Flexible, efficient and interactive retrieval for supporting in-silico studies of endobacteria

Stefania Montani, Giorgio Leonardi
Dipartimento di Informatica
University of Piemonte Orientale
Alessandria, Italy
stefania.montani, giorgio.leonardi@unipmn.it

Stefano Ghignone, Luisa Lanfranco
Dipartimento di Biologia Vegetale
University of Turin
Turin, Italy
stefano.ghignone,luisa.lanfranco@unito.it

Abstract— **Studying the interactions between arbuscular mycorrhizal fungi (AMFs) and their symbiotic endobacteria has potentially strong impacts on the development of new biotechnology applications. The analysis of genomic data and syntenies is a key technique for acquiring information about phylogenetic relationships and metabolic functions of such organisms.**

In this paper we describe a case-based retrieval tool, which supports customized comparative genomics searches, and which is part of a modular architecture meant to support in-silico genome sequence analysis, being developed within the project BIOBITS. From a methodological viewpoint, the tool takes advantage of an abstraction technique similar to Temporal Abstractions, thus allowing to neglect un-relevant details. Retrieval is made flexible by the use of such multi-level abstractions, and efficient by the use of proper taxonomical index structures. Moreover, end-users are allowed to progressively relax or refine their queries, in an interactive way. A case study taken from the application domain is used to illustrate the approach.

Keywords: Case-Based Reasoning, Flexible and Interactive Retrieval, Comparative Genomics

I. INTRODUCTION

Arbuscular mycorrhizal fungi (AMFs) are obligate symbionts which, to complete their life cycle, must enter in association with the root of land plants. Here, they become a crucial component of soil microbial communities, and exert positive impacts on plants health and productivity: in particular, they furnish a better mineral nutrition, and act as a means for increasing tolerance to stress conditions [1,2]. AMFs are thus a significant resource for sustainable agriculture; in addition, they could also be exploited as a still unknown resource to promote green (agriculture) and white (industrial) biotechnologies.

AMFs are often in further symbiosis with uncultivable bacteria, living inside the AMF itself [3]. The resulting tripartite system (i.e. (i) endobacterium; (ii) AMF; (iii) plant roots) is a complex biological object, whose extensive study requires a comparative genomics approach, in order to answer fundamental questions concerning the biology, ecology and evolutionary history of the system and of its composing elements. As a matter of fact, comparative genomics represent a key instrument to discover or validate phylogenetic relationships, to give insights on genome evolution, and to infer metabolic functions of a given organism, and is particularly useful when biochemical and physiological data are not available and/or hard to obtain.

While bacterial endosymbionts in the animal kingdom are excellent models for investigating important biological events, such as organelle evolution, genome reduction, and transfer of genetic information among host lineages [4], examples of endobacteria living in fungi are limited [5]. A key part of the study about the tripartite system mentioned above is therefore represented by the analysis of the genomic data of the endobacteria themselves. In particular, large-scale analysis and comparison of genomes belonging to phylogenetically related free-living bacteria can provide information about the events that led to genome down-sizing, and insights about the reason of the strict endosymbiotic life style of the bacteria at hand.

In this paper, we present the design of a computational environment for the genomic study of the AMF *Gigaspora margarita* (isolate BEG34) and of its endobacterium *Candidatus* Glomeribacter gigasporarum, which are currently used as a model system to investigate endobacteria-AMFs interactions. In particular, we are developing a modular architecture, composed by a database, in which massive genomic data are imported and stored, and by genomic comparison (synteny) and visualization tools.

In the implementation we are exploiting as much as possible the available tools built around GMOD, the Generic Model Organism Database project [6], which brought to the development of a whole, and still under expansion, collection of open source software tools for creating and managing genome-scale biological databases. In particular, we have chosen to resort to the GMOD database Chado [7], and to some freely available and freely adaptable GMOD facilities to search for syntenies like CMap, GBrowse_syn, SyBil (see section II).

However, we are extending the functionalities offered by GMOD, in order to properly meet the needs of our specific comparative genomics research.

In particular, we are working at a genome search and comparison tool, implementing the *retrieval* step of the Case-Based Reasoning (CBR) methodology [8]. Our tool allows to search in the genomes database by expressing

queries at different levels of abstraction detail, resorting to a technique similar to Temporal Abstractions (TA) [9,10]. This makes retrieval very *flexible*. Moreover, end-users are allowed to progressively relax or refine their queries, in an *interactive* way. Finally, retrieval is made *efficient* by the use of multi-dimensional orthogonal index structures, which allow for early pruning and focusing. The description of such a tool is the main objective of the current paper.

The work is supported by the BIOBITS project, a grant of Regione Piemonte, under the Converging Technologies Call, which involves the University of Turin, the University of Piemonte Orientale, the IPP-CNR and the companies ISAGRO Ricerca s.r.l., GEOL Sas, Etica s.r.l.

