PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT. © ICOT, 1992

Integrated System for Protein Information Processing

Hidetoshi Tanaka

Institute for New Generation Computer Technology (ICOT)
1-4-28, Mita, Minato-ku, Tokyo 108 Japan
htanaka@icot.or.jp

Abstract

This paper describes requirements for databases and DBMS in protein information processing, and an experiment on a privately integrated protein knowledge base. We consider an integrated DBMS-KRL system for an integrated protein KB-DB.

In order to clarify unknown functions of proteins via empirical approaches, existing public databases should be integrated as part of a private knowledge base, which can proceed biological knowledge discovery. And DBMS-KRL system should support representability, parallel processing, information retrieval, advanced query processing, and quality management techniques for protein information.

DBMS Kappa-P and KRL $\mathcal{QUIXOTE}$, both designed at ICOT, perform a useful role in processing protein information. Kappa-P is for efficiency by its parallel processing and extensibility, while $\mathcal{QUIXOTE}$ is for advanced query processing and quality management, by its representational flexibility of object identification and module.

1 Introduction

Molecular biological information processing is increasing in importance, as biological laboratories are improving in their computational environments and biological data are augmenting far more faster than biologists' understanding. To speed up converting such data into biological knowledge, biological database should help biologists by providing conveniences in storing, browsing, and query processing.

Molecular biological databases have two categories: public databases and private databases. Public databases have hundreds of Mbytes of various data: sequence, structure, functions, and other indispensable auxiliary information of DNA, RNA, and protein.

There are variation in their sizes and data structures. As for the sizes, PIR Release 30 (1991) has the proteins of 1 residue through to 6048 residues. GenBank Release 70 (1991) has the regions (*loci*) of the DNA sequences of 3 bases through to 229354 bases. As for the data structures, for example, the feature descriptions of proteins or loci require multiple nested structures. A protein often consists of plural amino acid sequences. A sequence of *eucaryote* is often coded in several separate DNA regions (*exons*) with separate *expression regulatory region*. The structure of regulatory regions is so unclear that we can only describe them as nested patterns of DNAs.

Public databases are maintained under international cooperation or specific volunteers, and provide most molecular biological data freely. Although the amount of data is increasing rapidly, recent dynamic improvements of machine environments allow biologists to store such data in their own small systems, and to use them as part of value-added private databases.

Such environments also allow them to create a database including their own experimental results, make cross-references between public and private databases, add customized query processing facilities, and try to conduct knowledge discovery by extracting rules from data.

In this paper, we focus on such privately integrated databases which are developed as part of the molecular biological information processing system of the FGCS project.

As an example, we are building an integrated protein knowledge base in the framework of deductive object oriented database (DOOD), which consists of a knowledge representation language (KRL) $\mathcal{QUIXOTE}$ and a DBMS Kappa-P. The reason why we choose protein information is due to their moderate amount for storage and study. As biological applications are very new, we had to check the appropriateness of the system and request to add several facilities to it.

We have developed Kappa-P and $\mathcal{QUIXOTE}$ on a parallel inference machine PIM. Kappa-P employs a nested relational model, and has a facility of extensible DBMS, which appears to be suitable for parallel processing and sequence retrieval.

$\mathcal{QUIXOTE}$ is based on a concept of DOOD. It provides a capability of advanced query processing, rich concepts such as module, identification, subsumption relation, and flexibility in describing knowledge.

This paper describes two types of system integration. One is database integration of protein information. The other is DB-KB (database and knowledge base) integration in Kappa-P + *QUIXOTE*. These are shown in Section 6.

We describe the requirements for databases and DBMSs in Section 2. Overviews of the protein databases we are focusing on are described in Section 3. The suitability of Kappa-P and *QUIXOTE* as ingredients of our integrated knowledge base system is discussed in Sections 4 and 5.

2 Requirements for Biological Database Systems

2.1 Requirements for Databases

Most of the biologists' requirements for existing molecular biological databases are concentrated into the problem: it is difficult to access several databases at once. It is because the differences between the databases: the attributes' meanings, the values' variations, and their relations, must be understood beforehand.

