An Evaluation of the Use of XML for Representation, Querying, and Analysis of Molecular Interactions

Lena Strömbäck and David Hall

lestr@ida.liu.se
Department of Computer and Information Science,
Linköping University, Sweden

Abstract. Currently, biology researchers rapidly generate new information on how genes, proteins and other molecules interact in living organisms. To completely understand the machinery underlying life it is necessary to integrate and analyze these large quantities of data. As one step in this direction, new standards for describing molecular interactions have been defined based on XML. This work evaluates the usage of the XML Query language XQuery for molecular interactions, as it would be of great benefit to the user to work directly on data represented in the new standards. We use and compare a set of available XQuery implementations, eXist, X-Hive, Sedna and QizX/open for querying and analysis on data exported from available databases. Our conclusion is that XQuery can easily be used for the most common queries in this domain but is not feasible for more complex analyses. In particular, for queries containing path analysis the available XQuery implementations have poor performance and an extension of the GTL package clearly outperforms XQuery. The paper ends with a discussion regarding the usability of XQuery in this domain. In particular we point out the need for more efficient graph handling and that XQuery also requires the user to understand the exact XML format of each dataset.

1 Introduction

During the past few years XML has become one of the most used formats for representation of information in a wide variety of domains and applications. In this paper we will discuss the current use of XML for molecular interactions, which is one important sub-area of bioinformatics. In this area the goal is to understand how proteins, genes, and other substances interact with each other within living cells. Proteins are the fundamental building blocks of life, and today biology researchers are gaining small pieces of information on each protein and how it interacts with other proteins and substances in the cell. To understand how the proteins and genes work together is the key to understanding the secret of life, and as such this has been set as a major goal for bioinformatics research by the Human Proteome Organization [8] and the US National Human Genome

Research Institute [5], since this would be the key to new medical treatments for many diseases.

Within the area of molecular interactions the tradition has been to publish results from experiments on the web, making it possible for researchers to compare and reuse results from other research groups. This has resulted in a situation with a large number of available databases on Internet [2, 9, 12–15, 20, 26] with information about experimental results. However, the information content, data model and functionality is different between the different databases, which makes it hard for a researcher to track the specific information he needs.

There is, however, ongoing development within the field with the goal of making the datasets from each of the databases available for downloading and further analysis. Evaluations [1, 16] have shown that XML is beneficial for information representation within bioinformatics. Most of the existing molecular interaction databases allow export of data in XML. Recently, there have also been proposals for XML-based exchange formats for protein interactions, e.g. SBML [10], PSI MI [8], and BioPAX [3]. However, to allow for easy analysis and understanding of these datasets there is still a need for software for integration, querying and analysis based on XML.

The aim of this paper is to evaluate the use of available XML tools for direct usage of molecular interaction data available in XML. The paper starts with a brief introduction to the chosen data formats. After that we report on two experiments on analysis of data with XQuery. Finally we conclude the paper with a discussion on future needs for XML tools for this application.

2 XML standards for molecular interactions

There are currently a number of different XML formats for molecular interaction data available from different databases. In this work we will focus on the two formats SBML [10] and PSI MI [8]. We've chosen these formats because they have been proposed as future standards and there are currently large datasets of data available in these formats. Here, we give a short introduction to these standards; for a more extensive description and comparison with other formats see [23, 24].

Systems Biology Markup Language (SBML) [10] was created by the Systems Biology Workbench Development group in cooperation with representatives from many system and tool developers within the bioinformatics field. A brief example of an SBML model is given in Figure 1. As we can see, an SBML model contains a number of compartments, each of which is a description of the container or environment in which the reaction takes place. The substances or entities that take part in the reactions are represented as species. The interactions between molecules are represented as reactions, defined as processes that change one or more of the species. Reactants, products and modifiers for reactions are specified by references to the relevant species.

