Prototyping a Genetics Deductive Database

Catherine Hearne, Zhan Cui, Simon Parsons and Saki Hajnal
Advanced Computation Laboratory,
Imperial Cancer Research Fund,
Lincoln's Inn Fields,
P.O. Box 123, London WC2A 3PX,
United Kingdom
ch,cui,sp,sjh@acl.lif.icnet.uk

Abstract

We are developing a laboratory notebook system known as the Genetics Deductive Database. Currently our prototype provides storage for biological facts and rules with flexible access via an interactive graphical display. We have introduced a formal basis for the representation and reasoning necessary to order genome map data and handle the uncertainty inherent in biological data. We aim to support laboratory activities by introducing an experiment planner into our prototype. The Genetics Deductive Database is built using new database technology which provides an object-oriented conceptual model, a declarative rule language, and a procedural update language. This combination of features allows the implementation of consistency maintenance, automated reasoning, and data verification.

Introduction

It is now widely perceived that there is a requirement for an 'intelligent laboratory notebook' in molecular biology and related laboratory sciences (Cui et al. 1993). The high rates of data production demand both efficient local storage and rapid and flexible access, while there is an growing need for so-called 'intelligent behaviour' such as automated reasoning and data verification and analysis. Such a database would provide storage and flexible access to the extensive and complex experimental data characteristic of genome research. Advanced database technology would also aid the lab worker in planning, recording and interpreting tasks and provide access to supporting analysis software. We are developing the Genetics Deductive Database (GDD) as a 'notebook' system to satisfy these requirements. Our prototypes in the course of this development are based on new, advanced database technology which provides the necessary combination of automation and deduction.

This paper reports the current state of the development, describing two prototype systems GDD1 and GDD2. These local laboratory knowledge base systems provide data storage, flexible retrieval, consistency maintenance, active behaviour and an interactive graphical display for browsing and querying. They can be seen as complementary to other systems for bioinformatics. By contrast with established public domain systems, providing interpreted, consensus public information using conventional databases (Pearson 1991) (Rawlings et al. 1991), we are using new technology to support local data and data interpretation with a view to lab management.

The underlying technology for GDD is being developed in the European Community funded IDEA project (ESPRIT Project 6333), which intends to integrate traditional database management features such as efficient and high-level data access, data sharing, reliability and security, with newer deductive, object-oriented and active database techniques. It is anticipated that at a later stage the IDEA technology will provide database interoperability, allowing interaction between GDD and conventional public domain databases and data analysis packages. We are using these technologies as they are made available, as well as the logic programming language Prolog and the production rule language Sceptic (Hajnal et al. 1989).

Development Requirements

There are a number of technical requirements which are necessary for a database system to support an 'intelligent laboratory notebook'. We can comment on these in terms of the IDEA technology we are using and how it meets these demands.

Database capacity is crucial since the database must be capable of coping with the massive volume of raw data produced in labs. These comprise results from sequencing, genotyping, hybridisation fingerprinting or other mapping techniques to inform on the various combinations of 3×10^9 elements in the human genome. The data from a lab require persistent central storage in a local laboratory database, providing the user with efficient access and automatic interpretation. The parallel implementation envisaged for the IDEA system will clearly be of benefit when dealing with such large quantities of data.

Knowledge base complexity makes demands on technology. Current relational databases have been

designed from the requirements of simple applications featuring simple data types and data structures. Biology is an area of complex knowledge, raising design requirements for the knowledge base and for knowledge representation in the data model. The powerful rule language supported by IDEA makes it possible to express this complex knowledge.

Formal specification is one approach to good design in building massive knowledge base support systems. For this it is necessary to design at the level of tasks, domain datasets and inference methods, as distinct from the lower level design of rules and facts. For part of our prototyping work we make use of our Specification Language for Object Theories (SLOT) in partnership with IDEA technology. This provides a design perspective and combines formal specification for an object-oriented system with deductive and active methods (Cui et al. 1993).

Object-oriented conceptual modelling provides a natural means of representing the complex, often composite structures in molecular biology. The data model groups objects into classes and subclasses which share properties by inheritance, a mechanism that permits a superclass to confer its attributes and behaviours on its subclass. According to a recent paper (Kochut et al. 1993), there are very few well-developed examples of the use of object-oriented database systems in molecular genetics. This situation can only be assisted by the development of robust technology, and our use of IDEA testbed systems suggests that, when mature, this should offer a unique and suitable platform for the development of such systems.