Such requirements are solved by integrating them. There are three approaches to database integration.

Standardization

Standardization is the most fundamental integration. It provides the simplest environment for the wide use of databases.

CODATA (Committee on Data for Science and Technology) in ICSU (International Council of Scientific Unions) proposed standardization of attributes to realize the virtual integrated database [JIPID 90]. The schema of every public database should be a subset of the virtual schema. NLM (National Library of Medicine) provides GenInfo Backbone Database [NCBI 90]. According to [NCBI 90], it is built as a standardized primary database, which is assumed to be a basis for secondary, value-added databases for the specialized interests of different biologists.

Determining a standard, however, is expensive. It is almost impossible to make the widest virtual schema that covers all attributes of protein information. We should accumulate experiences in creating and using most private databases as before. Moreover, both standardizations started so recently that they have not so widely distributed yet.

Integrated User Interface

Making an integrated user interface is the fastest way of getting an integrated environment. It generally provides not only query processing facilities but also visual browsing facilities, which are quite attractive and useful for biologists. It used to need a lot of cost, however, to remake an interface when a new database is to be added.

GeneWorks[1] and Entrez[2] provide integrated environments to enable access to existing DNA / amino acid sequence databases, although they are packaged for browsing only, from a PC or Mac. They are not for adding new applications or new databases.

S. Smith et al. (Harvard U.) are developing an environment for genetic data analysis (GDE[3]) which will help access several databases at once by providing data exchange tools between representative databases. The first version is based on a *multiple alignment editor* and allows tools for sequence analysis to be included in the system. It can reduce efforts for the interface remaking by rich widgets. It has also just started and further improvement is expected.

Integrated Knowledge Base

The integrated knowledge base is our approach. It consists of two stages: to represent all facts in one language, and to supplement the rules necessary to get and use the facts (see Fig. 1). The former corresponds to standardization, and realizes a syntactically integrated database. Not only existing public databases but also private databases are integrated. In order to provide an useful integrated database system, efficient DBMS which allows to store and to access complex data easily.

The latter stage converts a database into a knowledge base, by accumulating supplementary knowledge, which are rules or facts recognized by biologists themselves. It seems almost impossible to define common operations to all knowledge just as relational algebra to relational database. Thus, new concepts had better be introduced to the mechanism, so that each (or a cluster of) knowledge can have intrinsic methods. DOOD is a promising concept for the mechanism.

2.2 Requirements for DBMS

Among the requirements for databases we can find ones for DBMS or data models which require improvement in retrieval and identification. Traditional ways are not so appropriate for some molecular biological applications, e.g., sequence retrieval and quality management.

Information Retrieval

DBMS is expected to support the facilities of information retrieval for the sequences of DNA, RNA, and amino acids. It is partly because a concept of DBMS is rather wider for biologists than traditional one for database researchers.

[1]IntelliGenetics, Inc., Mountain View, CA
[2]National Institute of Health, Bethesda, MD
[3]Genetic Data Environment

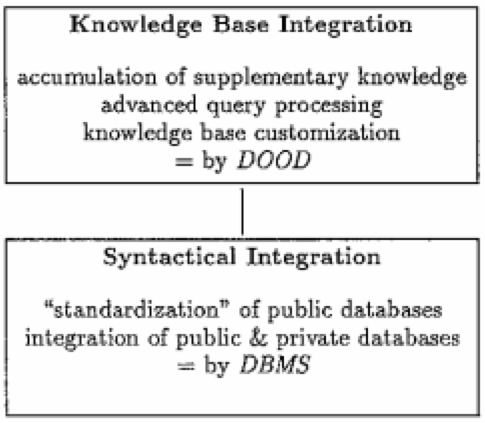
Knowledge Base Integration
accumulation of supplementary knowledge
advanced query processing
knowledge base customization
= by *DOOD*
Syntactical Integration
"standardization" of public databases
integration of public & private databases
= by *DBMS*

Figure 1: Integrated Knowledge Base of Proteins

Although sequence retrieval is like full text search, there are some differences in the search criteria: similarity search via dynamic programming (DP) or other algorithms (ex. BLAST[Altshul *et al.* 90]), with given similarities between characters (namely, amino acids).

