BAAQ: An Infrastructure for Application Integration and Knowledge Discovery in Bioinformatics

Xiujun Gong, Kensuke Nakamura, Hua Yu, Kei Yura, Nobuhiro Go

***Abstract*—The emerging grid computing technologies enable bioinformatics scientists to conduct their researches in a virtual laboratory, in which they share public databases, computational tools as well as their analysis workflows. However, the development of grid applications is still a nightmare for general bioinformatics scientists, due to the lack of grid programming environments, standards and high-level services. Here, we present a system, which we named Bioinformatics: Ask Any Questions (BAAQ), to automate this development procedure as much as possible. BAAQ allows scientists to store and manage remote biological data and programs, to build analysis workflows that integrate these resources seamlessly, and to discover knowledge from available resources. This paper addresses two issues in building grid applications in bioinformatics: how to smoothly compose an analysis workflow using heterogeneous resources and how to efficiently discover and re-use available resources in the grid community. Correspondingly an intelligent grid programming environment and an active solution recommendation service are proposed. Finally, we present a case study applying BAAQ to a bioinformatics problem.**

***Index Terms*—Active Solution Recommendation (ASR), Bioinformatics, Grid application, Task Mapping Editor (TME).**

I. Introduction

IN silico bioinformatics experiments involve integration of and access to computational tools and biological databases[1]. Bioinformatics scientists need to orchestrate a growing number of these resources to perform their analysis[2-4]. Bioinformatics is a collaborative discipline, in which bioinformatics scientists need to share their ideas and analysis methods not only through manuscripts, but also through information repositories, such as public databases, web services and analysis workflows[5]. The emerging grid technology[6], whose initiative is to enable scientists of similar interests to conduct their researches in a virtual organization, is forming the base of the new generation collaboration platform.

Although numerous grid middleware are being made available[7-9], the development of grid applications is still a nightmare for general bioinformatics scientists, due to the lack of grid programming environments, standards and high-level services. To ease the use of available resources without concerning the low level details of how the individual grid components operate, many researches have focused on studying high level services. Asia Pacific BioGrid (http://www.apbionet.org/grid/) project is trying to build a customized, self-installing version of the Globus Toolkit, a distributed environment for designing and managing grid. It comprises well-tested installation scripts and avoids dealing with Globus details. Bio-Grid working group[10] is developing an access portal for modeling biomolecular resources. The project develops various interfaces for biomolecular applications and databases that will allow biologists and chemists to submit works to high performance computing facilities, hiding grid programming details. However, both of them put more efforts on transparently accessing to a single resource, little on how to connect them to form a meaningful workflow as a whole. Additionally, scientists usually prefer learning development experiences from existing analysis, because the existing analysis workflows can tell users not only how programs and databases are used, but also how they are glued together. In some sense, knowing how the result is derived is more important than the result itself. To this end, myGrid[11] has implemented a service discovery framework using semantic description. Chimera[12] uses a virtual data language interpreter to derive data generation procedure. Preserv[13] allows developers to integrate process documentation recording into their applications, to search those documents using XQuery. However, to use their systems, users have to learn their specific languages, which are still difficult works for bioinformatics scientists.

In this paper, we present a system, which we named

Manuscript received October 17, 2005, revised on July 20, 2006

This work was supported in part by the Special Coordination Funds Promoting Science and Technology from Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan and IT Based Laboratory (ITBL) Project.

X. Gong is with the Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192 Japan and the School of Computer Science and Technology, Tianjin University, No. 92, Nankai, Tianjin, 300072, China. (email: gongxj@tju.edu.cn)

H. Yu is with the School of Computer Science and Technology, Tianjin University, No. 92, Nankai, Tianjin, 300072, China. (email: yuhua@tju.edu.cn)

K. Nakamura, K. Yura and N. Go are with Center for Promotion of Computational Science and Engineering, Japan Atomic Energy Research Institute, 8-1 Kizu, Souraku, Kyoto, 619-0215 Japan. N. Go is also with the first affiliation and the Neutron Science Research Center, Japan Atomic Energy Research Institute, 8-1 Kizu, Souraku, Kyoto, 619-0215 Japan (email: nakamura.kensuke@jaea.go.jp, yura.kei@jaea.go.jp , go.nobuhiro@jaea.go.jp)

"Bioinformatics: Ask Any Questions (BAAQ)". BAAQ allows scientists to access and manage biological databases and computation tools in the grid, and to ease building analysis workflows. The system is also capable of managing and sharing these workflows, and discovering knowledge from available resources. Specifically, BAAQ has the following features:

1) It is a flexible, grid-based infrastructure that allows integration of data and computational intensive applications;
2) It provides an intelligent grid programming environment that eases the works of composing analysis;
3) It provides a search engine called active solution recommendation (ASR) service that recommends workflow-based solution candidates considered as references or parts of new analysis for users' requirements through a natural language interface.