Deductive rule languages provide inference in the database and flexible information retrieval. These have already been used in molecular biology, in the form of the logic programming language Prolog (Kazic et al. 1990) (Yoshida et al. 1990) and in protein structure prediction (Rawlings et al. 1985). Prolog has also been successfully combined with object-oriented database design (Gray et al. 1990). IDEA technology provides a deductive database language less general than Prolog but which has the advantage of being integrated with object-oriented and active functions and being optimized for efficient database access. This is used to derive values of object attributes and to implement constraints defining restrictions on objects, classes, attribute values and legal states of the database.

Active behaviour is vital for an advanced data handling system to respond to external events impinging on the database, such as the addition of data, user queries and internal database events such as the violation of constraints. Such active behaviour is achieved through event-oriented functions using data-directed production rules or "triggers", and these are being widely incorporated into relational, object-oriented and deductive database products. These are used to define specific reactions to particular events in the database (Bayer 1993). IDEA technology makes triggers available to the database designer, in the form of event, condition and action specifications. The full expression of triggers in the IDEA technology will allow the GDD2 database to be programmed to respond automatically to events and prevailing conditions as desired.

Database interoperability is required by GDD2. This is the interaction of IDEA technology with conventional databases, allowing GDD2 users access to existing public data. Access to data stored conventionally means that new lab notebook systems do not need to duplicate the public domain resources. We hope that IDEA technology will provide this necessary access through the approach of federated databases. It is similarly undesirable to reimplement standard data analysis software and it is preferable to provide interaction from advanced technologies.

Specialised reasoning methods are not supplied by IDEA but are required in our design to retrieve, derive and manipulate genetic and other data stored in the database. More will be said about these in later sections. Here we can summarise the basic requirements. Temporal reasoning is a necessary formalism for the development of the GDD2 system which offers representation and reasoning with genome maps. Genome maps can considered as linear orders of one-dimensional intervals and distances between intervals. These can therefore be represented by temporal logic, especially interval logic (Guidi and Roderick 1993) (Cui, 1994). For GDD2 we are using an extended interval logic which supports the necessary partial orders, local orientation and distance. We are working on a special inference mechanism to exploit the spatial structure present in these maps and to provide new inference rules like those described in CPROP (Letovsky and Berlyn 1992). In addition we need a formal treatment for reasoning under the uncertainty of many of these data, which can be incorporated into the map representation language of GDD2. We need not only a way to represent the associated imperfections of biological data, but also a method to reason with uncertain or ambiguous data. The IDEA technology does not support this activity and in order to do this we are applying argumentation (Fox et al. 1993).

Genetics Deductive Databases

To date we have developed two prototype versions of the Genetics Deductive Database, GDD1 and GDD2. GDD1, as the first version, reflected the limitations of the early version of the IDEA technology with which it was built. GDD2 is the planned refinement of this utilising a later more robust and advanced testbed from the database technology development.

GDD1

GDD1 represents our initial use of IDEA technology to create a laboratory notebook system with active and

deductive functions and a persistent knowledge base (KB) for genetics laboratory and public domain data.

GDD1 Knowledge Base. The scientific contents of the GDD1 knowledge base are laboratory observations, interpretations and theories. The scientific theories required to explain observations are general scientific knowledge and account for and give structure and context to lab observations. In particular the GDD1 KB consists of genetic theories expressed as deductive rules, facts such as sequence data, laboratory mapping data and instances of associated concepts, experiments and their components and interpretations of data, for example, as genetic maps.

GDD1 User Interface. The GDD1 graphical user interface provides an interactive display of the knowledge base schema. The class browser provides viewing, querying and editing of instances of classes in the GDD1 knowledge base, mediated by a graphical user interface built in XPCE (a user interface builder included in the ECL^{i}PS^e logic programming system). All display windows have window scroll bars and respond to the mouse, providing pop-up forms for completion and window specific or data specific menus. Interactive graphical objects respond to mouse movements by providing text, graphics, or with pop-up boxes or menus. The class browser provides the menu contents for each graphical object based on the relationships of objects and their attributes in the data model. Figure 1 illustrates the user interactive session with the graphical display. The queries for a display of human chromosome 2 and details of regional map data have been answered using public and private data and the results displayed as interactive diagrams.

GDD1 Functionality. The GDD1 interface allows the user to query and edit data, by graphical interaction or text entry, but not the database schema. Queries can be issued concerning data from public and private sources stored in GDD1 including chromosomes, maps, markers and experimental data such as genome analyses. New data can be added, interpreted and displayed. Declarative integrity constraints are used to express biological and physical laws concerning the data. These provide checks on the consistency of the knowledge base contents and assist when new data are added and when inference from data is propagated within the database. For example new genotype data is checked for Mendelian inheritance and sequence data is checked for the correct base composition. In GDD1 data are stored temporarily before they are subjected to checking and possible persistent storage. IDEA technology provides a deductive database, and GDD1 uses passive deductive rules to propagate inference from data through the database. This provides a degree of automation in the system, allowing data interpretation and update propagation. The declarative constraints also serve to detect inconsistencies in automated processes.

