A Protein Patent Query System Powered By Kleisli

Jing Chen Limsoon Wong Louxin Zhang
BioInformatics Centre & Kent Ridge Digital Laboratories
21 Heng Mui Keng Terrace, Singapore 119613
Email: {cjing,limsoon,lxzhang}@krdl.org.sg

1 Introduction

Kleisli [5] is an integration technology that is rather suitable in the bioinformatics arena. Many bioinformatics problems (1) require access to data sources that are highly heterogeneous, geographically distributed, highly complex, constantly evolving, and high in volume; (2) require solutions that involve multiple carefully sequenced steps; and (3) require information to be passed smoothly between the steps. Kleisli is designed to handle these requirements directly. In particular, Kleisli provides the high-level query language CPL [4] that can be used to express complicated transformation across multiple data sources in a simple way.

We developed a prototype system to more effectively query protein patents. This system uses Kleisli to tie together the following sources to answer queries on protein patents that are considerably more demanding than simple free-text search: (1) the protein section of the Entrez system at the National Center for Biotechnology Information [11]; (2) the BLAST sequence homology service at the the National Center for Biotechnology Information [2]; (3) the WU-BLAST2 sequence homology software from Washington University [1]; (4) the Isite system at the US Patent and Trademark Office (`http://patents.uspto.gov`); and (5) the structural classification of protein database SCOP at Cambridge MRC Laboratory of Molecular Biology [8].

The system is aimed at these three questions: Is my protein sequence already patented? What are the prior arts? How to broaden my patent claims? The system provides the following information in response to these questions: It can find patented sequences that are similar to yours, it can try to determine the superfamily of your sequence and can find patented sequences that are similar to some sequences in the same superfamily, it can also find sequences that are similar to unpatented members in the same superfamily. An on-line demo of this system can be accessed at `//adenine.iss.nus.sg:8080/examples/patent`.

SIGMOD '98 Seattle, WA, USA

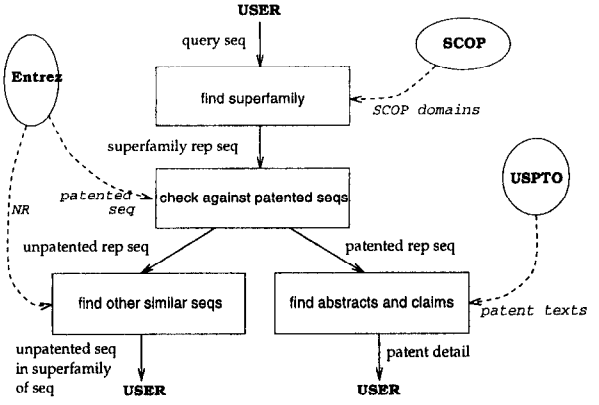

Figure 1: "Dataflow" of the patent query system.

2 Motivation

Consider a pharmaceutical company that has a large choice of protein sequences to work on (ie. to determine their functions.) A criterion for selection is patent potential. A process involving many data sources and steps is required; see Figure 1.

At the initial stage, we know only the amino acid sequence of these sequences and very little else. A question at this point would be: Which of them have already been patented? Existing patent search systems are IR systems that rely on English words. These systems suffer from the dichotomy of recall vs precision [6]. They either return only the highly relevant information at the expense of missing out a large proportion of them, or return most of the highly relevant information at the expense of returning also a lot of irrelevent information. So searching patents using them is laborious and is not always fruitful. Furthermore, at this early search, we do not even have any English word to use for searching—we have only the actual amino acid sequences! Thus, we need more reliable technology for comparing our protein sequences to those sequences already patented.

There are two things in our favour. First, protein patents are generally based on protein function *and* primary sequence structure (ie. the linear string of 20 letters in the amino sequence.) Second, bioinformatics is sufficiently advanced and tools that can reliably identify homology at the primary sequence structure level are available [10, 3]. Therefore, if patented protein sequences can be extracted from some database and prepared in a form suitable for such tools

```
1. webblast-blastp (#name: "nr-blast", #db: "nr", #level: 2);
2. localblast-blastp (#name: "patent-blast", #db: "patent-aa/blast/fasta");
3. localblast-blastp (#name: "scop-blast", #db: "scop-aa/blast/fasta")
4. seqindex-scanseq(#name:"scop-index", #index:"scop-aa/seqindex", #level: 1);
5. scop-add "scop";
6. readfile scop-summary from "scop-aa/data/summary" using stdin;
7. primitive scop-accn2uid == (set-index' (#accession, scop-summary)).#eq;
8. materialize "scop-accn2uid";
9. {(#title: z.#title, #accession: z.#accession, #uid: z.#uid,
10   #class: i.#desc.#cl, #fold: i.#desc.#cf, #superfamily: i.#desc.#sf,
11.  #family: i.#desc.#fa, #protein: i.#desc.#dm, #species: i.#desc.#sp,
12.  #scop-pscore: x.#pscore, #nr-pscore: z.#p-n)
13. | \x <- process SEQ using scop-blast, x.#pscore <= PSCORE-SCOP,
14.  \i <- process <#sidinfo: x.#accession> using scop,
15.  \sf <- process <#numsid: i.#type.#sf> using scop,
16.  \sfuid <- scop-accn2uid (sf),
17.  \y <- process <#get: sfuid.#uid> using scop-index,
18.  {} = { x | \x <- process y.#seq using patent-blast, x.#pscore <= PSCORE-PATENT },
19.  \z <- process y.#seq using nr-blast, z.#p-n <= PSCORE-NR };
```

