Published in the proc. of UIDIS'99 international workshop

The Design of Query Interfaces to the GPCRDB Biological Database

Dunren Che, Karl Aberer and Yangjun Chen
GMD-IPSI, Dolivostr. 15, 64293 Darmstadt, Germany
{che, aberer, yangjun}@darmstadt.gmd.de

Abstract

In this paper, we describe the query interfaces of a practical biological database system - GPCRDB. Distinguishing features of the system include: an embedded smart query engine (for query relaxation), smooth integration of navigation with the more conventional SQL-based query mechanisms, and the top-down style of incremental query result presentation combined with flexible navigation capabilities. Query relaxation is important due to the fact that queries tend not to be expressed exactly by the users, particularly when complex domain knowledge is involved. Navigation capability is desired because it can be an ideal supplement to SQL-based query mechanisms when large, complex data sets are concerned, especially in the WWW environment where hyperlinks are heavily used. Top-down incremental presentation is one of the best ways for a user to conduct the data presentation/retrieval process more reasonably and efficiently toward the point of interest of the user without being lost in (unwanted) details

1. Introduction

Interfaces to databases have been recognized as one of the most important topics for future research by the DB community [KB96]. Interface design can overwhelm the effects of better or worse database engines [Ch96]. It is "best considered not as science, but as craft", as argued by Matthew Chalmers [Ch96]. Consequently, one of the best ways of dealing with this issue is to *conquer it by doing it*. In this paper we report our experience in the development of the GPCRDB query system, and in particular its interfaces. The conclusions we derived from this experience are believed to be good guidelines for our future work, and are hopefully also of some reference value to others involved in similar areas.

GPCRDB has been developed as an advanced data management system for G-Protein Coupled Receptors (GPCRs) within the European Bioinformatics Program. Nowadays, biological databases are receiving more attention due to the practical use of them for human society. A variety of biological databases have been developed that provide database support for both the research and the application in different biological disciplines, e.g., GDB [PMF92], GenBank [NCB92, WZ98], GSDB (Genome Sequence Data Base), GCRDb [GC98], GPCR mutant database [GPM98] and GPCRDB [GPC98]. Protein and DNA sequence data are the primary data that reside in these databases while various related data such as annotations, mutant information, and physico-chemical characteristics are often added to the databases along with the main sequence data. Although these systems are usually equipped with information retrieval facilities to help biologists to search for specific information relating to their domain problems, two issues still bother the users frequently: they are either overwhelmed by an unreasonably large amount of matching data or frustrated by the "nothing found" response of the systems.

To deal with the first issue in the GPCRDB query system, we adopt the so-called *top-down* result organization style combined with flexible *navigation* mechanisms. In this way, our system first presents to a user a top-level *summary page* containing only very general information about the matching data. In addition, a variety of hyperlinks are embedded that can help the user navigate to a more detailed result page concentrating on a specific point of interests of the user or to other related data pages. Furthermore, the family tree (a classification hierarchy of protein sequences) in our system serves as a reasonable navigation hierarchy supporting the navigation capability.

To deal with the second issue, we developed a so-called *smart query engine* (embedded in summary pages) that can intelligently and iteratively relax an input query exploiting pertinent domain knowledge, and let users get more relevant data regarding an input query if that is desired.

The approaches adopted are especially suitable for GPCRDB. In this system, queries submitted by users are usually domain knowledge related and users may often fail to appropriately formulate the queries to exactly match their information needs. *Top-down* presentation combined with flexible *navigation* gives users the convenience to reasonably and efficiently attain the data sub-spaces just fitting their interests in an *explorative* manner, while the most criticized aspect - low efficiency - of navigation is well compensated for by its combination with the more conventional SQL-based query mechanism. Within this framework, navigation is subsidiary to SQL queries, i.e., SQL queries are evaluated to data sub-spaces, from which navigation starts. *Smart query relaxation* is important to data-intensive systems that involve complex domain knowledge (this is increasingly common with scientific databases in particular) since even a rather experienced domain expert may have difficulty in enumerating all the alternate terms regarding his/ her query (condition) according to the domain knowledge, and manually reformulating the query.

Essential features of our query system and interfaces are the following:

(1) The *top-down* presentation style is adopted in the query system that allows a user to first get a general impression about the results of his/her query as quickly as possible. Then the initial result can be used as a piece of starting-out knowledge to efficiently conduct the later-on presentation/retrieval process toward the user's particular interest.

(2) *Navigation* capability is smoothly integrated into the query system. Under the *top-down* presentation framework, *navigation* helps a user quickly become familiar with the complex domain knowledge and the relevant data (sub-)space.

(3) A *smart query engine* is developed and integrated into the query system that exploits pertinent domain knowledge.

(4) *Over-adaptation* is a common problem with regard to adaptive systems. In our case, this issue mainly concerns *over-relaxation*, and is overcome by the *iterative* relaxation approach.

Different from most related work, in our system we integrated many good design criteria (or ideas) proposed recently such as query relaxation/restriction [YK98], user-friendliness [LC96], incremental information presentation [Tara96], and reconciliation of prompted and unprompted system behavior [BMT96], as well as reuse of history queries [Chal96]. We proposed a top-down presentation style and smoothly integrated a flexible navigation capability into this framework. Our query relaxation technique [CCA99] is also different - it uses more deep domain knowledge and is accomplished through an iterative approach, by which over-relaxation is prevented. History queries can also be reissued, re-edited or combined (forming new compound queries).

The remainder of the paper is organized as follows. First, in Section 2, a general description of the system is given. Then, in Section 3, we describe our advanced query interfaces that are believed to be representative and can reflect most of the design styles and features of our system. Last, some conclusions drawn from our experience are set forth in Section 4.

2. Overview of GPCRDB query system

In this section, we describe the GPCRDB query system in general. Specifically, we describe the system's goals, GPCR data features, query system architecture, and the main functions implemented.

2.1. Goals

The biological goal of the GPCRDB project is to rationalize the drug discovery process. The system will be routinely queried by academic users (biologists) and industrial users (pharmacists) and is intended to be accessible on the WWW. Within this cooperative project we are mainly responsible for the query system.