"Keyword" extraction from the sequence is far more difficult than from the text. In order to process large sequences, they should be preprocessed by an information retrieval technique, as strings are preprocessed to make an alphabetical index. Keyword extraction corresponds to such preprocessing.

However, we should consider first, what word is in DNA or amino acid sequences. A gene is a sentence, and a DNA is a character. Thus, a word might be a specific DNA pattern closely related to some function, which is not so clear at present. An amino acid sequence is a sentence, and an amino acid is a character. So, a word may be a structural block or a functional block, either of which is represented as an amino acids' pattern including some varieties. Determining and extracting "keywords" is one of the big problems in biology.

At present, the sequences should be regarded as character strings, and not as paragraphs or sentences which consists of words. Moreover, we have to consider the following features of the sequences.

- DNA and RNA sequences consists of only four characters.
- Proteins mostly consist of 20 characters, but there are some exceptions.
- Similarity between the characters are defined.

Identification Facilities

DBMS is required to provide rich identification facilities. In treating molecular biological data, we should consider at least two kinds of errors: *experimental errors* and *identification errors*. Experimental errors are inevitable in molecular biological databases. It is necessary to repeat experiments to reduce them. In reading DNA sequences, for example, the *same region* should be repeatedly read to verify the result. In such case, GenBank is useful in reducing efforts of verifying. though it has sequences with various qualities. It contains many staff-reviewed sequences with many references, while it also contains a lot of sequences each of which has been just registered by a researcher.

Identification errors possibly occur in this verification process. It is not so clear whether we can get the sequence of the *same region* because of slightly different repeating regions, or natural errors such as diseases or mutations.

Representation of relations between proteins and functions are more ambiguous than relations between loci and DNA sequences in the example above. We should always consider identification errors in both proteins and functions. As experimental facts are accumulated, for example, "cytochromes transfer electrons" may turn into relations such as "cytochromes and ubiquinone transfer electrons" (protein identification is relaxed) or "cytochromes transfer electrons to generate energy" (function identification is detailed).

A concept of object identity is important in such cases. Results including experimental errors should be treated as different objects to store them avoiding integrity problems, whereas they should be treated as an object when we ask the "verified" result. Furthermore, identifiers should be so flexible that we can change them with as few difficulties as possible.

3 Protein Databases

In Section 2, general issues on molecular biological databases and DBMS are shown. This section focuses on protein databases and overviews the reasons for using existing public databases, as well as their general use, in order to consider the necessity of an integrated knowledge base.

3.1 Public Protein Databases

Protein information includes amino acid sequences, 3D structures, and functions. Protein functions include thermodynamic, chemical, and organic functions of total or partial proteins. In addition, there is important auxiliary information such as the authors, titles, and journals of the references relating to the data, source organisms, and experimental conditions.

Public protein databases have been trying to cover all protein information: amino acid sequences (PIR, Swiss-Prot), structures (PDB), partial patterns (ProSite), enzyme functions, and restricted enzymes (REBASE). All databases contain the auxiliary information mentioned above.

The amount of information contained for each area is shown in Table 1. It seems possible that whole databases can be held privately in order to catalyze a change in their use: from databases accepting biological applications to knowledge bases including public databases and processing advanced biological queries.

Table 1: Public Protein Databases

database	release	entries	size
PIR	30.0 (9/91)	33,989	9,697,617 residues
Swiss-Prot	18.0 (5/91)	20,772	6,792,034 residues
PDB	(7/91)	688	153 Mbytes
ProSite	7.0 (5/91)	508	1 Mbytes
REBASE	(1/92)	1975	16 Mbytes

3.2 Purposes of Protein Databases

The final goal of using protein databases is to predict the unknown functions of a protein. Biologists gather enough of a known relations between functions and proteins to predict the unknown functions of a known/unknown protein as accurately as possible.

Its subgoals are important for molecular biology:

- Understanding of the relation between protein structure and function

  Most of protein functions are due to their structure. Structure might be predicted from their amino acid sequences, by *molecular dynamics* or several kinds of empirical approaches.