The paper is organized as follows: section II sketches the general architecture we are implementing for genome analysis in the project; section III focuses on our retrieval tool; section IV discusses related works, and section V is devoted to conclusions.

II. System Architecture

In our project, genome sequence analysis is supported by a modular architecture, which permits:

- to store and access locally all the information regarding the organisms to be studied, and
- to provide algorithms and user interfaces to support the researchers' activities (e.g.: search and retrieval of genomes, comparison and alignment with a reference genome, investigation of syntenies and local storage of potential new annotations).

The system architecture has been engineered exploiting the standard modules and interfaces offered by the GMOD project [6], and completed with custom modules to provide new functionalities (see Figure 1).

The main module of the system contains the database which provides all the data needed to perform the in-silico activities. We adopted the Chado database schema [7], to take advantage of its completeness and of its support for controlled vocabularies and ontologies. Furthermore, Chado is the standard database for most of the GMOD modules, therefore we can reuse these modules to support the main activities of the project, and extend the system incrementally as the researchers' needs evolve. The database in this module stores and provides all the information about the organisms to be studied (mainly bacteria), their genomes, their known annotations, their proteins and metabolic pathways, and the newly discovered annotations, which can be stored and managed locally until they are confirmed and published.

As explained, our database contains information to be used and stored locally, but we have added the possibility to populate and update the database with information retrieved from the biological databases accessible through the Internet. This feature is provided by the set of modules in the *Import modules* section (see Figure 1). The main module (RRE - Queries), which is built on the basis of a previously published tool [11], performs queries to different biological databases through the Internet (e.g. the GenBank [12]) and converts the results into a standard format. Afterwards, the Import module inserts or updates the retrieved information into the Chado database. This process can be started on-demand, or performed automatically on a regular basis, in order to maintain the local database up-to-date.

Chado also acts as the data interface for the software layers implementing the functionalities and tools used by the researchers. From the architectural point of view, we offer two types of services: the services implemented through existing modules of GMOD (*GMOD modules* section in Figure 1), and new services implemented through new modules, developed ad-hoc (*New applications* section).

The latter is composed, at the time of writing, of:

- a set of clustering and other data mining modules (whose description is outside the scope of this paper). Such tools rely on BioMart [13], a GMOD facility able to reorganize the information stored in the Chado database into a data warehouse;
- a flexible retrieval module, described in section III, which supports efficient and interactive retrieval in the context of the search for genomic similarity.

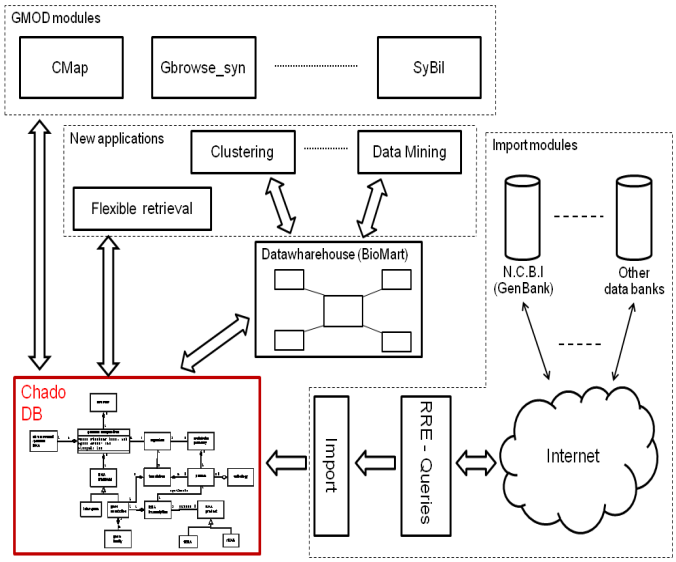

Fig. 1. System architecture.

The existing GMOD modules, on the other hand, exploit the Chado database to provide the researchers with additional techniques for comparative genomics needed in the BIOBITS project. In particular:

- CMap allows users to view comparisons of genetic and physical maps. The package also includes tools for maintaining map data;
- GBrowse is a genome viewer, and also permits the manipulation and the display of annotations on genomes;

- GBrowse_syn is a GBrowse-based synteny browser designed to display multiple genomes, with a central reference species compared to two or more additional species;
- SyBil is a system for comparative genomics visualizations.

All the tools in this system use a web-based interface to be more user-friendly and easy to use. Many GMOD modules can be reused as they are, but they can be customized to meet the researchers' recommendations before being integrated in our software architecture. Furthermore, every new module added in the *New application*s section of our architecture, or every customized module in the *GMOD modules* section, connects to the other modules of our architecture using GMOD standard interfaces. Therefore, every new or customized module can be published to the GMOD community, in order to extend and enrich this platform.