We can figure out from the expectations of the intended users (biologists) the goals that our system is to achieve. They are summarized as follows:

(1) Appropriate handling of various, domain-specific, *non-standard* data types, e.g., sequence data, secondary structures (sub-structures of sequences), and hierarchical classifications.

(2) A simple, intuitive user-computer interface.

(3) Flexible query processing, in particular, *smart query relaxation* when the users expect to get more relevant matches.

(4) Support for a suitable amount of *user interaction* so that the users can conduct the relaxation process according to their particular interests.

(5) The capability of *navigation* among the obtained result sets of queries. Navigation gives users the convenience of drilling into any specific data sub-spaces of a large result set.

Navigation is critical to the GPCRDB query system because various hyperlinks (and cross references) are heavily used in the GPCRDGB environment, and complex, large data items of non-standard data types are frequently involved in query results that are best put into separate pages and accessed through navigation.

Accordingly, we designed simple *form-based* access interfaces to the database under the *top-down* presentation style, where complex underlying query functions can be easily manipulated by the users.

2.2. GPCR Data Features

To show the features of our system, we need briefly to clarify the data features of GPCRs and some important notions exploited in our system.

Data to be handled in GPCRDB involves various domain-specific data items of *non-standard* data types and, in particular, of large volume and complex structure. Among others, secondary structure motifs, the 3D snake-like plots of the GPCRs, mutants, ligand (or binding information), and a variety of annotations made by the domain experts, the hierarchical classification information, as well as the GPCR protein sequences themselves, are the important data items to be dealt with by the GPCRDB query system.

Queries (for sequences or related data) can be concerning any such features or the sequences. In the following, we explain three important concepts regarding the features of GPCRs, namely, *fuzzy equivalence classes*, *family tree*, and *generalization hierarchy*, based on which, our query system accomplishes one essential function - *intelligent query relaxation* (exploiting pertinent domain knowledge for this purpose).

Materially, GPCRs proteins are sequences of residues. To denote different residue types, a set of characters has been used by the domain experts with each representing a specific residue type. Suppose $\mathfrak{R}$ is such an alphabet. From an abstract point of view, we simply define our GPCRDB as a triple: $<S, H_F, H_G>$, where S is a set of sequences belonging to $\mathfrak{R}^*$, H_F is a family tree imposed upon S and H_G is a thesaurus hierarchy - a kind of generalization hierarchy.

As similarity generally exists among the residue types, alphabet $\mathfrak{R}$ can be partitioned into a certain number of the so-called *fuzzy equivalence classes* (***FEC***) $I_1, ..., I_m$, $m \geq 1$. (Currently, $m = 35$, and overlaps meaningfully occur among the partitions.) For each *FEC* I_k, a membership function f, playing the role of a *belief factor*, is associated, e.g.,

I_1: {A};	$f(\{A\}) = 1$.
I_2: {S, T};	$f(\{S, T\}) = 0.9$.

In terms of the biological classification, S can be partitioned into several subsets which, in turn, can be organized into a hierarchy, called a *family tree* (H_F). Each leaf node of the tree corresponds to a subset of S, while the root node covers all the GPCR sequences stored in the database.

On the other hand, we can also group S according to a hierarchically organized thesaurus, i.e., the so-called *generalization hierarchy*, H_G. Each node in H_G corresponds to a term affiliated into the thesaurus, and is associated with a set of synonyms that constitute another kind of *fuzzy equivalence class*, characterizing one type of similarities. Further similarities exist between both sibling nodes and parent-child node pairs. All the information is used as sound criteria for relaxing (and restricting) query conditions. Different from H_F, overlaps are common with regard to the partitions made according to H_G.

In addition to *set* partitions, each individual sequence is partitioned into a number of sub-sequences, called *secondary structures*, reflecting specific biological properties. Very often, queries are formulated using such sub-sequences as search conditions.

2.3. System architecture

As an implementation platform we chose the Informix Universal Server (IUS) (see [IUS97]) database management system. For illustration, the architecture of the GPCRDB query system is depicted in Fig. 1. It is designed to run on the WWW.

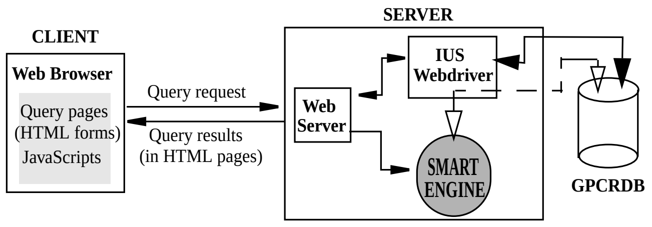


Fig. 1. GPCRDB query system architecture.

On the client side, users formulate their queries by simply filling out the query forms embedded in HTML pages. When submitted, the queries are first preprocessed (optimized) on the spot by calling a corresponding *Java script*. Valid queries are then transmitted to the Web server that further forwards the queries either to the IUS Webdriver or to the ***smart query engine***. The IUS Webdriver works as a gateway between Informix databases and the Web server while the smart query engine performs query relaxation and evaluation optimization [CCA99].

Queries submitted to the Web server are normally interpreted by the IUS Webdriver. However, if a user activates the smart query engine explicitly, control will be taken over by the smart engine. Implemented outside IUS, the smart query engine uses IUS only for retrieving data from GPCRDB database. It intelligently and iteratively relaxes a query by exploiting relevant domain knowledge, then presents to the user an enlarged result set as long as the query is relaxable according to some criteria [CCA99]. Users may also conduct the relaxation process along a specific direction by making additional choices from an additional option button (see Fig. 3) when the smart engine is activated.

2.4. Main functions

Two essential functions implemented in our system make our work different from other's [BMR96, KB96] and welcomed by our biologist users: smart query relaxation and flexible navigation capability. In the remainder of this section, we briefly address these two issues.

2.4.1. Query relaxation

Generally speaking, relaxation can be done either syntactically or semantically. Typical syntactic relaxation replaces a strict operator/term with a less strict one, e.g., an *AND* with an *OR*. For semantic relaxation, pertinent domain knowledge is used.

In our case, we concentrate on three aspects of relaxation: keyword (concept) generalization, residue pattern relaxation, and secondary structure expansion, which are all carried out according the domain knowledge.

