iDigPhylo: An API For Building Phylogenetic Trees From iDigBio and GenBank Data

CAP5510 Bioinformatics Final Project
Matthew Collins, 9827-1841
2015-12-07

Abstract

Constructing phylogenetic trees from sequence data is an important part of many biological studies. Many on-line tools exist (Bioinformatics services 2015) for alignment and tree building but they require pre-formating data to use them. iDigPhylo aims to minimize the work involved in making trees to speed up the scientific inquiry process. The iDigPhylo API rests directly on top of iDigBio specimen data and pre-fetched GenBank sequences so scientists can work by expressing their research question as an iDigBio record query instead of having to manipulate data.

The iDigPhylo API runs as a Python web service and uses a distributed worker architecture to align sequences with Clustal Omega and build trees with MrBayes from the alignments. Trees are drawn with the ETE Python tree library and returned as scalable vector graphics. Performance of the API to return a graphic ranges from three seconds for four sequences to 52 seconds for 32 sequences. The API is currently most limited by lack of availability of GenBank identifiers in iDigBio data and by the variation in the types of sequences in GenBank.

Source Data

iDigPhylo uses data digitized from museum specimens stored in iDigBio (Matsunaga 2010), a 10 year NSF-funded project, as a base. This data is collected by natural history museums in the United States and aggregated into a central database. Access to the database is provided through a REST API. The data includes information such as the geographic location where the specimen was collected, date, and taxonomic identifications.

GenBank (Benson 2013) is a collection of public DNA sequences managed by International Nucleotide Sequence Database Collaboration. The sequences can be of any part (gene, fragment, whole genome, etc) of an organism and can vary in quality. Sequences are annotated with features of interest but not necessarily with information about the organism the sequence came from. All data is available through an API or through a simple web query interface.

Data Pre-Processing With Spark

Combining the two data sources required locating GenBank sequence identifiers in iDigBio data. The ontologies used to represent data in iDigBio lack a concept for a gene sequence

identifier so applying a regular expression to the free-text Darwin Core associatedSequences field is the best available method for finding which specimen records relate to which Genbank sequences.

iDigBio is currently about 40 GB in size when represented as comma separated text suitable for regular expressions. A 5-node (40 core) Cloudera cluster running Apache Spark (Matei 2010) was used to efficiently generate a list of iDigBio specimen and Genbank sequence identifier pairs. These pairs were stored in a database table in the MariaDB relational database.

Sample GenBank Sequences

A small sample of alignable sequences was downloaded from GenBank and matched up to specimen queries in iDigBio. There were a total of 7,653 records in iDigBio in the order squamata (scaled reptiles) with a populated associatedSequences field. From those, 5,251 sequences were detected by the Spark processing. Those sequences were downloaded and their metadata reviewed manually to make sure they referred to the same gene and were alignable with each other. Only sequences from the NADH dehydrogenase subunit 2 (ND2) gene were selected for use in the demonstration and benchmarking.

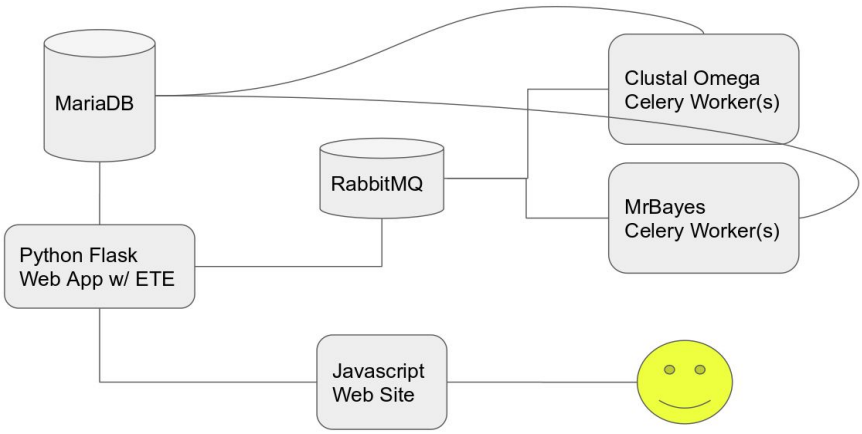
Service Architecture

A distributed systems approach was chosen for the computation services behind the API. This pattern allows for horizontal scaling of individual tasks according to need. It also allows the type of computer to be matched to each type of task: large memory tasks like alignment can be run on computers with lots of RAM and computationally expensive tasks like tree building can be run on computers with many cores or GPU accelerators.