Figure 2: Kleisli/CPL program to find unpatented sequences in the same superfamily as a user-supplied sequence.

to operate on, we will have a means to reliably identify which of our sequences have not yet been patented. We obtain the patented sequences from the protein section of Entrez [11]. These are "warehoused" locally for greater efficiency. We use WU-BLAST2 [1] for comparing our sequences against this warehouse for primary sequence structure homology.

After the unpatented protein sequences have been identified, the second question at this point is: Which ones of these have the potential for wider patent claims? To understand this question it is necessary to recognize that protein patents are generally granted on a sequence *and* its function. While we do not know the functions of our proteins, because we have not done work on them yet, we know that proteins of the same evolutionary origin tend to have similar functions even if they differ significantly in their primary structure [7]. Using the terminology of SCOP, these proteins are in the same superfamily [8]. So one way to identify protein sequences that have potential for wider patent claims is to find those having large number of unpatented sequences in the same families. Homology searching algorithms based on primary structure are generally not sufficiently sensitive to detect the majority of sequences in a typical superfamily [10], as the primary structure of distance members of the family are likely to have mutated significantly. So we need tools for homology at the tertiary structure level.

No reliable automatic tools for this purpose exists at this moment, because structural similarity at the tertiary level does not necessarily imply similarity in function. Nevertheless, reliable manually constructed databases of superfamilies exist. A very nice one is the SCOP database [8]. Therefore, if we screen our unpatented sequences against SCOP, we can pull out other *representative* sequences in their superfamilies and check which ones have already been patented, thus identifying superfamilies with good potential. We use WU-BLAST2 for the screening. After unpatented representatives of superfamilies have been found, it is still necessary to use them to fish out the rest of the unpatented members of superfamilies. This step can be accomplished by using BLAST [2] to remotely compare these representatives against the huge nonredundant protein database (NR) curated at the National Center for Biotechnology Information.

Having found these potentially good protein sequences, we are ready to work on them and to hopefully patent the results in due course. We are ready to ask the third question: What are the relevant prior arts? The examination of prior arts is necessary to determine directions when we begin work on our proteins, if we decide to work on them at all. Retrieving the texts of patented sequences in the same superfamilies as our proteins would be very helpful here. This step is complementary to the previous step and can be carried out uisng exactly the same technology!

3 Execution

We describe the query to find unpatented sequences in the same superfamily of a user-supplied protein sequence, as it is the most complicated of the three questions we mentioned. The Kleisli/CPL program is shown in Figure 2.

As several data sources are used, we connect up to them first. We establish a connection `nr-blast` to BLAST at National Centre for Biotechnology Information for searching its NR database (line 1). The concurrency level of this connection is set to 2 for more efficient parallel access. We establish a connection `patent-blast` to WU-BLAST2 for searching against our local warehouse of patented proteins (line 2). We establish a connection `scop-blast` to WU-BLAST2 for searching against our local warehouse of SCOP representative sequences (line 3). We establish a connection `scop-index` to the index of SCOP representative sequences (line 4). Both warehouses and the index were constructed previously using Kleisli. We establish a connection `scop` to the SCOP classification database (line 5). These different connections to SCOP are needed because the SCOP database (line 5) contains only names and classification but not the actual representative sequences. We keep our sequences using a proprietary sequence indexing technology SeqIndex [9]. This index (line 4) allows us to quickly retrieve a sequence given an identifier or a pattern. Unfortunately, WU-BLAST2 requires the sequences to be stored in a different format. Hence we need the third connection to SCOP (line 3). We have to deal with one more problem. The identifiers used by SCOP (lines 3, 5) are different from the identifiers used by SeqIndex (line 4). We need a map between these identifiers. This map is the relation `scop-summary` (line 6). For quick access we create a memory-resident index `scop-accn2uid` to map SCOP identifiers to

SeqIndex identifiers (lines 7-8).

After setting up connections to various data sources as described above, we are ready to issue our query to retrieve information about unpatented sequences in the same superfamily as our protein sequence SEQ. The information returned include title, accession, unique identifer, and classification of these sequences (lines 9–11). Also returned with each unpatented protein sequence is its *pscore* with respect to SCOP and NR (line 12). The pscore is a reliable estimate of the corresponding sequence being a false positive, given by BLAST and WU-BLAST2 [2, 1]. Eg., if BLAST returns a hit with pscore of 0.001 to your sequence, then there is a once in a thousand chance of the hit being a fluke.