Keyword generalization

Terms affiliated in the thesaurus are all carefully selected, with each representing a meaningful biological concept. The H_G hierarchy characterizes three kinds of relationships among the terms in the thesaurus: strong (between synonyms) and weak (between sibling nodes) similarities, and generalization (between a child-parent node pair). Consequently, for keyword relaxation (via an iterative process), first, we introduce the synonyms of a keyword into the query, next, try the left sibling (if any), then the right sibling (if it exists), and last, use the parent of the keyword as the substitute for it. This process repeats until the expected results of a user are obtained or the keywords involved are not further generalizable, i.e., neither synonyms, brothers nor parents exist.

Residue pattern relaxation

Residue pattern relaxation concerns the matching conditions of both secondary structures and the entire sequences (Residue patterns are represented as regular expressions in our system.) This kind of relaxation is desired due to some biological similarities existing between different residue types. This type of domain knowledge has been formally defined [CCA99] as *fuzzy equivalence classes* of residue types for efficient implementation of residue pattern relaxation. We summary the basic ideas of this type of relaxation as follows:

1. Generate all *fuzzy regular expressions* (***FRE****s*) with regard to an input residue pattern by replacing certain residue type(s) with corresponding *FEC(s)*.
2. Calculate the *belief factor* of each generated *FRE,* i.e., the minimum of all the belief factors assigned to each *FEC* involved.
3. Sort the *FREs* according to belief factors.
4. Generate the first alternative relaxation of the input residue pattern by selecting the *FRE* possessing the largest belief factor (during next iteration, select the next largest one).

This process iteratively repeats until no further alternative is available.

Secondary structure expansion

For queries involving secondary structures, in addition to residue pattern relaxation, the borders of secondary structures may also be expanded. That is, if this function is switched on, during each relaxation iteration, each of the two borders of an involved secondary structure gets an expansion of one residue position. This process can be repeated up to five times in our current implementation as further expansion is considered less interesting from a biological viewpoint.

2.4.2. Navigation

Navigation among the data space of a database system has been widely accepted as a very useful supplement to conventional SQL-based query mechanisms. The two strategies represent completely different styles of accessing information repositories. Navigation is more explorative and user-friendly compared with language based mechanisms, but is criticized for its low efficiency. The two strategies are expected to benefit from the combination of them in one system which appears still to be a knotty problem for designing specific query systems, and quite often to be more craft than science.

To smoothly integrate the navigation capability into our query system, random navigation within the entire data space is not allowed. That is, navigation in our system is restricted in the following way:

(1) Navigation within the sequence space is only allowed along the family tree, i.e., climbing up (to visit a super family) or drilling down (to focus on a specific subfamily). For both, a starting point (i.e., family tree entrance) needs first to be located (in the way described below).

(2) The family tree entrances are determined by the evaluation of a SQL-formulated query, i.e., a SQL query is first optimized and evaluated and the results obtained are then grouped according to their family information serving as entrances into the family tree.

In other words, the navigation supported by our system is first among the result set obtained, then along the family tree which the obtained result set serves as the navigating entrances for visiting all related super- and sub-families.

Navigation among the result set of a SQL query is supported as follows: diverse hyperlinks in the first summary page (at the top level) is offered with each pointing to a sub-summary page (at a lower level). Each sub-summary page contains all the found matches of the query of a specific subfamily (corresponding to an entrance node into the family tree). Each sub-summary page at a further lower level may again contain various hyperlinks, e.g., links to relevant entrances in a related (remote) GPCR database such as tinyGRAP [GPM98], and links to complex data fields of the sequences such as the 3D structures and the sequence data themselves which are normally large and cannot well fit into the same page.

3. Interfaces

In addition to the domain requirements mentioned in 2.1, attention should also be paid to some epistemological and psychological principles to have the intended users work with the system easily. Following are the additional elements of our consideration:

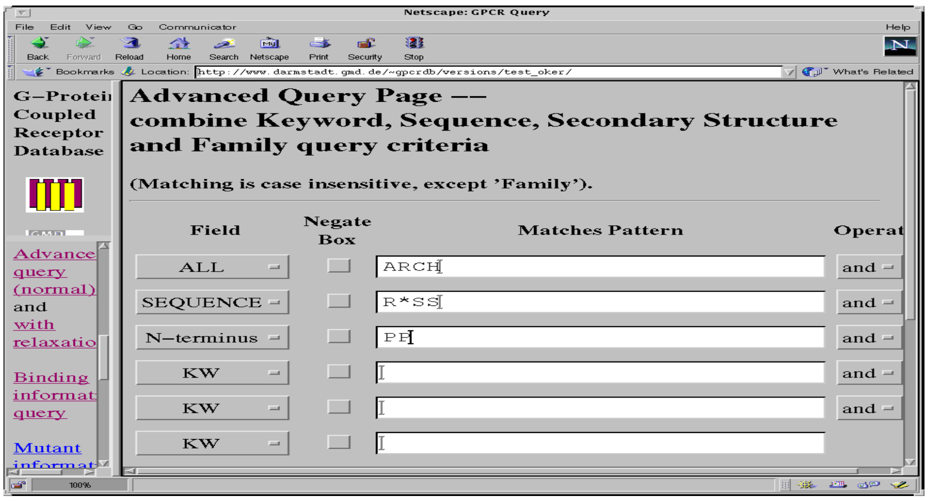
1. A frequently encountered troublesome situation during an information searching is the response of "*nothing found*" to a user query. In practice, frequent "nothing found" responses frustrate a user more than any other response and gives rise to the biggest complaint of a user although it may be due to the query being improperly formulated. To make the matter worse, he/she may lack any clue as to how to appropriately reformulate the query - this is especially the case in GPCRDB, which has complex domain knowledge. Intelligent query relaxation (reformulation) appears to be the right solution to this issue.
2. An important issue with regard to query relaxation is effective prevention of the so-called *over-relaxation*. In our system, *prompted iterations* have been adopted as a practical approach to this issue. (An action is *prompted* if it follows an explicit user request [BMR96].)
3. The behavior of a query system should first strictly conform to the original form of a user's query; the decision whether or not to apply the relaxation function on a query is best left to the user. Whether to accept some other's help is one of the basic rights of a person. Here, a useful metaphor for the role of our smart query engine is to only ask "May I help?" and be prepared to offer the help to a user.
4. Navigation has to take place in the *right situation* and at the *right time*, i.e., when it is just necessary. Smooth integration of this function into a language-based query system is still a knotty problem for designing a concrete system.

