Hindawi Publishing Corporation
ISRN Biomathematics
Volume 2013, Article ID 615630, 14 pages
http://dx.doi.org/10.1155/2013/615630

Review Article

An Overview of Multiple Sequence Alignments and Cloud Computing in Bioinformatics

Jurate Daugelaite,[1] Aisling O' Driscoll,[2] and Roy D. Sleator[1]

[1] *Department of Biological Sciences, Cork Institute of Technology, Rossa Avenue, Bishopstown, Cork, Ireland*
[2] *Department of Computing, Cork Institute of Technology, Rossa Avenue, Bishopstown, Cork, Ireland*

Correspondence should be addressed to Roy D. Sleator; roy.sleator@cit.ie

Received 24 May 2013; Accepted 23 June 2013

Academic Editors: M. Glavinovic and X.-Y. Lou

Multiple sequence alignment (MSA) of DNA, RNA, and protein sequences is one of the most essential techniques in the fields of molecular biology, computational biology, and bioinformatics. Next-generation sequencing technologies are changing the biology landscape, flooding the databases with massive amounts of raw sequence data. MSA of ever-increasing sequence data sets is becoming a significant bottleneck. In order to realise the promise of MSA for large-scale sequence data sets, it is necessary for existing MSA algorithms to be run in a parallelised fashion with the sequence data distributed over a computing cluster or server farm. Combining MSA algorithms with cloud computing technologies is therefore likely to improve the speed, quality, and capability for MSA to handle large numbers of sequences. In this review, multiple sequence alignments are discussed, with a specific focus on the ClustalW and Clustal Omega algorithms. Cloud computing technologies and concepts are outlined, and the next generation of cloud base MSA algorithms is introduced.

1. Introduction

Multiple sequence alignments (MSA) are an essential and widely used computational procedure for biological sequence analysis in molecular biology, computational biology, and bioinformatics. MSA are completed where homologous sequences are compared in order to perform phylogenetic reconstruction, protein secondary and tertiary structure analysis, and protein function prediction analysis [1]. Biologically good and accurate alignments can have significant meaning, showing relationships and homology between different sequences, and can provide useful information, which can be used to further identify new members of protein families. The accuracy of MSA is of critical importance due to the fact that many bioinformatics techniques and procedures are dependent on MSA results [1].

Due to MSA significance, many MSA algorithms have been developed. Unfortunately, constructing accurate multiple sequence alignments is a computationally intense and biologically complex task, and as such, no current MSA tool is likely to generate a biologically perfect result. Therefore, this area of research is very active, aiming to develop a method which can align thousands of sequences that are lengthy and produce high-quality alignments and in a reasonable time [2, 3]. Alignment speed and computational complexity are negatively affected when the number of sequences to be aligned increases. The recent advances in high throughput sequencing technologies means that this sequence output is growing at an exponential rate, the biology, landscape being punctuated by a number of large-scale projects such as the Human Genome Project [4], 1000 Genomes Project [5], and Genome 10K Project [6]. Indeed, technologies such as Roche/454 [7], Ilumina [8], and SOLiD [9] are capable of producing Giga basepairs (Gbp) per machine per day [10]. The entire worldwide second-generation sequencing capacity surpasses 13 Pbp per year (recorded in 2011) and is continuing to increase yearly by a factor of five [11]. Other large-scale data is emerging from high-throughput technologies, such as gene expression data sets, protein 3D structures, protein-protein interaction, and others, which are also generating

huge sequence data sets. The analysis and storage of the growing genomic data represents the central challenge in computational biology today.

As the protein alignment problem has been studied for several decades, studies have shown considerable progress in improving the accuracy, quality, and speed of multiple alignment tools, with manually refined alignments continuing to provide superior performance to automated algorithms. However, more than three sequences of biologically relevant length can be difficult and time consuming to align manually; therefore, computational algorithms are used as a matter of course [2]. Sequences can be aligned using their entire length (global alignment) or at specific regions (local alignment). Multiple sequence alignment for protein sequences is much more difficult than the DNA sequence equivalent (containing only 4 nucleotides) due to the fact that there are 20 different amino acids. Global optimization techniques, developed in applied mathematics and operations research, provide a generic toolbox for solving complex optimization problems. Global optimization is now used on a daily basis, and its application to the MSA problem has become a routine [12]. Local alignments are preferable; however, they can be challenging to calculate due to the difficulty associated with the identification of sequence regions of similarity. The two major aspects of importance for MSA tools for the user are biological accuracy and the computational complexity. Biological accuracy concerns how close the multiple alignments are to the true alignment and are the sequences aligning correctly, showing insertions, deletions, or gaps in the right positions. Computational complexity refers to the time, memory, and CPU requirements. Complexity is of increasing relevance as a result of the increasing number of sequences needed to be aligned. The complexity of a primal MSA tools was always $O(L^n)$, where O is complexity, L is length of the sequence, and n is the number of sequences to be aligned. Until recently, this was not a problem because n was always smaller than L; therefore, most algorithms concentrated on how to deal with lengthy sequences rather than the number of sequences, and now the situation has changed, where a lot of alignments have n larger than L; therefore, new and more recent MSA algorithms are concentrating not only on the length of sequences but also on the increasing number of sequences [13].

A wide range of computational algorithms have been applied to the MSA problem, including slow, yet accurate, methods like dynamic programming and faster but less accurate heuristic or probabilistic methods. Dynamic programming (DP) is a mathematical and computational method which refers to simplifying a complicated problem by subdividing it into smaller and simpler components in a repeated manner. The dynamic programming technique can be applied to global alignments by using methods such as the Needleman-Wunsch algorithm [14] and local alignments by using the Smith-Waterman algorithm [15]. Up to the mid-1980s, the traditional multiple sequence alignment algorithms were only best suited for two sequences, so when it came to producing multiple sequence alignment with more than two sequences, it was found that completing the alignment manually was faster than using traditional dynamic programming algorithms [16]. Dynamic programming algorithms are used for calculating pairwise alignments (two sequence alignments) with the time complexity of $O(L^n)$. In theory, this method could be extended to more than two sequences; however, in practice, it is too complex, because the time and space complexity becomes very large [17]. Therefore, producing multiple sequence alignment requires the use of more sophisticated methods than those used in producing a pairwise alignment, as it is much more computationally complex. Finding a mathematically optimal multiple alignment of a set of sequences can generally be defined as a complex optimization problem or NP-complete problem as it must identify an MSA with the highest score from the entire set of alignments; therefore, heuristic ("best guess") methods must be used.