The reminder of this paper is organized as follows: Section 2 describes the system architecture forming functionality of BAAQ; Section 3 presents an intelligent grid programming environment; Section 4 introduces the active solution recommendation service; Section 5 reports a simple case study using BAAQ; and section 6 concludes this paper and discusses future works.

II. System Architecture

The components of BAAQ can be divided into four groups: communication infrastructure group, information service group, service assistant group and application resource group, as shown in Fig 1.

1) The communication infrastructure group provides a grid-based computational environment that is responsible for authentication, authorization, communication and computational resource location and allocation. Its implementation is based on Seamless Thinking Aid (STA) [14], which is IT-based Laboratory (ITBL) environment for assisting parallel programming. STA uses nexus developed at Argonne National Laboratory as communication library and can employ diverse communication protocols such as TCP/IP, AAL5 and MPL. STA provides a file manager through which users manage remote files as if they are in a single machine. The tool manager of STA allows managing and integrating grid-based middleware.
2) The information service group consists of grid middleware such as Task Mapping Editor (TME) [8], Active Solution Recommendation (ASR), TextBrowser, and PluginTool. These middleware modules interact with users through a GUI interface and communicate with STA by corresponding adaptors. TME (see Section III) is the core component for managing resources and building workflows. ASR (Section IV) recommends solution candidates by looking up existing resources. TextBrowser is used to browse the text-based content of data resources. PluginTool is designed to visualize the content of data resources using client side software.
3) The service assistant group provides a set of Application Program Interfaces (APIs) to the service group for helping accessing and managing application resources. Metadata management module is designed to collect and maintain a metadata repository which contains information about available resources and system configuration. Filter management module is responsible for manipulating filters that mediates data formats. Wrapped Program Toolkits (WPT) is used to help users encapsulate bioinformatics programs with uniform interfaces by utilizing filter repository. Active Service Provider (ASP) has the following three functions: 1) provide guidelines for choosing a resource and a set of parameters for program resources; 2) aid connecting resources in the process of workflow generation of TME; and 3) recommend a visualization tool (such as TextBrowser or PluginTool) for viewing analysis results.

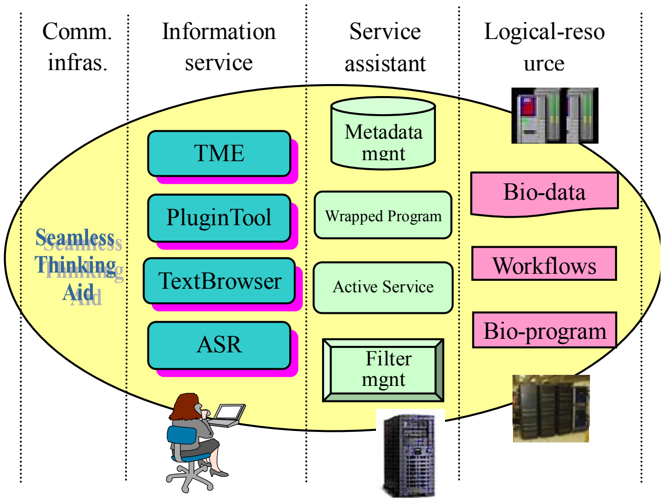


Fig 1 Architecture of BAAQ

4) The application resource group consists of data and program distributed on heterogeneous computers and their descriptions, as well as workflows that describe how data and programs are connected to perform a certain analysis. Hereafter, we call these three kinds of resources as bio-resources. The information service group accesses these resources by the help of service assistants.

In this architecture, the biological data and analysis programs (and their replicas) distributed on different computation nodes are registered through TME. These resources are encoded with knowledge rich metadata and are stored under the tree-like structure of TME workspaces. Before registering the analysis tools, they are wrapped so that they have uniform interfaces. To compose an analysis workflow, a user only needs to select corresponding resources and link them by drawing a line between them. ASP is helpful for this procedure as described above. The analysis results are usually stored as remote files, managed as icons in the TME workspace, and can be viewed using customized visualization tools such as TextBrowser and PluginTool. An alternative way for building a new analysis is to use ASR. ASR recommends solution candidates by looking up available resources. Once a user finds candidates relevant to his/her requirements, one can import them based on a certain proxy and reconfigure their parameters to meet his/her specific needs.

III. An Intelligent Grid Programming Environment

Task Mapping Editor (TME) [8], one of the components of STA, is a visual programming tool developed at the Center for Promotion of Computational Science and Engineering at Japan Atomic Energy Research Institute. TME has been developed for handling distributed resources and supporting the integration of distributed applications. All the resources are represented as icons on TME workspaces and data dependency is defined by a line linking the icons. Using TME, a user can design a workflow diagram of the distributed applications, just like using a drawing tool. We equip TME with the following functions to handle bioinformatics problems:

A. *Scalable filter library*

Data format transformation is a tedious work for bioinformatics scientists; hence it is becoming an active research area in bioinformatics. Two typical methods are data warehousing and federation[15], both of which adopt SQL-like languages as interfaces with end users. Our method, data filter approach, is quite similar to the federation method in principle. Instead of providing SQL interfaces, we use the filter manager [Fig 2] to call corresponding filters to perform intended transformation.

