Genopage : A database of all protein modules encoded by completely sequenced genomes

Sarah COHEN BOULAKIA[1], Christine FROIDEVAUX[1], Emmanuel WALLER[1] and Bernard LABEDAN[2]

[1] Laboratoire de Recherche en Informatique, CNRS UMR 8623, Bâtiment 490
[2] Institut de Génétique et Microbiologie, CNRS UMR 8621, Bâtiment 409,
Université Paris-Sud, 91405 Orsay Cedex, France

Abstract

Comparative analysis of the whole set of proteins encoded by completely sequenced genomes allowed to identify segments of homology (modules) in many proteins. These modules are then clustered into families in order to progressively trace back the history of present-day proteins. We plan to assemble all these modules and their families in a relational database. In this paper, we present a brief description of the structure and functionalities of this base, Genopage. The database schema contains all required informations in a concise structure built using PostgreSQL. Yet, it allows to pose complex queries which can be efficiently answered, providing relevant understanding of the biological phenomena. We think that Genopage might be useful not only to the community of genomicists but to many other biologists. This database is expected to further evolve into a knowledge base in order to become even more valuable.

Keywords : *relational database, protein homology, protein history.*

1. Introduction

We are interested in the progressive disclosing of the mechanisms of protein evolution. We have previously shown [14, 16] that the homology between proteins is often limited to *long* (around 200 residues) structural segments. These segments which are far longer than what is generally called *domains* (see for example [2, 5, 12], or *motifs* or signatures [7, 15], have been called *modules* [14, 16]. We have shown that the concept of modular structure is crucial to the understanding of the mechanisms which have shaped protein evolution (gene ancestry) as well as to the reconstruction of the chromosome evolution of microorganisms. Indeed, if domains and motifs (listed in several databases such as Prosite [7], ProDom [5], Domo [12], Pfam [2], SMART [15]), are nice tools to define protein families on the basis of functional criterions, modules which often contained several such domains and motifs are helpful to trace back the events of gene duplication and gene fusion, two fundamental events in the invention of new proteins [14, 16].

In order to cope with the overflow of data released by the whole-genome sequencing programs, we have devised a suite of automatic programs [1] which allows us to obtain in a few steps the whole set of modules which constitute the proteome of any organism for which the complete sequence is available (Table 1). This set contains both *detected* (found directly by an homology search) and *distant* modules. Distant modules are logically deduced from the analysis of groups of families of multimodular proteins sharing common detected modules [1]. By difference, we also identified all proteins unique to an organism and all nonhomologous modules.

We intend to assemble our whole set of modules in a relational database made using PostgreSQL on a Linux server [4]. To do this, we designed a data base schema using a simple structure, containing all required informations without any redundancy. This base might be queried via the site *Genopage*, which will be accessible through a Web server. In this paper, we present already a brief description of the structure and functionalities of Genopage and of how this base may be useful not only to the community of genomicists but to many other biologists. Along this presentation we will point out how the collaboration between computer scientists and biologists has contributed to provide good features to Genopage.

2. Experimental design

2.1. Finding out all homologous modules

To assess the homologous relationships among proteins inside the same proteome (paralogy) or between various proteomes (orthology), we set up the following two-steps strategy as shown below.

Main experimental step	Programme	Language	Written by
1. Collecting protein dataset and translation in SGML language	sgml	C	Karim Abou-Merhy
2. Detecting homologous proteins	Darwin AllAll	Maple and C	Renaud De Rosa
3. Detecting homologous modules	Module	Maple	Hichem Arfaoui
4. Assembling homologous modules in families	Families	C	Karim Abou-Merhy
4a. Grouping families sharing adjacent unrelated modules	Adjacent Families	C	Adrien Bazureau Karim Abou-Merhy
5. Inferring distant modules in multimodular proteins after grouping families and clustering them in new refined families	Module Sorter Module Cluster SortClust	C C C++	Stéphanie Langevin Isabelle Montaland
6. Challenging deduced modules by computing distance matrix and evolutionary tree for each refined family	DistTree	C	Adrien Bazureau

TABLE. 1 – Our suite of programmes to find out all modules

2.2. Assembling the whole set of modules

Finding out all detected modules : Proteins encoded by completely sequenced genomes of various microbial organisms (step 1) are compared in an exhaustive way using the AllAll program [10, 11] of the DARWIN package (step 2). Segments of homology are detected using stringent thresholds for evolutionary distance and minimal length of alignments (step 3) and then grouped transitively in families (step 4).