2. Multiple Sequence Alignment Algorithms

The most popular heuristic used from which the majority of multiple sequence alignments are generated is that developed by Feng and Doolittle [18], which they referred to as "progressive alignment" [16, 18]. Progressive alignment works by building the full alignment progressively, firstly completing pairwise alignments using methods such as the Needleman-Wunsch algorithm, Smith-Waterman algorithm, k-tuple algorithm [19], or k-mer algorithm [20], and then the sequences are clustered together to show the relationship between them using methods such as mBed and k-means [21]. Similarity scores are normally converted to distance scores and guide trees are constructed using these scores by guide tree building methods such as Neighbour-Joining (NJ) [22] and Unweighted Pair Group Method with Arithmetic Mean UPGMA [23]. Once the guide tree is built, the multiple sequence alignment is assembled by adding sequences to the alignment one by one according to the guide tree, that is, the most similar sequences added first and then gradually adding more distant sequences. Unfortunately, this heuristic has a greedy nature; that is, it only looks at two sequences at a time and ignores the remaining data and therefore cannot guarantee an optimal solution. Also, if mistakes are made in the initial stages of the alignment, they cannot be fixed in later stages, and the mistake will continue throughout the alignment process with the problem worsening as the number of sequences increases. Progressive alignment is the foundation procedure of several popular alignment algorithms such as ClustalW [24], Clustal Omega [21], MAFFT [25], Kalign [26], Probalign [27], MUSCLE [13], DIALIGN [28], PRANK [29], FSA [30], T-Coffee [31, 32], ProbCons [33],and MSAProbs [34]. Different methods for producing multiple sequence alignment exist, and their use depends on user preferences and sequence length and type, as shown in Table 1.

An improved version of the progressive alignment method was developed called "iterative progressive algorithms." These algorithms work in a similar manner to progressive alignment; however, this approach repeatedly applies dynamic programming to realign the initial sequences in order to improve their overall alignment quality, also at the same time adding new sequences to the growing MSA.

Table 1: Types of multiple sequence alignment and corresponding algorithms.

Types of MSA alignment	MSA algorithms
Pairwise alignment	Needleman-Wunsch, k-mer, k-tuple, and Smith-Waterman algorithms.
Progressive alignment	Clustal Omega, ClustalW, MAFFT, Kalign, Probalign, MUSCLE, Dialign, ProbCons, and MSAProbs.
Iterative progressive alignment	PRRP, MUSCLE, DIALIGN, SAGA, and T-COFFEE.
Homology search tools	BLAST, PSI-BLAST, and FASTA.
Structure incorporating alignment	3D-COFFEE, EXPRESSO, and MICAlign.
Motif alignment	PHI-Blast, GLAM2.
Short-read alignment	Bowtie, Maq, and SOAP.

The iteration benefits the alignment by correcting any errors produced initially, therefore improving the overall accuracy of the alignment [35]. Iterative methods are able to give 5%–10% more accurate alignments; however, they are limited to alignments of a few hundred sequences only [21]. The most used iterative alignment algorithms include PRRP [36], MUSCLE [13], Dialign [28], SAGA [37], and T-COFFEE [32, 38].

Multiple sequence alignments can also be constructed by using already existing protein structural information. It is believed that by incorporating structural information to the alignment, the final MSA accuracy can be increased; therefore, most structure-based MSA are of higher quality than those based on sequence alignment only. The reason for structure-based MSA being of better quality is not due to a better algorithm but rather an effect of structures evolutionary stability that is, structures evolve more slowly than sequences [39]. The most popular structure and based MSA is 3D-COFFEE [40], and others include EXPRESSO [41] and MICAlign [42].

Motif discovery algorithms are another type of MSA algorithms that are used. These methods are used to find motifs in the long sequences; this process is viewed as a "needle in a haystack" problem, due to the fact that the algorithm looks for a short stretch of amino acids (motif) in the long sequence. One of the most widely used tools for searching for motifs is PHI-Blast [43] and Gapped Local Alignments of Motifs (GLAM2) [44].

Short sequence alignment algorithms are also beginning to emerge, primarily due to advances in sequencing technologies. Most genomic sequence projects use short read alignment algorithms such as Maq [45], SOAP [46], and the very fast Bowtie [47] algorithms.

3. Top Multiple Sequence Alignment Algorithms

The number of multiple sequence alignment algorithms is increasing on almost monthly bases with ~1-2 new algorithms published per month. The computational complexity and accuracy of alignments are constantly being improved; however, there is no biologically perfect solution as yet. ClustalW (one of the first members of the Clustal family after ClustalV) is probably the most popular multiple sequence alignment algorithm, being incorporated into a number of so-called black box commercially available bioinformatics packages such DNASTAR, while the recently developed Clustal Omega algorithm is the most accurate and most scalable MSA algorithms currently available. ClustalW and Clustal Omega are described later, and also a brief description is provided for the T-Coffee, Kalign, Mafft, and MUSCLE multiple sequence alignment algorithms.

3.1. ClustalW. ClustalW [24] was introduced by Thompson et al. in 1994 and quickly became the method of choice for producing multiple sequence alignments as it presented a dramatic increase in alignment quality, sensitivity, and speed in comparison with other algorithms. ClustalW incorporates a novel position-specific scoring scheme and a weighting scheme for downweighting overrepresented sequence groups, with the "W" representing "weights." Firstly, the algorithm performs a pairwise alignment of all the sequences (nucleotide or amino acid) using the k-tuple method by Wilbur and Lipman [19] which is a fast, albeit approximate, method or the Needleman-Wunsch method [14] which is known as the full dynamic programming method. These methods calculate a matrix which shows the similarity of each pair of sequences. The similarity scores are converted to distance scores, and then the algorithm uses the distance scores to produce a guide tree, using the Neighbour-Joining (NJ) method [22] for guide tree construction. The last step of the algorithm is the construction of the multiple sequence alignment of all the sequences. The MSA is constructed by progressively aligning the most closely related sequences according to the guide tree previously produced by the NJ method (see Figure 1 for an overview).

3.1.1. Pairwise Alignment. The k-tuple method [19], a fast heuristic "best guess" method, is used for pairwise alignment of all possible sequence pairs. This method is specifically used when the number of sequences to be aligned is large. The similarity scores are calculated as the number of k-tuple matches (which are runs of identical residues, usually 1 or 2 for protein residues or 2–4 for nucleotide sequences) in the alignment between a pair of sequences. Similarity score is calculated by dividing the number of matches by the sum of all paired residues of the two compared sequences. Fixed penalties for every gap are subtracted from the similarity score with the similarity scores later converted to a distance

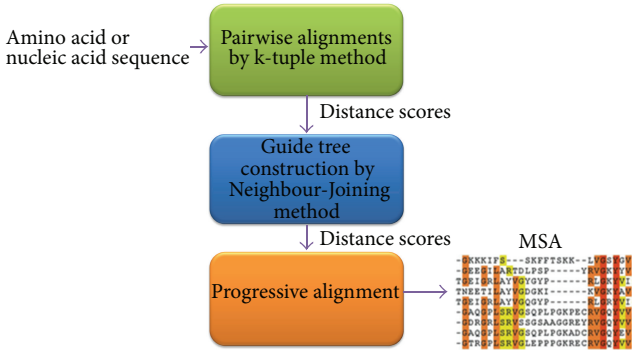

Figure 1: ClustalW algorithm, which works by taking an input of amino acid or nucleic acid sequences, completing a pairwise alignment using the k-tuple method, guide tree construction using the Neighbour-Joining method, followed by a progressive alignment to output a multiple sequence alignment.

score by dividing the similarity score by 100 and subtracting it from 1.0 to provide the number of differences per site.