The Proteomics Standards Initiative Molecular Interaction XML format (PSI MI) [8] was developed by the Proteomics Standards Initiative, one initiative of

SBML

```
<model name="Example">
 <listOfCompartments>
  <compartment name="Mithocondrial Matrix"
             id="MM">
 </listOfCompartments>
 <listOfSpecies>
  <species name="Succinate"
      compartment="MM" id="Succinate">
  <species name="Fumarate"
      compartment="MM" id="Fumarate">
  <species name="Succinate dehydrogenase"
      compartment="MM" id="Succdeh">
 </listOfSpecies>
 <listOfReactions>
  <reaction name="Succinate dehydrogenas
       catalysis" id="R1">
   <listOfReactants>
    <speciesReference species="Succinate">
   </listOfReactants>
   <listOfProducts>
    <speciesReference species="Fumarate">
   </listOfProducts>
   <listOfModifiers>
     <modifierSpeciesReference
                 species="Succdeh">
   </listOfModifiers>
  </reaction>
 </listOfReactions>
</model>
```

PSI MI

```
<entry>
 <interactorList>
  <proteinInteractor id="Succinate>
   <names>
    <shortLabel>Succinate</shortLabel>
    <fullName>Succinate</fullName>
   </names>
  </proteinInteractor> ...
 </interactorList>
 <interactionList>
  <interaction>
   <names>
    <shortLabel> Succinate dehydrogenas
         catalysis </shortLabel>
     <fullName>Interaction between ....
     </fullName>
    </names>
   <participantList>
    <proteinParticipant>
     <proteinInteractorRef ref="Succinate">
     <role>neutral</role>
    </proteinParticipant>
    <proteinParticipant>
     <proteinInteractorRef ref="Fumarate">
     <role>neutral</role>
    </proteinParticipant>
    <proteinParticipant>
     <proteinInteractorRef ref="Succdeh">
     <role>neutral</role>
    </proteinParticipant>
   </participantList>
  </interaction>
 </interactionList>
</entry>
```

Fig. 1. Examples of data in SBML and PSI MI

the Human Proteome Organization (HUPO). An abbreviated example pathway represented in PSI MI is shown in Figure 1. In PSI MI the *experimentList* describes experiments and links to publications where the interactions are verified. The pathway itself is described via the *interactorList*, which is a list of proteins participating in the interaction, and the *interactionList*, a list of the actual interactions. For each *interaction* it is possible to set one or more names. The participating proteins are described by their names or by references to the *interactorList*. Note that, where the intention of SBML is to describe an actual interaction, i.e. that interacting substances produce some product, the purpose of PSI MI is to describe the result of an experiment, i.e. that there is some chemical interaction between the substances but roles of the substances in the interaction are not always known.

As we can see, the two formats are similar. Even so, there are several important differences between them. As previously discussed PSI MI contains more detailed information and there are differences in how participants in an interac-

1.1 *Find all information on a given compartment. Compartment id is given.*
1.2 *Find all information on a given species. Species id is given.*
1.3 *Find all information on a given reaction. Reaction id is given.*
2.1 *Find all reactions which a given modifier participates in. The species of the modifier is given.*
3.1 *Find all the reactions whose reactants are the products of some reactions which a given modifier participates in. The species of the modifier is given.*
3.2 *Find all the reactions whose reactants and products are the same but modifiers are different.*
4.1 *Count the number of species in the database.*
4.2 *Count the number of reactions in the database.*

Fig. 2. Queries for the SBML dataset

tion are represented. In addition to this there is also an important difference in the fact that SBML makes more use of XML attributes while PSI MI prefers to represent information as extra children in the tree structure. In the remainder of this paper we will look at possibilities for the researcher to work directly on the dataset, i.e. to analyze it by querying directly against the XML document.

3 Experiment 1: XQuery querying

For our first experiment we want to test some common queries within the molecular interaction domain. For the experiments we use XQuery [32], the proposed standard query language for XML. The experiment consists of three parts: first the selection of queries and datasets for the test, next the formulation of the XQuery queries, and finally execution on XQuery implementations.

3.1 Definition of queries and datasets

Since the two standards SBML and PSI MI contain partly different information we define one set of interesting queries for each of the standards. The selected queries are based on an investigation of the query possibilities within available databases or investigating what are interesting questions from a biological point of view. The queries are divided into four different groups:

1. Simple selection of one data item.
2. Combination of information on two kinds of data types.
3. Complex queries, combination of several items.
4. Counting information in the datasets.

The selected queries for SBML are presented in Figure 2, the first number of each query shows which of the query groups it belongs to. Since the PSI MI data model is richer than the one for SBML we could use more queries compared to the SBML dataset. Here we also wanted to test some combined queries, i.e.