- Prediction of the 3D structure from the amino acid sequence

  Theoretically, most of protein 3D structures are calculated by molecular dynamics. It costs, however, enormous time to compute at present. Thus various empirical approaches have been tried, and would be tried.

The sequence similarity search, especially for extraction of common sequence patterns or similar regions (which are often called 'consensus sequences' in molecular biology) is a first step of empirical approaches. How to represent and how to use biological knowledge to predict unknown structures or unknown functions are other early problems.

3.3 General Use of Protein Databases

Similarity Search

Most traditional uses of protein databases are supported by traditional DBMS, except for similarity searches in the sequence database. Biologists ask the database to get a set of proteins whose name is, for example, "cytochrome c", or proteins which are found in "E.Coli." This type of retrieval is supported by the traditional DBMS.

They often want to examine such a set of sequences to discover a description of the similarity of a certain protein set, such as the existence of consensus sequences. They use *multiple alignment* [Ishikawa *et al.* 92] after they get all sequences they want. In this case, we consider interaction between application and database in rather higher level (see Section 6).

Another important use is similarity search. They search for amino acid sequences in the database that are similar to the unknown sequence they have. The unknown sequence may be a fragment or a whole sequence. The former is *motif search*, which is regarded as text content search, while the latter is *homology search*.

Biologists want to get accurate results in these searches and examination. Because the accuracy affects the quality of function prediction and structure prediction. They would like to retrieve the several of the best sequences of similar functions in the database.

In order to improve recall and precision ratios in protein similarity search, plenty of biologists' empirical knowledge and experimental results are indispensable. In addition to them, two problems have to be solved: finding an efficient algorithm for the homology and motif searches, and speeding up basic retrieval. The former needs the cooperation of biologists and computer scientists, whereas the latter could be devised independently by computer scientists, for some basic operations might be taken from techniques of the partial string match in the text database.

Data management

Data management, such as designing schema, storing data, and checking integrity, are owing to great efforts of the staffs of public databases.

Recently, schema of existing public databases are gradually standardized (as shown in Section 2), however, each existing database still employs independent naming rules using alphanumeric symbols such as 'P08478'(PIR), 'AMD1$XENLA'(Swiss-Prot), and '1.14.17.3'(Enzyme DB). Biologists are annoyed by updating cross-references among public databases and private ones.

As for storing, public databases accept an electronic form of registration to reduce staffs' efforts for quick storing. The U.S. National Institute of Health proposes a standard format for data exchanging (ASN.1 [NCBI 90]), which simplifies registration procedures and is useful in gathering them into personal systems.

In order to distribute recent data as quickly as possible, PIR distributes less verified data for biologists. Thus, it reduces staffs' efforts for quick checking. When such data are used, verification process is owing to the biologists who would like to use them. PIR has three kinds of indications by their verification level: 'Annotated and Classified', 'Preliminary', and 'Unverified'. It is obvious that such indications are not enough for biologists' private data management.

Cooperation with biologists is indispensable in settling how to identify data with their quality and make cross reference data, although some management can be independently devised for advanced uses.

4 Kappa-P: An Extensible Parallel DBMS

We use Kappa-P as an ingredient in our integrated system. Kappa-P provides several facilities suitable for protein information. The efficiency of the nested relational model of Kappa is shown in [Yokota *et al.* 89], where efficient usage of storage and flexibility of schema evolution are described. In this section, we show the effectiveness of Kappa-P as an extensible DBMS for protein information processing and how to embed information retrieval facilities into Kappa-P.

4.1 Parallel DBMS

As the sequence search is executed exhaustively on a full sequence, its parallel processing is obviously effective. Fig. 2 shows the cofiguration of our system.