Then, all of the k-tuples between the 2 sequences are located using a hash table. A dot matrix plot between the two sequences is produced with each k-tuple match represented as a dot. The diagonals with the most matches in the plot are found and marked within a selected "Window Size" of each top diagonal. This sets the most likely region for similarity between the two sequences to occur. The last stage of the k-tuple method is to find the full arrangement of all k-tuple matches by producing an optimal alignment similar to the Needleman-Wunsch method but only using k-tuple matches in the set window size, which gives the highest score. The score is calculated as the number of exactly matching residues in the alignment minus a "gap penalty" for every gap that was introduced.

3.1.2. Guide Tree Construction. ClustalW produces a guide tree according to the "Neighbor-Joining" method. The NJ method is often referred to as the star decomposition method [48]. The NJ method keeps track of nodes on a tree rather than a taxa (a taxonomic category or group, such as phylum, order, family, genus, or species) or clusters of taxa. The similarity scores are used from the previous k-tuple method and stored in a matrix. A modified distance matrix is constructed in which the separation between each pair of nodes is adjusted by calculating an average value for divergence from all other nodes. The tree is then built by linking the least distant pair of nodes. When two nodes are linked, their common ancestral node is added to the tree and the terminal nodes with their respective branches are removed from the tree. This process allows the conversion of the newly added common ancestor into a terminal node tree of reduced size. At each stage in the process, two terminal nodes are replaced by one new node. The process is completed when two nodes remain separated by a single branch. The tree produced by the NJ method is un-rooted and its branch lengths are proportional to divergence along each branch. The root is placed at the position at which it can make the equal branch length on either side of the root. The guide tree is then used to calculate weight for each sequence, which depends on the distance from branch to the root. If a sequence shares a common branch with another sequence, then the two or more sequences will share the weight calculated from the shared branch, and the sequence lengths will be added together and divided by the number of sequences sharing the same branch.

3.1.3. Progressive Alignment. ClustalW's progressive alignment uses a series of pairwise alignments to align sequences by following the branching order of the guide tree previously constructed by the NJ method. The procedure starts at the tips of the rooted tree proceeding towards the root. At each step, a full dynamic programming algorithm is used with a residue weight matrix (BLOSUM) and penalties for opening and extending gaps.

3.2. Clustal Omega. Clustal Omega is the latest MSA algorithm from the Clustal family. This algorithm is used to align protein sequences only (though nucleotide sequences are likely to be introduced in time). The accuracy of Clustal Omega on small numbers of sequences is similar to other high-quality aligners; however, on large sequence sets, Clustal Omega outperforms other MSA algorithms in terms of completion time and overall alignment quality. Clustal Omega is capable of aligning 190,000 sequences on a single processor in a few hours [21]. The Clustal Omega algorithm produces a multiple sequence alignment by firstly producing pairwise alignments using the k-tuple method. Then, the sequences are clustered using the mBed method. This is followed by the k-means clustering method. The guide tree is next constructed using the UPGMA method. Finally, the multiple sequence alignment is produced using the HHalign package, which aligns two profile hidden Markov models (HMM) as shown in Figure 2.

3.2.1. Pairwise Alignment. Pairwise alignment of Clustal Omega is produced using the k-tuple method, the same technique as employed by ClustalW, described earlier.

3.2.2. Sequence Clustering. After the similarity scores are determined from the pairwise alignment, Clustal Omega employs the mBed method which has a complexity of $O(N \log N)$. mBed works by "emBedding" each sequence in a space of n dimensions, where n is proportional to $\log N$. Each sequence is then replaced by an n element vector. Each element is the distance to one of n "reference sequences." These vectors can then be clustered extremely quickly by methods such as k-means or UPGMA [49].

3.2.3. k-Means Clustering. Clustal Omega uses the k-means++ clustering method by Arthur and Vassilvitskii [50]. The k-means method is a widely used clustering technique which seeks to minimise the average squared distance between points in the same cluster. This method is very simplistic and fast at clustering sequences. k-means++ successfully overcomes the problems of defining initial

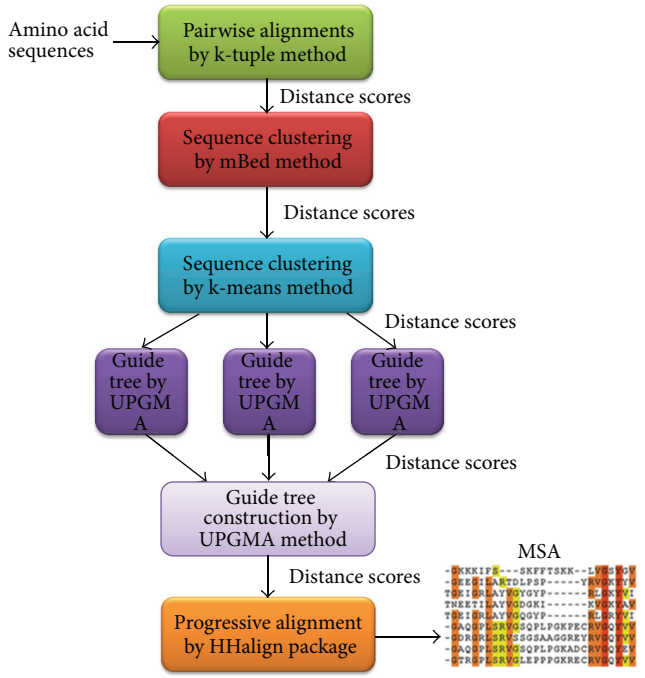

Figure 2: Clustal Omega algorithm, which works by taking an input of amino acid sequences, completing a pairwise alignment using the k-tuple method, sequence clustering using mBed method, and k-means method, guide tree construction using the UPGMA method, followed by a progressive alignment using HHalign package to output a multiple sequence alignment.