III. Flexible and Interactive Retrieval of Similar Genomes

The flexible retrieval module we have designed in the project architecture implements the retrieval step of the CBR [8] cycle. CBR is a reasoning paradigm that exploits the knowledge collected on previously experienced situations, known as *cases*. The CBR cycle operates by:

1. *retrieving* past cases that are similar to the current one and by
2. *reusing* past successful solutions after, if necessary, properly
3. *adapting* them; the current case can then be
4. *retained* and put into the system knowledge base, called the *case base*.

Purely retrieval systems, leaving to the user the completion of the reasoning cycle (steps 2 to 4), are however very valuable decision support tools [14], especially when automated adaptation strategies can hardly be identified, as in biology and medicine [15]; in BIOBITS we are following exactly this research line.

In the flexible retrieval module, the information stored in cases is related to genomes expressed as sequences of nucleotides, each one taken from a different organism, and properly aligned with the genome of a reference organism.

Completing the alignment task is therefore a prerequisite for representing cases in our library.

Details of our alignment strategy, and of case representation, are described in sections A and B, respectively. Case retrieval is then described in section C.

A. Sequence alignment

In our approach we rely on BLAST (http://blast.ncbi.nlm.nih.gov/Blast.cgi) to deal with the alignment task. BLAST is in fact a state-of-the-art local alignment algorithm, specifically designed for bioinformatics applications.

BLAST can take as an input any type of nucleotide sequence, i.e. a whole chromosome or plasmid, contigs or single genes, and properly aligns it to a database of strings belonging to (different) organisms of interest.

From a typical BLAST output (figure 2) one can extract basic information (% of similarity and length of sequence alignment) that can be easily plotted as represented in figure 3, providing a piecewise constant function, which graphically represents the alignment itself.

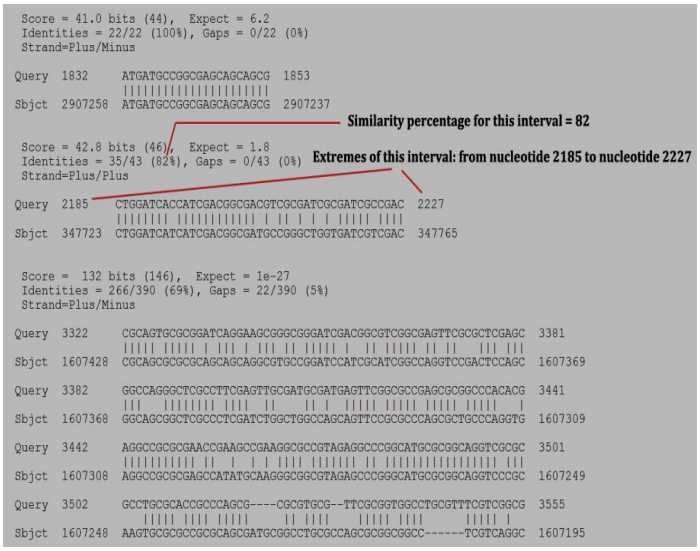

Fig. 2. BLAST sequence alignment.

B. Case representation

Quantitative similarity values provided by BLAST can be converted into a set of qualitative levels (e.g. low, medium, high similarity), thus providing a "higher level" view of the information, able to abstract from unnecessary details. To perform such a conversion, we propose a semantic-based abstraction process, similar to the Temporal Abstractions (TA) techniques [9,10].

TA is an Artificial Intelligence methodology which allows to move from a *point-based* to an *interval-based* representation of a time series, where time series points are converted into intervals (*episodes*), aggregating adjacent points sharing a common behavior, persistent over time. In particular, *state* abstractions [10] allow to extract episodes associated with qualitative levels of the variable represented by the time series, where the mapping between qualitative abstractions and quantitative values has to be parameterized on the basis of domain knowledge.

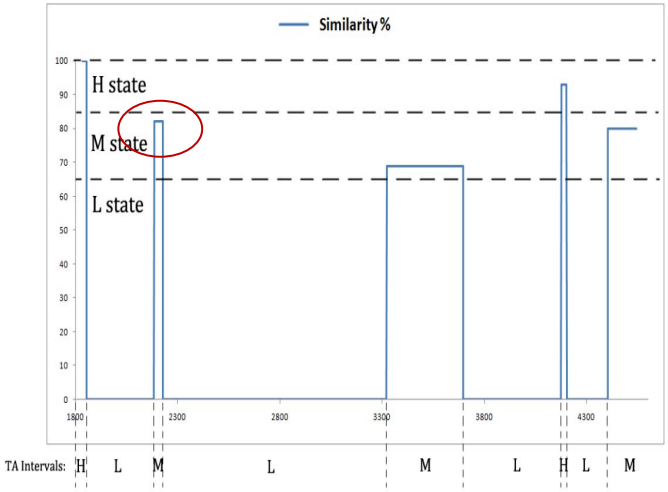

Fig. 3. A graphical visualization of sequence alignment (x-axis: nucleotide position of the alignment with respect to the reference string; y-axis: % of similarity). The specific substring with an 82% similarity identified in figure 2, corresponding to a medium (M) qualitative value, is highlighted.