We define a uniform XML schema with enough capability of describing biological data sources and typical analysis results of bioinformatics tools. To exchange different data formats with the defined XML format, we design a set of filters (Fig 2). Upon a user's requirement, the filter manager accesses corresponding filters to perform intended transformation. Using this filter library, the data filter service can transform formats between any two different data sources. The filter library also contains two other classes of filters. One is the filter that extracts some results from program outputs into XML format and is used to wrap programs. The other is to convert some XML analysis results into PostScript format or RasMol script format and is used in the visualization module. Each filter has the name SOURCE2XML, XML2SOURCE (SOURCE represents the name of the individual source) so that the filter manager can identify it easily. The filter is coded using C++ and complied and distributed to all the computation nodes. The system will select an appropriate one at the time of workflow execution. One of the advantages of this architecture is that developers only need to change the individual filter when a data source is changed.

B. *Wrapped Program Toolkits*

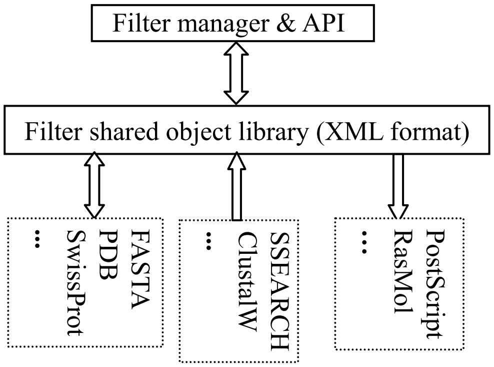
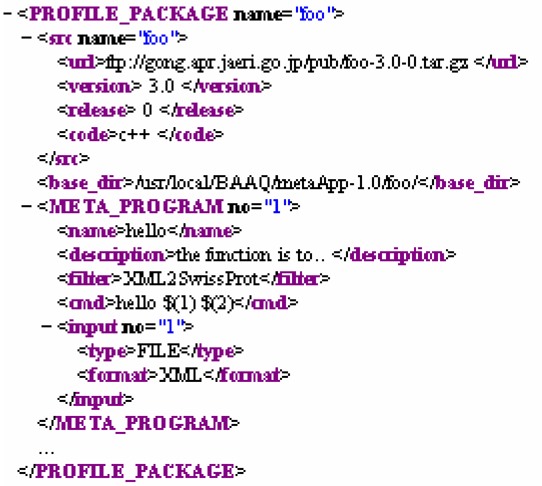
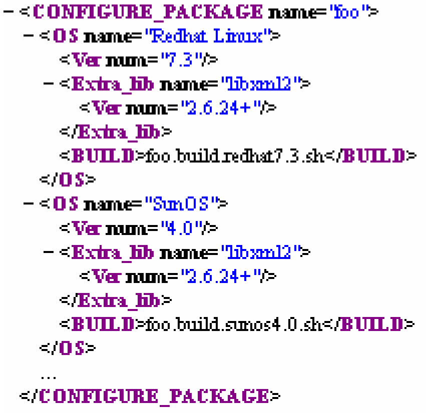


Fig 2 structure of the filter library



```
- <PROFILE_PACKAGE name="foo">
  - <src name="foo">
      <url>ftp://gong.apr.jaeri.go.jp/pub/foo-3.0-0.tar.gz </url>
      <version> 3.0 </version>
      <release> 0 </release>
      <code>c++ </code>
    </src>
    <base_dir>/usr/local/BAAQ/metaApp-1.0/foo/</base_dir>
  - <META_PROGRAM no="1">
      <name>hello</name>
      <description>the function is to.. </description>
      <filter>XML2SwissProt</filter>
      <cmd>hello $(1) $(2)</cmd>
    - <input no="1">
        <type>FILE</type>
        <format>XML</format>
      </input>
    </META_PROGRAM>
    ...
  </PROFILE_PACKAGE>
```

Fig 3 a sample for a program profile



```
- <CONFIGURE_PACKAGE name="foo">
  - <OS name="Redhat Linux">
      <Ver num="7.3"/>
    - <Extra_lib name="libxml2">
        <Ver num="2.6.24+"/>
      </Extra_lib>
      <BUILD>foo.build.redhat7.3.sh</BUILD>
    </OS>
  - <OS name="SunOS">
      <Ver num="4.0"/>
    - <Extra_lib name="libxml2">
        <Ver num="2.6.24+"/>
      </Extra_lib>
      <BUILD>foo.build.sunos4.0.sh</BUILD>
    </OS>
    ...
  </CONFIGURE_PACKAGE>
```