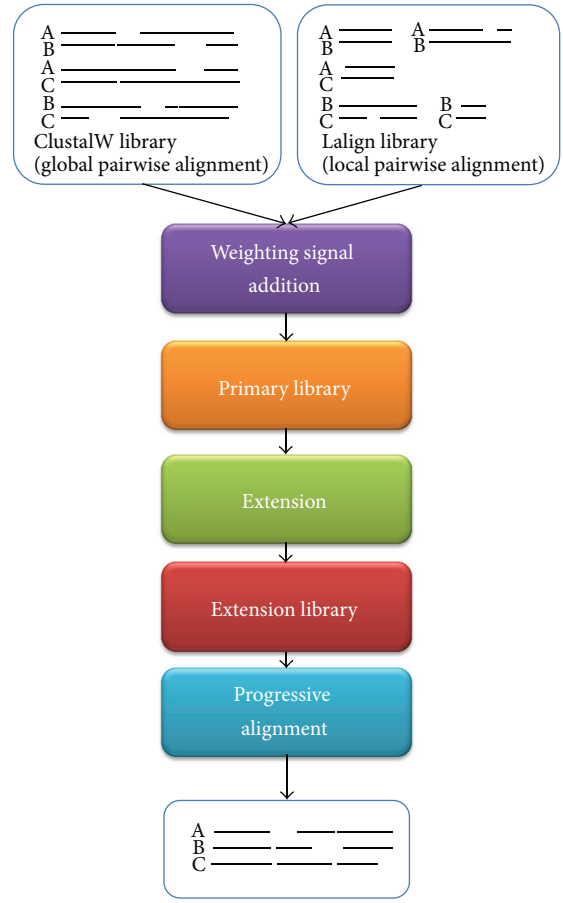

Figure 3: T-COFFEE diagram. Steps involved in producing multiple sequence alignment by T-Coffee method.

cluster centres for k-means and improves the speed and accuracy of the k-means method [50].

3.2.4. Guide Tree Construction. Clustal Omega uses the UPGMA method for sequence guide tree construction. UPGMA is a straightforward method of tree construction which uses a sequential clustering algorithm in which local homology between operational taxanomic units (OTUs) is identified in order of similarity. The tree is constructed in a stepwise fashion. Pairs of OTUs that are most similar are first determined and then are treated as a new single OTU. Subsequently, from the new group of OTUs, the pair with the highest similarity is identified and clustered. This process continues until only two OTUs remain [20].

3.2.5. Progressive Alignment. Clustal Omega uses the HHalign package by Johannes Soding 2005 [51] for completing progressive alignments. This method aligns two profile hidden Markov models, instead of a profile-profile comparison; this improves the sensitivity and alignment quality significantly. All sequence-profile and sequence HMM comparison methods are based on the log-odds score. The log-odds score is a measure for how much more probable it is that a sequence is emitted by an HMM rather than by a random null model.

3.3. T-Coffee. T-Coffee, which stands for tree-based consistency objective function for alignment evolution, is an iterative MSA algorithm. T-Coffee provides a simple and flexible means of producing multiple sequence alignments by using heterogeneous data sources which are provided to T-Coffee via library of global and local pairwise alignments. In the progressive alignment, pairwise alignments are completed first in order to produce a distance matrix. This matrix is then used to produce a guide tree using the Neighbour-Joining method. The tree is then used to group the sequences together during the multiple sequence alignment process. The closest two sequences on the tree are aligned first using normal dynamic programming method. The alignment uses weighting in the extended library as shown in Figure 3. This is done in order to align the residues in two sequences. The next two closest sequences suggested by the guide tree or prealigned group of sequences are always joined. This continues until all the sequences have been aligned. To align two groups of prealigned sequences, the scores from the extended library are used; however, the average library scores in each column of existing alignment are taken. T-Coffee increases the accuracy of the alignments 5–10% in comparison to ClustalW; however, the algorithm presents disadvantages such as weak scalability. T-Coffee can only align maximum 100 sequences without loss of accuracy [52].

3.4. MAFFT. Another good quality, highly accurate multiple sequence alignment is an algorithm called MAFFT. MAFFT uses two novel techniques; firstly, homologous regions are identified by the fast Fourier transform (FFT). In this method, the amino acid sequences are converted to a sequence composed of volume and polarity values of each amino acid residue. Secondly, a simplified scoring system is introduced which reduces CPU time and increases the accuracy of alignments. MAFFT uses two-cycle heuristics, the progressive method (FFT-NS-2) and iterative refinement method (FFT-NS-i). In the (FFT-NS-2) method, low-quality all-pairwise distances are rapidly calculated, a provisional MSA is constructed, refined distances are calculated from the MSA, and then the second method (FFT-NS-i) is performed. (FFT-NS-i) is a one cycle progressive method; it is faster and less accurate than the FFT-NS-2. Part tree option is available to alignments of ~50,000 sequences, and this method allows scalability [53].

3.5. Kalign. Kalign is yet another good quality multiple sequence alignment algorithm. The algorithm follows a strategy that is very similar to the standard progressive methods for sequence alignments, such as pairwise distances which are calculated firstly by using k-tuple method adopted from ClustalW. The guide tree is constructed using either UPGMA or Neighbour-Joining method, and progressive alignment is completed by following the guide trees. In contrast to the existing methods, what makes this algorithm different is the use of Wu-Manber approximate string-matching algorithm. This method is used in the distance calculation and in the dynamic programing used to align the profiles. This method allows string matching with mismatches. Also, the distances between two strings are measured using Levenshtein edit distance.

3.6. MUSCLE. MUSCLE stands for multiple sequence comparison by log expectation. MUSCLE uses two distance measures, kmer distance for unaligned pairs of sequences and the Kimura distance method for aligned pairs of sequences. Guide trees are produced using UPGMA method. A progressive alignment is then constructed following the order of the guide tree. This process produces an initial multiple sequence alignment. The program carries out stage two which is completed in order to improve the progressive alignment. The initial guide tree is reestimated using Kimura distance method, and this method is known to be much more accurate than kmer, and however it requires an alignment. Once the distances are computed, the UPGMA method reclusters the sequences producing second guide tree. A progressive alignment is calculated following second tree, producing second multiple sequence alignment. A new multiple sequence alignment is produced using both the first multiple sequence alignment and the second one. New multiple sequence alignment is produced by realigning the two profiles. If the SP score is improved on the second MSA, then the new alignment is kept and the old is discarded; otherwise, it is deleted and the first alignment is used [20].