1.1 *Find information on a given protein. Protein id is given.*
1.2 *Find information on a given experiment description. Experiment description id is given.*
1.3 *Find information on a given interaction. Interaction id is given.*
2.1 *Find the protein information for the proteins that participate in a given interaction. Interaction id is given.*
2.2 *Find the experiment description information for an experiment description that is part of an interaction. Interaction id is given.*
2.3 *Find all interactions that a given protein participates in. Protein id is given.*
2.4 *Find all interactions that a given experiment description is part of. Experiment description id is given.*
2.5 *Find any interactions that two given proteins are a part of. Protein ids for the two proteins are given.*
3.1 *Find information on the proteins that could interact with a given protein. Protein id is given.*
3.2 *Find the description of the experiments which involve some interactions which a given protein participate in. Protein id is given.*
3.3 *Find the interactions which some given proteins participate in. The proteins secondary attribute is given.*
4.1 *Count the number of proteins in the database.*
4.2 *Count the number of interactions in the database.*

Fig. 3. Queries for the PSI MI dataset

forcing XQuery to join information from different parts of the data file. The selected queries for the PSI MI dataset are presented in Figure 3.

The database currently providing the largest subsets of SBML data is Reactome [12]. It is a database on biological pathways, mainly human but there are pathways from other species as well. We selected two datasets from Reactome of sizes 3 and 6 MB. They are available in SBML level 2, version 1. PSI MI is the most supported format for protein interaction databases. It is available as an alternative download format in a number of databases, for instance DIP [20], MINT [26], and IntAct [9]. Here, the IntAct database is the one currently providing the largest portions of PSI MI data. It is an open source database and toolkit for protein interactions. It currently contains nearly 40,000 interactions. We selected three datafiles from IntAct of sizes 9.5, 29.3 and 37.3 MB.

3.2 Expressing queries in XQuery

In this section we discuss some issues in formulating queries for the first experiment. An extensive description of the queries is given in [7]. Many of the queries are written as simple path expressions with arbitrary depth (nesting) using // and some conditional. This is used in, for instance, the first three queries for both test cases which are very basic queries consisting of paths. Here we exemplify this with query 2.1 for PSI MI:

```
document("rat_small.xml")//proteinInteractor[@id="EBI-77471"]
```

For the more complicated queries, join over path expressions or in some cases the XQuery FLWOR expressions are used, since these make the queries more readable and easier to express. As an example of this we present query 2.1 for PSI MI which uses a FLWOR expression. Here we find the interaction with the appropriate ID and iterate over the proteins participating in this interaction. For each of these proteins we find the desired information in the list of protein-interactors.

```
for $ref in document("rat_small.xml")//interaction
  [names/shortLabel="interaction1"]
  /participantList/proteinParticipant/proteinInteractorRef/@ref
return document("rat_small.xml")//proteinInteractor[@id=$ref]
```

The last two queries for each standard use the count aggregate function to give a measure of the size of the datasets. Here we exemplify this with query 4.1 for PSI MI counting the number of proteins in the database.

```
count(document("rat_small.xml")//proteinInteractor)
```

From this discussion we can conclude that for a user that is accustomed to the concepts and constructions in the molecular interaction standards, and who has a reasonable knowledge of XQuery, these queries can easily be expressed.

3.3 Efficiency

Having formulated the queries, we were interested in the performance of the different XML database systems. There are a large number of implementations of XQuery, ranging from implementations for direct querying on XML files to systems aiming at more efficient storage and treatment of larger XML files, so-called native XML databases. We selected three native XML database implementations: eXist [27], Sedna [30] and X-Hive [31] and the XQuery API QizX/open [29] which does not support indexing and thus is expected to yield lower performance than the other systems.

