Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006

PDB Data Curation

Yanchao Wang and Rajshekhar Sunderraman

Abstract— **In this paper, we propose two architectures for curating PDB data to improve its quality. The first one, PDB Data Curation System, is developed by adding two parts, Checking Filter and Curation Engine, between User Interface and Database. This architecture supports the basic PDB data curation. The other one, PDB Data Curation System with XCML, is designed for further curation which adds four more parts, PDB-XML, PDB, OODB, Protin-OODB, into the previous one. This architecture uses XCML language to automatically check errors of PDB data that enables PDB data more consistent and accurate. These two tools can be used for cleaning existing PDB files and creating new PDB files. We also show some ideas how to add constraints and assertions with XCML to get better data. In addition, we discuss the data provenance that may affect data accuracy and consistency.**

I. Introduction

It is well known that the Human Genome Project has resulted in rapidly growing databases in bioinformatics. And bioinformatics provides some opportunities to develop novel data analysis methods, but there are still some challenges including protein structure prediction, genomic sequence analysis, etc. For example, in protein structure prediction, scientists are using PDB file to apply amino acid sequence to determine the secondary and tertiary structure of proteins.

PDB (Protein Data Bank) is a centralized repository of protein structures founded in 1971 at Brookhaven National Laboratory, USA [8]. PDB file specifies the positions in space of every atom in a molecule which includes atomic coordinates, a header which gives information about the model embodied in the coordinates. The PDB files for X-ray crystallography vary widely in quality, and they are rarely totally incorrect. Unfortunately, PDB supports several different data formats, thus it suffers from inconsistencies in how the data formats are constructed over time. PDB files do not contain complete bond information for biopolymers, which chemical bonds must be reconstructed by the reading/viewing software based on chemistry tables and known bond rules. Since different software can interpret those rules differently, software can be inconsistent the way it draws bonds in PDB-based structures and may produce undesired redundancy [9]. For example, the sequence given in the PDB SEQRES records is compared against the sequence derived from the coordinate records that can show redundant data. In addition, the PDB is inconsistent in many other aspects, such as lack of EC numbers, compound name and inconsistent chain labeling [2].

PDB format is old, ambiguous, and inadequate, but it is still the most widely used format since all relevant software can read it. Although some users claim they do not mind the errors of PDB file, which may result in the partial or wrong information to affect the accuracy of derived and propagated data. Thus it is very necessary to develop a system to curate data in PDB file to provide easy understanding and better services for protein scientists and computer scientists.

In this paper, we propose two architectures for PDB data curation. One simply uses Checking Filter and Curation Engine to curate data. The other one improves the first one by adding XCML curation part to further curate data, which can deeply clean the PDB data by using more constraints. These two tools can be used not only for cleaning existing PDB files also for creating new PDB files.

This paper is organized as follows. First the PDB Data Curation System is explained in Section 2. Then Section 3 lists the issues of PDB data and the possible solutions. Next, Section 4 covers the architecture of PDB Data Curation System with XCML and some examples. Section 5 describes data provenance. Finally, conclusion and future work will be given in section 6.

II. The PDB Data Curation System

Figure 1 shows the architecture of PDB Data Curation System. From User Interface, Checking Filter gets PDB file, which checks the errors and reports them to Curation Engine. And then curated data by Curation Engine are sent to Database storage. The users can get better PDB file from our database after this process.

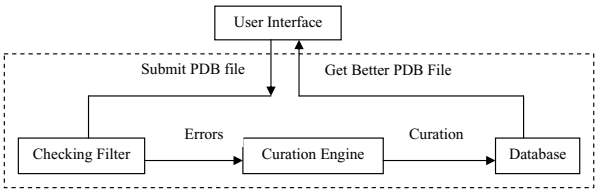

Fig. 1. PDB Data Curation System Architecture

Most errors are fixed in original PDB file after it is through this PDB Data Curation System. Although some curations are not automatic, they help improve the PDB data quality.

III. Major Issues of PDB Data

As we described above, the inconsistency, redundancy and other problems may occur in PDB file. Following shows these issues and possible solutions.