4. Cloud Computing Technologies for the MSA Problem

In order to realise the promise of MSA for large-scale sequence data sets, it is necessary for existing MSA algorithms to be executed in a parallelised fashion with the sequence data distributed over a computing cluster or server farm. High performance computing has become very important in large-scale data processing. The message passing interface (MPI) and graphics processing unit (GPU) are the primal programming APIs for parallel computing. In recent years, cloud computing has received significant attention, though many different definitions of the technology exist. As an often mistakenly used analogy for the Internet or anything "online," the "cloud" is a familiar buzzword. Alternatively, some analysts tend to provide a very narrow definition of cloud computing as an updated version of utility computing [54]. While such a definition is not inaccurate, it does not describe the whole picture. A more precise definition is provided by the National Institute of Standards and Technology (NIST) who describe it as "*a pay-per-use model of enabling available, convenient and on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [48]. The term cloud computing was cocoined by an Irish entrepreneur Sean O'Sullivan, cofounder of Avego Ltd., along with George Favaloro from Boston, Massachusetts [55]. It should be noted that while cloud computing is a recent technology, some of the concepts behind cloud computing are not new, such as distributed systems, grid computing, and parallelised programming. Cloud services remove the need for the user to be in the same physical location as the hardware that stores its data and/or applications. The cloud can do both, own and store the hardware and the software needed for a user to run their applications or processes. Truly, one of the biggest enablers of cloud computing is the virtualisation technology. Virtualisation is a layered approach for running multiple independent virtual machines (VM) on a single physical machine, sharing the resources yet running on its own operating systems and applications [49]. The concept of virtualisation can be applied to devices, servers, operating systems, applications, and networks. Virtualisation is beneficial due to providing easy access to data, the ability to share applications from central environment, and it reduces the cost associated with data backups, maintenance personnel, and software licensing [56]. A virtual machine (VM) is a piece of software that runs on a local machine emulating the properties of a computer. The emulator provides a virtual central processing unit (CPU), network card, and hard disk. Popular products such as VMware [57] and KVM [58] provide virtual machines to customers. The differences between traditional and virtual server models can be seen in Figure 4.

Cloud computing is an information technology discipline, which provides computing, such as the necessary storage space and processing power, on demand and as a service. Such a "rental" model is often referred to as Infrastructure as a Service (IaaS) or utility computing. However, another aspect of cloud computing is Platform as a Service

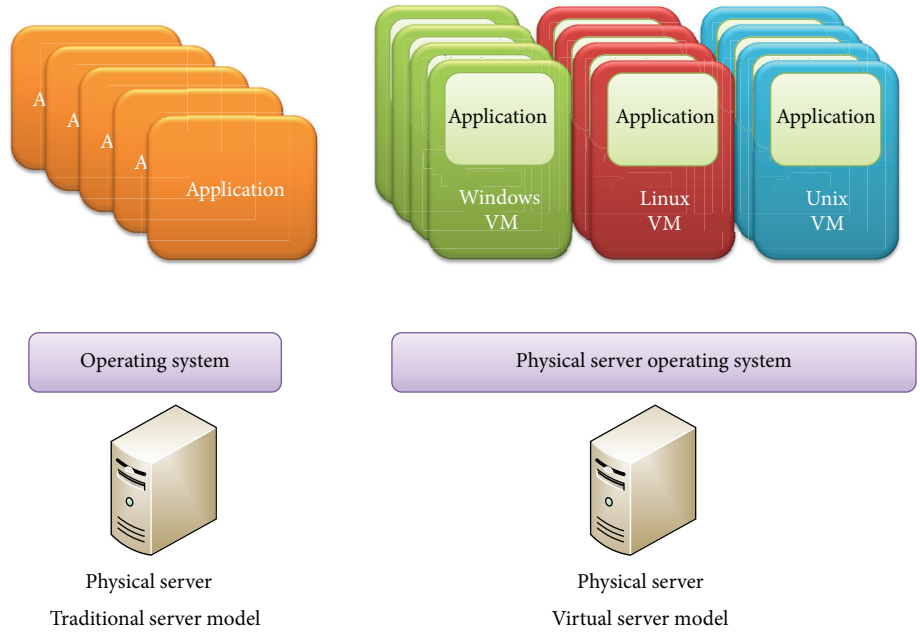

Figure 4: Transition from traditional computing where applications interact with the hardware via one instance of the operating system (OS) to virtualised computing where multiple OS images share the hardware resources.

Table 2: Cloud computing deployment models.

Deployment model	Description
Public	A cloud infrastructure that is owned by a cloud provider, who made resources such as infrastructure, software, and platform available to general public for "pay-per-use" basis, via Internet.
Private	A cloud which is owned and used by a single organisation. This model provides more security measures in comparison to other models.
Community	A cloud which is shared amongst several users or organizations.
Hybrid	A cloud that is a combination of public, community, and private clouds.

(PaaS) which allows users to build software applications by building on software libraries or development platforms already developed by the cloud provider. Another enabler includes advances in "Big Data" technologies that have realised the potential of distributed systems, grid computing, and parallelised programming enabling developers to focus on solving the problem at hand rather than maintaining the robustness of the distributed system and the parallelised programming structure. The negative consequences caused by a failure of a single machine in a distributed system have been eliminated by improving the overall distributed systems structure [59]. Cloud computing provides four deployment models. Selecting a deployment model depends on the users, organisations requirements according to their suitability, as shown in Table 2.

In order to use cloud computing services, one requirement has to be met, which is Internet connection. Clouds are accessed via the Internet which is of benefit; it enables users or organisations to access their stored data, to download or upload data at any given time or place through any device which has wireless or wired Internet connection.

Cloud computing is often considered to provide only rental of computing storage and power; however, cloud computing provides many service models according to an "XaaS" paradigm, representing "X as a Service," "Anything as a Service," or "Everything as a Service." The acronym refers to an increasing number of services that are provided over the Internet rather than the local services. XaaS is the essence of cloud computing. The primary service models of cloud computing are Software as a service (SaaS), Infrastructure as a service (IaaS) and Platform as a service (PaaS) as illustrated in Figure 5.

4.1. Infrastructure as a Service (IaaS). An IaaS provider allows subscribed users to completely outsource the storage and resources, such as hardware and software that the user may require. IaaS companies provide offsite servers, storage, and also networking hardware which users can rent and access on demand. An IaaS service offers benefits to users such as no maintenance, no up-front capital costs, 24/7 accessibility to applications and data, and elastic infrastructure that allows the user to scale up and down on demand [60]. IaaS is often referred to as "elastic computing" as a consequence of this ability to scale up and down on demand. As seen in Figure 5, users maintain significant management capability when it comes to this service model. The user can create their own VM, with a specific operating system and applications. This can be custom built or chosen from an IaaS catalogue.

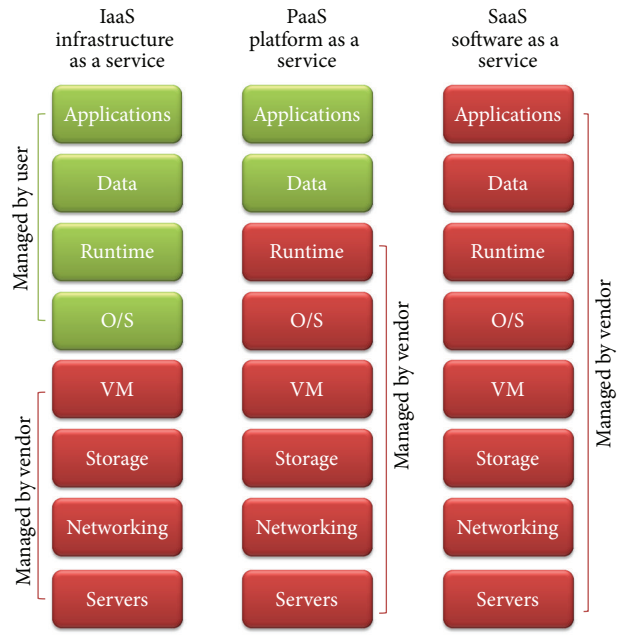

Figure 5: Cloud computing service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). This figure highlights the functionality provided by the vendor to the user which largely depends on the user's requirements.