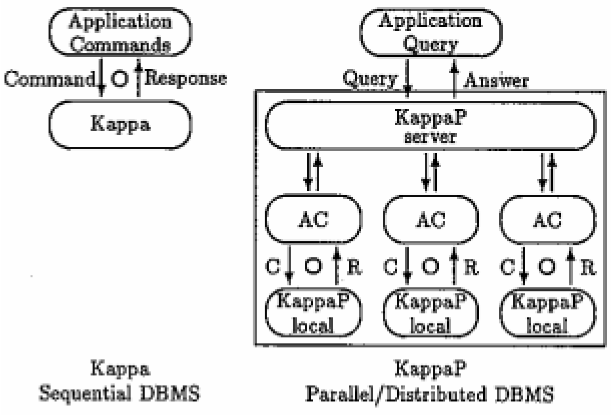

Figure 2: Configuration of Kappa-P

The aim of the extensibility of Kappa (sequential DBMS) is to reduce interaction with applications, and to customize the command interface for each application. The modules of an application which frequently use Kappa commands are included in Kappa system so that the number of communication among processes on the left-hand side of Fig. 2 decreases.

Beside these facilities, another kind of extensibility must be considered in a parallel system. Parallel DBMS Kappa-P consists of server DBMS and local DBMS, where server DBMS has a global map of local DBMSs and coordinates local DBMSs to deal with users' request, while local DBMS holds users' data [Kawamura *et al.* 92]. In our environment, most applications work on the same PIM with Kappa-P. So, if the server DBMS merges all the answers from local DBMSs into one answer, the effectiveness of parallel processing is reduced.

In order to avoid such a situation, the user defined commands, for example, DP or BLAST, are thrown to every local DBMS, and they play a role of filter from local databases to their server. The filters select data satisfying the given conditions, and send them to the server processor.

It is obviously efficient to throw application procedures to every local DBMS. The extensibility in Kappa contributes to efficient parallel processing of sequence search as in Fig. 2.

4.2 Information Retrieval

Extensible DBMS is also suitable for supporting information retrieval, obviously because it allows to customize command interface for applications. Sequence similarity searches, which correspond to full text searches, are implemented easily as "Application Commands (AC)" in Fig. 2.

We have developed a character-pair based index system, especially for motif search. This kind of index system is also implemented as application commands, while indexes are held in each local DBMS. Thus, the number of communications between local and server DBMS decreases.

Motif dictionary such as the public database ProSite could also be used as another useful index for sequence similarity searches. Extensible DBMS is so flexible that when such an improved index is developed it could be easily added in the system.

5 *QUIXOTE*: A Deductive Object Oriented Database

We use *QUIXOTE* as another ingredient in our system. Though advanced query processing is available by any logic programming language, facilities of *QUIXOTE* are more suitable to represent protein information, especially protein functions [Tanaka 91].

In this section we focus on its representation facilities in data management: schema flexibility and powerful identification.

5.1 Objects and Modules of quixote

Object Identifier

Objects in $\mathcal{QUIXOTE}$ are represented by extended terms called *object terms* [Yasukawa *et al.* 92]. An object term is of the following form:

$$o[l_1 = v_1, \ldots, l_n = v_n]$$

where o is called the *head* of the object term, l_i is a label, and v_i is the value of the l_i of the object term.

The labels and their values of an object term represent the *properties* of the object which are intrinsic to identify it. In such sense, object terms play a role of *object identifiers*. An object may have properties other than those specified in its object term. To represent such properties (extrinsic properties) of an object, special form of term representation called an *attribute term* is used:

$$o[l_1 = v_1, \ldots, l_n = v_n]/[l'_1 = v'_1, \ldots, l'_m = v'_m].$$

This attribute term represents that the object identified by the object term $o[l_1 = v_1, \ldots, l_n = v_n]$ has properties $[l'_1 = v'_1, \ldots, l'_m = v'_m]$. It is important to distinguish intrinsic properties with extrinsic ones.

Simplified examples are shown in Fig. 3. (1) and (2) describe the same object and their attribute terms contradict each other, while (1') and (2') represent different objects.

```
object_head [ol1 = ov1, ol2 = ov2, ...] /
            [al1 = av1, al2 = av2, ...]

  (1)  fact [label1=v1] / [label2=v2].
  (2)  fact [label1=v1] / [label2=v3].

  (1') fact [label1=v1, label2=v2].
  (2') fact [label1=v1, label2=v3].
```

Figure 3: Examples of $\mathcal{QUIXOTE}$ objects

Such representation of protein information is quite useful, for only the attributes whose values are determined can be used for identification. It is also useful in repeating local integrity checking, as data set would not stop increasing in amount.