Fig 4 a sample for the configuration of a program

TME organizes a program as an icon with configurable interfaces. Its function and semantics of its interfaces are still hidden from users. Users have to shift more efforts for studying the function and understanding the meaning of each interface. Most bioinformatics tools are developed by different researchers or commercial units. The data formats that programs can accept are quite different from one another. In BAAQ, each program is associated with two descriptive files: profile (Fig 3) and configuration(Fig 4). The profile encodes three kinds of information: 1) source information about the program, for example the URL, from which the program is downloaded, version, release number, and code types(C/C++, Perl, Fortran and so on), is used to maintain consistence of different versions; 2) function summary information, for example analysis types (sequence alignment, structure prediction), methods and performance evaluation, function description, for human understanding or retrieval purpose; and 3) function and interface description of each program for semantic check. The configuration file ensures that programs in the package are executable on different platforms (Linux, SunOS and IRIX and so on). Fig 3 and Fig 4 list samples for profile and configuration of a program "foo".

Program interfaces play a particular role in composing analysis workflows. Heterogeneity and diversity in program interfaces exhausts most users. To solve this problem, we classify the interfaces into three types: switch, real-value and file parameters. We can list all possible values for the switch parameters and limit the maximum and minimum for the real-value parameters. For the file parameters, the situation becomes a little more complex, because of the diversity in biological data formats. Our solution is to unify data formats of the file parameters as XML formats as much as possible. Since XML is more flexible to develop machine understandable code, we encode related information into XML files so that the system can identify them automatically. To this end, we design many filters that extract related information from the original outputs of the analysis tools into XML files [Fig 2]. Using these filters, the programs are wrapped easily.

C. *Visualization for analysis procedure / result*

Visualization plays a vital role in understanding the analysis procedures and results. Using the graphically-enabled task editor environment in TME, a user can visually design and view the workflow, monitor its execution status and trigger corresponding tools to browse the content of an individual icon in the workflow. In the visualization module, we focus on the visualization of analysis results through triggering the third party visualization tools. There is a large amount of software for presenting biological data graphically, for example, Protein Explorer, RasMol, Chime for protein structure, gsview and GNU gv for many kinds of printable pictures and GnuPlot and SciLab for statistical data. To integrate those kinds of programs, we face two challenging problems: 1) Although most of them have different versions working on different platforms, for example RasMol can run on Windows, Mac and Unix-based platform, few of them are platform-independent; and 2) There are many programs for visualizing the biological data in a similar way. However, users usually prefer the one they are most accustomed to.

In BAAQ, users can use the programs hosted in client side by the way of Plugin, which most web browsers provide. What the system does is to provide targeted data and to recommend content type, by which the browser decides which program on the client side will be triggered. Once content type is identified, Active Service Provider will call the corresponding filter (Fig 2) to transform the XML file into the corresponding format (i.e. PostScript, RasMol), then trigger the PluginTool to view the data. In this way, users can customize their preferred visualization tools.

D. *Active service provider*

TME has already provided abundant utilities for composing and managing a workflow with distributed resources. Each resource is represented as an icon. Building a connection between different resources is just to draw a line between icons. For specific application areas, however there are still some problems: 1) What data and programs should a user choose to perform a specific bioinformatics analysis? 2) How does a user keep semantic consistency when building a connection between two icons? For example what would happen if a user connects two icons with heterogeneous formats?

For the first problem, we provide a tree-based browser for biological databases and tools. In TME window, each data or program resource is represented as an icon under the corresponding taxonomy tree. A user can search for the target objects by traveling through the taxonomy tree. Also one can use ASR discussed in the next section to search for the available resources by a natural language interface.

For the second problem, although TME can check some syntactic errors for the whole workflow, little emphasis is put on semantics. We provide a method, called Active Service Provider, for solving the problem. Our Active Service Provider includes two procedures: semantic check and service provider. When a user builds a connection, Active Service Provider will be triggered to check whether this connection is admissible by examining the corresponding profile, which is an XML formatted file to describe resource's attributes, such as public interfaces, accessibility and functionality. If allowed, the system will detect where extra services are needed. For example, if a user wants to connect two icons with different data formats, the data filter service will be provided to perform data format transformation.

IV. AN ACTIVE SOLUTION RECOMMENDATION SERVICE

In the previous section, we addressed the issue how the bio-resources are glued together to compose an analysis workflow. The issues we address here include how to choose the available bio-data and program resources and how to discover and reuse existing workflows for the developers of grid applications. We believe that existing workflows enclose abundant knowledge. By investigating these workflows, we can, not only learn how bio-resources are used, but also know how the analysis results are derived. In some sense, knowing how the result is derived is more important than the result itself. We propose a prototype called Active Solution Recommendation (ASR) to solve these problems. As the name suggests, the goals of ASR are to discover the bio-resources, to allow the reuse of the most relevant workflows, and to recommend solution candidates for user's questions.