The GPCRDB query system allows users to query the GPCR database through a variety of ways: "family", "keyword", "secondary structure" (sub-sequence), "ligand" (binding information), "mutants". In addition, an advanced (compound) query page is provided for formulating query expressions consisting of up to six different conditions that can be logically interconnected with "$\wedge$" or "$\vee$", constituting an "and-or" normal form. For example, if $c_1 \vee c_2 \wedge c_3$ is input, it will be treated as $c_1 \vee (c_2 \wedge c_3)$. That is, "$\wedge$" takes precedence over "$\vee$".

To show our design style and the features of our query system, in the following, it suffices to narrow our description to only the advanced query page as that can be a representative for our query system.

The interface shown in Fig. 2 contains a query that can be restated as follows:

*Find GPCR sequences that have the keyword 'ARCH' (contained in any one of the annotation fields; to each sequence a number of annotation fields are attached) and contain the pattern 'R*SS'; furthermore, the secondary structure (sub-sequence) "N-terminus" of the sequences contain the pattern 'PP'.*


Netscape: GPCR Query
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: What's Related
G–Protei
Coupled
Receptor
Database
Advance query (normal) and with relaxatio
Binding informat query
Mutant informat
100%

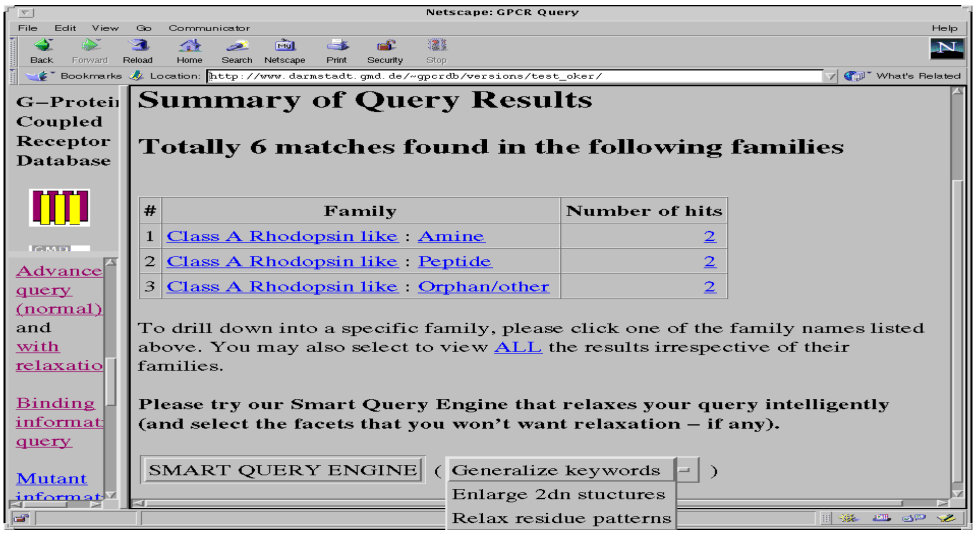
http://www.darmstadt.gmd.de/~gpcrdb/versions/test_oker/

Advanced Query Page ––
combine Keyword, Sequence, Secondary Structure and Family query criteria

(Matching is case insensitive, except 'Family').

Field	Negate Box	Matches Pattern	Operat
ALL	☐	ARCH	and
SEQUENCE	☐	R*SS	and
N–terminus	☐	PP	and
KW	☐		and
KW	☐		and
KW	☐		

Fig. 2. The Advanced query page (forms) of GPCRDB.


Netscape: GPCR Query

Location: http://www.darmstadt.gmd.de/~gpcrdb/versions/test_oker/

G–Protein Coupled Receptor Database

Advance query (normal) and with relaxation

Binding information query

Mutant information

Summary of Query Results

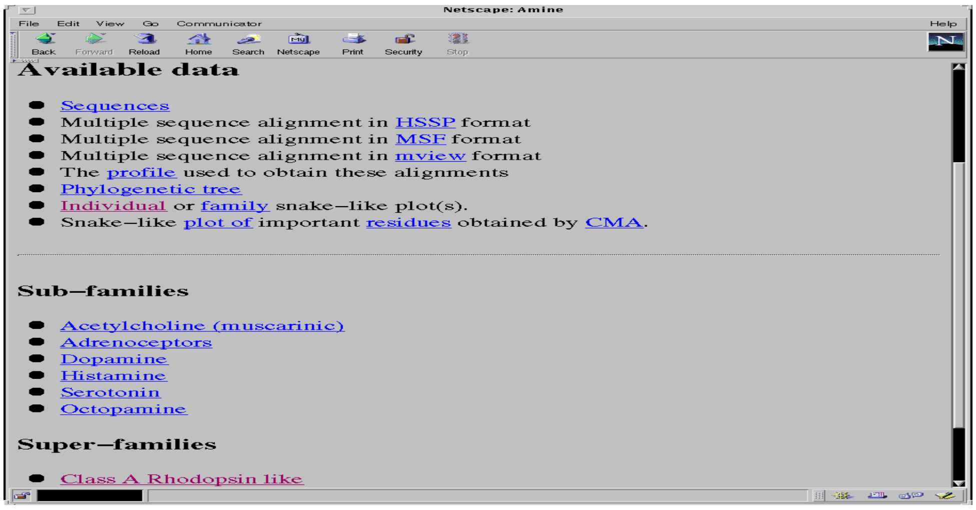
Totally 6 matches found in the following families

#	Family	Number of hits
1	Class A Rhodopsin like : Amine	2
2	Class A Rhodopsin like : Peptide	2
3	Class A Rhodopsin like : Orphan/other	2

To drill down into a specific family, please click one of the family names listed above. You may also select to view ALL the results irrespective of their families.

Please try our Smart Query Engine that relaxes your query intelligently (and select the facets that you won't want relaxation – if any).

Fig. 3. Result page of ordinary query (without relaxation).


Netscape: Amine

Available data

- Sequences
- Multiple sequence alignment in HSSP format
- Multiple sequence alignment in MSF format
- Multiple sequence alignment in mview format
- The profile used to obtain these alignments
- Phylogenetic tree
- Individual or family snake–like plot(s).
- Snake–like plot of important residues obtained by CMA.