In our framework, we adopt this methodology with a main difference: the independent variable is the symbol position in the aligned strings, instead of time. The values to be converted into qualitative levels are then the percentages of similarity between the two strings themselves. An example is provided in figure 3, where e.g. a similarity of 82% is considered to be *medium* (M).

More specifically, on the basis of domain knowledge we define a whole taxonomy of qualitative levels, where *low*, *medium* and *high* values can be further specialized (see figure 4). Abstractions of quantitative similarity values at different levels of detail can support different requests of the researchers.

We also allow to abstract the information about the "localization" of the aligned substrings along the nucleotide sequences at different granularities (see figure 5), visualizing genome information at the level of genes, regions, chromosomes or even complete genomes[1]. In particular, since, in our application, the full genome of an organism is typically subdivided into one or more chromosomes/plasmids, similarity between two genomes has to be calculated by applying an aggregation operation to the similarities at the single chromosome/plasmid level. We are currently using the arithmetic mean as an aggregation operator.

This granularity change makes sense from a biological point of view: consider e.g. that a region may be conserved among relative organisms, while a specific gene within the region may not. Thus, a high similarity at the region level might be difficultly identified at the level of single genes (see the example in section C).

[1] Note that a more specific visualization at the contig level may be required in some situations; this need could be straightforwardly covered in our framework as well.

Note that the taxonomy of granularities is strongly influenced by domain semantics. For instance, the number of nucleotides which composes a gene depends on the specific organism, and on the specific gene. Domain knowledge also strongly influences the conversion of a string of symbols from a given granularity to a different one, as required for flexible retrieval (see section C).

To summarize, in our framework *case representation* is obtained as follows. First, a pair of nucleotide strings, optimally aligned as calculated by BLAST, is taken. In particular, for each subsequence of nucleotides, a percentage of similarity with the aligned nucleotides in the paired string is provided. Abstractions on such quantitative levels are then calculated, and allow to convert these values into qualitative ones. Abstractions are calculated at the ground level in the symbol taxonomy (and operate also at the ground level in the granularity taxonomy, since they work on nucleotides, see figure 5). The resulting string of symbols is then stored in the case library as a *case*.

Despite the fact that cases are stored as abstractions at the ground level, they could be easily converted at coarser levels in both dimensions (i.e. in the dimension of the taxonomy of symbols, and in the one of granularities). Such conversion is actually the means by which we support flexible case retrieval, and will be described in the next section.

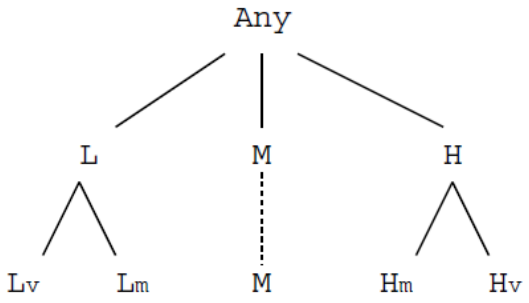

Fig. 4. An example taxonomy of state abstraction symbols; for instance, the high (H) symbol specializes into very high (Hv) and moderately high (Hm).

C. *Case retrieval (query answering)*

As described in section B, our retrieval framework allows for *multi-level abstractions*, according to two dimensions, namely a taxonomy of state abstraction symbols, and a variety of granularities.

Taking advantage from this data representation, we support flexible retrieval.

In particular, we allow users to express their queries at any level of detail, both in the dimension of data descriptions (i.e. at any level in the taxonomy of symbols) and in the dimension of granularity. Obviously, since cases are stored at the ground level in both dimensions, in order to identify the cases that match a specific query, a function for scaling up (*up* henceforth) two or more symbols expressed at a specific granularity level to a single symbol expressed

at a coarser one must be provided. Moreover, a proper distance function must be defined for retrieval.