Since the exact times can depend on external factors such as other processes running on the system the computer results would differ from time to time for the same queries. To decrease the influence of sudden spikes in measured time all queries were run several times and we base our values on the mean times. We have run several sets of tests on our selected datasets on two different computers. Here we will present a selection of results that represents our general findings. Figure 4 shows a general comparison of the systems for queries on the IntAct 29.3 MB dataset run on a IBM X40, Intel Pentium 1200 MHz processor with 512 MB RAM. All the Native database systems have similar and good performance, where X-Hive has slightly higher response time than the other systems. QizX/open performs worse than the other systems for all queries, as expected

We also wanted to compare how the response time varied if we varied data size. Figure 5 show a comparison of response times on eXist and Sedna on the

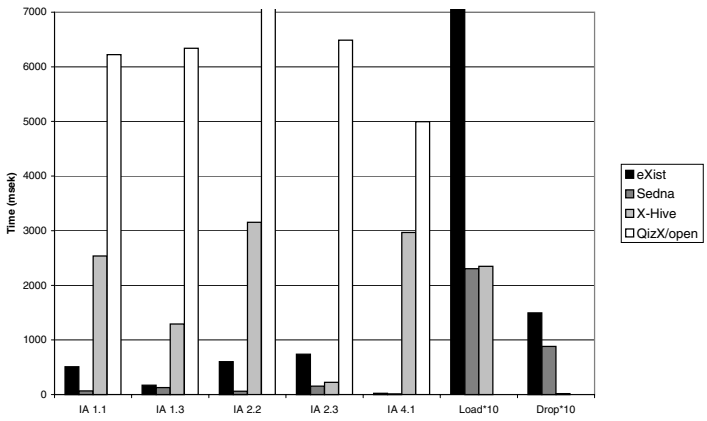

Fig. 4. Query times for different systems.

9.5 and 37.3 IntAct datasets. This queries were run on an AMD Athlon 1000 MHz computer with 512 MB RAM. From the figure we can conclude that Sedna is the better performing system, with some exceptions where we have a very high response time for Sedna. For both systems we can also see an increase in response time when complexity of queries increase. Here performance of the systems is highly dependent on the number of intermediate results generated by a query, and thus by evaluation order on parts of the query. In general, the response time increases when the data size increases. There are however some exceptions to this. These exceptions can be explained with the different composition of datasets. For a particular query the data size for an intermediate step can be higher even if the total data size is smaller.

Finally we wanted to compare the performance between SBML and PSI MI. For this we used the Reactome 6MB and IntAct 9.15 MB datasets, which are reasonably comparable in number of items stored. A comparison on corresponding queries are given in Figure 6, this time run on the AMD computer. The figure shows that there is no larger difference between the datasets in terms of performance.

To conclude this section we can see that all the queries run with a reasonable response time on the selected datasets. Comparing the systems we can see that all the native databases provide similar performance with Sedna being the fastest and X-Hive being the most stable implementation.

4 Experiment 2: Pathway analyses

In addition to queries similar to those currently available through conventional systems, we also wanted to test advanced analysis on the datasets. In this case,

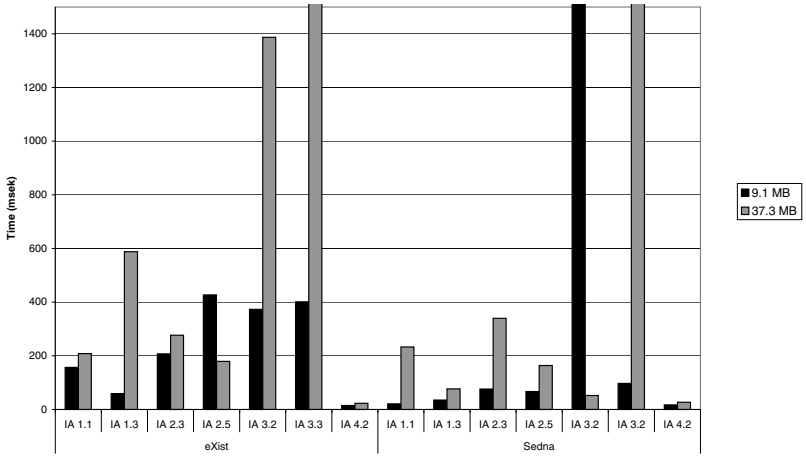

Fig. 5. Query times for different data sizes.

we want to search for interaction chains between given proteins. As explained, it is not possible to make out reactant and products in PSI MI interactions since there is no order of the reactions defined in this format. Therefore we decided to concentrate the pathway searching efforts to the SBML dataset and used the following query:

> *Given two proteins find out if there is a given pathway between them in maximum n steps. Protein ids are given for the two proteins.*

This analysis is very important, since it is often important to identify connections between interacting proteins in a given dataset. To express this in XQuery, recursion is required. The query is shown in Figure 7. As we wanted to be able to test the query for various lengths of the pathways the query contains a recursive function *findMolecule* that returns elements for found connected proteins within a specified maximum length of the path. The function takes the start and goal reactants together with the cut-off depth as parameters.