Automation is provided as active behaviour in the form of production rules or triggers. These active rules may also effect the propagation of inference, but require an event, such as storage of data, to fire an active rule. If a given set of conditions are true when the triggering event occurs then the active rule is said to be fired and a course of database action is initiated, for example to call an application program for data analysis. Figure 1 is a snapshot of the GDD1 system taken the moment after an active rule is triggered. Here the user is adding new data on family genotyping which the system is checking data for consistency with Mendelian Laws, using declarative constraints, before these are stored permanently. This prototype system has been designed such that the conditions for the rule to fire are that sufficient new and consistent data have been added. The triggered actions include a message to the user and the simulation of a call to an external data analysis program.

The return of results from such a data analysis constitutes another triggering event. In this case the system interprets the results automatically as a new genetic map which may be stored. In the figure a dialogue box allows the user to name this map in the database. The addition of the new data may cause further triggers to fire. The final effects of this kind of active propagation in the knowledge base cannot be forecast—it constitutes part of the intelligent activity of the system we are building. It is easy to envisage cycles of triggering and propagation which do not require an external connection and which may influence the user in a new cycle of investigation and experiments.

GDD2

The GDD2 system is a development from the GDD1 prototype and features improved technology and increased functionality. GDD2 will allow the user to record details of lab activities, that is experiments or procedures, as these are performed, and record their results as they are observed. The GDD2 system will provide representation and reasoning with raw experimental data to provide automatic analysis and a biologically useful interpretation in the database.

GDD2 Knowledge Base. The GDD2 knowledge base consists of data specific to the domain, data concerning tasks such as experiments, analyses or access to other database systems, and reasoning capabilities using these data. The GDD2 knowledge base should support a wider and more detailed representation of domain knowledge and lab practice than GDD1. In particular, the GDD2 knowledge base provides support for a range of genome map representations including cytogenetic, contig, genetic, sequence and restriction maps. GDD2 also provides inference from these data to enable reasoning in map construction, to integrate data from diverse sources and to give a systematic means of

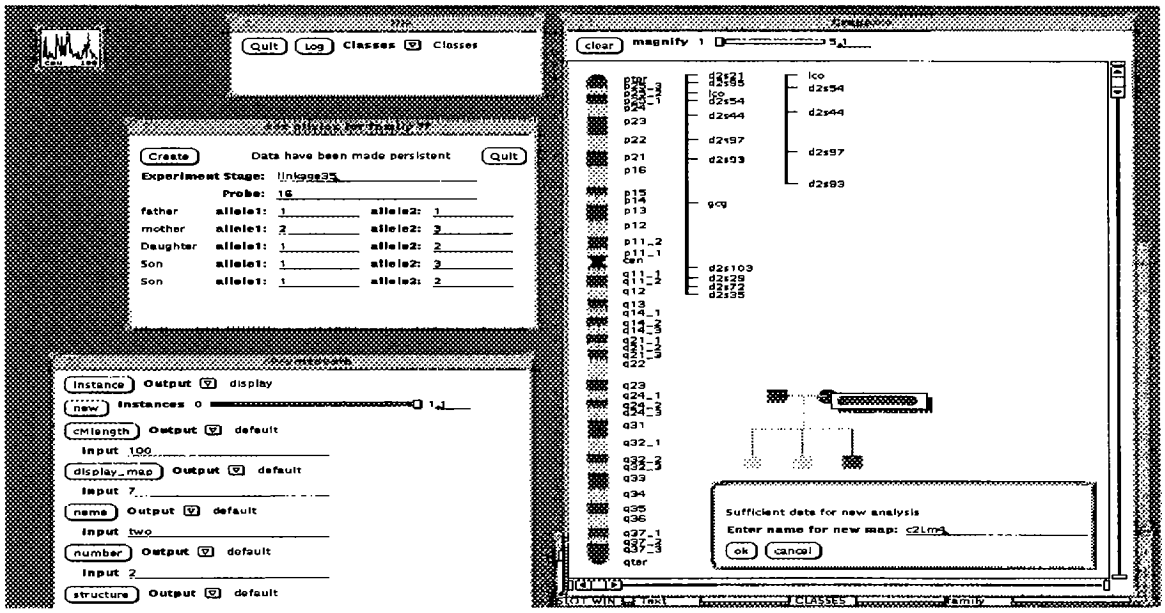

Figure 1: Snapshot of the GDD1 User Interface

handling imperfect data. The GDD2 model of laboratory experiments is being designed to provide scheduling for experimental processes and the management of lab resources such as reagents and equipment.