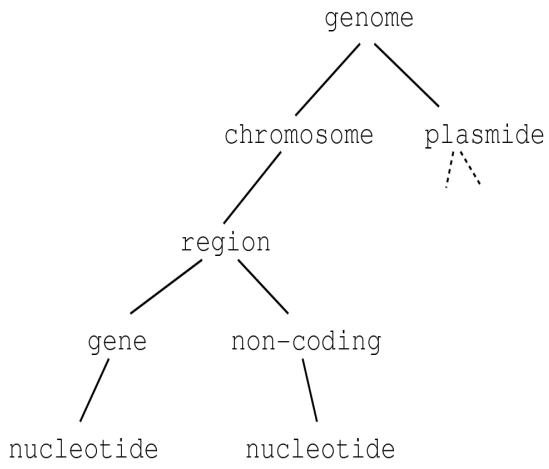

Fig. 5. A taxonomy of sequence granularities**.**

The data structures described in section B, as well as the *up* and the distance functions, have to be detailed on the basis of the semantics of the specific application domain. However, we have identified a set of general "consistency" constraints, that any meaningful choice must satisfy, in order to avoid ambiguous or meaningless situations. For instance, we enforce the fact that distance monotonically increases with ordering in the symbol domain - if any (e.g. the distance between L (*low*) and M (*medium*) is smaller than the distance between L (*low*) and H (*high*), see figure 4). Moreover, distance "preserves" ordering also in case *isa* relationships between symbols are involved (e.g. the distance between L (*low*) and M (*medium*) is smaller than the distance between L (*low*) and H_v (*very high*)). The exhaustive presentation of such constraints is outside the scope of this paper, but can be found in [16].

In order to increase efficiency, our framework also takes advantage of multi-dimensional orthogonal index structures, which allow for early pruning and focusing in query answering. Indexes are built on the basis of the data structures described in section B. The root node of each index is a string of symbols, defined at the highest level in the symbol taxonomy (i.e. the children of "Any", see figure 4) and in the granularity taxonomy. A (possibly incomplete) index stems from each root, describing refinements along the granularity and/or the symbol dimension. An example multi-dimensional index, rooted in the H symbol, is represented in figure 4. Note that, in the figure, granularity has been chosen as the *leading dimension*, i.e. the root symbol is first specialized in the granularity dimension. From each node of the resulting index, the sequence of symbols of the node itself is then orthogonally specialized in the *secondary* (i.e. the symbol) *dimension*, while keeping granularity fixed. However, the opposite choice for instantiating the leading and the secondary dimensions would also be possible.

Each node in each index structure is itself an index, and can be defined as a *generalized case*, in the sense that it summarizes (i.e. it indexes) a set of cases. This means that the same case is typically indexed by different nodes in one index (and in the other available indexes). This supports flexible querying, since, depending on the level at which the query is issued, according to the two taxonomies, one of the nodes can be more suited for providing a quick answer.

Technically speaking, to answer a query, in order to enter the more proper index structure, we first progressively generalize the query itself along the secondary dimension (i.e. the symbol taxonomy in the example), while keeping the leading dimension (i.e. granularity in the example) fixed. Then, we generalize the query in the other dimension as well. Following the generalization steps backwards, we can enter the index from its root, and descend along it, until we reach the node which fits the original query leading dimension level. If an orthogonal index stems from this node, we can descend along it, always following the query generalization steps backwards. We stop when we reach the same detail level in the secondary dimension as in the original query. If the query detail level is not represented in the index, because the index is not complete, we stop at the most detailed possible level. We then return all the cases indexed by the selected node.

We will now illustrate query answering by means of a specific case study. We take as a reference organism a bacterium belonging to the Burkholderia genus. All bacteria belonging to this family share a region, called DCW cluster, which is involved in the synthesis of peptidoglycan precursors and cell division. The DCW cluster is composed by 14 genes: FtsA, FtsI, FtsL, FtsQ, FtsW, FtsZ, mraW, mraY, mraZ, murC, murD, murE, murF, murG. The prominent feature of this cluster is that it is conserved with a high (H) similarity in many bacterial genomes over a broad taxonomic range. Notwithstanding some bacteria belonging to the studied family simply miss one of the 14 genes (specifically the third), while all of the others maintain a high similarity at the DCW region level with their relatives. Therefore, it makes sense to de• ne the *up* function as follows: *up*(HHLHHHHHHHHHHH)=H (where the absence of a gene is identi• ed by a low similarity value in the gene position).

Suppose that, more precisely, the user expresses the query HvHvLvHvHvHvHvHvHvHvHvHvHv, looking for the specific bacteria missing the third gene, but very similar to the reference one as regards the other genes. Our system will first generalize the query in the symbol taxonomy dimension, providing the string HHLHHHHHHHHHHH (see figure 6), and then in the granularity dimension, providing the query H at the region level. This allows to enter the index in figure 6 from its root. Then, following the generalization step backwards, a node identical to the query can be found, and the ground cases indexed by it can be retrieved. The index search steps are highlighted in the figure.

Interactive and progressive query relaxation (or refinement) are supported as well in our framework. Query relaxation or refinement can be repeated several times, until the user is satisfied with the width of the retrieval set. In the Burkholderia example, the user may generalize the initial query as an H at the region level, and retrieve also the cases indexed by HHHHHHHHHHHHHH at the gene level (the other siblings of HHLHHHHHHHHHHH do not index any real case in this specific situation). The cases indexed by HHHHHHHHHHHHHH can thus be listed, clarifying that their distance (calculated by any distance function which satisfies the constraints illustrated in [16], and quickly described above) from the original query is greater than zero.