The response times for this query run on the IBM 1200 GHz computer on the 3MB Reactome file are presented in Table 1. eXist, Sedna and QizX/open ran out of memory at 4 steps. Sedna query times increase more slowly than for X-Hive, but X-Hive is the only XML tool reaching 4 steps.

These results are disappointing, both in the sense that formulating recursive XQuery queries gets rather complicated and that the response times are too high, especially taking into account that a real application would often need to do queries on larger datasets and longer paths than the ones we used.

For the molecular interaction applications there is a need for more efficient handling of these kinds of queries. One possibility would be to include an existing graph package into the XQuery language. For this reason we made an experi-

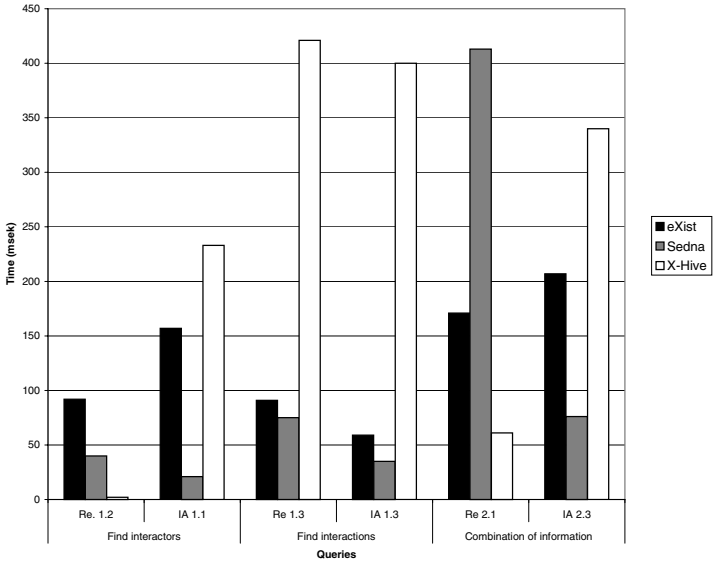

Fig. 6. Comparison between SBML and PSI MI

ment with the graph package GTL (Graph Template Library) [28]. GTL is an extension of the Standard Template Library for graphs and graph algorithms in C++. The function we wanted to test, finding all paths between two proteins, was not implemented in GTL and we had to extend the package [7]. Before GTL can be used the protein interaction, data in SBML format is transformed to GML, a non-XML-based graph representation format provided by GTL. In our translation nodes are substances and edges are reactions. The transformation is done using the MSXSL command line transformation utility from Microsoft and it takes about 2 seconds to transform the 3 MB Reactome file.

Table 1. Test case 3: Pathway searches with XQuery.

Steps	eXist	X-Hive	Sedna	QizX/open
	Mean time	Mean time	Mean time	Mean time
1	422 ms	334 ms	121 ms	906 ms
2	951 ms	646 ms	245 ms	1125 ms
3	33323 ms	9053 ms	3493 ms	7172 ms
4	-	700443 ms	-	-

```
declare function local:findMolecule($molecule as xs:string,
            $goalMol as xs:string, $n as xs:integer) {
  for $i in document("sbml.xml")//reaction
        [listOfReactants/speciesReference/@species=$molecule]
        /listOfProducts/speciesReference/@species
  return <item>{$i} {
    if($i = $goalMol) then <found/> else
      if($i = $molecule) then <loop /> else
        if ($n = 1) then <max/> else
          local:findMolecule($i, $goalMol, $n - 1)}
    </item>};
<path>{local:findMolecule("H2O", "sodium ion",2)}</path>
```

Fig. 7. Query for path searches

Table 2. Test case 3: Pathway searches using the extended GTL package

Steps	1	2	3	4	5	6	7	8
Time	0.3 ms	3 ms	15 ms	101 ms	823 ms	5,55 s	35,8 s	215 s

The first step of our algorithm would be a search using GTL's built-in breadth-first search algorithm to verify that the end node really can be reached from the start node. The search between the start and end node is done by a recursive function, which works outwards from the start node and follows outgoing edges. In addition to this we use a cut-off depth at which to stop searching, as with the corresponding XQuery. The graph sent as an argument has already-visited nodes marked as hidden to avoid loops. Table 2 shows the performance of the pathway searches using our extension of GTL. The numbers given in the relevant table are based on a mean value of ten iterations. Figure 8 shows a comparison between the GTL implementation and the tested databases.