Finding out all deduced modules : Families which are connected by a chain of neighbouring unrelated homologous modules are further grouped (step 4a). The analysis of these groups allows us to break up into their component parts many fused modules and/or to deduce by logic more distant modules (step 5). Then, all detected and deduced modules are assembled in refined families. Finally, the soundness of these deduced modules was systematically challenged using independent and complementary tests (step 6).

Differentiating the modules : When this methodology is applied to one organism (*intragenomic analysis*), we obtain exhaustive informations about the whole set of paralogues (homologues descending from ancestral duplications, as defined by Fitch [8]) and we identify by contrast the genes which are unique to this organism. We have shown previously that paralogues are good tools to unravel protein history independently of the speciation events [14, 16]. When we compare several organisms (*intergenomic analysis*), we add supplementary informations about the orthologous relationships between each category of genes (paralogues and uniques) for each organism. As already underlined by Fitch [8], these orthologues are epitomizing the speciation events and thus are good tools to study the phylogeny of organisms. Thus, we can differentiate, for each proteome, the relative proportions of genes (1) which are unique to this genome and which either have an orthologue (1a) or have not (1b) and those (2) which have paralogue(s) and which either have an orthologue (2a) or have not (2b).

3. Making and using Genopage, a relational database for proteomics and molecular evolution

3.1. Need for a SQL database

In the last years, we used the relational database application FileMakerPro to manage the data present in the various files of results obtained at each experimental step (Table 1). This application was also used to publish on a Web server the data obtained on the paralogous genes of the bacterium *Escherichia coli*. This *Colipage* server (http://www-colipage.igmors.u-psud.fr) was progressively built in parallel with the addition of new programs,

leading to the additions of more and more tables. When we began to accumulate data on both *intragenomic* and *intergenomic* analysis, it became evident that (i) we had to restructure entirely our database, (ii) we needed to upgrade to an SQL application to cope with an enormous increase of the data, a potentially increasing granted number of modules per protein and to allow more complex and sophisticated queries. Thus, we decided to turn to PostgreSQL, an open source DBMS, on a Linux machine.

3.2. Structuring the database

Preliminary studies of the most commonly used queries led us to the following database schema.

Tables : The nine more or less redundant tables present in the Colipage version have been drastically reduced to only three tables (Fig 1). Although the presence of redundancies could allow fast answers to complex queries, they are often unnecessarily expensive and may lead to potential inconsistencies due to update anomalies.