GDD2 Functionality. In the GDD2 system the user can record daily activity, rather as in a paper notebook. The user will be able to query the GDD2 database for details of the methods and materials of laboratory procedures and records of completed procedures and their outcome as observations and end products. Active rules in GDD2 enable the system to assist in preparation and planning experiments and managing lab records by triggering automatic responses such as checking reagent stocks and booking equipment. In creating records for example, the system will create a new instance of an experiment by automatically querying the knowledge base for details of that experiment class and adding information such as reagents, input material, creation date and time to the new instance. In addition, a query to the system can remind the user of a particular method, unusual reagent or safety hazard and anticipate the addition of new data. The user interacts with the system during the experiment to enter data as observations are made, reagents are taken from stock, equipment used, freed or occasionally broken. If an experiment is abandoned due to early failure the system must be informed and must roll back data entry, plans and preparations already made.

Deductive rules in the system provide consistency checking to ensure that data match the expected category for a given experiment, have the correct syntax, and are consistent with the KB content. New data must not introduce conflicting information although it may be desirable to maintain conflicting data while awaiting further experimental evidence. Also some allowance is required for the inherent ambiguity in experimental observations.

GDD2 User Interface. The GDD2 user interface remains similar to that of GDD1, with the possibility of reimplementing this in Tcl/Tk rather than XPCE.

Database Interoperability Database interoperability is highly desirable for a laboratory notebook system and the IDEA project is addressing the general question. We are hopeful that GDD2 will provide this facility using future IDEA technology releases. This would permit an extended definition of the functionality of the laboratory notebook. The external sources of data and applications that are required include conventional databases and analysis programs in routine use in the public domain. The user may require the knowledge base to query for data held at remote databases and request services external to the application to complement and assist in local interpretation. An example of this need was seen in the GDD1 implementation, where sample data was obtained in part from the Genome Database (GDB, which is a SyBase system). In a real lab situation, dynamic access to this data would require an SQL interface with the GDB server.

Knowledge Representation

The data model for GDD2 is both an expansion and a refinement of the data model employed by GDD1, and raises some interesting issues in knowledge representation.

Representing Experiments

Laboratory observations are given their framework by the daily activities of the experimenter who records these data alongside the detailed description of the experiment in a paper notebook. We are building an electronic version of this in the GDD system. As an active extension, GDD2 may provide support for the scheduling of experiments, as has been achieved in medical decision support.

Currently we are using an object-oriented model to describe tasks of varying complexity, generally consisting of composite activities, as has been used by Baclawski et al. (1993).Each stage of a task is an activity with associated conditions before and after, a set of input materials, duration and conditions, and results for each stage in the form of material output and observations. For the purposes of the class hierarchy in GDD2 we have included processes as a class of experimental_objects. These processes may be performed by lab personnel and frequently consist of chemical or biological reactions, each involving a series of steps by which substances are transformed from initial states to final states. Within the biological domain we can also distinguish natural transformations such as growth and cell division and disturbances to these, such as cancer, which is unregulated cell growth. These are useful concepts for future extensions to the data model to describe disease states.

In GDD2 many instances of experimental process will be biological or chemical processes (or a combination). These may be used in tasks and as part of testing hypotheses, and experiments may also consist of mathematical or logical processes, for example for data analysis.Various kinds of equipment must also be modelled for the user to record experimental processes such as DNA synthesis, centrifugation, refrigeration and heating. This also allows the system to provide automatic stock control and reordering for items such as disposable plastic ware, such as microtitre plates and pipette tips as is required for the biochemical substances participating in reactions.

A description of each experimental procedure is stored in the database as a set of basic components of the procedure and the higher order structures (experiment plans) into which they can be assembled. This database of experimental methods can be queried and modified by the user. These are automatically retrieved when the user creates a new instance of an experiment. The user may also wish to design new experiments or steps in experiments, and the system must allow the creation of new combinations and new components. As a first stage, active rules will allow the system to react to the consumption of reagents during experiments and trigger their automatic reordering. Initially GDD2 is intended to provide a knowledge base as a reference source for experiment details which may eventually support an active experiment planner.

Representing Genetic Maps

Genome mapping is concerned with constructing an ordered map of genetic loci based on analysis of fragmentary ordering and metric data. Various methods (Guidi and Roderick 1993) (Mott et al. 1993) have been used and the input data take various forms. In general, a genetic database must be flexible enough to support the diversity of raw data required by and the interpreted data produced by different map construction algorithms. Some of the key requirements are: supporting the concepts of partial order and distance, accommodating uncertainty of measurement and ambiguity of results and handling local orientations and derived data (Guidi and Roderick 1993). As far as we are aware, there is no system which meets all of these requirements.