Module

Modules in $\mathcal{QUIXOTE}$ help object management. Simplified examples are shown in Fig. 4.

```
(1)  module1 :: object1.

(2)  module2 >- module1.
(3)  module2 :: {{ object2. object3. }}

(4)  module3 >- module1.
(5)  module3 :: object3.
```

Figure 4: Examples of $\mathcal{QUIXOTE}$ modules

"object1 is an object in module1" is represented as (1). (3) is an abbreviation form. (2) represents an order between modules specifying that module2 inherits all the objects from module1, and (4) represents another inheritance. Therefore, module2 has object1, object2, and object3, whereas module3 has object1 and object3.

Although object3 is in both module2 and module3, it may have different properties in each module, because any relations between module2 and module3 is not defined. We can give different properties to the same object in different modules. Thus, we can use different modules to avoid database inconsistency when we get different results by different experiments.

5.2 Identifiers of Proteins

Requirements

Since it is impossible to give the clearest identifier instantly, identification requires that the following be satisfied.

(1) Subsumption relation

An identifier sometimes has to be generalized or specialized. For example, the sentence "cytochromes have a certain feature" sometimes has to be reconsidered as "cytochromes and hemoglobins have a certain feature" or "cytochrome c has a certain feature." It seems rare to misidentify completely different objects. Most erroneous identifiers have to change only their abstraction level, and need not to be altered completely.

(2) Flexibility

In the process of determining the clearest identifier, we feel it useful if DBMS accepts tentative identifiers which can be specialized or generalized at anytime. We could use trial and error to determine the proper identifier.

(3) Module

To distinguish tentative identifiers from fixed ones, or experimental results from derived ones, a facility for making modules is required. It allows local integrity to be checked within the module, and for the global uniqueness of the labels of the identifiers to be ignored.

Flexibility along Subsumption Relation

Proteins need identification transition along subsumption relation, as shown above. Fig. 5 is an example of how they are represented in *QUIXOTE*.

```
(fact1) cytochrome[lifename= E.Coli] /
                             [feature = featX].
(fact2) cytochrome[type= c] /
                             [feature = featX].
(hyp1)  cytochrome / [feature = featX].

(cf.1)  cytochrome( E.Coli, _, featX )
        cytochrome(         _, c, featX )
(cf.2)  cytochrome( [lifename(E.Coli),type(c)],
                    featX )

(hyp2) protein[name={cytochrome,hemoglobin}] /
                 [feature = featX].
```

Figure 5: Proteins as objects in *QUIXOTE*

Provided that there is a feature named 'featX'. As experiments are repeated, the identifier of the protein whose feature is 'featX' may be changed.

QUIXOTE expressions (fact1) and (fact2) are examples of the identification of experimental results. (fact1) mentions nothing about the (fact2) attribute "type", and vice versa.

In the relational data model or Prolog, it is necessary to redesign the attributes of tables or arguments of facts to reflect such schema changes, since attributes have to be fixed. This is shown in (cf.1). In Prolog, we can reflect them by using lists as shown in (cf.2) [Yoshida *et al.* 91]. However, it is necessary to support a particular unifier for the list, and users must manage the meaning of the list (e.g., connected by 'and' or 'or') carefully. *QUIXOTE* allows set concepts with particular semantics to avoid such mismatches.

When we consider what sort of protein has 'featX' and get (fact1) and (fact2), we can easily think of a hypothesis (hyp1). We can also get this hypothesis by *QUIXOTE* , using object lattice of subsumption relation.

Moreover, if we give some relations among 'cytochrome', 'hemoglobin', and 'protein', another hypothesis such as (hyp2) is available.

Modules for Data Management

Objects of experimental results, verified results, and public databases have to be distinguished by modules, to be checked by different integrity checking methods.