Y. Wang is with the Computer Science Department, Georgia State University, 34 Peachtree Street, Suite 1454, GA 30303, USA ywang17@student.gsu.edu

R. Sunderraman is with Faculty of Computer Science Department, Georgia State University, 34 Peachtree Street, Suite 1452, GA 30303, USA raj@cs.gsu.edu

A. Data Identification

PDB users usually want to know who generates the data, which paper, which lab, which date so that users can identify the confidence of PDB data. In PDB file, first 6 columns are not changeable in the process of data generation, but some users may change the spelling without any intention such as changing AUTHOR to AUTHRO, TITLE to ITTLE, and propagate these errors to others who pull out data from them. Therefore our system is designed to automatically check those spelling errors and provide users the authority to change them. In addition, our data curation system also gives information about missing data such as no HEADER, no AUTHOR, no Date and so on. Since header of PDB file is very important that contains information about the original literature citation, full names of ligands etc, it is very necessary to include it in PDB file. And author, paper and lab give us information about respects of different data resources which can confirm the confidence of data. Also date in PDB file headers shows the time that the atomic coordinates were deposited, modified and published at the Protein Data Bank, which helps keep data up-to-date. Therefore, identifying these data will give users very useful and helpful information.

B. Data Errors

1. Data errors in original PDB file: The users can report to our system if they find the errors of PDB file. After we get original authors' approval and correct them with marking reason, author, date for changed part, then we resubmit. The reason we do in this way is that different parts of PDB file have different authorities, some parts are generated automatically by system, some are only used by primary authors, some are managed by PDB staff, others are controlled by submitters.
2. Data errors generated when users change data: The size of PDB file is very huge such that users who are using them may remove or change some data without any intention. We store the original data and changed parts of PDB file in our storage to avoid this kind of errors. If users have changed data, the system sends alert signal which can avoid users to change data without any attention. But if users really want to change data, our system compares the current file and previous file to avoid the users to make mistake. The system will suggest users to use the previous data file if it finds the errors from new data file.
3. Data format errors: Our system retains a normal PDB file format which other PDB files can be very easily converted to. If the current PDB file has any incompatible items against it, system will report to the users such that users can correct them to make data conform to the common format.

C. Redundant Data

PDB data may be redundant which wastes storage and users' time, therefore our system provides filters to automatically or manually check data redundancy. The system can remove redundant data without any loss of accuracy of data to make sure that data file does not have repeatable data by using following method–the system firstly checks REMARK part of PDB file and sees whether there is redundant data, if it has, the system will compare protein sequence and atomic structure to get rid of duplicated part in protein sequence by applying structures matching method.

D. Ambiguous Data

PDB data may have different names for the same object. For example, some PDB files have ambiguous side chain atoms ("AE1", "AE2" atoms on Gly residues in 1ECO and 5CYT). It can be fixed by defining those ambiguous side chain atoms in a hash table which lists all different names for each object. The system will automatically check them when they appear in the protein structures. In addition, the different objects may have the same name in PDB file that are difficult to check. But we can check them by saving all corresponding objects for each name in a table. We manually check them according to the protein structures, if they are not the same object, we will give them different names.

E. Heterogeneous Data

There are so many PDB resources on the web sites. Different labs may apply different formats to save their experiments results which make them difficult be understood. Building a table to store the confidence of different labs provides the users convenience easy understanding to use the most trustful data to avoid confusion.

F. Inconsistent Data

The Validation Suite and Server (VSS) used by PDB can check the coordinate format and validate the overall structure before deposition which provides sequence/coordinate alignment and data inconsistencies [14]. For the simple inconsistent data such as the same protein has different names, which can be checked by collecting all synonyms in a dictionary, the system will automatically check them when PDB file is submitted into the PDB Data Curation System. For complicated data inconsistencies that are found by VSS of PDB, our system can manually amend according to the protein structures and data information. All data inconsistencies can be improved by changing PDB file to mmCIF format and converting mmCIF back to PDB file [10].

G. Conflict Data

Our system provides checker module which applies machine learning combing fuzzy method to check conflict data according to the protein data structures, and automatically correct them without users' involvement. According to three protein structures, the system can remove the conflict data in any protein structure. For example, the amino acids of primary structure can be checked against protein tertiary structure by using some extra constraints and assertions in section 4.

H. Obsolete Data

As we know, life science has a tremendous amount of data and updates very often. In order to satisfy the users' need, our system tries to keep following the pace of PDB data appearance to update old data and add new data types once they appear. For this part, the system will manually check the new data entries every month and store both if it finds the new one for existing data so that users can conveniently keep track of the provenance for specific data.

We remove the results for this PDB Data Curation System because of the page limit.