As shown by this table the C++ program developed for graph searches is magnitudes faster than using the XQuery searches. This depends on a number of things with the most important probably being that the C++ program has graph and loop detection and that the representation is optimal for graph searches.

5 Discussion and implications for the future

The development of web databases and new standards within the area of molecular interaction indicates that XML representations will be of high importance for the area in the future. This means that there will also be a high degree of interest in existing XML technology as well as a need for development of new technology for the specific needs of the application. In this section we will put our results into context by providing a discussion on the generality of the results and requirements for the molecular interaction application.

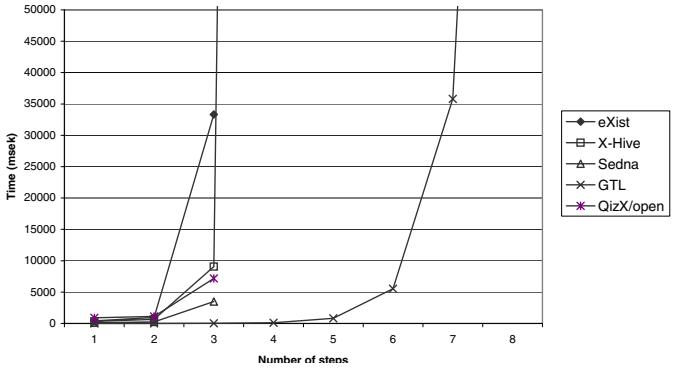

Fig. 8. Comparison of query times for pathway searches.

For our experiments the most central criteria has been to test whether XQuery is useful for finding relevant information from a molecular interaction dataset with reasonable simple query formulations and a reasonable level of efficiency. To determine this we have based our queries on an investigation of available queries on existing databases for molecular interactions and cellular pathways available over the Internet [2, 9, 12–15, 20, 26]. This ensures that our selected queries capture the most important features of the application.

Another measure of generality is to compare our queries to available test benches for XML [4, 17, 21, 25]. These test benches define either particular XQuery queries or sets of XQuery queries, where the idea is to cover as many features of XQuery as possible. Such a comparison shows that our selected test queries cover the query groups relevant to this domain. Certain kinds of queries that were not relevant to this application or these datasets were naturally excluded from our tests, for example queries based on order.

For the queries in experiment 1, it was feasible to write queries on the protein-interaction data using the XQuery language. For a small dataset all three tested NXD's gave response times acceptable for interactive use with the exception for pathway queries. Our previous comparison with relational databases [22] also shows that these results are comparable with what can be achieved using a relational approach.

However, an interesting question is whether XQuery is a suitable query language for the domain. Even though it is possible to express queries, querying with XQuery requires a solid knowledge of the specific XML format of a dataset, which requires the user to have a high degree of knowledge about the specific XML formats. Since it is very likely that a typical user would need to handle several of these formats, there is a need for developing higher-level query languages for the domain.

In the case of more complex analyses, e.g. pathway analysis, the search times become large after just a few steps and the tested XQuery implementations

do not cope with pathway queries longer than three steps. A C++ program for performing this query was developed using a graph package and resulted in searches that were orders of magnitude faster than for the XML databases, making it possible to search even larger graphs.

XQuery does not provide any special support for graph processing, while queries of this kind are doubtless very interesting in many applications. XGMML (Extensible Graph Markup and Modeling Language) [18] is a general format for describing graphs using XML based on GML, used in these tests. Other more specialized formats, such as the already-existing PSI MI and SBML formats for biological data, may emerge in a number of different subject areas.

This means there is a need for graph-capable XML in combination with XQuery. To be able to perform larger graph searches there must be special support for this. One possibility would be to extend XQuery with a number of internal functions for path searches and graph analyses. This is possible by using, for example, a Java binding such as those offered by eXist and QizX/open, which makes it possible to call functions in Java in the same way as XQuery's internal functions.

