BIOINFORMATICS APPLICATIONS NOTE Vol. 24 no. 16 2008, pages 1823–1825
doi:10.1093/bioinformatics/btn325

Databases and ontologies

BioMAJ: a flexible framework for databanks synchronization and processing

Olivier Filangi[1], Yoann Beausse[2], Anthony Assi[1], Ludovic Legrand[3], Jean-Marc Larré[2], Véronique Martin[3], Olivier Collin[1], Christophe Caron[3], Hugues Leroy[1] and David Allouche[2,*]

[1]IRISA/INRIA, Symbiose Project, [2]Unité de Biométrie et Intelligence Artificielle, UR875, INRA, F-31320 Castanet-Tolosan and [3]INRA Unité Mathématique Informatique et Génome, UR1077, INRA, F-78350 Jouy-en-Josas, France

Received on March 14, 2008; revised and accepted on June 20, 2008

Advance Access publication June 30, 2008

Associate Editor: Dmitrij Frishman

ABSTRACT

Large- and medium-scale computational molecular biology projects require accurate bioinformatics software and numerous heterogeneous biological databanks, which are distributed around the world. BioMAJ provides a flexible, robust, fully automated environment for managing such massive amounts of data. The JAVA application enables automation of the data update cycle process and supervision of the locally mirrored data repository. We have developed workflows that handle some of the most commonly used bioinformatics databases. A set of scripts is also available for post-synchronization data treatment consisting of indexation or format conversion (for NCBI blast, SRS, EMBOSS, GCG, etc.). BioMAJ can be easily extended by personal homemade processing scripts. Source history can be kept via html reports containing statements of locally managed databanks.

Availability: http://biomaj.genouest.org. BioMAJ is free open software. It is freely available under the CECILL version 2 license.

Contact: biomaj@genouest.org

1 INTRODUCTION

Biological knowledge, within the context of proteomics and genomics, is mainly based on bioinformatic analyses consisting of periodic comparisons between newly produced data and the current set of known information. This approach requires both locally installed bioinformatic programs and collections of heterogeneous biological databanks, which are available through numerous servers located around the world (NCBI, EBI, DDBJ, expasy, etc.). Local integration of all of these data begins with data mirroring and indexation. This essential, preliminary step represents a major bottleneck for data annotation and comparative genomics projects due to the fact that the databanks are large (measured in terabytes) and in heterogeneous formats. Once downloaded, the data must undergo various post-processing steps before they can be used. These steps consist of intensive processing tasks corresponding to data indexation, format conversion and data concatenation or extraction. Additionally, the frequency with which various databanks require updating is not constant and, depending upon the source, can vary from daily to several times per year. In most cases, the only way to determine the existence of a new release is a periodic check of the files stored on the remote server. Data maintenance is a complex and labor-consuming task that is absolutely necessary for projects that perform intensive local analyses on that data. Thus, a flexible and robust software tool that maintains workflows associated with the various stages of local mirroring of databanks is essential to automate the numerous iterative updates that are required. While several commercial solutions are available, free software solutions range in complexity from server specific shell scripts to the more sophisticated application Biodownloader (Shapovalov *et al.*, 2007). Unfortunately, this program was not designed to manage large numbers of sources, cannot monitor the global repository and is not compatible with UNIX. The best solution evaluated was from the GMOD project, called Citrina (Goodman, 2004). While less user-friendly than Biodownloader, it provides data synchronization and the ability to launch sequential post-processing tasks on various UNIX-based operating systems. Since the Citrina project is no longer being developed, we began the process of creating a new project, called BioMAJ, using the Citrina source as inspiration. BioMAJ, where MAJ stands for 'mise a jour', which is French word for 'update', is designed to automate and manage data workflows associated with updating and processing local mirrors of large biological databases. The software can be used by both large-scale bioinformatics projects and administrators of large computational infrastructures that provide services based on well-known bioinformatics suites such as EMBOSS, Sequence Retrieval System (SRS) (Etzold *et al.*, 1996), (http://www.biowisdom.com/navigation/srs/srs) and GCG. This article describes how BioMAJ provides a flexible framework for databank synchronization and processing.

2 COMPATIBILITY AND PACKAGE

BioMAJ was developed in JAVA and ANT. The application is compliant with various UNIX operating systems. It provides:

(1) A reliable engine, that automatically downloads remote data, provides for error correction, synchronizes local and remote

*To whom correspondence should be addressed.

data, formats/post-processes this data and publishes the data for all users and/or applications.

(2) A group of predefined templates for synchronizing common biological databanks.

(3) A processing script set than can be applied to the fetched data for indexation and/or format conversion.