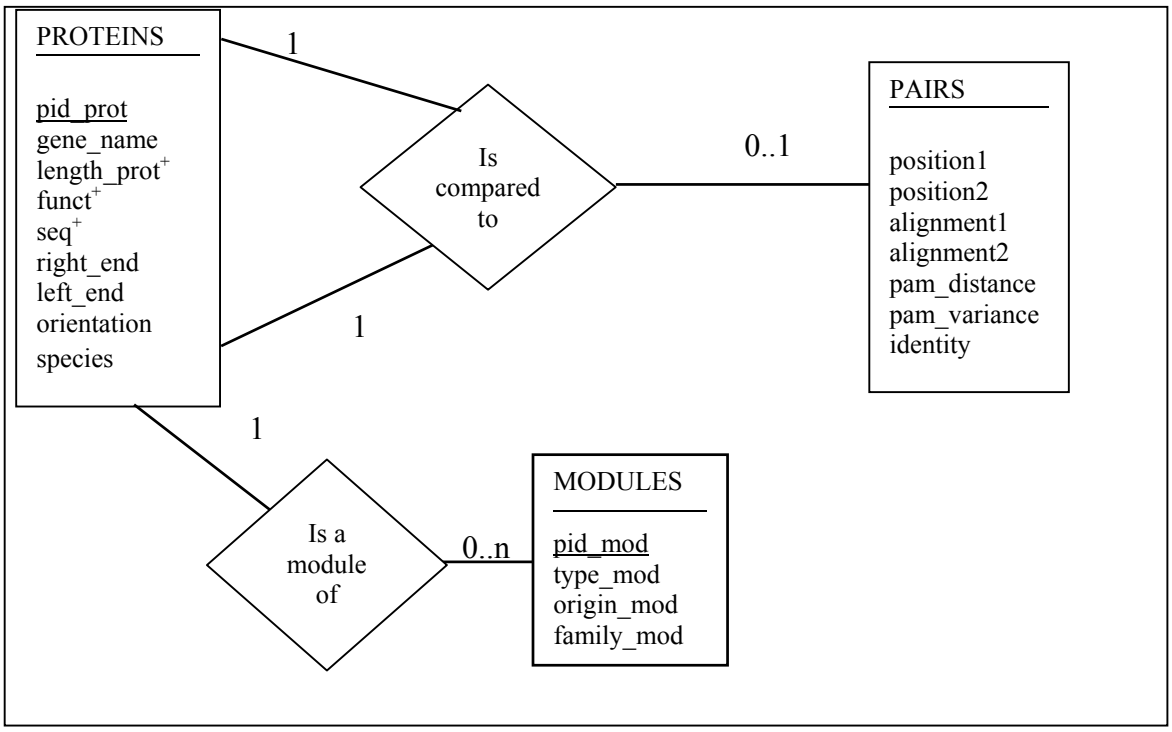


Fig. 1. Structure of the Genopage database [4]. Note that *sequence*, *length* and *function* have been replaced by *seq*, *length_prot* and *funct* respectively since these words are already used by PostgreSQL.

Fig 1 shows the general structure of Genopage with the relations between these three main tables [4]. Table Proteins contains various informations extracted from GenBank/EMBL files describing complete genome sequences (step 1 of Table 1). For each protein, we store the protein identification number and the name of its coding gene, its length (number of residues), what is known on its function, its amino acid sequence and where its coding gene is located on the genome. Table Pairs contains the fields describing the alignment between each pair of homologous modules and the informations about their evolutionary distance (steps 2 and 3 of Table 1). The evolutionary distance is expressed both in PAM units (a PAM unit being defined as the number of *accepted* point mutations per 100 residues separating two sequences [6, 17]) and in percent of identical residues. Table Modules presents the data obtained on each detected and deduced module (steps 4 to 6 of Fig.1). To be independent of the total number of modules granted per protein this Table Modules has been built. For each module, are stored the type (*e.g.* module_33), the origin (detected or deduced) and the number of its family.

We found that such a database structure contained all required informations without any redundancy.

Views : Our three-tables structure was found to be powerful enough to extract temporarily pertinent informations through more or less complex queries. However, we also found necessary that frequently posed and important queries which require several tables to be answered must be stored for more efficient use of our database. To fulfill this kind of requirement, we used the DBMS facilities of PostgreSQL to set up *views* for often required data. Such intricate queries might even replace some of the previously used programs (described in Table 1). For example, a

frequent query would ask for modules belonging to a subset of species and coding a specific class of function. Accordingly, a virtual table will group corresponding necessary informations about pid_mod, family_mod, (from Table Modules) and funct, species (from Table Proteins).

3.3. Using the database

Simple queries : Our database allows to give crucial informations about each of the found modules, their families, their relative proportion in various proteomes. Therefore, we may recover for example homologous families of paralogous modules for closely or distantly related organisms and then compare these families and use this comparison to estimate the phylogenetic relationships between these organisms. Below is an example :

What are the families made of less than 10 members…more than 20 members ?

```
Select family_mod as nb_modules
  From MODULES
  Group by family_mod
  Having count(pid_mod)<10  …>20
```

More sophisticated queries : We may also build more complex queries, in order to recover for example sets of families of modules with information about the location of their encoding genes and use these data to measure the conservation or not of the neighborhood of these genes with respect to their orthology. Some of these complex queries could be views, as described above. Following is an example of a more complex query :