The Python distributed task manager library Celery was used to write and schedule the alignment and tree construction jobs. Celery uses the RabbitMQ distributed messaging server to hold the work queue and coordinate the workers. The Celery workers can be run on any machine with a network connection to the RabbitMQ messaging server. Separate workers and queues were used for the alignment and tree tasks.

Results from the workers are stored back into the MariaDB relational database. This is where the API picks them up to be served back to the client. Figure 1 summarizes the workings of the API.

Figure 1: Service Architecture



Clustal Omega and MrBayes Workers

Phylogenetic trees were constructed from multiple sequence alignments generated on the fly. The sequences that are related to iDigBio specimen queries change as the queries change so no pre-alignment can be done. The Clustal Omega program (Sievers 2011) was found to be the best balance for speed and accuracy in a recent comparison (Asp 2015). This program is multi-threaded, memory efficient, and easy to run. The alignment Celery worker uses it to output a FASTA format multiple sequence alignment that is then stored in the database and passed on to the tree building worker.

MrBayes (Huelsenbeck 2001) was chosen for constructing the phylogenetic trees. It uses Bayesian inference methods to build a tree whose likelihood can be calculated directly. It is multithreaded and supports GPU acceleration. The MrBayes tree Celery worker generates a NEXUS formatted tree that is stored in the database.

API Structure and Sample Application

The iDigPhylo API consists of three endpoints in this initial prototype. The alignment and tree construction processes can be long-running so the API uses an asynchronous pattern of submitting jobs, querying their status, and then returning results when completed.

The Flask web application framework for Python was used to implement the following REST endpoints:

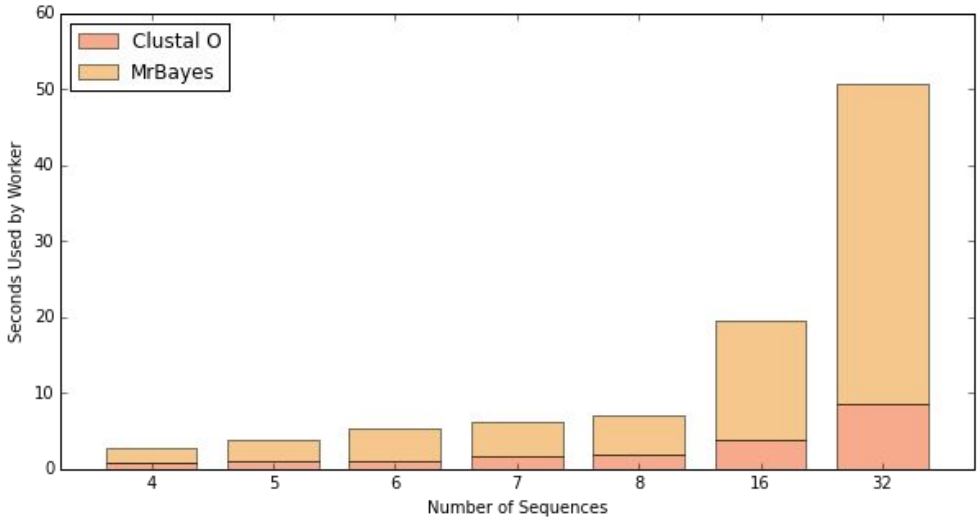
1. */tree/view/<job_id>* - Returns a JSON-wrapped MyBayes NEXUS format file containing the tree generated by the given job or the current status of the job if the result is not ready
2. */tree/build* - POST endpoint that takes an iDigBio record query as *rq* and an optional *limit* parameter to start the construction of an tree, returns the job_id in JSON

3. */tree/render/<job_id>* - Returns an SVG image of the tree produced by the given job using the ETE Python library (Huerta-Cepas 2010)

API Performance

Queries designed to use a known number of sequences were entered into the sample web application and the amount of time spent by the alignment and tree building workers was measured. The time spent passing messages and data was negligible compared to the time spent by the workers. Figure 2 shows the average of 10 samples of building trees from four through 32 sequences. The measurements were made on 8 cores of a Xeon E5640 processor.