(4) XSLT Scripts that generate reports describing repository contents and statistics.

3 ENGINE BEHAVIOR

BioMAJ has been specifically designed to manage databank update cycles. It permits flexible data synchronization, controls execution of local post-download processing tasks and logs all activity for ulterior usage. All processing tasks are highly configurable and can be executed serially or in parallel. The engine supervises the execution of all tasks declared within each processing stage. In case of an error, only the faulty sub-parts of a treatment are re-executed, which is extremely useful when a treatment requires extensive computational resources. BioMaJ's features have been developed to iteratively execute workflows in order to routinely update huge and/or numerous databanks in batch mode.

The engine follows a predefined template mapped onto the processes of updating and indexing. Some parts of the template are static and only need custom properties to define the remote server address, the file transfer protocol, regular expressions that select remote files and whether or not downloaded files should be uncompressed. Other stages of BioMAJ templates are more open and can utilize a meta-scheduler for deferred program execution and a basic description language that enables one to implement personalized databank processing.

Each personalized databank update cycle is divided into five stages: initialization, reprocessing, synchronization, post-processing and deployment. The exact steps to be executed are described in a text file, referred to as the properties file. The standard behavior of each stage is as follows:

The initialization stage consists of setting up the session, loading the workflow and checking the state of both the current and previous databank releases.

The preprocessing stage handles various actions that must be performed prior to synchronization, such as sending email alert messages and ensuring available disk space. This stage is customizable and has the same features as the post-processing stage discussed subsequently.

During synchronization, the version of the latest databank release is extracted from the remote server using regular expressions on a specified remote file or through interrogation of the remote file's timestamp. Selected files are fetched using various protocols (ftp, http, rsync, local copy) and transfer integrity checks are performed to ensure that valid local copies have been made. Remote databank tree hierarchies can be preserved and the organization of both local and global repository contents is managed by the application. BioMAJ can perform multiple downloads or updates simultaneously. After the files are downloaded, BioMAJ can automatically uncompress files and reconstruct the desired local release. All file attributes as well as history and provenance information are stored in log files that can also be used to back trace local files and determine which files require resynchronization.

The post-processing stage consists of performing various tasks on synchronized data. Integration of processing programs is easy and flexible, as BioMAJ relies on system calls to execute shell scripts. Information about the context of the databank update cycle, such as input files, parameters and output locations, is transferred to each processing task using parameters declared in the template or shell variables, which are automatically set during system calls. Thus, generic wrappers for bioinformatic programs can be easily developed and reused by multiple workflows. In addition, BioMAJ provides a facility whereby task execution can be organized on a local machine or on a cluster using an external scheduler system. Description of the post-processing stage is handled by three hierarchical elements. The most basic unit of processing is a task, usually a wrapper script containing a set of serial processing commands. Tasks are grouped into meta-processes, which can be further organized into blocks. Blocks, meta-processes and processing tasks allow one to describe customized topologies for data processing. It also permits one to control the order in which data processing tasks are executed. Blocks are launched serially following their declaration order. In a given block, each meta-process is associated to a specific thread so that individual processing tasks can be run in parallel. This allows one to easily design a directed acyclic graph in which each vertex is a processing task with specific attributes and edges are the chronology of execution. Unlike more sophisticated workflow engines such as Taverna (Oinn *et al.*, 2004), neither explicit dependencies of data nor specific semantic have been formalized for the input and output channels of treatments. Users need a priori knowledge of the location of produced data, but this job is greatly facilitated by the default tree directory proposed by the application. Thus, BioMAJ makes it possible to define dependencies between different stages of data processing and to take into account relationships and inter-dependencies between treatments. Tasks can be executed either sequentially or in parallel to optimize execution time.

As an example, many sites routinely index all GenBank divisions both for EMBOSS and BLAST. One can parallelize this task with BioMAJ by defining two sequential blocks that each contains an independent meta-process for each GenBank division. In the first block, each meta-process includes an indexation task for EMBOSS. The parallel BLAST block includes a meta-process for each division that contains two sequential processing tasks: conversion from GenBank to FASTA format and formatdb indexation for BLAST. Each task can be executed on an individual cluster node via an external scheduler such as pbs or sge. In this example, the same wrapper script is called by each process. Behaviors are changed by personalizing the call parameters in the workflow description. Details about the syntax of the current example can be found on the website. The resulting DAG organization, which has also been used in grid application such as Dagman (Condor Team, 1990–2007), allows one to significantly reduce the amount of time required to process a new databank release. In the previously example i.e.: the indexation of the 18 divisions of Genbank both for EMBOSS and BLAST, if 18 CPUS (Xeon 5140 Woodcrest 2.3GHz, sharing data with Network File System) are used in parallel for data processing . The elapsed time takes 8h30, which is roughly 10 times faster than when run sequentially on a single processor. Due to IO contention and the unbalanced size of GenBank divisions, the speed increase does not evolve linearly with the number of meta-processes. However, the benefit when processing

large databanks is often measured in hours. Nevertheless, with certain tasks, such as indexation for SRS, chunk decomposition of data processing is not feasible, or difficult to perform. In the best case, the parallelization can only be incorporated in subsequent, sequential tasks. For SRS indexation, the parallelization has been directly included into the wrapper script using a parallel makefile program.