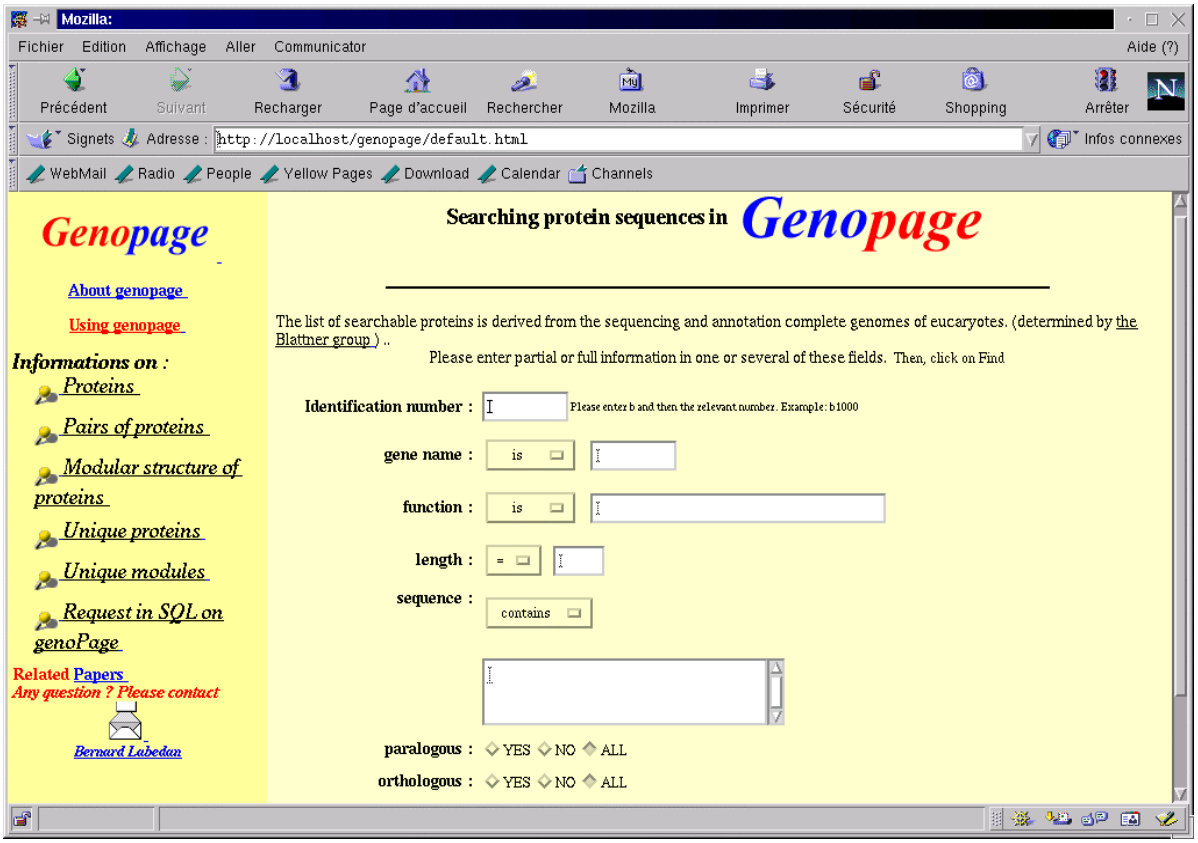
Find out all orthologous proteins made of two modules and separated by an evolutionary distance bounded by 50 and 75 PAM units, and give their corresponding species name

```
Select pid_prot1, pid_prot2, p1.species as species1,
p2.species as species2
    From PAIRS, PROTEINS P1, PROTEINS P2
    Where pam_distance >= 50
         And pam_distance <= 75
         And pid_prot1 in
                  (Select pid_prot
                      From MODULES
                      Where MODULES.type_mod = 'mod12'
                         Or MODULES.type_mod = 'mod22'
                   )
         And pid_prot2 in
                  (Select pid_prot
                      From MODULES
                      Where MODULES.type_mod = 'mod12'
                         Or MODULES.type_mod = 'mod22'
                )
         And P1.pid_prot = pid_prot1
         And P2.pid_prot = pid_prot2
         And P1.species <> P2.species
```