Sub–families

- Acetylcholine (muscarinic)
- Adrenoceptors
- Dopamine
- Histamine
- Serotonin
- Octopamine

Super–families

- Class A Rhodopsin like

Fig. 4. Family tree entrance page.

The first response of the query system to an input query is a top-level summary page that presents to the user links to various pages containing more details about all hits of the query (the matching sequences). Fig. 3 is an example of such a summary page. In the displayed *Family* column, two links are provided pointing to the top level family and a subfamily, respectively, and serve as entrances to the family tree. In the *number of hits* column, the numbers (also as links) are clickable, navigating to a lower level summary page about the matches found in a specific subfamily. This summary page also contains a button at the bottom that the user can use to invoke the *smart query engine* to relax the query if he/she wants to get more relevant results. And of course, "going back" to the query input page (i.e., Fig. 2) is always supported via an additional option button (not displayed in Fig. 3) below the smart engine button for the user to manually reformulate his/her query if he/she prefers that.

Suppose the "Amine" subfamily (hyperlink) has been clicked; the user will get a response page as shown in Fig. 4. This page serves mainly as an entrance point into the family tree where all relevant sub-families and super-families are listed as clickable links, by which the user can easily navigate to any one of the listed super-families and/or sub-families, as well as the sibling families (by first climbing up to a super-family, then down to a sub-family). In addition, with regard to the current (sub)family - "Amine", the "available data" section in this page collects important information

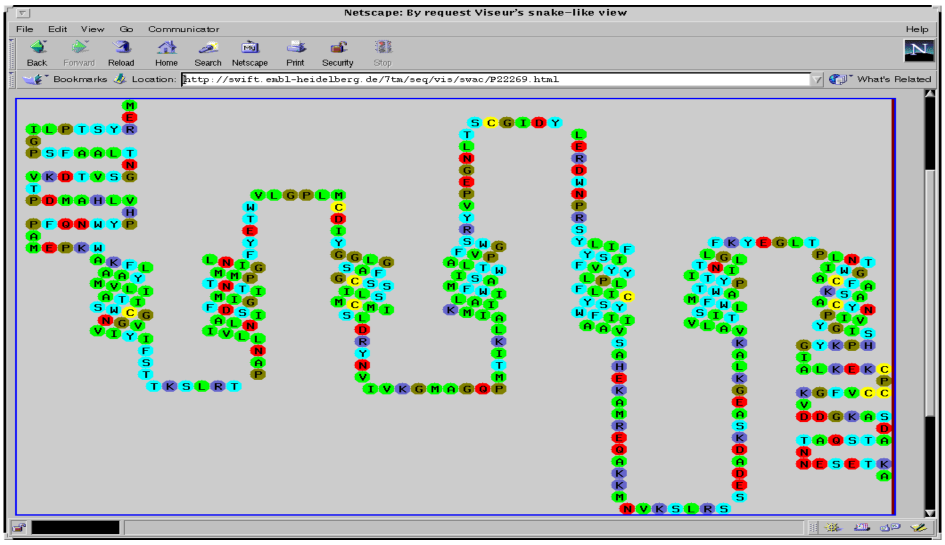
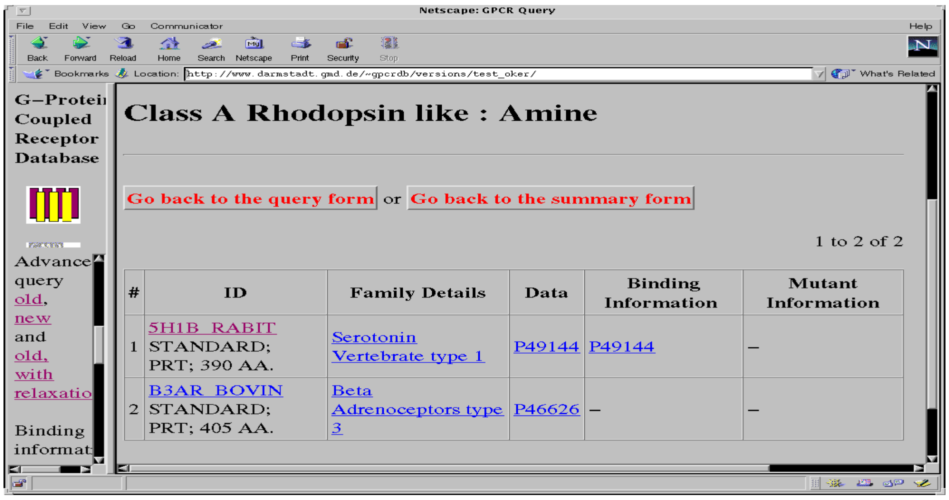

Netscape: By request Viseur's snake-like view
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://swift.embl-heidelberg.de/7tm/seq/vis/swac/P22269.html
What's Related

Fig. 5. 3D structure (snake-like plot) of a found sequence.


Netscape: GPCR Query
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.darmstadt.gmd.de/~gpcrdb/versions/test_oker/
What's Related

G–Protei
Coupled
Receptor
Database

Advance
query
old,
new
and
old,
with
relaxatio

Binding
informat

Class A Rhodopsin like : Amine

Go back to the query form or Go back to the summary form

1 to 2 of 2

#	ID	Family Details	Data	Binding Information	Mutant Information
1	5H1B_RABIT STANDARD; PRT; 390 AA.	Serotonin Vertebrate type 1	P49144	P49144	–
2	B3AR_BOVIN STANDARD; PRT; 405 AA.	Beta Adrenoceptors type 3	P46626	–	–