One of the difficulties is the lack of a suitable formalism to provide adequate representation and enable the construction of inference rules to allow reasoning with map data in the knowledge base.

The fact that most of the data related to genetic map construction is one dimensional suggests that temporal logic is a suitable formalism. Indeed, several systems based on temporal logics have been built. Letovsky and Berlyn (1992) use point set ontology for their constraint-based system CPROP which only uses one relation **B** (before). Local ordering windows are introduced to handle different orientations and an interesting set of inference rules are formed to contruct global ordering. Guidi and Roderick (1993) use Allen's (1983) interval-like logic to represent partial and uncertain ordering in DNA fragments but give no details in their short survey paper. Honda et al. (1993) propose an object-oriented data model which is also based on temporal logic. Graves' (1993) approach differs in using a general knowledge based system based on connection graph but the idea appears still to be based on temporal logic.

There are some problems in using temporal logic in that it is not entirely appropriate for map representation and it makes no provision for uncertain data. In addition, valuable metric information is not dealt with in most temporal logics. Although both DNA fragments and probes can be viewed as intervals, they do not always have precise location information or known distal extent. Partial orderings from experiments are true only in some local frame of reference whereas in the temporal domain there is a simple global orientation. Many experiments, for example hybridisation approaches to physical mapping, are designed to produce fragments overlapping to an unknown degree. These partial orders are also usually uncertain due to unavoidable experimental error and conflicting orders may result from different experiments.

In temporal logic, imprecise information is represented as disjunction of 13 mutually exclusive base relations. This is undesirable in our case because of the computational cost of representing disjunction. What

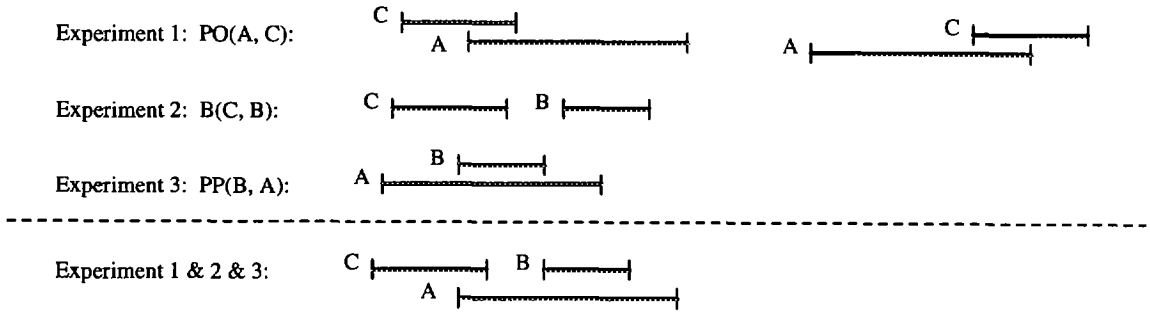

Figure 2: Diagram of Database Mapping Inference using Interval Logic

is needed is some higher order representation to cope with the alternative possibilities of interval relations. Recently, Freksa (1992) has used semi-intervals to represent this coarser knowledge but these relations do not have a obvious hierarchy which we can use in our data model.

The spatial logic described by Randell et al. (1992) defines base relations by a single primitive C (connected[1]) which can be represented in a hierarchy, corresponding to the level of coarseness of knowledge. For example, we may initially only observe two DNA fragments are connected, but not know whether they are adjacent or overlapping without further experimentation. Randell et al.'s logic can be used to define all the 13 interval relations in Allen's logic by introducing only one primitive relation B(x, y); x being before y. For the definitions of this logic and the complete refinements see Randell (1991).

Our basic model for map objects has the same hierarchy as this spatial logic. Specifically, each node represents a class of relations between map objects in the database. We consider the two components 'local orientation' and 'distance' (representing the two regions or fragments) as map object class attributes. The lowest level of information is represented in the root class (corresponding to weakest relation in the logic,) where the relation between two fragments is unknown. The highest level of information is encoded by the leaf nodes where the precise relation between two fragments is known. In our particular model, the fragments can be genetic entities such as DNA fragments, chromosomes, probe, clone and restriction sites as well as partial maps. Partial maps are represented in GDD2 as tuple linked lists, where each tuple consists of left and right pointers and the current fragment and heterogeneous maps are represented by multiple instances. In this way we are representing the coarseness of our knowledge about two DNA fragments in a general refinement hierarchy.

In this model, the inferences used in deriving new data and constructing maps are based on reasoning on the topological relations. Binary relations represented as metric data (known map distances) can be translated into topological relations in order to reason with these data in the knowledge base. From this logic we are able to develop a set of inference rules, similar to those of CPROP (Letovsky and Berlyn 1992) but making use of more information and thus allowing greater powers of deduction.

