Computational and Structural Biotechnology Journal xxx (2017) xxx–xxx

COMPUTATIONAL AND STRUCTURAL BIOTECHNOLOGY JOURNAL

journal homepage: www.elsevier.com/locate/csbj

Scalability and Validation of Big Data Bioinformatics Software

Q2 Q1 Andrian Yang [a,b], Michael Troup [a], Joshua W.K. Ho [a,b,*]

[a] Victor Chang Cardiac Research Institute, Darlinghurst, NSW 2010, Australia
[b] St. Vincent's Clinical School, University of New South Wales, Darlinghurst, NSW 2010, Australia

A R T I C L E I N F O

Article history:
Received 30 March 2017
Received in revised form 30 June 2017
Accepted 17 July 2017
Available online xxxx

A B S T R A C T

This review examines two important aspects that are central to modern big data bioinformatics analysis – software scalability and validity. We argue that not only are the issues of scalability and validation common to all big data bioinformatics analyses, they can be tackled by conceptually related methodological approaches, namely divide-and-conquer (scalability) and multiple executions (validation). Scalability is defined as the ability for a program to scale based on workload. It has always been an important consideration when developing bioinformatics algorithms and programs. Nonetheless the surge of volume and variety of biological and biomedical data has posed new challenges. We discuss how modern cloud computing and big data programming frameworks such as MapReduce and Spark are being used to effectively implement divide-and-conquer in a distributed computing environment. Validation of software is another important issue in big data bioinformatics that is often ignored. Software validation is the process of determining whether the program under test fulfils the task for which it was designed. Determining the correctness of the computational output of big data bioinformatics software is especially difficult due to the large input space and complex algorithms involved. We discuss how state-of-the-art software testing techniques that are based on the idea of multiple executions, such as metamorphic testing, can be used to implement an effective bioinformatics quality assurance strategy. We hope this review will raise awareness of these critical issues in bioinformatics.

© 2017 The Authors. Published by Elsevier B.V. on behalf of Research Network of Computational and Structural Biotechnology. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

UNCORRECTED PROOF

1. Introduction

The term big data is used to describe data which are large with respect to the following characteristics: volume (amount of data generated), variety (type of data generated), velocity (speed of data generation), variability (inconsistency of data) and veracity (quality of captured data) [1]. Sequencing data is the most obvious example of big data in the field of bioinformatics, especially with the advancement in next-generation sequencing (NGS) technology and single cell capture technology. Other examples of big data in bioinformatics include electronic health records, which contain a variety of information including phenotypic, diagnostic and treatment information; and medical imaging data, such as those produced by magnetic resonance imaging (MRI), positron emission tomography (PET) and ultrasound. Furthermore, emerging big data relevant to biomedical research also include data from social networks and wearable devices.

One particularly major advancement in experimental molecular biology within the last decade has been the significant increase in sequencing data available for analysis, at a cheaper cost [2]. The cost of sequencing per genome has reduced from $100,000,000 in 2001, to $10,000,000 in 2007, down to a figure close to $1000 today. The $1000 genome is already a reality [3]. Currently, the data that comes out of a NGS machine are in the order of several hundred gigabytes for a single human genome. With the rapid advancement in single-cell capture technology and the increasing interest in single-cell studies, it is expected that the amount of sequencing data generated will increase substantially as each single-cell run can generate profiles for hundreds to thousands of samples [4]. In this review, we will focus specifically on bioinformatics software that deals with NGS data as this is currently one of the most prominent and rapidly expanding source of big data in bioinformatics.

In this review, we argue that the two main issues that are fundamental to designing and running big data bioinformatics analysis are: the need for analysis tools which can scale to handle the large and unpredictable volume of data (Scalability) [4–7], and methods that can effectively determine whether the output of a big data analysis conforms to the users' expectation (Validation) [8,9]. In general, there are many other issues associated with bioinformatics big data analysis, such as storage, security and integration [10]. However, these issues have existed even before the rise of big data in bioinformatics, and are these issues are typically targeted to specific use cases, such as the storage of sensitive patient data and integration of several specific types of data. Solutions to these specific issues are available [11,12], though there may be additional challenges associated in implementing

* Corresponding author at: Victor Chang Cardiac Research Institute, Lowy Packer Building, 405 Liverpool Street, Darlinghurst, NSW 2010, Australia.
E-mail address: j.ho@victorchang.edu.au (J.W.K. Ho).

http://dx.doi.org/10.1016/j.csbj.2017.07.002
2001-0370/© 2017 The Authors. Published by Elsevier B.V. on behalf of Research Network of Computational and Structural Biotechnology. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

ARTICLE IN PRESS

the solution due to the increased volume and noise. Nonetheless, these issues are mostly specific to individual application areas. We believe that if we can effectively deal with the scalability and validation problem, it will go a long way in terms of making big data analysis more widespread in practice. This review aims to provide an overview of the technological development that deals with the scalability and validation problems in big data bioinformatics for sequence-based analysis tools.