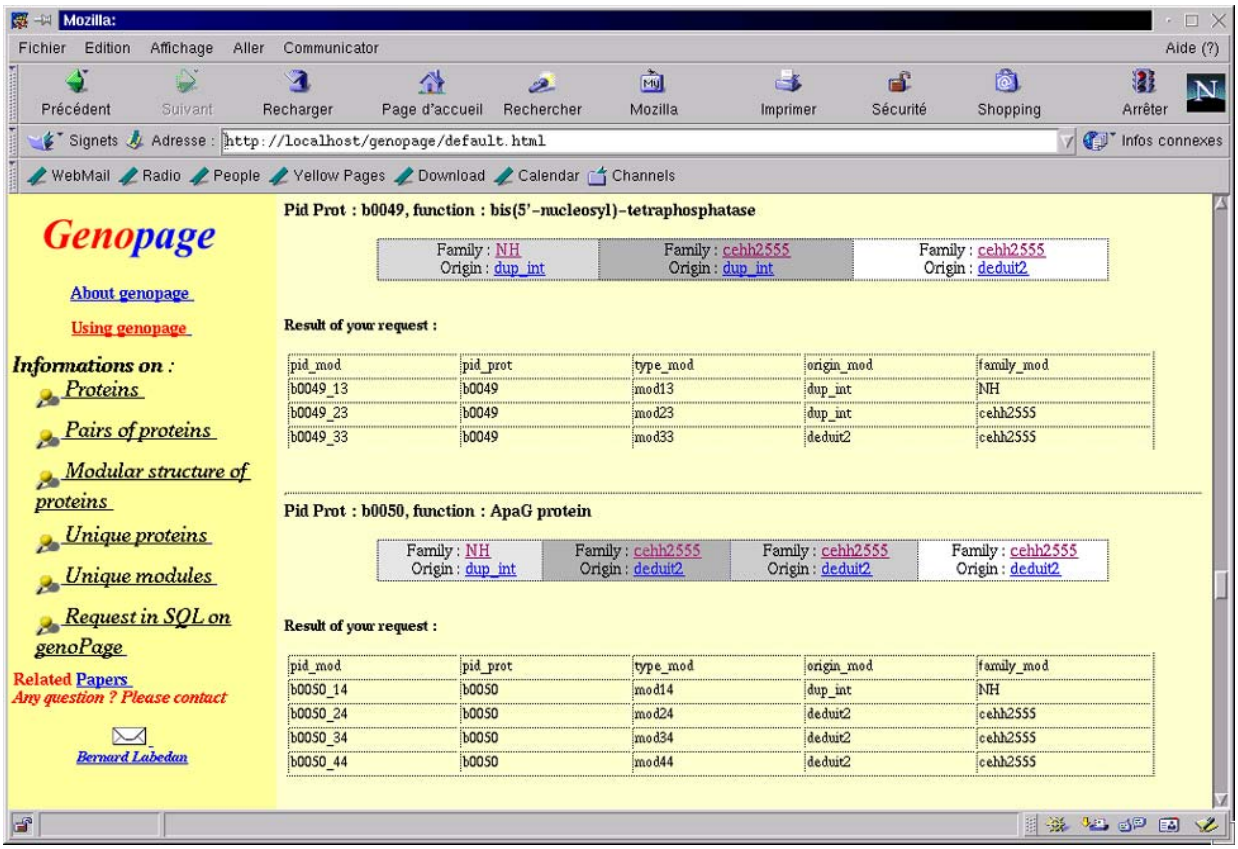
Internal use : As we have to consider more and more new entirely sequenced genomes, such a database appears to be an excellent tool to manage increasingly complex data in order to address various questions in the different fields of molecular evolution we are studying.

External use : The data we are introducing into Genopage will be progressively accessible through a Web server. They may interest a large public and be useful in rather different fields such as annotating newly sequenced genomes, studying the structure and function of present-day proteins, exploring molecular evolution (history of proteins, phylogeny, genome dynamics), comparing organisms life style, looking for various pathogenicity factors, etc …

Below are examples of available pages of our future Web server :



Below is the result of a request about proteins having more than two modules. A primitive modular structure is shown for each protein beside the results table.



4. Conclusions and perspectives

Our PostgreSQL approach allows to pose complex queries which can be efficiently answered, providing relevant understanding of the biological phenomena. The new data base schema is quite simple and the query language chosen proved to be expressive enough till now. Moreover, besides many preprogrammed queries allowing any biologist to retrieve many data, we offer the possibility to any people knowing the SQL language to write unforeseen queries.

4.1. Improving the biological significance

Adding more informations on genome organization : In a next future, we would like to introduce new data related to gene context, lateral gene transfer (LTG), species relationships, phylogeny of procaryotes…

Adding more informations on proteins : We also plan to display (i) genealogical trees for each family of modules and (ii) more dynamical graphical representation of the modular structure for each homologous protein.

4.2. Improving the database

Beyond the relational model : We would like to improve the graphical form of the data as we are concerned by genealogical trees and phylogenetic trees. We could envisage to integrate these trees with the relational data. On the other hand, to facilitate efficient data communication between our base and other databases using the XML format, and to increase data queriability, scalability and availability, we contemplate using XML. Indeed, there is already an international effort to establish new DTDs for biology (see for example, http://www.ai.sri.com/~pkarp/xol/xol.html, [13, 18]). For this, we have to investigate other non relational data models which would be more expressive than relational data models.

Towards Oracle : Since it is expected that the amount of genomics data will continue to increase unwaveringly, we may presume to reach the limits of PostgreSQL in a short interval of time. Accordingly, we plan to migrate as soon as necessary to another DBMS such as Oracle.