In the example shown in Figure 2 we show how deduction in the GDD2 knowledge base can operate on the data from three separate experiments (1, 2 and 3). In this case we are not considering metric data, only order relations. In all three loci (A,B and C) are studied and their data entered into the system. Experiment 1 reveals a partial overlap between A and C : PO(A, C), which may have the two local orientations as shown. Experiment 2 informs the worker that locus C lies before B : Before(C, B), and Experiment 3 detects that locus A contains locus B, expressed as B is a proper part of A : PP(B, A). The automated deduction carried out by the system can reason, as the worker would, that there is only one solution for the local orientation, as shown.

Each partially ordered set of fragments in the knowledge base is associated with a local orientation. Constructing a new partial order out of two partially ordered fragments presents a problem of ambiguity in the data which requires further use of the inference rules and the maximal utilisation of available data. The GDD2 system, through the use of IDEA technology and reasoning methods to supplement this, is attempting to provide a high degree of automated deduction and propagation of inference. These are necessary elements in the formation of an intelligent system for molecular biology.

Representing Imperfect Data

Throughout the design and implementation of the Genetics Deductive Database we have been aware that biological data are characteristically imperfect (Allison 1993), and that some mechanism must be provided in order to represent these imperfections. There are a number of different issues concerning imperfection (Motro 1992) including inconsistency, where labs

[1]Informally, two regions are connected if they share a point in common.

give differing results to the same question, ambiguity, where more than one interpretation is possible, and incompleteness, where values for attributes do not exist in the data for whatever reason.

When imperfect, data must be appropriately marked and handled in the knowledge base, data analysis and the propagation of updates. Inconsistent biological data are relatively common and their resolution must often await further results. In cases where a decision is needed on inconsistency, some criterion must be used to favour one of the competing pieces of data. Similarly, it will be necessary at times to disambiguate data to obtain an interpretation that fits with other related information. Such conclusions drawn may require revision at any time. Incomplete or missing data can be handled in a number of ways—it is possible to fill in the gap with some reasonable assumption, explicitly represent the fact that the item is missing, or discard the item with the missing value (Mott et al. 1993)—while the uncertainty surrounding data must be propagated and communicated to the user.

Many of the problems in representing imperfect information have been addressed. The usual approach to handling uncertainty is to attach some numerical measure, typically a probability, to every fact (Barbara et al. 1990). This permits the certainty of information to be represented in a reasonably intuitive way. Vague and ambiguous data may be modelled using fuzzy predicates (Buckles and Petry 1987) which permit the expression of the degree to which given attributes have values. Inconsistent and incomplete information can be addressed by the provision of null values (Imieliński and Lipski 1984), or by applying methods from non-monotonic logic (Ginsberg 1987) to fill in the gaps. Such techniques may also be applied to resolve inconsistency by specifying default choices, and any implementation of a non-monotonic method will require some form of truth maintenance system (Doyle 1979),(de Kleer 1986) to ensure consistency as reasoning progresses. For example, newly derived data from reasoning processes may require an update of the database which includes removing previously derived data no longer held to be true.

Thus if we wish to handle all these different aspects of imperfection, we will either need to combine a number of different techniques together in some way, as argued by Umano (1983), or to use a means of representing imperfection that can be applied to all the different varieties. We are doing the latter, adopting a method of argumentation (Fox et al. 1993).

Argumentation provides a general framework for representing knowledge that is closely related to the notion of labelled deductive systems (Gabbay 1990), and extends the usual logical notion of what constitutes an argument. In classical logic an argument is a sequence of inferences that lead to a conclusion. The basis on which the conclusion is reached is implicitly the rules of inference of the logic and the full set of facts that are known to the reasoner. Fox et al. (1993) describe a logic of argumentation LA which extends this idea by annotating the conclusion with an explicit statement the set of facts that are used in the deduction, the reason for believing the conclusion, and the degree to which the conclusion is believed. The triple of conclusion, the degree of belief and the reason for believing it are called an argument. LA clearly generalises classical logic, but goes further since the degree to which the conclusion is believed can be stated numerically, or symbolically to represent for example that a piece of information is from an authoritative source.

Argumentation also allows the representation of inconsistent and non-monotonic information. Inconsistency, which defeats normal logical methods, may be handled because LA makes it possible to represent and resolve the conflicts caused by inconsistency. Thus it is possible to express the idea that a conclusion is rebutted, when it is possible to construct an argument that has the opposite conclusion. Since the stages in reaching a conclusion are also represented, as the 'reasons to believe' it is also possible to record the fact that there are arguments against these, in which case the conclusion is said to be discounted. By considering all the possible discounting and rebuttings that may be derived, an overall conclusion may be reached. The fact that an existing conclusion can be rebutted by a new one, derivable from new data, makes LA capable of non-monotonic reasoning.

