Analysis and visualization of phylogenetic xml trees

Robin Paul, Jayendra Khandare and John Leite

**Abstract – Biological organisms have evolved from single-celled bacteria to multicellular animals such as humans in a time-span of 4 billion years. During this evolutionary time, millions of different biological species have evolved from a single common ancestor. Such diversity in life has made the understanding and visualization of biological evolution a challenge. Phylogeny and relationship of organisms is in the form of a tree structure which can be represented in the form of an xml tree structure. Here we characterize and visualize a phylogenetic xml file by displaying its various branches using XSLT and converting part of the phylogenetic tree into SQL. Finally, we also visualize the entire phylogenetic tree using the software archaeopteryx.**

**Introduction**

Biological evolution of living organisms has occurred over a span of four billion years giving rise to wide variety of living species. The best data structure to store and analyze this massive amount of biological diversity is in the form of an xml tree.

However, there are several challenges in the construction and visualization of an xml database. First, in many cases, the confidence in delineating a particular lineage of biological species can be difficult. Secondly, in many cases many species are either unnamed or haven't been characterized completely. Such drawbacks make storing of phylogenetic data difficult in traditional SQL databases making NoSQL databases an obvious choice.

We have made use of XSLT to parse out those branches of the XML tree which have confidence above a certain threshold at various levels of the phylogenetic tree. We have also converted part of the XML tree into SQL using a custom python script to enable us firing query in traditional relational databases.

**Methods**

a) Parsing of xml database with XSLT

An XSLT file was created which prints the branch length, confidence, and scientific name of a particular branch in the xml database. It then uses a series of nested for-loops to print out all the sub-branches in a tree using a threshold confidence of 80%. It prints the output in HTML tabular format, which can be easily viewed using a web browser.

b) Conversion of xml tree into SQL database

For this task, we chose Python language to write a script to perform the necessary task. We used two external Python libraries named BioPython and Psycopg2. Bio.Phylo contains a submodule ‘PhyloXML’ which was used to handle parsing and tokenizing the XML trees. Psycopg2 was used to connect to the PostgreSQL database. First, we tokenized the XML tree into tags using Phylo.read() function. Then we had to decide which tags are essential and omit the others, so we created a database to store those values. Finally, we connected to the PostgreSQL database and feed it the tokens we generated in the previous phase.

c) Visualization of phylogenetic tree using archaeopteryx

**Results and Discussion**

a) XSLT results

b) SQL conversion and query

SQL conversion of XML proved to be a tough feat to achieve. Unlike SQL, XML is not well structured and the XML database we used proved to be impossible to work with. Most of the species identified don’t have a name ascribed to them. Some species don’t have a confidence level mentioned in the XML. Some have scientific names while some don’t. So, to avoid a lot of empty entries we decided to lose some information like ‘scientific name’ which have very rare occurrence. Converting this data manually is ineffective because it will require more time. Also, if we don’t drop a few details, SQL transformation is possible but it will waste a lot of space since most of the columns will be empty. This is the list of the properties which we chose to include:

BRANCH\_NAME VARCHAR(20), BRANCH\_LENGTH FLOAT(20), TOTAL\_BRANCH\_LENGTH FLOAT(20), CONFIDENCE INTEGER, NODE\_PATH\_LENGTH INTEGER, TERMINALS\_NUMBER INT,

PARENT\_NODE VARCHAR(20)

If a species doesn’t have a name, we denote it as “Bx” where x stands for an integer value.

We were able to extract data of 134 species from the XML and convert it into SQL format with the above approach.

The output for query “select \* from phyloxml” can be seen in the snapshots given at the end of this report. Furthermore, we could fire up a query to find information with specific constraints like (confidence > 50 AND confidence < 75) or descendants of a species by firing simple SQL queries like

1. “select \* from phyloxml

where confidence > 50 AND confidence <75;”

1. “select \* from phyloxml

where parent\_node = ‘B40’;”

c) Visualization of xml file