Some of the leading companies in the IaaS space include Amazon Web Services (AWS) [61], Microsoft [62], Rackspace [63], and VMware [57]. Amazon Web Services (AWS) is the leading IaaS provider, widely recognized for providing the most reliable, scalable, cost-efficient, and user friendly web infrastructure. In 2011, alone the company has earned $6 billion based on providing public cloud services. Examples of the services offered by AWS are Amazon Elastic Compute Cloud (EC2), Amazon Simple Storage Service (S3), Amazon SimpleDB, Amazon Simple Queue Service (SQS), Amazon Simple Notification Service (SNS), Amazon CloudFront, and Amazon Elastic MapReduce (EMR).

Bioinformaticians use IaaS for building databases, storing data, and in developing a pipeline for comparative analysis on various genes and/or proteins. They can now also access databases through AWS, who have started to provide bioinformatics data sets in their publically hosted datasets such as Ensembl [64] and Genbank [65].

4.2. Software as a Service (SaaS). SaaS refers to cloud based delivery of software applications which are hosted by cloud providers. SaaS eliminates the need to install software locally on computers with the user instead accessing software via the Internet. The vendors own the applications and the users may pay a subscription fee to access them via a VM, where all the applications are installed, without the necessity for the user to have a physical copy of the software installed on their own device. SaaS providers run and maintain all necessary hardware and software. One of the top SaaS providers, Salesforce.com, offers on-demand customer relationship management (CRM) software solutions built on its own infrastructure. Salesforce.com does not sell licence for this software, instead it charges a monthly subscription fee starting from $65 per user per month and delivers this software directly to users via Internet [66]. The Economist estimates that the market for SaaS is growing at 50% each year [67]. SaaS vendors have a complete management control of applications, O/S, Runtime, VM, and Servers as shown in Figure 5; however, as a user is simply using the software and does not require control over the OS, and this model is not restrictive and meets the user's requirements.

An example of SaaS used in bioinformatics is Cloud BioLinux, which was developed at the J. Craig Venter Institute. Cloud BioLinux is a publicly accessible Virtual Machine (VM) which offers an on-demand, cloud computing solutions for the bioinformatics field. Users have access to a range of preconfigured command line and graphical software applications, documentation, and more than 135 bioinformatics tools for applications such as sequence alignments, clustering, tree construction, editing, and phylogeny. Cloud BioLinux is stored on Amazon EC2 and is freely available to its users [49].

4.3. Platform as a Service (PaaS). PaaS providers allow subscribed users to access the components that are required for the user to develop or operate applications. These PaaS systems are web-based application development platforms, providing either end-to-end or partial environments for implementing full programs/algorithms online. This includes tasks such as editing code, debugging, deployment, and runtime. As seen in Figure 5, users can build the application with the vendor's on-demand tools and collaborative development environment. The benefits of PaaS include the elimination of complex evaluation, configuration, and management and cost reduction in buying, updating, and maintaining of all hardware and software needed for custom built applications [68]. PaaS also saves time and resources, that is, no need to reinvent the wheel; developers simply build more complex systems using existing platforms. Some of the biggest PaaS platforms today are Google App Engine, Microsoft Azure, and MapReduce/Hadoop. Google App Engine was first released in 2008 and is used for developing and hosting web applications. First created by Google in order to process vast amounts of data, MapReduce is a programming model and an implementation for storing and processing large data sets. Using the MapReduce paradigm, the user specifies a map function which analyses data with the reduce function merging all the results associated with the values from the map phase [69].

One such PaaS technology, Hadoop and Map/Reduce, driven by big data, distributes the data over commodity hardware and provides parallelised processing and analytics. MapReduce developed by Google is a general purpose, relatively easy-to-use parallel programming model that is perfect for carrying out analysis of large data sets on commodity hardware clusters. Additionally, Apache Hadoop is a software framework that implements the distributed processing of big data sets across cluster farms based on the MapReduce model. With the combination of MapReduce and Hadoop Distributed File System (HDFS), Hadoop intends to enable

Table 3: Related software and projects on MapReduce.

Projects	Description
Avro	A data serialization system. URL: http://avro.apache.org/
Cassandra	A highly scalable, consistent, distributed, and structured multimaster database. URL: http://cassandra.apache.org/
Chukwa	An open source data collection system for monitoring large distributed systems. URL: http://incubator.apache.org/chukwa/
Dryad	An infrastructure which allows the use of resources of a computer cluster for running data-parallel programs. URL: http://research.microsoft.com/en-us/projects/dryad/
Hadoop Common	The common utilities that support the other Hadoop subprojects. URL: http://hadoop.apache.org/
Hadoop MapReduce	A programming model and an associated implementation for processing and generating large data sets. URL: http://research.google.com/archive/mapreduce.html
HaLoop	A modified version of the Hadoop MapReduce framework, which supports iterative applications by making the task scheduler loop-aware and by adding various catching mechanisms. URL: https://code.google.com/p/haloop/
HBase	A Hadoop database, a distributed, scalable big data store. URL: http://hbase.apache.org/
HDFS	Hadoop Distributed File System is a distributed file system designed to run on commodity hardware. URL: http://hadoop.apache.org/docs/r1.0.4/hdfs_design.html
Hive	A data warehouse system for Hadoop that facilitates data summarization and ad hoc queries. URL: http://hive.apache.org/
Mahout	A scalable machine learning and data mining library. URL: http://mahout.apache.org/
MapR	A complete distribution for Apache Hadoop and HBase that includes Hive, Mahout, Pig, Cascading, and many other projects. URL: http://www.mapr.com/
Pig	A platform for analysing large data sets that consists of high-level language for expressing data analysis programs. URL: http://pig.apache.org/
Pregel	A system for large-scale graph processing. URL: http://kowshik.github.com/JPregel/pregel_paper.pdf
Twister	A support for iterative MapReduce computations. URL: http://www.iterativemapreduce.org/
YARN	Next Generation Apache Hadoop MapReduce Framework. URL: http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html
ZooKeeper	A high-performance manager for distributed applications. URL: http://hadoop.apache.org/zookeeper/

reliable, scalable, and distributed computing. Hadoop [94] was initiated by Doug Cutting, who worked on the Apache Nutch project (Hadoop is named after his son's toy, a stuffed yellow elephant). It has been designed to scale out from as little as one server to thousands of machines each offering local computation and storage. One of the biggest advantages of Hadoop is speed, being able to process data stored in billions of records, overnight, a process which would have taken several weeks to process [95]. Apache offers other projects similar to Hadoop, these include HBase, HaLoop, Chukwa, Cassandra, Hive, Mahout, Pig, ZooKeeper, and others as outlined in Table 3.