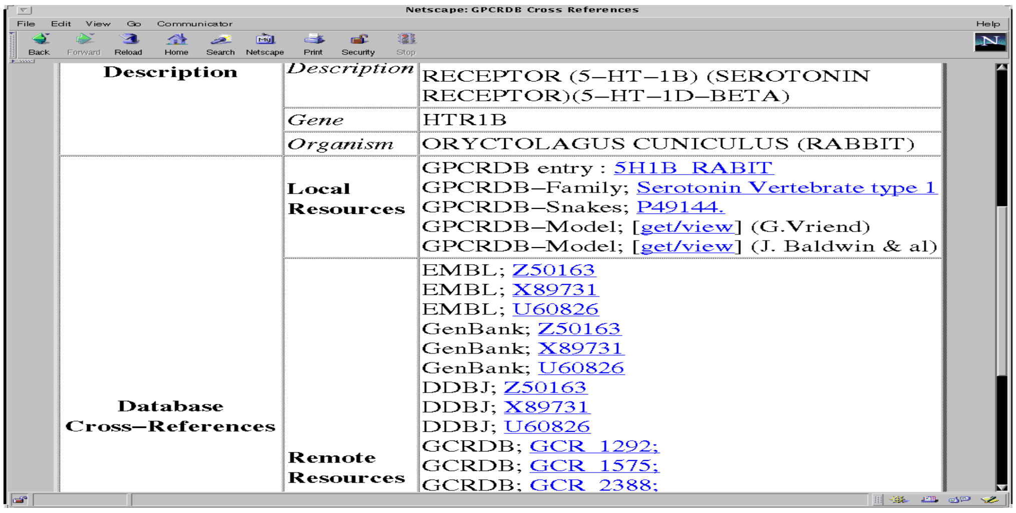
Fig. 6. One sample subsummary page of query results.

about the family (rather than only the individual sequences found), e.g., all available sequences of this family, and individual or family snake-like plots of the sequences.

Snake-like plots of sequences are 3D structures. If the user clicks the “family” snake link (Fig. 4), the corresponding snake-like plot will be displayed as shown in Fig. 5.

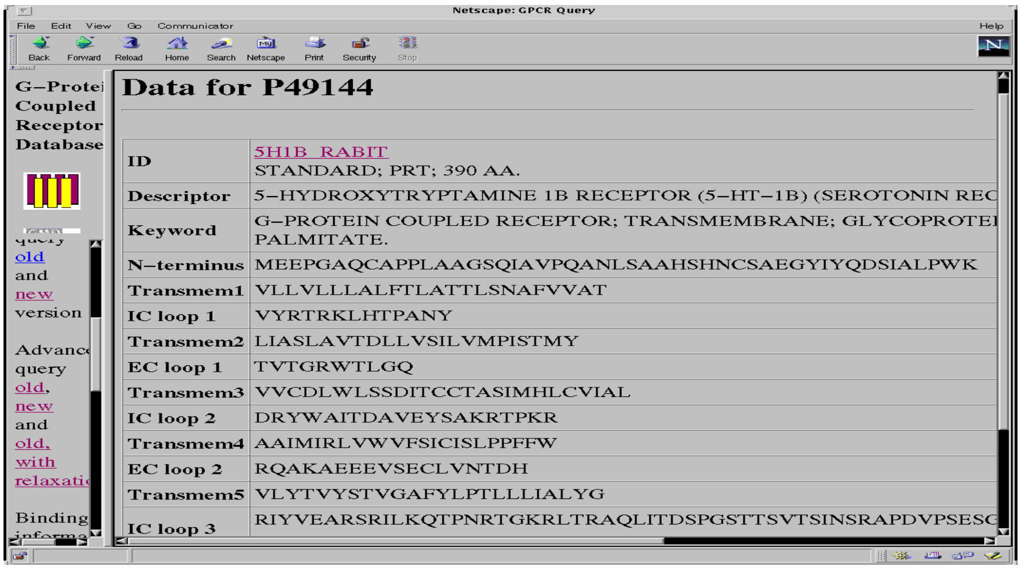
Each of the clickable numbers listed in the *number of hits* column indicates how many matches have been found in the (sub)family displayed in the *family* column at the same line (see Fig. 3). If, say, the number “2” displayed at the first line of the summary table, is clicked, the user will navigate to a lower-level summary page as shown in Fig. 6. In this page, the IDs displayed in the *ID* column are external IDs of sequences, and the first one in each cell is again a hyperlink, and if clicked, will lead to another page called the *cross reference* page (see Fig. 7) that contains all available references (also hyperlinks) to related local and remote data resources; the *Family Details* column provides links to detailed family information about each found sequence; links in the *Data* column point to pages (see Fig. 8) containing discrete data items of sequence; the links displayed in *Binding Information* and *Mutant Information* columns provide access to the binding and mutant information pages, respectively, if that information exists.

To invoke the smart query engine, a user simply clicks on the corresponding button (Fig.3). Before that, he/she may


Netscape: GPCRDB Cross References
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop

Description	*Description*	RECEPTOR (5–HT–1B) (SEROTONIN RECEPTOR)(5–HT–1D–BETA)
	Gene	HTR1B
	Organism	ORYCTOLAGUS CUNICULUS (RABBIT)
Database Cross–References	**Local Resources**	GPCRDB entry : 5H1B_RABIT GPCRDB–Family; Serotonin Vertebrate type 1 GPCRDB–Snakes; P49144. GPCRDB–Model; [get/view] (G.Vriend) GPCRDB–Model; [get/view] (J. Baldwin & al)
	Remote Resources	EMBL; Z50163 EMBL; X89731 EMBL; U60826 GenBank; Z50163 GenBank; X89731 GenBank; U60826 DDBJ; Z50163 DDBJ; X89731 DDBJ; U60826 GCRDB; GCR_1292; GCRDB; GCR_1575; GCRDB; GCR_2388;

Fig. 7. A page of cross references to local and remote relevant data resources about a matched sequence.