It is worth noting that indexes may be incomplete with respect to the taxonomies. Index refinement can be automatically triggered by the memorization of new cases in the case base, and by the types of queries which have been issued so far. In particular, if queries have often involved e.g. a symbol taxonomy level which is not yet represented in the index(es), the corresponding level can be created. A proper frequency threshold for counting the queries has to be set to this end. This policy allows to augment the indexes discriminating power only when it is needed, while keeping the memory occupancy of the index structures as limited as possible.

Flexibility and interactivity are also supported by a user-friendly graphical interface, which has been designed by following software engineering principles, in order to enhance usability and user friendliness in the interaction with the system. Through the interface, we provide a graphical representation of the indexes (conceptually depicted as in figure 6), whose nodes can be exploded or iconified, facilitating index navigation (see figure 7). Moreover, the graphical interface can support the user in selecting the proper navigation direction, providing him/her with quantitative and qualitative information about the cases indexed by sons and siblings of the currently visited node (i.e. the number of indexed cases, the sequence of abstractions representing the cases and the distance from the sequence of abstractions representing the node currently visited by the user).

The interested reader may find additional technical details about our framework in [16]. Very encouraging experimental results have already been obtained by resorting to the same framework, in the field of haemodialysis [17].

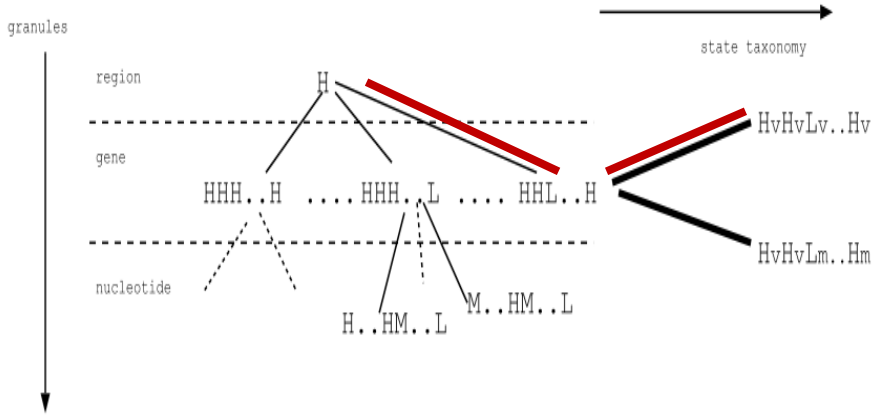

Fig. 6. An example multi-dimensional orthogonal index

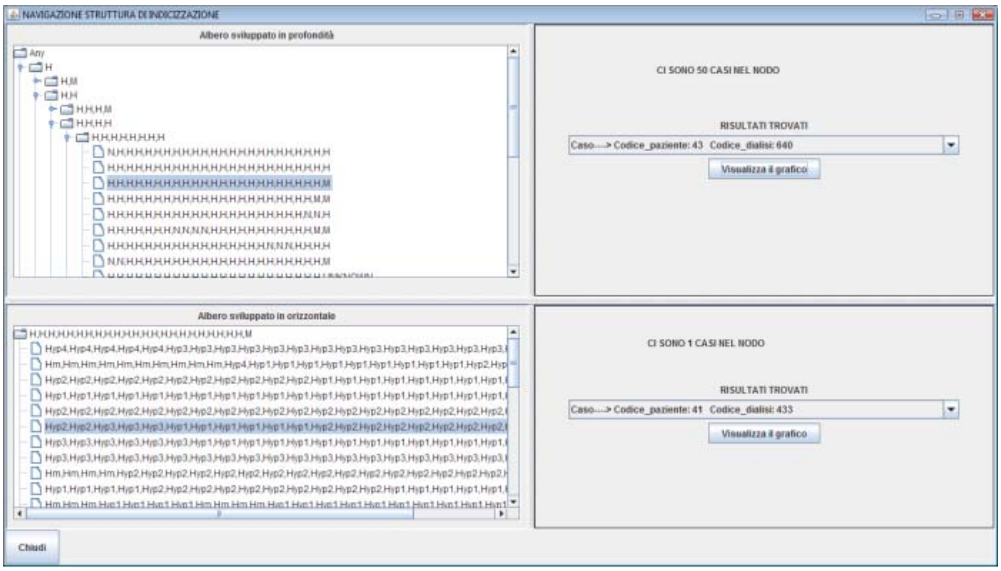

Fig. 7. A snapshot of index navigation as rendered by the system graphical interface.