Such technology provides a scalable and cost-efficient solution to the big data challenge. Recent years have shown a massive increase in the size of biological data sets and the growth of new, highly flexible on-demand computing technologies. The data from genomic, proteomic, and metagenomic sequencing projects are increasing at exponential rates, providing information for widening the overall insight of genomes and proteins; however, it is also introducing new challenges such as need for increased storage space, higher power computation, and large data analysis. Cloud computing resources have the potential to aid in solving these problems, by offering a utility model of computing and storage, such as almost unlimited storage capacity, anytime usage, and cheap flexible payment models. Effective use of cloud computing on large biological datasets requires dealing with nontrivial problems of scale and robustness, since performance-limiting factors can change substantially when a dataset grows. New computing paradigms are thus often needed. The use of cloud platforms also creates new opportunities to make data widely available and share it amongst different research laboratories by uploading data to the cloud. Cloud Web services such as Amazon Elastic Compute Cloud (EC2) and Amazon Elastic MapReduce are commercially available, but there are also clouds that provide free service; IBM/Google Cloud Computing University Initiative and the United States Department of Energy's Magellan provide free services, so the users can upload their data by using a web interface, and then they can perform all of their operations on a remote client webpage.

A case in point is Amazon Web Services (AWS) which provides a centralized repository of public data sets, including archives of GenBank, Ensembl, 1000 Genomes, Model Organism Encyclopedia of DNA Elements, Unigene, and Influenza Virus. As a matter of fact, AWS contains multiple

Table 4: Parallel bioinformatics software tools.

Function	Algorithm	Description
Genomic sequence mapping	CloudAligner [70]	A MapReduce-based application for mapping short reads generated by the next-generation sequencing machines. URL: http://cloudaligner.sourceforge.net/
	CloudBurst [71]	A parallel read mapping algorithm used for mapping next-generation sequence data to the human genome and other genomes. URL: http://sourceforge.net/apps/mediawiki/cloudburst-bio/index.php?title=CloudBurst
	SEAL [72]	A suite of distributed applications for aligning, manipulating, and analysing short DNA reads. URL: http://biodoop-seal.sourceforge.net/
	BlastReduce [73]	A parallel short DNA sequence read mapping algorithm optimized for aligning sequence data for use in SNP discovery, genotyping, and personal genomics. URL: http://www.cbcb.umd.edu/software/blastreduce/
Genomic sequencing analysis	Crossbow [74]	A scalable software pipeline, which combines Bowtie and SoapsSNP for whole genome resequencing analysis. URL: http://bowtie-bio.sourceforge.net/crossbow/index.shtml
	Contrail [75]	An algorithm for de novo assembly of large genomes from short sequencing reads. Contrail relies on the graph-theoretic framework of de Bruijn graphs. URL: http://sourceforge.net/apps/mediawiki/contrail-bio/index.php?title=Contrail
	CloudBrush [76]	A distributed genome assembler based on string graphs. URL: https://github.com/ice91/CloudBrush.
RNA sequencing analysis	Myrna [77]	A cloud computing pipeline for calculating differential gene expression in large RNA-Seq data sets. URL: http://bowtie-bio.sourceforge.net/myrna/index.shtml
	FX [78]	RNA sequence analysis tool for the estimation of gene expression levels and genomic variant calling. URL: http://fx.gmi.ac.kr/
	Eoulsan [79]	An integrated and flexible solution for RNA sequence data analysis of differential expression. URL: http://transcriptome.ens.fr/eoulsan/
Sequence file management	Hadoop-BAM [80]	A novel library for scalable manipulation of aligned next-generation sequencing data in the Hadoop distributed computing framework. URL: http://sourceforge.net/projects/hadoop-bam/
	SeqWare [81]	A tool set used to work with next generation genome sequencing technologies (Illumina, ABI SOLiD, 454) which includes a LIMS, Pipeline, and Query Engine. URL: http://sourceforge.net/projects/seqware/
	GATK [82]	A MapReduce framework for analysing next-generation DNA sequencing data. URL: http://genome.cshlp.org/content/20/9/1297
Phylogenetic analysis	MrsRF [83]	A scalable, efficient multicore algorithm that uses MapReduce to quickly calculate the all-to-all Robinson-Foulds (RF) distance between large numbers of trees. URL: https://code.google.com/p/mrsrf/
	Nephele [84]	A set of tools, which use the complete composition vector algorithm in order to group sequence clustering into genotypes based on a distance measure. URL: https://code.google.com/p/nephele/
GPU bioinformatics software	GPU-BLAST [85]	An accelerated version of NCBI-BLAST which uses general purpose graphics processing unit (GPU), designed to rapidly manipulate and alter memory to accelerate overall algorithm processing. URL: http://eudoxus.cheme.cmu.edu/gpublast/gpublast.html
	SOAP3 [86]	Short sequence read alignment algorithm that uses the multiprocessors in a graphic processing unit to achieve ultrafast alignments. URL: http://soap.genomics.org.cn/soap3.html
Search engine implementation	Hydra [87]	A protein sequence database search engine specifically designed to run efficiently on the Hadoop MapReduce framework. URL: http://www.webcitation.org/query.php?url=http://code.google.com/p/hydra-proteomics/&refdoi=10.1186/1471-2105-13-324
	CloudBlast [88]	Scalable BLAST in the cloud. URL: http://ammatsun.acis.ufl.edu/amwiki/index.php/CloudBLAST_Project
Miscellaneous	PeakRanger [89]	A multipurpose, ultrafast ChIP sequence peak caller. URL: http://ranger.sourceforge.net/
	YunBe [90]	A gene set analysis algorithm for biomarker identification in the cloud. URL: http://lrcv-crp-sante.s3-website-us-east-1.amazonaws.com/
	BioDoop [91]	A set of tools which modules for handling FASTA streams, wrappers for BLAST, converting sequences to the different formats, and so on. URL: http://dc.crs4.it/projects/biodoop
	BlueSNP [92]	An algorithm for computationally intensive analyses, feasible for large genotype-phenotype datasets. URL: https://github.com/ibm-bioinformatics/BlueSNP
	Quake [93]	DNA sequence error detection and correction in sequence reads. URL: http://www.cbcb.umd.edu/software/quake/

public datasets for a variety of scientific fields, such as biology, astronomy, chemistry, climate, and economics [96]. All public datasets in AWS are delivered as services and therefore can be easily integrated into cloud-based applications. Also, using cloud platforms would reduce duplication and provide easy reproducibility by making the sequence datasets and computational methods easily available [97]. Big data technology algorithms are increasing on monthly bases, facilitating different functional sequence analysis, as outlined in Table 4.