A final issue is the situation where the user needs to query over several datasets and integrate the resulting information into one query. Here we see several solutions. One is to provide a higher level query language capable of translating the query into several specific query languages. This is similar to what has been proposed for general databases within bioinformatics [11]. Another option would be to provide tools for fast data integration between the different XML formats in the line of the work within schema matching [6, 19].

6 Summary

XML is more and more commonly used within the area of molecular interactions and new XML standards are arising within the area. Because of this it would be very appealing if existing XML technology could be used for querying and analyses on this data. This work evaluates the use of XQuery and Native XML databases on datasets in two of the available standards, SBML and PSI MI.

Our experiments show that XQuery is a suitable language for most of the queries expected for the domain. We also saw a reasonable level of efficiency in the tested native XML implementations. There are, however, several obvious points for future research. One is the need for extended query languages and methods for graph analyses. A second issue is methods for the user to query over several different standard formats without having an exact knowledge of the specific XML format for each of the datasets.

Acknowledgements Thanks to Patrick Lambrix for valuable comments on this work. We are also grateful to Liu Ke for performing some of the tests.

References

1. Achard F, Vaysseix G, and Barillot E: XML, bioinformatics and data integration. *Bioinformatics* **17**(2):115-125, 2001.
2. Bader GD, Donaldson I, Wolting C, Oulette BF, Tony BF, Pawson TBF, and Hogue CWV: BIND - The Biomolecular Network Database. *Nucleic Acids Research* **29**(1):242-245, 2001.
3. BioPAX working Group: BioPAX – Biological Pathways Exchange Language. Level 1, Version 1.0 Documentation. Available at http://www.biopax.org, 2004.
4. Bressan S, Lee M-L, Li YG, Lacroix Z, Nambiar U: The XOO7 Benchmark. EEXTT 2002: 146-147, 2002.
5. Collins FS, Green ED, Guttmacher AE, and Guyer MS: A vision for the future of genomics research: A blueprint for the genomic era. *Nature* **422**: 835-847, 2003.
6. Doan, A, Halevy AY: Semantic Integration Research in the Database Community: A brief Survey. *AI Magazine, Special Issue on Semantic Integration, Spring 2005*, 2004.
7. Hall D. *An XML-based Database of Molecular Pathways.* Master Thesis, Department of Computer and Information Science, Linköpings universitet, LITH-IDA-EX–05/049–SE, 2005.
8. Hermjakob H, Montecchi-Palazzi L, Bader G, Wojcik J, Salwinski L, Ceol A, Moore S, Orchard S, Sarkans U, von Mering C, Roechert B, Poux S, Jung E, Mersch H, Kersey P, Lappe M, Li Y, Zeng R, Rana D, Nikolski M, Husi H, Brun C, Shanker K, Grant SGN, Sander C, Boork P, Zhu W, Akhilesh P, Brazma A, Jacq B, Vidal M, Sherman D, Legrain P, Cesareni G, Xenarios I, Eisenberg D, Steipe B, Hogue C, and Apweiler R: The HUPO PSI's Molecular Interaction format - a community standard for the representation of protein interaction data. *Nature Biotechnology* **22**(2):177-183, 2004.
9. Hermjakob H, Montecchi-Palazzi L, Lewington C, Mudali S, Kerrien S, Orchard S, Vingron M, Roechert B, Roepstorff P, Valencia A, Margalit H, Armstrong J, Bairoch A, Cesareni G, Sherman D, Apweiler R: IntAct - an open source molecular interaction database. *Nucl. Acids. Res.* 32: D452-D455, 2004.
10. Hucka M, Finney A, Sauro HM, Bolouri H, Doyle JC, Kitano H, Arkin AP, Bornstein BJ, Bray D, Cornish-Bowden A, Cuellar AA, Dronov S,. Gilles ED, Ginkel M, Gor V, Goryanin II, Hedley W-J, Hodgman TC, Hofmeyr J-H, Hunter PJ, Juty NS, Kasberger JL, Kremling A, Kummer U, Le Novere N, Loew LM, Lucio D, Mendes P, Minch E, Mjolsness ED, Nakayama Y, Nelson MR, Nielsen PF, Sakurada T, Schaff JC, Shapiro BE, Shimizu TS, Spence HD, Stelling J, Takahashi K, Tomita M, Wagner J, and Wang J: The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* **19**(4):524-531, 2003.
11. Jakoniene V, Lambrix P: Ontology-based Integration for Bioinformatics. Proceedings of the VLDB Workshop on Ontologies-based techniques for DataBases and Information Systems - ODBIS 2005, pp 55-58, Trondheim, Norway.
12. Joshi-Tope G, Gillespie M, Vastrik I, D'Eustachio P, Schmidt E, de Bono B, Jassal B, Gopinath GR, Wu GR, Matthews L, Lewis S, Birney E, Stein L: Reactome: a knowledgebase of biological pathways. *Nucleic Acids Res.* 1;33 Database Issue:D428-32. PMID: 15608231, 2005.
13. Kanehisa, M and Goto, S: KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.* **28**, 27-30, 2000.