Although the architectures, contents and services to be provided in grid information retrieval (GIR)[16], are still not well defined in the grid research community, we believe that discovering workflows and easing the deployment of retrieved workflows should gain special priority in the evolution of GIR.

The implementation of ASR is based on mature web searching and data mining technologies. But it distinguishes the traditional search engines by two aspects:

1) Compared with the great diversity of web contents, its sources are well organized, so that it allows more efficient

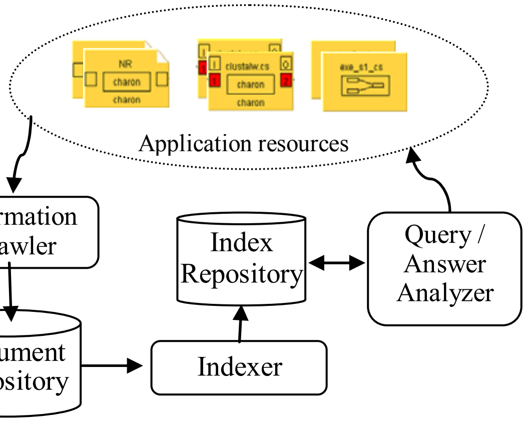


Fig 5 Architecture of ASR

search algorithms.

2) ASR is not only responsible for finding the targeted resources, but also provides a set of utilities for manipulating the candidates to be shared and reused in the grid environment. However, the concern of a general web search engines focuses on how to locate the sources. It is a user's task to find out how to use them.

The overall architecture of ASR is shown in Fig5. Its main modules are Information Crawler, Indexer and Query/Answer Analyzer.

A. *Information Crawler module*

The Information Crawler module is responsible for automatically retrieving bio-resources into a central repository. In BAAQ, each bio-resource is described by an XML file and associated with a Uniform Resource Identifier (URI). And each bio-data or program resource binds to at least a source identified by a Uniform Source Identifier (USI). The system will select a suitable USI at the runtime of a workflow. A workflow links other resources together for performing an analysis. Intuitively, if a bio-resource is referred frequently, probably it is more likely to be significant than others. Thus, the linking information here plays the role as a hyperlink does in traditional search engines, such as Google. The system also allows users to make notes based on their opinions about bio-resources' capabilities. The Information Crawler collects information above into a document repository.

B. *Indexer module*

The Indexer module is responsible for building and maintaining the index data structure of ASR. This step is very important, because some information about the relevance of bio-resources within the grid must be integrated and indexed for efficient search and discovery. The information we consider to be important includes:

1) functional information;
2) annotation information;
3) linking information.

The functional information provides a more complex but precise description of functionality offered by the bio-resources. For instance, the functional information of a bio-data resource describes the characteristic of bio-data such as data format, published method, and a simple abstract showing how the data is generated. The description of bio-program resource includes its published method, copywriter information, classification in TME workspace and a profile showing how the program works. For the case of a workflow, the situation becomes more complex due to the complexity in its components. We propose a workflow description framework through using a controlled vocabulary, the similar idea which has been used to describe experimental procedures and common process in the Human Proteome Organization Protein Standards Initiative [17]. The organization of our vocabulary follows a hierarchical structure. It provides a clear and comprehensive functional description of a workflow.

The annotation information involves the evaluation from different users who have obtained and used the resource.

The linking information regards how the resource is referred by other resources and who have visited the resource. All the information is used to provide the significance of its quality.

The Indexer builds an index repository by parsing and weighting the information above.

C. *Query/Answer Analyzer module*

The third module of ASR is the Query/Answer Analyzer, which actually solves a user's requirement based on the index repository that has been built in the Indexer module. This module takes a user's query as well as his/her preference, which has been captured from his/her query history, as input, and outputs a ranked list of candidate answers.

Specifically, the similarity between a bio-resource and a user's query is measured by the cosine of a dot product between them, which can be calculated using the following formulas:

For a data/program resource,

$$<Q,R>=<w_1Q+w_2P,\ R> \quad (w_1+w_2=1,\ w_1,\ \ w_2>0)$$

where Q, R , P are the vector representations of a query, bio-resource description and user's preference, w_1 , w_2 are experience weights for Q, P(here w_1=0.8, w_2=0.2).

For a workflow resource

$$<Q,R>=<w_1Q+w_2P,\ \lambda_0 R_0+\sum_{i=1}^{n}\lambda_i R_i>$$

Where R_0 is the vector representation of the workflow description, R_i (i=1…n) is the vector representation of the description of the component consisting of the workflow, λ_0, λ_i (i=1…n) are the experience weights (here, λ_0=0.6, λ_i=0.4/n).