4.3. Evolution of Genopage towards a knowledge base

The present base could be a good first stage for a more distant aim. We would like to progressively introduce more precise terms using an ontology approach to conceive a knowledge base. In order to do this, we plan to extract many pieces of knowledge from different sources and to combine them in building a hierarchy [3, 9, 13]. Accordingly, we hope to construct a specialized ontology, allowing for more complex and integrative questions in the field of protein evolution.

References

[1] ARFAOUI H., DE ROSA R., LE BOUDER S. and LABEDAN B. (2000) *A suite of programs to find out all modules present in homologous proteins encoded by completely sequenced genomes*. JOBIM 2000, Recueil des Actes, pp. 31-38

[2] BATEMAN A., BIRNEY E., CERRUTI L., DURBIN R., ETWILLER L., EDDY S.R., GRIFFITHS-JONES S., HOWE K.L., MARSHALL M. and SONNHAMMER E.L. (2002) *The Pfam protein families database*. Nucleic Acids Res, **30**: 276-280

[3] BIDAULT A., FROIDEVAUX C. and SAFAR B. (2000)., *Repairing Queries in a Mediator Approach*, *in* Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, 406-410

[4] COHEN BOULAKIA S. (2001). *Génomique fonctionnelle et Bases de données*. Mémoire de Maîtrise. Université Paris-Sud

[5] CORPET F., SERVANT F., GOUZY J. and KAHN D. (2000) *ProDom and ProDom-CG: tools for protein domain analysis and whole genome comparisons*. Nucleic Acids Res, **28**: 267-269

[6] DAYHOFF M. O., SCHWARTZ R. M. and ORCUTT B. C. (1978) . A model for evolutionary change, pp. 345-352. *In* M.O. Dayhoff ed., *Atlas of Protein Sequence and Structure*, vol.5, suppl.3. National Biomedical Research Foundation, Washington, D.C

[7] FALQUET L., PAGNI M., BUCHER P., HULO N., SIGRIST C.J.A., HOFMANN K. and BAIROCH A.(2002). *The PROSITE database, its status in 2002*. Nucleic Acids Res. **30**: 235-238

[8] FITCH W. D. (1970) *Distinguishing homologous from analogous proteins*. Systematic Zool. **19**:99-113

[9] FRIDMAN NOY N. and HAFNER C. (1998). *Representing scientific experiments : implications for ontology design*

and knowledge sharing, *in* Proceedings of the 15th National Conference on Artificial Intelligence, Madison, 615-622

[10] GONNET G.H., COHEN M.A. and BENNER S. A.. (1992) *Exhaustive matching of the entire protein sequence database*. Science **256**:1443-1445

[11] GONNET G.H. and HALLETT M. (1997) *The DARWIN Manual*

[12] GRACY J. and ARGOS P. (1998) *DOMO: a new database of aligned protein domains*. Trends Biochem Sci **12**:495-497

[13] KARP P. (2000) *An ontology for biological function based on molecular interactions*. Bioinformatics **16**:269-285

[14] LABEDAN B. and RILEY M. (1999) *Genetic inventory: Escherichia coli as a window on ancestral proteins* Chapter 17, pp 311-329 *In* "Organization of the Prokaryotic Genome" (Robert Charlebois, editor) ASM Press, Washington

[15] LETUNIC I., GOODSTADT L., DICKENS N.J., DOERKS T., SCHULTZ J., MOTT R., CICCARELLI F., COPLEY R.R., PONTING C.P. and BORK P. (2002). *Recent improvements to the SMART domain-based sequence annotation resource* Nucleic Acids Res. **30**: 242-244

[16] RILEY M. and LABEDAN B. (1997) *Protein Evolution Viewed through Escherichia coli Protein Sequences: Introducing the Notion of a Structural Segment of Homology, the Module*. Journal of Molecular Biology **268**:857-868

[17] SCHWARTZ R. M. and DAYHOFF M. O. (1978). Matrices for Detecting Distant Relationships, p. 353-358. *In* M.O. Dayhoff ed., *Atlas of Protein Sequence and Structure*, vol.5, suppl.3. National Biomedical Research Foundation, Washington, D.C

[18] THE GENE ONTOLOGY CONSORTIUM. (2000). *Gene Ontology : Tool for the unification of biology*. Nat. Genet. **25**:25-29