Finally, after all post-processing tasks are complete; the deployment stage makes the new release available and removes all temporary files and obsolete releases, based upon specified retention/release parameters. Deployment concludes a successful update cycle but BioMAJ can re-execute any faulty steps through its exception handling facilities.

4 BANK ADMINISTRATION AND MONITORING

BioMAJ also has many administrative functions such as online querying tools that interrogate repository contents and management commands that import, delete, rename and move databank releases. Therefore, it is possible to manage the local repository using mainly BioMAJ administrative functions.

Each session for a specific database is registered in an xml state file, which can be exploited under different time scales for monitoring and querying/updating. Within the short time scale, xml state files are used for exception handling. This functionality enables the recovery or restarting of an update cycle. Under the long time scale, xml state files are treated as history files, containing useful information about the database life cycle. BioMAJ can generate html reports including graphs describing statistics for each source and global statistics for all databanks included in the repository. Web reports are structured around meta-data information included in the workflow descriptions, which are text variables that annotate properties such as bank type, file format and descriptions of the databank and its associated processing. The resulting report offers a user-friendly way to browse the current and past state of the local repository. Thus, another significant benefit of using BioMAJ is that one can achieve a good level of quality of service (QoS) by obtaining a clear and precise state of the local data repository. QoS and traceability are essential in both large and small infrastructures to ensure reproducibility of data analysis pipelines that make use of the downloaded databanks.

5 RESULTS AND PROSPECT

The BioMAJ package currently provides the required functionality to mirror over 100 public domain databases (from servers such as NCBI, EBI, Expasy, tiger, etc.). New databases can easily be added through the configuration of a single properties file. Samples for the most common bioinformatics databases are available in the package. Each sample describes a dedicated workflow of databank synchronization, including in some case data post-processing. The project website has been specifically designed to share properties files and post-processing scripts between BioMAJ users.

Concerning data processing, multiple indexation post-processes are supported for various applications: NCBI blast, SRS, EMBOSS, GCG. Furthermore, post-processing scripts for format conversion and testing the index integrity after data processing are also available. The BioMAJ architecture is open; so that users can also easily integrate their personal homemade processing scripts (independent of programming language). Full guidelines on how to develop and integrate scripts can be found in the application manual.

In conclusion, our work was motivated by two main ideas. On one hand, academic research needs free software for managing public databanks. On the other hand, this project has been inspired by the workflow approach, an obvious way to capture knowledge for practical usage. Applied to databank synchronization and log reporting, workflows represent a way to normalize data replication and reduce the entropy associated with their dissemination. We believe that the normalization introduced by using a single application for databank synchronization is very likely the only way to efficiently control this process.

In the future, BioMAJ will integrate the bittorrent file transfer protocol and provide Rich Site Summary support. Together, these technologies will enable BioMAJ workflows to be executed when the software detects a torrentcast announcing a new release of the databank. Such automation will facilitate local synchronization as well as distribute the network bandwidth requirements associated with moving the large repositories from the remote centers to the local installations.

ACKNOWLEDGEMENTS

The authors would like to thank Jérome Gouzy and Jason S. Iacovoni for fruitful article comments.

Funding: This work has been funded by the RNG (Reseau National des Genopoles), the ReNaBi network, Région Bretagne, INRA and INRIA

Conflict of Interest: none declared.

REFERENCES

Condor Team,UOW. (1990–2007) Dagman terminology. Copyright © 1990–2007. Available at http://www.cs.wisc.edu/condor/manual/v6.8/2_11DAGMan_Applications.html.

Etzold,T. *et al*. (1996) SRS: information retrieval system for molecular biology data banks. *Meth Enzymol*, **266**, 114–128.

Goodman,J. (2004) Citrina. Available at http://www.gmod.org/wiki/index.php/Citrina.

Oinn,T. *et al*. (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, **20**, 3045–3054.

Shapovalov,M.V. *et al*. (2007) Biodownloader: bioinformatics downloads and updates in a few clicks. *Bioinformatics*, **23**, 1437–1439.