IV. Related Work

In 1993, Aaronson [18] suggested that analogical reasoning (which includes CBR) is particularly applicable to the biological domain, because biological systems are often homologous, and because biologists often design and perform experiments based on the similarity between features of the new system to be investigated, and already known ones. As a matter of fact, since then, some CBR applications in biology and bioinformatics have been published. The paper in [19] is an interesting survey on the topic. The surveyed papers are mostly related to experimental design in protein crystallization and protein structure prediction. However, one contribution [18] also makes the hypothesis of using a CBR approach for predicting unknown regulatory regions. More recently, a hybrid method (resorting to Bayesian techniques and CBR) for feature selection in microarray data analysis has been presented [20] .Except for the work in [18], however, we are not aware of CBR works in genomic comparison. Moreover, the work in [18] does not support any flexible and interactive case retrieval, as we are able to do by means of the abstraction mechanism.

As stated in section 4, our abstraction mechanisms resembles the one of TA [9,10]. In fact, the present work has been developed starting from our previous experience on TA-based time series retrieval [16,17], and properly adapting the characteristics of the framework described in [16,17] to the biological domain. Actually, such a framework was designed in a modular and domain independent way. The adaptation to the biological domain has been realized by acquiring the specific domain knowledge, which is the basis for a proper definition of the taxonomies and of the distance and up functions. As regards TA, they have been extensively resorted to in the literature, especially in the medical field, from diabetes mellitus [21,22], to artificial ventilation of intensive care units patients [23] (see also the survey in [24]), but typically with the aim to solve a data interpretation task [9], and not as a flexible retrieval support facility. The goal of our proposal is to try to fill this gap, by exploiting an abstraction mechanism for supporting data interpretation, as well as case exploration and retrieval; this idea thus appears to be significantly innovative in the recent literature panorama.

As previously observed, one of our main goals is also interactivity. It is worth noting that, in classical CBR systems, interactivity is typically not supported: the user is asked to input the entire, precise problem description as a query for case retrieval. This means s/he must know the relevance of every case feature - which is not always straightforward in practice. A research direction meant to overcome this limitation indeed exists in the CBR literature, and is known as Conversational CBR (CCBR) (see e.g. [25,14,26]). In CCBR, the user is allowed to input just a brief free text description of the case, to start. The system then supports a progressive query refinement through a conversation, in which best matching cases are listed, and further questions meant to reduce and specialize the retrieval set are asked by the system itself. Our framework thus loosely resembles CCBR. However, CCBR is characterized by some strong challenges, namely related to case authoring, and dialog inferencing. Both aspects are non-trivial, and should be solved by specific modules (based e.g. on machine learning [27] or model-based reasoning [28] techniques). Such an additional effort is not required in our framework.

As a last remark, a number of (non CBR) tools to support comparative genomics studies are available in the literature: e.g. VISTA (http://genome.lbl.gov/vista/index.shtml) allows the visualization of pre-computed pairwise and multiple

alignments of whole genome assemblies. However our tool also allows to mine genomes at multiple levels: customized searches can be performed, to retrieve genomes and/or genomic segments matching specific features as described by the query at the desired granularity. Furthermore, thanks to our tool, queries can be performed efficiently and potentially on very large databases.

V. CONCLUSIONS

In this paper, we have described a modular architecture for supporting in-silico comparative genomics analysis, being developed within the BIOBITS project. In particular, we have focused on the main features of a genome search tool, which implements the retrieval step of the CBR cycle. Such a tool provides researchers with flexible retrieval capabilities, in an interactive fashion. Flexibility and interactivity are also supported by a user-friendly graphical interface. Moreover, retrieval performances are optimized by resorting to multi-dimensional orthogonal index structures, allowing for a quick query answering.

In the next months, we will work at the system evaluation phase, by initializing the tool case base with a selection of genomes (in particular, from symbiotic microorganisms) available in the RefSeq NCBI database. The newly sequenced Candidatus Glomeribacter gigasporarum will be made available soon. The tool will then support genomic comparison studies between the new genome and the ones of relative organisms already known to the scientific community, which is one of the key goals of BIOBITS.

REFERENCES

[1] J. Marx. The roots of plant-microbe collaborations. *Science*, 304:234236, 2004.

[2] P. Bonfante, A. Genre, . Mechanisms underlying beneficial plant-fungus interactions in mycorrhizal symbiosis, *Nature Communications*., 1:48, 2010.

[3] I.A. Anca, E. Lumini, S. Ghignone, A. Salvioli, V. Bianciotto, P. Bonfante, The ftsZ gene of the endocellular bacterium 'Candidatus Glomeribacter gigasporarum' is preferentially expressed during the symbiotic phases of its host mycorrhizal fungus, *Molecular plant-microbe interactions : MPMI*;22(3):302-10, 2009

[4] N.A. Moran, A.J. McCutcheon, and P. Nakabachi. Genomics and evolution of heritable bacterial symbionts. *Annu. Rev. Genet*., 42:165190, 2008.