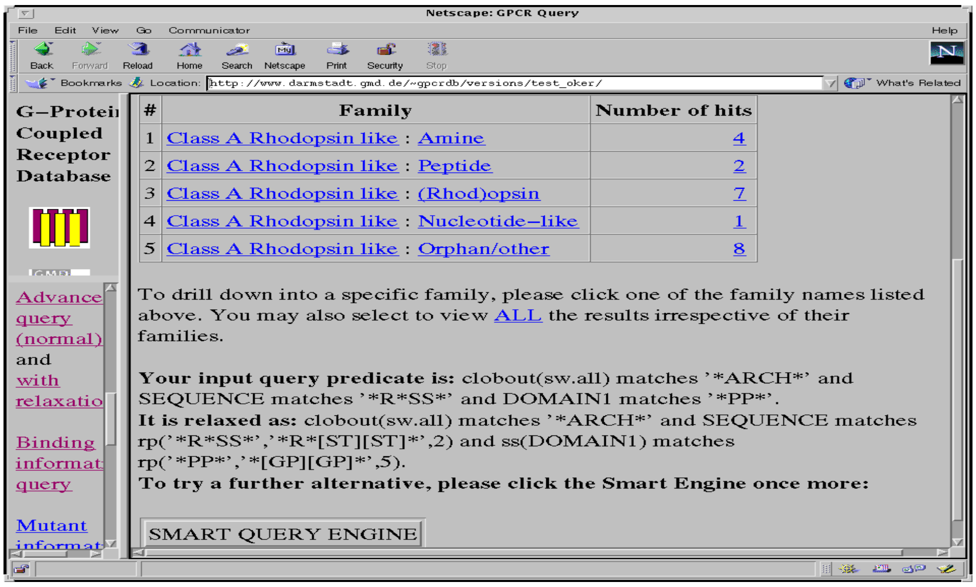
Data for P49144

ID	5H1B_RABIT STANDARD; PRT; 390 AA.
Descriptor	5–HYDROXYTRYPTAMINE 1B RECEPTOR (5–HT–1B) (SEROTONIN REC
Keyword	G–PROTEIN COUPLED RECEPTOR; TRANSMEMBRANE; GLYCOPROTEI PALMITATE.
N–terminus	MEEPGAQCAPPLAAGSQIAVPQANLSAAHSHNCSAEGYIYQDSIALPWK
Transmem1	VLLVLLLALFTLATTLSNAFVVAT
IC loop 1	VYRTRKLHTPANY
Transmem2	LIASLAVTDLLVSILVMPISTMY
EC loop 1	TVTGRWTLGQ
Transmem3	VVCDLWLSSDITCCTASIMHLCVIAL
IC loop 2	DRYWAITDAVEYSAKRTPKR
Transmem4	AAIMIRLVWVFSICISLPPFFW
EC loop 2	RQAKAEEEVSECLVNTDH
Transmem5	VLYTVYSTVGAFYLPTLLLIALYG
IC loop 3	RIYVEARSRILKQTPNRTGKRLTRAQLITDSPGSTTSVTSINSRAPDVPSESG

Fig. 8. The discrete data page of a matched sequence.

make extra selections from an additional option button just beside the *smart engin*e button (see Fig. 3). This additional button allows a user to specify one or two aspects of his/her query that should not be relaxed. This means that the three kinds of relaxation (each operating on a different aspect of queries) discussed in the last section can be freely switched on/off. By default, all three kinds of relaxation are to be performed on a query when the smart engine is started.

Once activated, the smart query engine will return to the user the first alternative (query results) of the relaxation of his/her query, as is illustrated in Fig. 9 (where the *keyword generalization* aspect has been switched off). With the same input query as inputted in Fig. 3, the result page of Fig. 9 contains a much enlarged set of matches (22 hits *vs.* the original 6 hits in Fig. 3). This page also presents the original query predicate and the relaxed (and just executed) one as an *explanation* to the accomplished relaxation. This information helps the user not only to understand the relaxation performed but also to formulate more pertinent queries in the future. In the presented relaxed query predicate (see Fig. 9), expression *rp(‘*R*SS*’, ‘*R*[ST][ST]*’, 2)* shows that the input residue pattern ‘**R*SS**’ is relaxed as ‘**R*[ST][ST]**’ using the fuzzy equivalence class numbered 2 (i.e., residue type ‘*S*’ is replaced in this pattern by its fuzzy equivalence class ‘*[ST]*’), and *ss(DOMAIN1)* indicates that the border expansion for secondary structure *DOMAIN1* in this query has been performed


Netscape: GPCR Query
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.darmstadt.gmd.de/~gpcrdb/versions/test_oker/
What's Related
G–Protei
Coupled
Receptor
Database
Advance
query
(normal)
and
with
relaxatio
Binding
informat
query
Mutant
informat
SMART QUERY ENGINE

#	Family	Number of hits
1	Class A Rhodopsin like : Amine	4
2	Class A Rhodopsin like : Peptide	2
3	Class A Rhodopsin like : (Rhod)opsin	7
4	Class A Rhodopsin like : Nucleotide–like	1
5	Class A Rhodopsin like : Orphan/other	8

To drill down into a specific family, please click one of the family names listed above. You may also select to view ALL the results irrespective of their families.

Your input query predicate is: clobout(sw.all) matches '*ARCH*' and SEQUENCE matches '*R*SS*' and DOMAIN1 matches '*PP*'.
It is relaxed as: clobout(sw.all) matches '*ARCH*' and SEQUENCE matches rp('*R*SS*','*R*[ST][ST]*',2) and ss(DOMAIN1) matches rp('*PP*','*[GP][GP]*',5).
To try a further alternative, please click the Smart Engine once more:

Fig. 9. The result page of a relaxed query returned by the *Smart Query Engine*.

once (if the border expansion has been performed twice, we will have *ss(ss(DOMAIN1))* instead of *ss(DOMAIN1)*). (Notice that *DOMAIN1* is the old name of *N-terminus* that is still used internally in our system, but an update will be made soon.)

4. Conclusions

In this paper, we described the GPCRDB query system and in particular its query interfaces. The query system embeds a smart query engine that takes charge of the query processing from the IUS SQL-based query engine upon user request - it iteratively relaxes (according to domain knowledge) and optimizes a query, then evaluates it and presents the results to the user in a *top-down* incremental presentation style. In addition, *navigation* capability is smoothly integrated into the system and the query interfaces. Navigation is conducted mainly by climbing-up or climbing-down the H_F hierarchy, by which any detail on any aspect of a hit relevant sequence can be obtained.

The GPCRDB [GPC98] and the query system are now operable and available on WWW. The last published working version can be invoked at URL: http://www.darmstadt.gmd.de/~gpcrdb/. The internal test version, which is under frequent updation, at the moment is also accessible on the Web (URL: http://www.darmstadt.gmd.de/~gpcrdb/versions/test _0499/).

