1, and has the same ranking as *map(a,y,b,x)* no matter which method is used, and so there is no means of identifying which is to be preferred. Thus there is some information which this scheme distinguishes as being truly ambiguous.

5 SUMMARY

This paper has addressed some of the problems of modelling information about genetic maps. In particular it has discussed the use of a system of argumentation to represent some typical genetic data which illustrate the kinds of imperfections with which such data are pervaded. The uncertainty of data introduced by the fact that they are the results of experiments which are usually less than completely reliable was handled by numerically quantifying the data. The use of heuristic default rules which, while they are not always reliable, make it possible to make useful deductions on many occasions, were handled by providing a symbolic quantification for default rules that interacts in a suitable way with certain and uncertain information. Finally, the fact that genetic data is inconsistent was handled by making use of the fact that the system of argumentation that has been adopted is capable of drawing meaningful conclusions from contradictory information, and a means by which this capability may be integrated with the handling of default and uncertain information was proposed. This, then meets several of the requirements for handling genetic data laid down in the literature [12, 13], as well as going further in

[3] Again this is an informal semantic conflict. To get a formal syntactic conflict we would have to include rules such as $\neg map(X,W,Y,Z) \leftarrow map(W,X,Y,Z)$ to cover all possible variations. These are omitted for the sake of clarity.

integrating different types of information than anyone whose work the author aware of.

It is clear that the treatment presented here is rather tentative and informal. This is naturally the case given the ongoing nature of the work, and is a problem that hopefully will be remedied in the near future. In the short term it is also intended to implement the kind of reasoning that described in this paper. Such an implementation will involve the extension of the Argumentation Theorem Prover (ATP) [15], a system which differs from standard theorem provers in that it allows the automated construction of the argument supporting a proposition as well as deducing the proposition. The necessary changes to handle the work described here are not major—it is only necessary to slightly alter some of the combination and flattening functions which can already handle several different sets of quantifiers. Whilst the efficiency of this implementation will clearly be an issue in the long run, the success of the logic programming approach advocated by Harley and Bonner [13] and the rule-based approach of Letovsky and Berlyn [17] suggests that these problems will not be insurmountable.

Acknowledgements

This work was partially supported by ESPRIT Basic Research Action 6156 DRUMS II (Defeasible Reasoning and Uncertainty Management Systems).

References

[1] Benferhat S. (1994) Handling hard rules and default rules in possibilistic logic, *Proceedings of the International Conference on Information Processing and the Management of Uncertainty*, Paris.

[2] Clark, D. A., Rawlings, C. J., and Doursenot, S. (1994) Genetic map construction with constraints, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, Stanford.

[3] Cui, Z. (1994) Using interval logic for order assembly, *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology*, Stanford.

[4] Cui, Z., Fox, J., and Hearne, C. (1993) Knowledge based systems for molecular biology: the role of advanced technology and formal specification, *Proceedings of the IJCAI Workshop on AI and the Genome*, Chambéry, France.

[5] Das, S. K. (1992) *Deductive databases and logic programming*, Addison-Wesley, Wokingham.

[6] Dubois, D., Lang, J., and Prade, H. (1991) Towards possibilistic logic programming, *Proceedings of the International Conference on Logic Programming*, Paris.

[7] Elvang-Gøransson, M., Krause, P., and Fox, J. (1993) Dialectic reasoning with inconsistent information, *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*, Washington D. C.

[8] Froidevaux, C. and Grossetête, C. (1990) Graded default theories for uncertainty, *Proceedings of the 9th European Conference on Artificial Intelligence*, Stockholm.

[9] Froidevaux, C. and Mengin, J. (1990) A theorem prover for free graded default theories, Technical Report LRI, Université Paris Sud, Orsay.

[10] Fox, J., Parsons, S., Krause, P., and Elvang-Gøransson, M. (1993) A generic framework for uncertain reasoning, in *Qualitative Reasoning and Decision Technologies*, N. Piera Carrete and M. G. Singh eds., CIMNE Press, Barcelona, pp 461–470.

[11] Ginsberg, M. L. (1988) Multivalued logics: a uniform approach to inference in artificial intelligence, *Computational Intelligence*, **4**, 265–316.

[12] Guidi, J. N. and Roderick, T. H. (1993) Inference of order in genetic systems, *Proceedings of the First Conference on Intelligent Systems for Molecular Biology*, Bethesda, MD.

[13] Harley, E. and Bonner, A. J. (1994) A flexible approach to genome map assembly, *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology*, Stanford.

[14] Hearne, C., Cui, Z., Parsons, S., and Hajnal, S. (1994) Prototyping a genetics deductive database, *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology*, Stanford.

[15] Krause, P., Ambler, S., and Fox, J. (1992) ATP User Manual, Technical Report 187, Advanced Computation Laboratory, Imperial Cancer Research Fund.

[16] Krause, P., Ambler, S., Elvang-Gøransson, M., and Fox, J. (1995) A logic of argumentation for reasoning under uncertainty, *Computational Intelligence*, **11**, 113–131.

[17] Letovsky, S. and Berlyn, M. B. (1992) CPROP: a rule-based program for constructing genetic maps, *Genomics*, **12**, 435–446.

[18] Mott, R., Grigoriev, A., Maier, E., Hoheisel, J., and Lehrach, H. (1993) Algorithm and software tools for ordering clone libraries: application to the mapping of the genome of *Schizosacchromyces pombe*, *Nucleic Acids Research*, **21**, 1965–1974.

[19] Ng, R. T. and Subrahmanian, V. S. (1992) Probabilistic logic programming, *Information and Computation*, **101**, 150–201.