2. Scalability

Scalability is not a unique challenge in big data analysis. In fact, software scalability has always been an issue since the early days of bioinformatics because of the high algorithmic complexity of some of the algorithms such as those involving global multiple sequence alignment. The early focus on scalability is on parallelising the computation, while a lot less attention is paid on optimally distributing the data. Efforts to make bioinformatics software scalable have continuously been made with the evolution of new hardware technologies, such as cluster computing, grid computing, Graphical Processing Unit (GPU) technology, and cloud computing. Currently in the age of big data bioinformatics, the focus is not only on parallelising computational intensive algorithms, but also on highly distributed storage and efficient communication among various distributed storage or computational units. Furthermore, the volume and variety of data can change dynamically in response to potentially unpredictable user demand. For example, in a medium-sized local sequencing centre, the volume of data can grow rapidly during certain unexpected peak periods, but remain constant during other periods. This variability of demand on computational resources is also a critical feature of modern big data bioinformatics analysis. In this section, we will review the evolution of parallel distributed computing technologies and how they have contributed to solving the issue of scalability of bioinformatics software. In particular, we will discuss how modern cloud computing technology and big data analysis frameworks, such as MapReduce and Spark, can be effectively used to deal with the scalability problem in the big data era.

2.1. Cluster Computing

Early attempts at scaling bioinformatics software beyond massively parallel (super) computers involved networking individual computers into clusters to form a parallelised distributed-memory machine. In this configuration, computations are performed by splitting and distributing tasks across Central Processing Units (CPUs) in a way that is similar to the symmetric multiprocessing (SMP) approach utilised in massively parallel computers. Unlike SMP, which relies on a shared main memory, clusters have distributed-memory, with each node having its own memory and hard drive, thus presenting a new challenge in developing software for cluster environments. To help with the development of cluster-based software, communications protocols and software tools, such as Message Passing Interface (MPI) [13] and Parallel Virtual Machine (PVM) [14], have been developed for orchestrating computations across nodes. An example of bioinformatics software designed for cluster computing is mpiBLAST, an MPI-based, parallelised implementation of the basic local alignment search tool (BLAST) algorithm which performs pairwise sequence similarity between a query sequence and a library or database of sequences [15]. The approach taken by mpiBLAST includes the use of a distributed database to reduce both the number of sequences searched and disk I/O in each node, thereby improving the performance of the BLAST algorithm. MASON is another example of MPI-based bioinformatics software for performing multiple sequence alignment algorithms using the ClustalW algorithm [16]. MASON speeds up the execution of ClustalW by parallelising the time- and compute-intensive step of calculating a distance matrix of the input sequences, and the final progressive alignment stage.

2.2. Grid Computing

The next approach in scaling bioinformatics software comes with the introduction of grid computing, which represents an evolution in the distributed computing infrastructure. Grid computing allows for a collection of heterogeneous hardware, such as desktops, servers and clusters, which may be located in different geographical locations, to be connected through the Internet to form a massively distributed high performance environment [17]. Although conceptually similar to a cluster, grid computing presents a different set of challenges for developing software. The comparatively large latency between nodes in a grid environment compared to a cluster environment means that software for grid needs to be designed with minimum communication between nodes. Furthermore, the heterogeneity of the grid environment means that software may need to take into account differences in the underlying operating system and the system architecture of the nodes. Development of bioinformatics software for a Grid typically uses a middleware layer which abstracts away the underlying grid architecture management. A widely-used middleware layer is the Globus Toolkit, a software toolkit for managing and developing in a grid environment [18]. An example of a bioinformatics software for the Grid environment is GridBLAST, an implementation of BLAST with Globus as the middleware layer for distributing BLAST queries across nodes in the grid [19]. Aside from Globus, there are also bioinformatics-specific grid middleware layers such as myGrid [20] and Squid [21].

2.3. GPGPU

The introduction of general-purpose computing on GPUs (GPGPUs) revived interest in the massively parallel approach initially used before the distributed computing approach becomes the mainstream. GPUs are specialised processing units designed for performing graphic rendering. Unlike a CPU, which has a limited number of multi-processing units, a GPU has a large number of processing unit in the order of hundreds and thousands, thus allowing for the high computational throughput required for rendering 3D graphics. Though the GPU is not a new technology, early GPU architectures were hardwired for graphics rendering and thus it was not until the development of a more generalised architecture which support general-purpose computing that GPU become more widely used for computation. As with other technologies, there are challenges associated with implementing bioinformatics software on GPUs due to the single instruction multiple data (SIMD) programming paradigm where data are processed in parallel using the same set of instructions. Due to its architecture, computation for GPU will need to designed with minimum level of branching (homogenous execution) with high computational complexity in order to fully take advantage of the high multiprocessing capability of the GPU. One of the early bioinformatics software utilising GPGPUs is GPU-RAxML (Randomized Axelerated Maximum Likelihood), a GPU based implementation of RAxML program for the construction of phylogenetic trees using a Maximum Likelihood method [22]. GPU-RAxML utilises the BrookGPU programming environment [23], which supports both OpenGL and DirectX graphic libraries, to parallelise the longest loop in the RAxML program, which accounts for 50% of the execution time. Another example of GPU-accelerated bioinformatics software is CUDASW++, an implementation of the dynamic-programming based Smith-Waterman (SW) algorithm for local sequence alignment [24]. CUDASW++ utilises the CUDA (Compute Unified Device Architecture) programming environment [25], developed for NVIDIA GPU, to implement two parallelisation strategies of the SW algorithm based on the length of the subject sequence.

2.4. Cloud Computing

Cloud computing is defined by the United States' National Institute of Standards and Technology as '…a model for enabling ubiquitous,

UNCORRECTED PROOF

Download English Version:

https://daneshyari.com/en/article/8408563

Download Persian Version:

https://daneshyari.com/article/8408563

Daneshyari.com