Within GDD2, LA can be used to represent the support for database facts by making use of its ability to handle any kind of object as a degree of belief. LA can also be used to handle missing information, since it is possible to represent explicitly the fact that a value has been used in the absence of information, and it can be used to manage inconsistency by providing a mechanism for deciding which of the various options has the most support. Thus it seems suitable for the representation of most kinds of imperfection that are necessary in GDD2.

As an example of the simplest use of LA we consider an example cited by Guidi and Roderick (1993). They discuss combining order and distance data from various experiments. LA allows the representation of the arguments for and against a combination of interpretations of the data in Figure 3 from five partial maps of a chromosome. LA allows the propagation of a record of the support for particular genetic maps when they are combined, for example using interval logic, and in resolving the inconsistency between different maps. Combining the results of Experiment 1 and Experiment 2 gives Map 3. Since the map obtained from Experiment 3 disagrees with that of Experiment 2, which is one of the steps in obtaining Map 3, Experiment 3 discounts Map 3. Similarly, the experiment that generated Map 5 rebuts Map 3. Thus, assuming that all the results are equally valid, Map 5 should be preferred to Map 3. We are also working on the evaluation of the strength

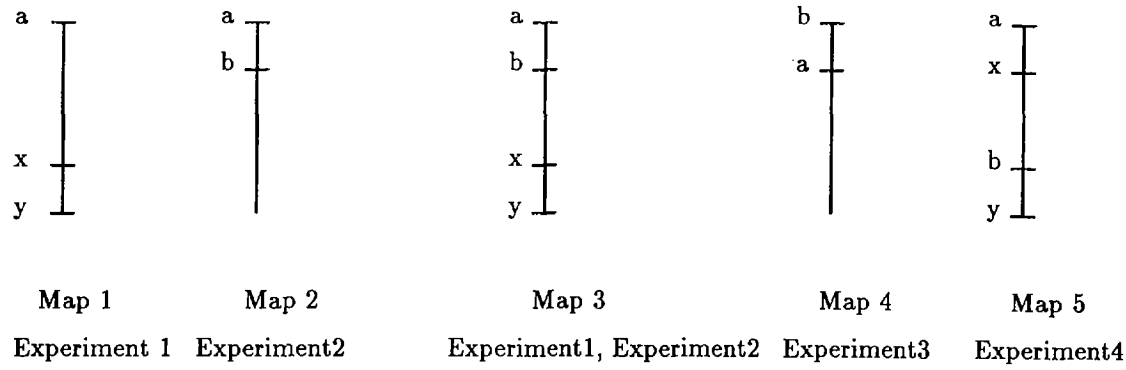

Figure 3: Inference with genetic maps

of arguments for reasoning with these kinds of contradictory data by attaching measures that reflect the quality of the results.

Conclusions and Future Research

In conclusion we can say that the GDD1 system clearly illustrates the potential for active behaviours in providing automated assistance to the database user and that the IDEA technology has the potential to satisfy this requirement. The IDEA technology being provided in the project for developing GDD2 is stronger and has optimized query functions to improve the overall performance of the application. The GDD2 system adds considerably to the reasoning capability of the knowledge base as well as improving its content. We are confident in the utility of the interval logic inference rules and the representation of the imperfections in the mapping data. The modelling and management of experiments is an area of ongoing effort. Initially at least, GDD2 is focussing on a foundational knowledge base for experimental methods and materials for the active management of resources rather than personnel. With later versions of IDEA technology GDD should provide a usable laboratory notebook system for molecular biology.

Acknowledgements

This research was partly supported by the European Commission under ESPRIT Project 6333 IDEA (Intelligent Database Environments for Advanced Applications). This paper reflects the opinions of the authors and not necessarily those of the consortium.

References

Allen, J. 1983. Maintaining Knowledge about Temporal Intervals. *ACM Transactions on Database Systems* 26(11):832–843

Allison, L. 1993. Methods for Dealing with Error and Uncertainty in Molecular Biology Computations and Data-Bases. In Proceedings of the 26th Hawaii International Conference on System Sciences, 604–604.

Baclawski, K., Futrelle, R., Fridman, N., and Pescitelli, M. 1993. Database Techniques for Biological Materials and Methods. In Proceedings First International Conference on Intelligent Systems for Molecular Biology, 21–28.

Barbara, D., Garcia-Molina, H., and Porter, D. 1990. A Probabilistic Relational Data Model. In Proceedings of the 1990 EDBT Conference, 60–74.

Bayer, P. 1993. State-Of-The-Art Report on Reactive Processing in Databases and Artificial Intelligence. *The Knowledge Engineering Review* 8(2):145–171.