Finally, we report some real-life experiences with regard to query interface design:

1) ***Top-down result presentation is a desirable presentation style to the users.***

The *top-down* style of query result presentation offers a user a general impression about the results of his/her query quickly (as only very general result information is presented, unnecessary network traffic and database burden are saved) so that the user won't be overwhelmed by the details of a large result set, and then can easily navigate toward the data subspace of their interests. The *summary page* is a good infrastructure to support this presentation style. In the GPCRDB query system, a summary page at the top level is first presented to a user who issued a query; the summary page involves the most general information about the results of the query and a variety of hyperlinks pointing to either lower-level summary pages containing more detailed information on specific aspects of the results or other related data resources available on the WWW.

For highlighting the traits of the "*top-down*" presentation style, let's use the metaphor of "finding interesting trees in a forest" - first, try to get some general knowledge about the forest as an overview; then using this knowledge efficiently localize to one part of the forest; next to a more restricted part, and so on, until the really interesting individual trees are reached.

2) ***Navigation can be a more desirable way of accessing databases if appropriately integrated with conventional query mechanisms.***

Navigation has been successfully used in the hypertext/hypermedia area. It has also been highly desired by the users of various data-intensive application systems. This mechanism also fits very well in the Web application environment where different information sources are inter-accessible through Web page links. Smooth integration of this functionality with the conventional SQL-based query systems is the key for its successful use in such systems. The way adopted by the GPCRDB query system is that *navigation is subsidiary to the conventional (language based) query mechanism*. That turns out to be a very satisfactory way - the low efficiency, the most criticized drawback of navigation, has been well compensated for by the SQL based query mechanism, while the more enjoyable aspects, e.g., user-friendliness and explorativeness, are maintained. Navigation performs even better when a domain-specific navigation hierarchy like our family tree is additionally provided.

Both the *top-down* presentation style and the *navigation* capability are specifically suitable for the applications that have large, complex data spaces and are associated with complex domain knowledge. They provide the greatest convenience for the users conducting information retrieval tasks toward their interests and solving their specific problems.

3) Intelligent query relaxation is the key ingredient of success.

Smart query relaxation is the key for a data-intensive system to ideally satisfy a broad spectrum of requirements of users, especially when the domain involves complex data set and is associated with complex domain knowledge. Equipped with such a facility, a query system will almost always present to its user a non-empty result set with more or less relevance if more exact matches for an input query cannot be found. Whether or not to apply relaxation to a query - the decision can only be made reasonably by the user according to the results obtained or at least after he/she gets a general impression about the results. So unprompted relaxation (without explicit user request) is not much desired. In the GPCRDB query system, the smart query engine that accomplishes intelligent relaxation is provided to the user as an embedded option in the top summary page of the obtained query results. The user always has the convenience of initiating the smart engine at will, sooner or later, according to the evaluation status of his/her original query.

4) *User controllable iterative query relaxation is a practical and effective way to prevent over-relaxation.*

A query system may suffer from *over-relaxation* when the system tries to relax a query. Therefore, the *user controllable iterative approach* is preferred since by it a user can stop a relaxation process at the end of an iteration once the expected match has been found.

In principle, after several iterations of a query relaxation process (each iteration tries one alternative), all relevant results (all relaxation alternatives) of a query will be obtained. That is, during each iteration, the smart query engine tries the relaxation along a different direction of relaxing the query and gets a correspondingly enlarged result set. The iterative relaxation approach adopted by the GPCRDB query system works exactly in this way. It incrementally presents to the user every alternative relaxation (and query results), and lets the user interrupt the process at the end of any iteration.

Acknowledgments. The authors are very glad to acknowledge that Hannelore Eisner has did a lot of implementation work in the GPCRDB query system. We also would like to show our gratitude to the many other partners of this project.

References

[BMR96]Brajnik, G., Mizzaro, S., Rasso, C., Evaluating User Interfaces to Information Retrieval Systems: A Case Study on User Support, in *Proc. of the 19th Anual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1996).

[CCA99] Che, D, Chen, Y. and Aberer, K., A Query System in a Biological Database. *11th International Conference on Scientific and Statistical Database Management*, Cleveland, Ohio, USA. July 28-30, 1999.

[Ch96] Chalmers, M., Interface Design: More Craft than Science? In *Proc. of the 3rd Int. Workshop on Interfaces to Databases*, Napier University, Edinburgh, 8-10 July 1996.

[GC98] GCRDb, URL: http://www.gcrdb.uthscsa.edu/.

[GPC98] GPCRDB, URL: http://www.sander.embl-heidelberg.de/7tm/.

[GPM98] *GPCR mutant database*, URL: http://www-grap.fagmed.uit.no/GRAP/homepage.html.

[IUS97] *Informix-Universal Server - Informix Guide to SQL*, Informix Press, Menlo Park, CA, USA, 1997.

[KB96] Kennedy, J. and Barclay, P. J. (Eds), *Proceedings of the 3rd International Workshop on Interfaces to Databases*, Napier University, Edinburgh, 8-10 July 1996.

[LC96] Lim, E.-P. and Cheng, S.-Y., LibSearch: A Window-Based Front End to Remote Bibliographic Databases on the Internet. In *Proc. of the 3rd Int. Workshop on Interfaces to Databases*, Napier University, Edinburgh, 8-10 July 1996.

[NCB92] National Center for Biotechnology Infromation, ENTREZ: Sequences User's Guide, *National Library of Medicine*, Bethesda, MD, 1992, Release 1.0.

[PMF92] Pearson, P., Matheson, N., Flescher, N., *et al*, The GDB human genome data base Anna 1992, *Nucleic Acids Research* 20 (1992), 2201-2206.

[Ta96] Tarantino, L., Hypertabular Representations of Database Relations in World Wide Web Front-Ends. In P*roc. of the 3rd Int. Workshop on Interfaces to Databases*, Napier University, Edinburgh, 8-10 July 1996.

[WZ96] Williams, H. and Zobel J., Indexing Nucleotide Databases for Fast Query Evaluation, in *Proc. of 5th Int. Conf. on Extending Database Technology*, Avignon, France, March 1998.

[YK98] Yoon, J. and Kim, S.-H. A., Three-Level User Interface of Multimedia Digital Libraries with Relaxation and Restriction, in *Proc. of IEEE Int. Forum on Research and Technology Advances in Digital Libraries* (*ADL'98*), April 22-24, 1998, Santa Barbara, California.