Fig. 6 shows an example of verification process. Upper modules inherit all facts and rules of their lower modules. **'PIR'**, **'Swiss-Prot'**, and **'Experimental Results'** are modules each of which allows local integrity checking. If identifiers are conflicted between these modules, they can be settled at their upper module.

'Sequence' has some rules and cross-references between PIR and Swiss-Prot so that it can select and reply a specific set of protein sequences contained in these public databases. **'Integrated'** has some rules to verify experimental results by merging the selected sequences from public databases. It also has cross-references between public databases and experimental results (but they are ignored in Fig. 6 to simplify the example).

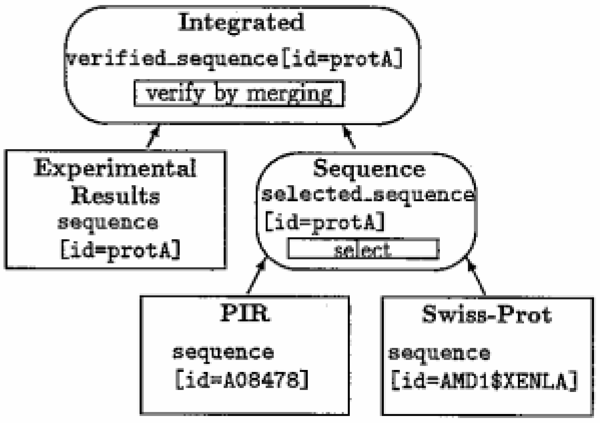

Figure 6: Modules for Verification Process

6 An Integrated System for an Integrated Knowledge Base

This section shows a system integration and a DB-KB integration, as to their configuration and their uses.

We are considering two kinds of integration: Kappa-P and *QUIXOTE* (DBMS and KRL), and existing public databases and biological knowledge (DB and KB).

6.1 Configurations of Integration

DBMS and KRL

There are three interactions in the integrated system of a database management system Kappa-P and a knowledge representation language *QUIXOTE* (see Fig. 7).

(1) Interactions between Kappa-P and *QUIXOTE*

All facts (non-temporal objects) in *QUIXOTE* are stored in Kappa-P, and Kappa-P activates necessary objects as the result of retrieval.

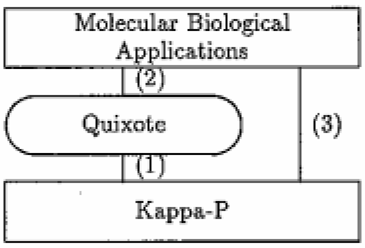

Figure 7: Integrated System of Kappa-P and Quixote

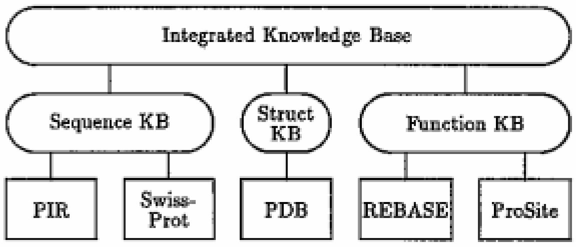

Figure 8: Integrated Knowledge Base of Proteins

(2) Applications and *QUIXOTE*

The followings are supported or will be provided (are under development).

- advanced query processing facility (inference)
  As *QUIXOTE* is an extension of Prolog, it provides more flexible and powerful query processing facility.
- a standard of the molecular biological data
  New databases and new rules (knowledge) are easily available by supporting ASN.1.
- graphical user interfaces
  Ad-hoc uses are quite important for biologists. The system should support ad-hoc queries, with graphical user-friendly interfaces. Kappa supports user interfaces for the nested relation and for PIR on X-window.
- class libraries for biological use
  This would include sequence retrieval and data management (see Sections 4 and 5).

(3) Applications and Kappa-P

The system should support direct access to databases for simple queries. It currently supports a graphical user interface to access amino acid sequences and some libraries to maintain biological data.

Protein Databases and Knowledge Bases

There are many public protein databases (see Section 3). We are holding several databases including such public ones as those shown at the bottom of Fig. 8.