Buckles, B. C., and Petry, F. E. 1987. Generalized Database and Information Systems, In *Anaysis of Fuzzy Information, Volume 2*, J. Bezdek (ed.), CRC Press.

Cui, Z. 1994 Using Interval Logic for Order Assembly In Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology (This volume)

Cui, Z., Fox, J., and Hearne, C. 1993. Knowledge based systems for molecular biology: the role of advanced technology and formal specification In Proceedings of the IJCAI Workshop on AI and the Genome 87-97.

Doyle, J. 1979. A Truth Maintenance System. *Artificial Intelligence* 12:231–272.

Fox, J., Krause, P., and Elvang-Gøransson, M. 1993. Argumentation as a General Framework for Uncertain Reasoning. In Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence, 428–434.

Freksa, C. 1992. Qualitative Spatial Reasoning In *Cognitive and Linguistic Aspects of Geographic Space.*, D.M.Mark and A.U.Frank (ed.s), Kluwer, Dordrecht.

Gabbay, D. 1990. Labelled Deductive Systems, CIS Technical Report 90-22, University of Munich.

Ginsberg, M. 1987. *Readings in Nonmonotonic Reasoning.* San Mateo, CA: Morgan Kaufmann.

Graves, M. 1993. Integrating Order and Distance Relationships from Heterogeneous Maps In Proceedings First International Conference on Intelligent Systems for Molecular Biology, 154–162.

Gray, P., Paton, N., Kemp, G., and Fothergill, J. 1990.

An object-oriented database for protein structure analysis. *Protein Engineering* 3:235–243.
Guidi, J. and Roderick, T. Inference of Order in Genetic System. In Proceedings First International Conference on Intelligent Systems for Molecular Biology, 163–169.
Hajnal, S., Krause, P., and Fox, J. 1989 Sceptic User Manual Technical Report 95, Advanced Computation Laboratory, Imperial Cancer research Fund.
Honda, S., Parrott, N., Smith, R., and Lawrence., C. 1993 An Object Model for Genome Information at All Levels of Resolution. In Proceedings of the 26th Hawaii International Conference on System Sciences, 564-573
Imieliński, T., and Lipski, W. 1984. Incomplete Information in Relational Databases *Journal of the ACM* 31(4):761–791.
Kazic, T., Lusk, E.,Olson, R., Overbeek, R., and Tuecke, S. 1990. Prototyping Databases in Prolog In *The Practice of Prolog*, Leon Sterling (ed.) Cambridge, Mass.: MIT Press.
de Kleer, J. 1986. An Assumption-based TMS. *Artificial Intelligence* 28:127–162.
Kochut, K., Arnold, J., Miller, J., and Potter, W. 1993. Design of an Object-Oriented Database for Reverse Genetics. In Proceedings First International Conference on Intelligent Systems for Molecular Biology, 234–242.
Letovsky, S. and Berlyn, M. 1992. CPROP: A Rule-Based Program for Constructing Genetic Maps. *Genomics* 12: 435-446
Motro, A. 1993. Sources of Uncertainty in Information Systems, In Proceedings of the 2nd Workshop on Uncertainty Management and Information Systems, Catalina Island.
Mott, R., Grigoriev, A., Maier, E., Hoheisel, J., and Lehrach, H. 1993. Algorithms and Software Tools for Ordering Clone Libraries. *Nucleic Acids Research*, 21:1965–1974
Pearson, P. L. 1991. The genome database (GDB) - a human genome mapping repository *Nucleic Acids Research* 19:2237–2239
Randell, D. 1993. Analysing the familiar. Reasoning about space and time in the everyday world. Ph.D Thesis University of Warwick 1991
Randell, D., Cui, Z., and Cohn, A. 1992. An Interval Logic for Space based on 'Connection'. In Proceedings of the 10th European Conference on Artificial Intelligence, 394–398
Rawlings, C., Taylor, W., Nyakairu, J., Fox, J., and Sternberg, M. 1985. Reasoning about protein topology using the logic programming language Prolog *Journal of Molecular Graphics* 3: 151 157.
Rawlings, C. J., Brunn, C., Bryant, S., Robbins, R. J., Lucier, R. E. 1991. Report of the Informatics Committee *Cytogenetics and Cellular Genetics* 58:1833–1838
Umano, M. 1983. Retrieval from fuzzy database by fuzzy relational algebra. In Proceedings of the IFAC Symposium, 1–6.
Yoshida, K., Smith, C., Kazic, T., Michaels, G., Taylor, R., Zawada, D., Hagstrom, R., and Overbeek, R. 1992. Toward a Human Genome Encyclopedia In Proceedings of the International Conference on Fifth Generation Computing Systems, 307–320.