The system also provides a set of utilities for manipulating the results. Specifically, the services provided include:

1) annotation service to enable a user to evaluate the candidate from the user's view;
2) reservation service to enable a user to make a reservation of the candidate so that she/he can import it;
3) authorization service to enable a user to authorize a candidate that has been reserved so that other users can import it;
4) import/export services to enable a user to import an authorized candidate and to export a reserved one. The import/export service is of great importance in the sense of collaborations among users, because this step makes

resources really shareable and reusable.

Once a user imports a workflow, the system will deploy all the components automatically. Also he/she can change their parameters or use it as a part of new analysis to meet his/her specific needs.

V. Case study

One of the most important tasks in the current field of bioinformatics is to predict the functions of proteins encoded in genome sequences. About 40 percent of proteins predicted from genome sequences are unknown function. *Deinococcus radiodurans* is well-known for its resistance to high dosage gamma ray irradiation [18], but the molecular mechanisms for this resistance remained to be disclosed. There must be mechanisms which respond to irradiation, because any DNA damage which results from irradiation is soon fixed in a while.

To find clues for these molecular mechanisms involved in the resistance out of proteome data, three programs: *garnier*, *helixturnhelix* and *pepcoil* in EMBOSS[19] package are wrapped as program resources, each of which has an input file and an output file in XML formats. They are distributed over three different machines: itbltasv (SunOS 5.8), charon (Red Hat 7.3) and gong (Red Hat 9.0). A user can select the available machines and manage remote files on them, as shown in Fig 6A. The final workflow to perform this analysis is shown in Fig 6B. In this workflow, the input sequence with SwissProt format is first transformed into XML format using the bioDatafilter service, which is triggered by Active Service Provider. Then the output is connected to three branches, each of which is processed by one of the three programs above. Finally, their outputs are transformed into three postscript files by the ps_maker service, which is a filter service designed for transforming XML format into postscript format. A user can browse the results graphically by just double clicking the results icons, which will trigger the PluginTool.

The results are shown in Fig 7. One of the proteins (the 1791st protein from the origin of the bacterium genome), which consists of 139 amino acid residues and is annotated as a hypothetical protein, is shown to have a helix-turn-helix motif, one of the well-known motifs for transcription factors (Fig 7B), and a coiled-coil region, one of the well-known structures for protein-protein interactions (Fig 7C). The helix-turn-helix motif and coiled-coil structure both consist of helix structures, and the regions predicted to have the motif are predicted as helix structures (Fig 7A). The simultaneous application of the genome annotation tools shows the consistency in the predictions. The regions predicted to have coiled-coil structures contains helix-turn-helix motifs (Fig 7B and Fig 7C). This overlap is, to our knowledge, quite a rare event. The result suggests that the region functions as a helix to interact with DNA and other proteins.

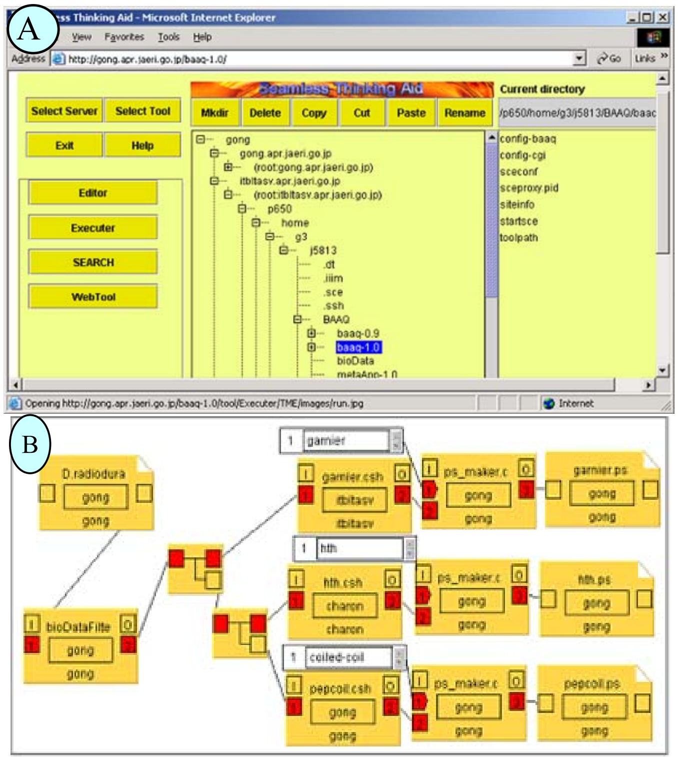


Fig 6 (A) Screen shot showing the configuration of grid environment; (B) workflow of the structure prediction for *D. radiodurans* proteome.