An oval represents a module of rules and facts, while a rectangle represents a Kappa-P database. Modules in the upper two levels are mostly rules in *QUIXOTE*, while ones at the bottom are mostly facts in Kappa-P. The user may ask the top-level module any queries.

It can also be integrated with private databases and customized to be a private knowledge base. An example of such integration and customization is shown in Fig. 6.

6.2 Use of the System

Application of Sequence Analysis

Ishikawa et al. (ICOT) have developed a parallel processing algorithm of protein multiple alignment [Ishikawa *et al.* 92]. When the multiple alignment system and the knowledge base are connected, and a new multiple alignment algorithm using motifs is developed, it becomes an integrated application and knowledge base system. This is expected to enable automatic motif extraction and motif accumulation.

Advanced Query Processing

The query processing facilities of *QUIXOTE* realize a data pool of experimental results with query processing. They act as a prototype database or knowledge base for the experiment, which accumulates queries and shows the tendency of its usage in the integrated environment.

Graphical User Interface

The system has an user interface which allows it to use both an advanced query processing interface to *QUIXOTE* and a browsing and query-by-example interface for Kappa-P. The query interface provides or will provide facilities of displaying examples of queries, or graphs of answers such as the relations of objects given by a recursive query.

The browsing interface also provides or will provide graphical displaying facilities. We have developed a visual feature exhibition of sequences of both GenBank and PIR.

7 Conclusion

The requirements of molecular biology, especially protein engineering, which is a brand-new DBMS/KRL field were overviewed. Biological applications are now shown to be stimulating for DBMS and KRL, which are required to have various functions: information retrieval, deduction, identification, module concepts, extensibility, and parallel processing.

Such facilities of DBMS/KRL had better be requested by (computer-)biologists. It is important to cooperate with them to conduct further research.

A private knowledge base including various existing public databases will proceed biological knowledge discovery. Although we have not mentioned in this paper, distributed DBMS is also necessary in case databases and knowledge bases exceed the personal system capacity. We think DOOD with extensible DBMS also play an important role, but it will be considered in future.

Acknowledgments

The author wishes to thank Kazumasa Yokota, Hideki Yasukawa, Moto Kawamura and other people in the *QUIXOTE* project and Kappa-P project for their valuable comments on earlier versions of this paper. The author also wishes to thank the people on the computer-biology mailing list for their suggestions from the viewpoint of biology.

References

[Altshul *et al.* 90] Altshul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman, D.J.: "Basic Local Alignment Search Tool", *J. Mol. Biol.* 215, pp.403-410, (1990).

[Ishikawa *et al.* 92] Ishikawa, M., Hoshida, M., Hirosawa, M., Toya, T., Onizuka, K. and Nitta, K.: "Protein Sequence Analysis by Parallel Inference Machine" *FGCS 92*, (Jun 1992).

[JIPID 90] PIR-International (JIPID): *PIR Newsletter*, No.3 (June 1990).

[Kawamura *et al.* 92] Kawamura, M., Sato, H., Naganuma, K. and Yokota, K.: "Parallel Database Management System : Kappa-P" *FGCS 92*, (Jun 1992).

[NCBI 90] NCBI: "GenInfo Backbone Database", Version 1.59, *Draft* (Apr 1990).

[Tanaka 91] Tanaka, H.: "Protein Function Database as a Deductive and Object-Oriented Database", *Database and Expert Systems Applications*, Springer-Verlag, pp.481-486 (Aug 1991).

[Yasukawa *et al.* 92] Yasukawa, H., Tsuda H. and Yokota, K.: "Objects, Properties, and Modules in Quixote", *FGCS 92*, (Jun 1992).

[Yokota *et al.* 89] Yokota, K. and Tanaka, H.: "GenBank in Nested Relation", *Joint Japanese-American Workshop on Future Trends in Logic Programming* (Oct 1989).

[Yoshida *et al.* 91] Yoshida, K., Overbeek, R., Zawada, D., Cantor, C.R. and Smith, C.L.: "Prototyping a Mapping Database of Chromosome 21", *Proceedings of Genome Mapping & Sequencing Meeting*, Cold Spring Harbor Laboratory, (1991).