IV. The Architecture of PDB Data Curation System with XCML

The eXtensible Markup Language (XML) is self-describing and is standardized by the World Wide Web Consortium (W3C). It is human and machine readable, flexible, extensible and has already become the standard format for exchanging information through the network [13]. It usually uses XML Schema or Document Type Definition (DTD) to define the syntax and data types. The data source will generate XML data according to their Schema definition or DTD when data are parsed. But it can not verify semantic constraints, avoid erroneous data and is domain dependent. Therefore, we will not get any better data if we just convert PDB file into XML format as PDBj-ML [1] did. In order to avoid the disadvantages of XML and make the use of XML's merits to get better data, we apply an XML constraint language, eXtensible Constraint Markup Language (XCML)[13], which is more powerful and supports the specification of dynamic and inter-relationship constraints, to add some extra constraints and assertions in XML format so that we can further curate data during the data conversion.

The following architecture shows how to improve our PDB Data Curation System by using XCML. Firstly, PDB file is submitted to PDB Data Curation System, after curation PDB file can be converted into XCML format (PDB-XML) by ConverterXML which uses some constraints to further curate. PDB-XML can be in Protein-OODB [15] format after it is handled by ConverterPDB and ConverterOODB. At this moment, users can get better data that are in domain specific object-oriented format.

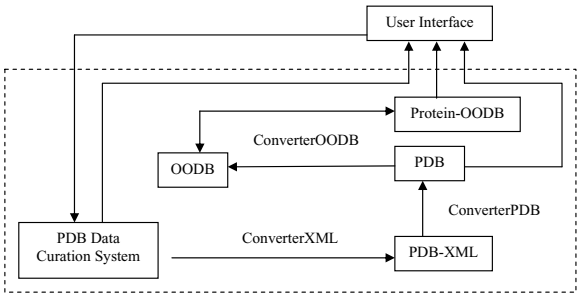

Fig. 2. The Architecture of PDB Data Curation System with XCML

The following will show some examples:

Example 1. We add one constraint to make sure that the length of each helix is equal to the difference of end position and begin position (pdbx_PDB_helix_length = end_auth_seq_id - beg_auth_seq_id + 1). We can check the correctness of length for each helix generated by using this constraint. But we need to define pdbx_PDB_helix_length, end_auth_seq_id and beg_auth_seq_id as integers instead of strings as PDBj_ML defined.

PDBj_ML format (Assume end_auth_seq_id and beg_auth_seq_id are already defined):

```
<xsd:element name= pdbx_PDB_helix_length
         minOccurs=0 maxOccurs = 1
         nillable=true type =xsd:string>
 ......
</xsd:element>
```

XCML format(beg_auth_seq_id and end_auth_seq_id constraints are not shown here):

```
<Constraint context=pdbx_PDB_helix
                    _length>
  <Parameter>
     <name>pdbx_PDB_helix_length</name>
     <type>number</type>
  </Parameter>
  <Assertion test=pdbx_PDB_helix_length
=($end_auth_seq_id-$beg_auth_seq_id+1)/>
</Constraint>
```

Example 2. We use constraints to ensure that the element of each sheet is exactly on the specific chain of protein sequences. PDBj_ML part:

```
<xsd:element name=pdbx_nonpoly_schem
  minOccurs = maxOccurs=unbounded>
<xsd:complexType>
  <xsd:all>
   <xsd:attribute name=ndb_seq_num
     use=required type=xsd:string>
     ......
 </xsd:all>
</xsd:complexType>
</xsd:element>
<xsd:element name= end_label_seq_id
    minOccurs=0 maxOccurs = 1
    nillable=true type =xsd:string >
</xsd:element>
```

XCML part:

```
<Constraint context=pdbx_nonpoly_schema>
  <Parameter>
     <name>ndb_seq_num</name>
     <type>number</type>
  </Parameter>
</Constraint>
<Constraint context=end_label_seq_id>
  <Parameter>
     <name>end_label_seq_id</name>
     <type>number</type>
```

```
  </Parameter>
<Assertion test=end_label_seqid=
                $ndb_seq_num/>
</Constraint>
```

The test part of assertion can check whether end_label_seq_id is equal to ndb_seq_num or not which can check the consistency of data.

V. Data Provenance

Data come from different sources and produced by different methods vary in the degree of real reliability. Also data that have been transformed multiple times are more likely to have been incorrectly transformed or lose an important context element. In order to evaluate data's reusability, it is necessary to understand for users the details of its collection. So data provenance is playing a crucial role in PDB file. In our system, we emphasize on solving the following two aspects of data provenance:

1. Same question different answers from different resources: As we know, the protein data may come from different experiments, computational techniques, and interpretation of primary data. This usually results in data sources using a variety of formats or referencing multiple data sources. Data sources themselves may include a lot of different information. Therefore same question may get different answers from different resources when we query from the web sites. The results from our system come from the most confidential data sources of our respects table. If the most confidential data is obsolete, the system will use greedy algorithm to search up-to-date data with the more confidence.