14. Kanehisa M, Goto S, Kawashima S, Okuno Y, and Hattori M: The KEGG resources for deciphering the genome. *Nucleic Acids Res.* **32**, D277-D280, 2004.
15. Karp P.D, Arnaud M, Collado-Vides J, Ingraham J, Paulsen IT, and Saier MH Jr: The E. coli EcoCyc Database: No Longer Just a Metabolic Pathway Database. *ASM News* **70**(1): 25-30, 2004.
16. McEntire R, Karp P, Abrenethy N, Benton D, Helt G, DeJongh M, Kent R, Kosky A, Lewis S, Hodnett D, Neumann E, Olken F, Pathak D, Tarzy-Hornoch P, Tolda L, and Topaloglou T: An evaluation of Ontology Exchange Languages for Bioinformatics. *Proceedings International Conference on Intelligent Systems for Molecular Biology* **8**:239-250, 2000.
17. Nambiar U, Lacroix Z, Bressan S, Lee M-L, Li YG: Current Approaches to XML Management. IEEE Internet Computing 6(4): 43-51, 2002.
18. Punin J, Krishnamoorthy M: XGMML (eXtensible Graph Markup and Modeling Language) http://www.cs.rpi.edu/∼puninj/XGMML/, 2001 (Accessed April 2005).
19. Rahm, E and PA Bernstein: A survey of approaches to semantic schema matching. *The VLDB Journal 10:334-350.* DOI 10-1007/s007780100057, 2001.
20. Salvinski L, Miller CS, Smith AJ, Bowie JU, and Eisenberg D: The Database of Interacting Proteins: 2004 Update. *Nucleic Acids Research* **32** Database Issue D449-451, 2004.
21. Schmidt AR, Waas F, Kersten ML, Florescu D, Carey MJ, Manolescu I, and Busse R: Why and How to Benchmark XML Databases. ACM SIGMOD Record, 3(30):27-32, 2001.
22. Strömbäck, L: Storage and Integration of Molecular Interactions: Evaluating the Use of XML Technology. *Proc of the 3rd International Workshop on Biological Data Management (BIDM'05).* Copenhagen, Denmark, 2005.
23. Strömbäck, L: XML representations of pathway data: a comparison. *Proc. of the ACM SIGIR'04 Workshop on Search and Discovery within Bioinformatics.* Sheffield UK, 2004.
24. Strömbäck, L and Lambrix P: Representations of molecular pathways: An evaluation of SBML, PSI MI and BioPAX. Accepted for publication in *Bioinformatics* **21**(24):4401-4407, 2005.
25. Yao BB, Özsu MT, and Khandelwal N: XBench Benchmark and Performance Testing of XML DBMSs. In *Proceedings of 20th International Conference on Data Engineering*, Boston, MA, pp 621-632, March 2004.
26. Zanzoni A, Montecchi-Palazzi L, Quondam M, Ausiello G, Helmer-Citterich M, and Cesareni G: MINT: a Molecular INTeraction database. *FEBS Letters* **513**(1):135-140, 2002.
27. eXist http://exist.sourceforge.net (Accessed May 2005)
28. GTL The Graph Template Library http://infosun.fmi.uni-passau.de/GTL/index.html (Accessed May 2005)
29. QizX/open http://www.xfra.net/qizxopen/ (Accessed May 2005)
30. Sedna http://www.modis.ispras.ru/Development/sedna.htm (Accessed May 2005)
31. X-Hivehttp://www.x-hive.com (Accessed May 2005)
32. XQuery http://www.w3c.org (Accessed May 2005)