Let us step through the body of the program. Given the user sequence SEQ, we compare it against representative sequences in SCOP; we keep only those hits x whose pscore is within the error threshold PSCORE-SCOP (line 13). Since each hit x is good, the superfamily of x can be taken as the superfamily of the input sequence SEQ. We find the superfamily of x by simply asking SCOP to return us the SCOP classification i of x (line 14). The name and identifier of x's superfamily is stored in the `#desc.#sf` and `#type.#sf` fields of i respectively. Next, we need to fish out all representatives of that family from SCOP. SCOP gives us their SCOP identifiers sf (line 15). We convert these identifiers into unique identifiers sfuid in the SeqIndex where the sequences are kept (line 16). The SeqIndex is then accessed to give us the actual representative sequences y (line 17). Each representative sequence y is compared against patented sequences. We retain those that have no hits within the error threshold PSCORE-PATENT (line 18). These are representatives of our superfamily that are dissimilar to every patented sequence. We compare them against the NR database to fish out all sequences z that are similar to them within the error threshold PSCORE-NR (line 19). These are all the desired unpatented sequences in our superfamily.

4 Demo

While it is easy to write Kleisli/CPL programs to query protein patent data, ordinary users would like a more friendly form-based interface. So in our prototype protein patent query system, we also provide form-based interface to our query programs. Figure 3 is a screendump of a form for finding unpatented proteins in the superfamily of a user-supplied protein sequences. When the user clicks on the "submit" button at the top half of the screen, his protein sequence and pscore thresholds are passed to the Kleisli/CPL program described in the previous section for execution. The results are then displayed at the bottom half of the screen, with links to various external sources for additional information. Similar forms to other programs in our protein patent query system have also been developed.

In conclusion, a high-level technology such as Kleisli [5] greatly simplifies the process of developing interesting integrated systems. Short and clear Kleisli/CPL programs that access multiple distributed databases and softwares can be easily written to return information on protein patents in a manner that is beyond the reach of existing patent information servers based on standard IR systems. Furthermore, convenient form-based interfaces can be put up to these programs.

Acknowledgements. We thank Steve Brenner for explaining and providing SCOP to us.

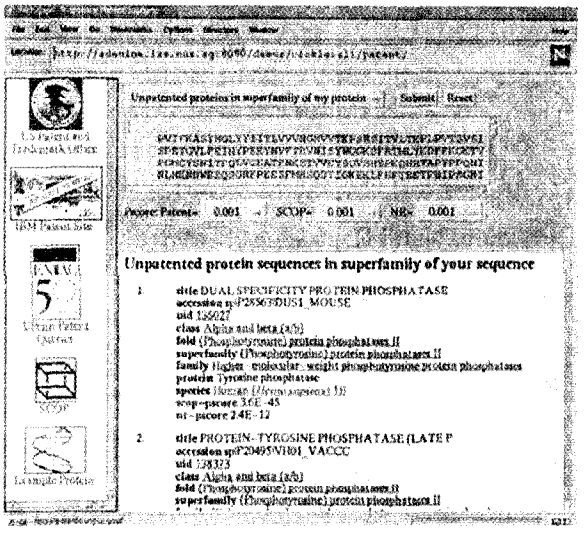

Figure 3: Screendump of the patent query system.

References

[1] S. F. Altschul and W. Gish. Local alignment statistics. *Methods in Enzymology*, 266:460–480, 1996.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[3] G. J. Barton and M.J.E. Sternberg. Evaluation and improvements in the automatic alignment of protein sequences. *Protein Engineering*, 1:89–94, 1987.

[4] P. Buneman, L. Libkin, D. Suciu, V. Tannen, and L. Wong. Comprehension syntax. *SIGMOD Record*, 23(1):87–96, March 1994.

[5] S. Davidson, C. Overton, V. Tannen, and L. Wong. BioKleisli: A digital library for biomedical researchers. *International Journal of Digital Libraries*, 1(1):36–53, April 1997.

[6] W. B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms.* Prentice Hall, 1992.

[7] H. Lodish, D. Baltimore, A. Berk, S. L. Zipursky, P. Matsudaira, and J. Darnell. *Molecular Cell Biology.* W. H. Freeman, New York, 1995.

[8] A. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: A structural classification of protein database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

[9] H.H. Pang, B.C. Ooi, and L. Wong. S-Hash: An indexing scheme for approximate subsequence matching in large sequence databases. Manuscript, October 1997.

[10] W. R. Pearson. Comparison of methods for searching protein sequence databases. *Protein Science*, 4:1145–1160, 1995.

[11] G. D. Schuler, J. A. Epstein, H. Ohkawa, and J. A. Kans. Entrez: Molecular biology database and retrieval system. *Methods in Enzymology*, 266:141–162, 1996.