VI. Conclusion and future direction

In this paper, we have presented an integrated framework for building grid applications in bioinformatics. In a grid-based environment, we developed an intelligent grid programming environment for composing an analysis workflow smoothly that integrates distributed and heterogeneous resources. Another contribution of this paper is the development of Active Solution Recommendation service, which allows users to search for and reuse the bio-resources they queried. This service is based on matured web search technologies and provides a set of utilities for manipulating results so that users

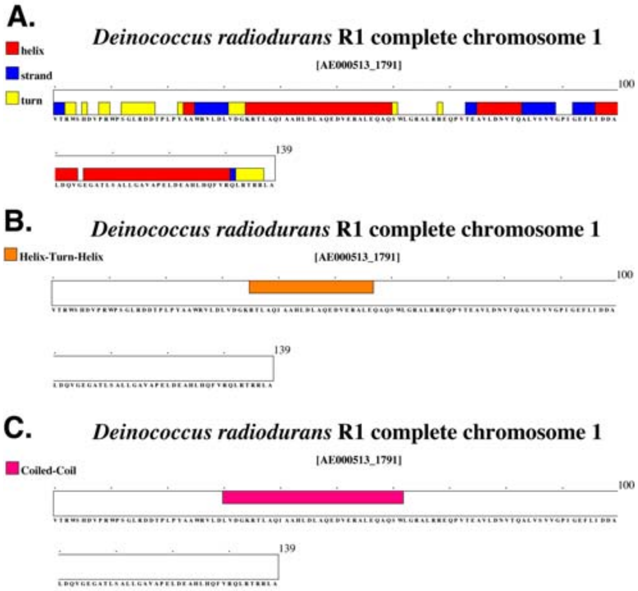


Fig 7 Results of structure prediction: (A) secondary structure; (B) helix-turn-helix structure; and (C) coiled-coil structure

can share and exchange their analysis. We believe that in the near future there will be a growing demand for this kind of services.

A prototype of the system has been implemented mainly using c++ for server side and java for client side on Redhat 9.0. Also the system has been successfully tested across SunOS4.0, FreeBSD 6.1 and Fedora Core 5. Currently, the system includes about 18 filters in the filter library, 30 bioinformatics tools and over 40 analysis workflows.

One of our future works would be to incorporate the ontology approach, which have been applied successfully in many bioinformatics applications[15, 20], into our framework. Clearly, an appropriate ontology is helpful for both the uniform access to heterogonous databases and the semantic description of bio-resources. Another future work is to refine the solution candidates of ASR. Ideally we expect that ASR can generate a new meaningful analysis for a user's query.

VII. Acknowledgement

Authors are grateful to Dr. Norihiro Nakajima, Dr. Yoshio Suzuki, Mr. Yukihiro Hasegawa and Mr. Nobuhiro Yamagishi.

VIII. References

[1] C. Wroe, C. Goble, M. Greenwood, P. Lord, S. Miles, J. Papay, T. Payne, and L. Moreau, "Automating Experiments Using Semantic Data on a Bioinformatics Grid," *IEEE Intelligent Systems*, vol. 19, pp. 48-55, 2004.

[2] J. Blythe, E. Deelman, and Y. Gil, "Automatically Composed Workflows for Grid Environments," *IEEE Intelligent Systems*, vol. 19, pp. 16-23, 2004.

[3] Y. Gil, E. Deelman, J. Blythe, C. Kesselman, and H. Tangmunarunkit, "Artificial Intelligence and Grids Workflow Planning and Beyond," *IEEE INTELLIGENT SYSTEMS*, vol. 19, pp. 26-33, 2004.

[4] M. Cannataro, C. Comito, F. L. Schiavo, and P. Veltri, "Proteus, a Grid based Problem Solving Environment for Bioinformatics: Architecture and Experiments," *The IEEE Computational Intelligence Bulletin*, vol. 3, pp. 7-18, 2004.

[5] K. H. Buetow, "Cyberinfrastructure: Empowering a 'Third Way' in Biomedical Research," *SCIENCE*, vol. 308, pp. 821-824, 2005.

[6] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Proc*. Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing, pp. 1-4, 2001.

[7] A. Sulistio, G. Poduvaly, R. Buyya, and C. K. Tham, "Constructing A Grid Simulation with Differentiated Network Service Using GridSim," *Proc*. Proceedings of the 6th International Conference on Internet Computing (ICOMP'05), Las Vegas, USA, pp., 2005.

[8] T. Imamura, N. Yamagishi, H. Takemiya, Y. Hasegawa, K. Higuchi, and N. Nakajima, "A Visual Resource Integration Environment for Distributed Applications on the ITBL System," *Proc*. ISHPC 200, Tokyo, Japan, pp. 258-268, 2003.

[9] A. Rowe, D. Kalaitzopoulos, M. Osmond, M. Ghanem, and Y. Guo, "The Discovery Net System for High Throughput Bioinformatics," *Bioinformatics*, vol. 19, pp. i225-i231, 2003.