Figure 2: Performance of API Over Different Numbers of Sequences



The alignment phase appears to follow a linear trend and accounts for less than 20% of the total time. The tree construction phase dominates the time and appears to be slightly greater than linear. Optimizing the performance of MrBayes through additional cores or GPUs is the best approach to improving the speed of the API.

Limitations and Next Steps

The current limitations of iDigPhylo are related to the source data availability and interpreting the source data in a biologically meaningful way to make sure appropriate trees are constructed. Resolving these limitations will require significant collaboration with biologists as well as changes to the API.

Only 25,289 out of 48 million specimen records in iDigBio contain information in the associatedSequences field. It is possible that some data providers use other fields for sequence information but they should be in the minority. This means that 99.95% of the results from iDigBio queries can not be used in tree construction. Data providers can be encouraged to provide this linking information in the future.

Of the sequences provided, selecting the ones that are appropriate to align, that is they are from the same gene, of the same type, etc., requires interpreting GenBank metadata and knowledge of the organisms involved. It is also easy to have a specimen query return very different organisms such as plants and animals that may not share any genes that can be aligned. Future development may involve clustering sequence data available based on biologically significant criteria and generating multiple trees. This limitations lead to iDigPhylo not being able to be used to construct trees that could be compared to published trees from TreeBase and similar repositories as originally planned.

Museums commonly collect many specimens from the same species so it is likely that there are clusters of sequences that are very similar to each other. The parameters and quality of the results for aligning and constructing trees from closely and more distantly related sequences are different. Adding parameters to the API to allow tuning of the alignment and tree construction programs will be done in future versions.

Source code for iDigPhylo as well as this document and presentations are available on Github: https://github.com/mjcollin/idigphylo/ . A running copy of the sample web application is availible temporarily at: http://badmatt.com/idigphylo//2015/12/07/demo.html .

References

Asp, A. (2015, December 1). Personal interview.

Benson DA, Cavanaugh M, Clark K, Karsch-Mizrachi I, Lipman DJ, Ostell J, Sayers EW.

Nucleic Acids Res. 2013 Jan;41(Database issue):D36-42. doi: 10.1093/nar/gks1195. Epub 2012 Nov 27.

Bioinformatics services. (n.d.). Retrieved December 9, 2015, from http://www.ebi.ac.uk/services

ETE: A Python Environment for Tree Exploration. Jaime Huerta-Cepas, Joaquín Dopazo and Toni Gabaldon. BMC Bioinformatics (2010) doi:10.1186/1471-2105-11-24

J. Huelsenbeck and F. Ronquist, 'MRBAYES: Bayesian inference of phylogenetic trees', Bioinformatics, vol. 17, no. 8, pp. 754-755, 2001.

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: cluster computing with working sets. In Proceedings of the 2nd USENIX conference on Hot topics in cloud computing (HotCloud'10). USENIX Association, Berkeley, CA, USA, 10-10.

A. Matsunaga, A. Thompson, R. Figueiredo, C. Germain-Aubrey, M. Collins, R. Beaman, B. MacFadden, G. Riccardi, P. Soltis, L. Page and J. Fortes, ‘A Computational- and Storage-Cloud for Integration of Biodiversity Collections’, 2013 IEEE 9th International Conference on e-Science, 2013.

Sievers F, Wilm A, Dineen DG, Gibson TJ, Karplus K, Li W, Lopez R, McWilliam H, Remmert M, Söding J, Thompson JD, Higgins DG (2011). Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Molecular Systems Biology 7:539 doi:10.1038/msb.2011.75