   ```
   Algorithm:
   Get result set S from data sources
   Check the confidence of set S
     according to the respectstable
     and get most confidential one MCS
   Loop
     If the MCS is obsolete, then find
       next most confidential one MCS
     Until get the more confidential
       and up-to-date result
   End loop
   ```

2. Uncertain probability: The PDB file has a very huge data size and is used by different-level users, there may be some errors in the file. And updates are often given to PDB file that may produce erroneous data by faulty experimental procedure or by a breakdown in the process to result in uncertain data and uncertain probability of data will propagate in the process. In order to solve this problem, we design a method to measure the uncertainty of data by using fuzzy method. In this method, we choose the most certain one for each data source from certainty table in order to get the most certain data from propagation process.

   ```
   Algorithm:
   For each level data source
      Use the biggest rate
      certainty/uncertainty from
      the certainty table
      Until the last level
   End for
   ```

VI. Conclusion and Future Work

In this paper, we propose two architectures for curating PDB data to improve its quality. One is PDB Data Curation System developed by adding Checking Filter and Curation Engine between User Interface and Database to curate PDB data which can detect all basic major data errors and amend them. The other curation architecture improves the first one by applying XCML to further curate PDB data, PDB Data Curation System with XCML, which adds four more parts, PDB-XML, PDB, OODB, Protin-OODB, into the previous one. This architecture can use XCML language to automatically check some errors of PDB file that enables PDB file more consistent and accurate. We also give some ideas by adding constraints and assertions with XCML to get better data. Finally, we present two data provenance issues that may occur in the PDB curation process and affect the accuracy and consistency of data, and give the algorithms to solve them. Now the curation is not totally automatic in our system. So our main future goal is to try to improve our system by using automatic methods to check data and curate them. We also plan to develop a scripting programming language like Java JDBC API to implement our systems based on current User Interface.

References

[1] N. Ito, H. Sakamoto, K. Kobayash, Y. Kaneta, Y. Kawaguchi, et al.: New Features of PDBj-ML, an XML Format for Protein Data Bank, 2002.

[2] T.N. Bhat, P. Bourne, Z. Feng, G. Gilliland, S. Jain, et al.: The PDB data uniformity project. Nucleic Acids Research, 2001, Vol.29, No. 1, pp214-218.

[3] P. Lord, A. Macdonald, L. Lyon, D. Giaretta: From Data Deluge to Data Curation.

[4] P. Lord and A. Macdonald: Data curation for e-Science in the UK: an audit to establish requirements for future curation and provision.

[5] T. Warnock, L. V. Einde and R. Moore: NEESit Data Curation Roadmap. http://it.nees.org/documentation/pdf/TR-2005-046.pdf

[6] J. Gray, A. S. Szalay, A. R. Thakar, C. Stoughton, J. vandenBerg : Online Scientific Data Curation, Publication, and Archiving.

[7] P. Buneman, S. Khanna and W. Tan: Data Provenance: Some basic issues. http://db.cis.upenn.edu/DL/fsttcs.pdf

[8] P. Buneman, S. Khanna, and W. Tan: Why and Where: A Characterization of Data Provenance.

[9] Y. L. Simmhan Beth Plale Dennis Gannon: A Survey of Data Provenance in e-Science.

[10] J. Chandonia, N. Walker, L. Conte, P. Koehl, M. Levitt, et al.: ASTRAL compendium enhancements, 2002.

[11] S. E. Brenner, P. Koehl and M. Levitt: The ASTRAL compendium for protein structure and sequence analysis.

[12] J. Hu and L. Tao: An Extensible Constraint Markup Language: Specification, Modeling and Processing, SchemaSoft, 2004.

[13] Extensible Markup Language (XML) 1.0, World Wide Web Consortium (W3C) Recommendation, Bray, T. et al., World Wide Web Consortium (W3C), 1998. http://www.w3.org/TR/1998/REC-xml-19980210

[14] http://deposit.pdb.org/xtal-struct-dep.pdf

[15] Yanchao Wang, Rajshekhar Sunderraman and Hao Tian: A Domain Specific Data Management Architecture for Protein Structure Data. The 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, New York, June 2006.