[10] J. Pytlinski, L. Skorwider, P. Bala, M. Nazaruk, and K. Wawruch, "BioGRID-Uniform Platform for Biomolecular Applications," *Proc*. Euro-Par200, Paderborn, Germany, pp. 881-884, 2002.

[11] S. Miles, J. Papay, C. Wroe, P. Lord, C. Goble, and L. Moreau, "Semantic Description, Publication and Discovery of Workflows in myGrid," Electronics and Computer Science, University of Southampton. ECSTR-IAM04-001, 2004.

[12] I. T. Foster, J. Vöckler, M. Wilde, and Y. Zhao, "Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation," *Proc*. the 14th International Conference on Scientific and Statistical Database Management, Edinburgh UK, pp. 37-46, 2002.

[13] P. Groth, S. Miles, and L. Moreau, "PReServ: Provenance Recording for Services," *Proc*. Proceedings of the UK OST e-Science second All Hands Meeting, Nottingham,UK, pp., 2005.

[14] H. Takemiya, T. Imamura, and H. Koide, "Development of a Software System (STA: Seamless Thinking Aid) for Distributed Parallel Scientific Computing," *IPSJ MAGAZINE*, vol. 40, pp. 1104-1109, 1999.

[15] Z. Lacroix, "Biological Data Integration: Wrapping Data and Tools," *IEEE Trans Inf Technol Biomed*, vol. 6, pp. 123-128, 2002.

[16] N. Nassar, G. B. Newby, M. J. Dovey, and J. Morris,"Grid Information Retrieval Architecture", 2003, http://www.gir-wg.org/papers/Grid_Information_Retrieval_Architecture.pdf.

[17] S. Orchard, L. Montecchi-Palazzi, H. Hermjakob, and R. Apweiler, "The Use of Common Ontologies and Controlled Vocabularies to Enable Data Exchange and Deposition for Complex Proteomic Experiments," *Proc*. Pacific Symposium on Biocomputing, Hawaii. USA, pp., 2005.

[18] M. J. Daly and K. W. Minton, "Resistance to Radiation," *Science*, vol. 270, pp. 276-277, 1995.

[19] P. Rice, I. Longden, and A. Bleasby, "EMBOSS: The European Molecular Biology Open Software Suite," *Trends in Genetics*, vol. 16, pp. 276-277, 2000.

[20] P. G. Bakera, A. Brassa, S. Bechhoferb, C. Gobleb, N. Patonb, and R. Stevensb, "Transparent Access to Multiple Bioinformatics Information Sources," *Bioinformatics*, vol. 16, pp. 184-185, 2000.

Xiujun Gong received the Ph.D. degree in computer science from Institute of Computing and Technology, Chinese Academy of Science, Beijing, China, in 2002.

He ever worked as a research fellow at the Department of Computer Science in National University of Singapore in 2002. Then, he worked as a postdoctoral fellow at Nara Institute of Science and Technology from 2003 to 2006. Now he is an associate professor at the School of Computer Science and Technology, Tianjin University, Tianjin, China. His research interests include Bioinformatics, data mining and grid computing.

Kensuke Nakamura received the Ph. D. degree in Chemistry from Keio University, Yokohama in 1990.

He had been a Researcher at the Chemistry department of UCLA (USA), then Key Molecular Inc. a drug-design company in Tokyo, Japan. Since then he has been working at Nara Institute of Science and Technology (Japan) and Japan Atomic Energy Agency, where his research has been focused on Bioinformatics. He has been mainly working on a project of structural and sequence analysis of metal-binding proteins, with developing computational tools for the efficient bioinformatics analysis

Hua Yu received the Master degree in Geography Information System (GIS) from Shandong University of Science and Technology, Tai'an, China, in 1999.

She has worked in a government agency for two years and industry for a year. Now she is an engineer in the School of Computer Science and Technology, Tianjin University, Tianjin China. Her research interests include GIS, information system design, and network engineering.

Kei Yura received the Ph.D. degree in Science in 1999 from Nagoya University, Nagoya, Japan.

He has been working at Quantum Bioinformatics Team, Japan Atomic Energy Agency at Kizu, Kyoto, Japan. He is now involved in building an integrated platform of bioinformatics tools and databases. His research interests cover a number of areas in computational biology including elucidation of mechanisms of how proteins interact with RNA molecules, how proteins form supramolecules, and how protein evolves to the current structure.

Nobuhiro Go received the Ph.D. degree in physics from the Tokyo University in 1966. His major field of study has been theoretical biophysics

He worked first as assistant professor at the Tokyo University, then as associate professor at Kyushu University, and further as professor at Kyoto University. Now he works as scientific advisor at Japan Atomic Energy Agency. He once worked as vice president of IUPAB (International Union of Pure and Applied Biophysics), as chairman of Commission for Biological Physics, IUPAP (International Union of Pure and Applied Physics), as head of Science Division, Science Council of Japan.