[5] E. Lumini, S. Ghignone, V. Bianciotto, and P. Bonfante. Endobacteria or bacterial endosymbionts? to be or not to be. *New Phytol*, 170:205208, 2006.

[6] Brian Osborne and GMOD Community. GMOD, 2000.

[7] C.J. Mungall, D.B. Emmert, The FlyBase Consortium. A chado case study: an ontology-based modular schema for representing genome-associated biological information. *Bioinformatics*, 23(13):i33746, July 2007.

[8] A. Aamodt and E. Plaza. Case-based reasoning: foundational issues, methodological variations and systems approaches. *AI Communications*, 7:3959, 1994.

[9] Y. Shahar. A framework for knowledge-based temporal abstractions. *Artificial Intelligence*, 90:79133, 1997.

[10] R. Bellazzi, C. Larizza, and A. Riva. Temporal abstractions for interpreting diabetic patients monitoring data. *Intelligent Data Analysis*, 2:97122, 1998.

[11] F. Lazzarato, G. Franceschinis, M. Botta, F. Cordero, and R. Calogero. Rre: a tool for the extraction of non-coding regions surrounding annotated genes from genomic datasets. Bioinformatics, 1;20(16): 2848-50,2004.

[12] http://www.ncbi.nlm.nih.gov/Genbank/

[13] http://www.biomart.org/

[14] I. Watson. Applying Case-Based Reasoning: techniques for enterprise systems. Morgan-Kaufmann, 1997.

[15] S. Montani. Exploring new roles for case-based reasoning in heterogeneous AI systems for medical decision support. *Applied Intelligence*, 28:275285, 2008.

[16] S. Montani, A. Bottrighi, G. Leonardi, L. Portinale, and P. Terenziani. Multi-level abstractions and multi-dimensional retrieval of cases with time series features. In L. McGinty and D. Wilson, editors, *Proc. International Conference on Case-Based Reasoning (ICCBR) 2009, Lecture Notes in Artificial Intelligence 5650*, pages 225 239. Springer-Verlag, Berlin, 2009.

[17] A. Bottrighi, G. Leonardi, S. Montani, L. Portinale, P. Terenziani, Intelligent data interpretation and case base exploration through Temporal Abstractions, Proc. International Conference on Case-Based Reasoning (ICCBR) 2010, LNCS 6176, I. Bichindaritz, S. Montani eds., Springer, Berlin, pp. 36-50

[18] J.S. Aaronson, H. Juergen, G.C. Overton, Knowledge discovery in GENBANK, Proc. International Conference on Intelligent Systems for molecular biology, L. Hunter, D. Searls, U. Shavlik eds., AAAI Press, pp. 3-11

[19] I. Jurisica, J. Glasgow, "Applications of Case-Based Reasoning in molecular biology", AI Magazine, vol. 25(1), pp. 85-95, 2004.

[20] I. Bichindaritz, A. Annest, Case based reasoning with Bayesian model averaging: an improved method for survival analysis on microarray data, Proc. International Conference on Case-Based Reasoning (ICCBR) 2010, LNCS 6176, I. Bichindaritz, S. Montani eds., Springer, Berlin, pp. 346-359

[21] Y. Shahar and M. Musen, "Knowledge-based temporal abstraction in clinical domains," Artificial Intelligence in Medicine, vol. 8, pp. 267–298, 1996.

[22] R. Bellazzi, C. Larizza, P. Magni, S. Montani, and M. Stefanelli, "Intelligent analysis of clinical time series: an application in the diabetes mellitus domain," Artificial Intelligence in Medicine, vol. 20, pp. 37–57, 2000.

[23] S. Miksch, W. Horn, C. Popow, and F. Paky, "Utilizing temporal data abstractions for data validation and therapy planning for artificially ventilated newborn infants," Artificial Intelligence in Medicine, vol. 8, pp. 543–576, 1996.

[24] P. Terenziani, E. German, and Y. Shahar, "The temporal aspects of clinical guidelines," in Computer-based Medical Guidelines and Protocols: A Primer and Current Trends, A. T. Teije, S. Miksch, and P. Lucas, Eds., 2008.

[25] D. Aha and H. Munoz-Avila. Introduction: interactive case-based reasoning. *Applied Intelligence*, 14:78, 2001.

[26] M. Manago, K.-D. Althoff, E. Auriol, R. Traphoner, S. Wess,N. Conruyt, and F. Maurer, "Induction and reasoning from cases," in Proc. European Workshop on CBR. Springer, 1993, pp. 313–318.

[27] D. Aha and L. Breslow, "Refining conversational libraries," in Proc International Conference on Case Based Reasoning. Springer, 1997, pp. 267–278.

[28] D. Aha, T. Maney, and L. Breslow, "Supporting dialogue inferencing in conversational case-based reasoning" in Proc European Workshop on Case Based Reasoning. Springer, 1998, pp.262–273.