4.4. Cloud-Based MSA-Next-Generation MSA. The multiple sequence alignment algorithms certainly need to be improved in order to be able to handle large amounts of DNA/RNA/protein sequences and most importantly produce multiple sequence alignments of high quality. At the moment, the only algorithm able to handle as much as 190,000 sequences is Clustal Omega, which even at that takes a few hours on a single processor to produce an alignment of all the sequences. In a recent study by Sievers et al., 2013, 18 standard automated multiple sequence alignment packages were compared with the main focus being on how well they scaled, aligning from 100 to 50,000 sequences. It was found that most of the algorithms, for example, T-Coffee, PSAlign, Prank, FSA, and Mummals, did not produce alignments above 100 sequences. Some of the algorithms produced alignment of max 1,000 sequences; these were Probcons, MUSCLE, MAFFT, ClustalW, and MSAProbs. Finally, the only MSA algorithms that completed alignment of 50,000 sequences were Clustal Omega, Kalign, and Part-Tree. Furthermore, it was noted that the quality of the alignment for each of the algorithms decreased progressively as the number of sequences increased [52]. This could possibly be explained by the nature of progressive alignments, which are heuristic in nature, therefore introducing noise and mistakes at the start of the alignment. The main concerns with scaling up and producing MSA of large sets of sequences are the computational complexity, the time it takes to produce the alignment and the accuracy of the final alignment. Also, at present, there are no systematic benchmarks tests that can handle testing alignments of massively increasing number of sequences; therefore, new benchmarks must be developed due to the fact that new algorithms are created on monthly bases and soon will be able to align massive numbers of sequences.

By employing big data technologies and cloud computing projects, the aim would be to develop an improved version of Clustal Omega which could produce alignments of very large data sets and in a shorter time frame than the original Clustal Omega algorithm. Also, other popular multiple sequence alignments could possibly be recoded, so it could complete MSA algorithm over a cluster of machines in a distributed, parallelised way by using the Hadoop/MapReduce framework. An example of multiple sequence alignment that is optimized in the cloud is the FASTA algorithm published by Vijaykumar et al. in 2012. This study presents an implementation of the FASTA algorithm built on the Hadoop/MapReduce framework and MPP Database. In this experiment it was observed that the time taken for sequence alignment increased as the number of sequences also increased. However, the experiment also showed that as the number of nodes increased the number of alignments that was executed in parallel also increased, resulting in a time decrease for alignment completion [98]. This proves the theory of parallelization and the use of the cloud computing technologies for improving multiple sequence alignment tools. Parallel, distributed multiple sequence alignments in the cloud is likely our only real means of keeping pace with today's sequence tsunami and will ultimately aid in the discovery of novel genes, entire metabolic pathways, novel proteins and potentially medically valuable end-products from the global metabolome [99]. The implication of such scalable worldwide analysis and sharing of data sets is enormous in that it will not only change lives but may also save them.

Conflict of Interests

No potential conflict of interests was disclosed.

Acknowledgment

Jurate Daugelaite is funded under the Embark Initiative by an Irish Research Council (IRC) Grant RS/2012/122. Aisling O' Driscoll and Dr. Roy D. Sleator are Principal Investigators on ClouDx-i an FP7-PEOPLE-2012-IAPP project.

References

[1] C. Kemena and C. Notredame, "Upcoming challenges for multiple sequence alignment methods in the high-throughput era," *Bioinformatics*, vol. 25, no. 19, pp. 2455–2465, 2009.

[2] R. C. Edgar and S. Batzoglou, "Multiple sequence alignment," *Current Opinion in Structural Biology*, vol. 16, no. 3, pp. 368–373, 2006.

[3] C. Notredame, "Recent evolutions of multiple sequence alignment algorithms," *PLoS Computational Biology*, vol. 3, no. 8, article e123, 2007.

[4] Human Genome Project Information, 2013, http://web.ornl.gov/sci/techresources/Human_Genome/home.shtml.

[5] "Home, 1000 Genomes," 2013, http://www.1000genomes.org/.

[6] G. K. C. O. Scientists, "Genome 10K: a proposal to obtain whole-genome sequence for 10, 000 vertebrate species," *Journal of Heredity*, vol. 100, no. 6, pp. 659–674, 2009.

[7] 454 Life Sciences, a Roche Company, 2013, http://www.454.com/.

[8] Illumina, Inc, 2013, https://www.illumina.com/.

[9] SOLiDTM 4 System, 2013, http://www.appliedbiosystems.com/absite/us/en/home/applications-technologies/solid-next-generation-sequencing/next-generation-systems/solid-4-system.html?CID=FL-091411_solid4.

[10] H. Li and N. Homer, "A survey of sequence alignment algorithms for next-generation sequencing," *Briefings in Bioinformatics*, vol. 11, no. 5, pp. 473–483, 2010.

[11] SourceForge.net: jnomics, 2013, http://sourceforge.net/apps/mediawiki/jnomics/index.php?title=Jnomics.

[12] C. B. Do and K. Katoh, "Protein multiple sequence alignment," *Methods in Molecular Biology*, vol. 484, pp. 379–413, 2008.

[13] R. C. Edgar, "MUSCLE: a multiple sequence alignment method with reduced time and space complexity," *BMC Bioinformatics*, vol. 5, article 113, 2004.

[14] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[15] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, 1981.

[16] I. M. Wallace, G. Blackshields, and D. G. Higgins, "Multiple sequence alignments," *Current Opinion in Structural Biology*, vol. 15, no. 3, pp. 261–266, 2005.

[17] K. Katoh and H. Toh, "Recent developments in the MAFFT multiple sequence alignment program," *Briefings in Bioinformatics*, vol. 9, no. 4, pp. 286–298, 2008.

[18] D.-F. Feng and R. F. Doolittle, "Progressive sequence alignment as a prerequisitetto correct phylogenetic trees," *Journal of Molecular Evolution*, vol. 25, no. 4, pp. 351–360, 1987.

[19] W. J. Wilbur and D. J. Lipman, "Rapid similarity searches of nucleic acid and protein data banks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 80, no. 3, pp. 726–730, 1983.

[20] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[21] F. Sievers, A. Wilm, D. Dineen et al., "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega," *Molecular Systems Biology*, vol. 7, article 539, 2011.

[22] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees," *Molecular Biology and Evolution*, vol. 4, no. 4, pp. 406–425, 1987.

[23] I. Gronau and S. Moran, "Optimal implementations of UPGMA and other common clustering algorithms," *Information Processing Letters*, vol. 104, no. 6, pp. 205–210, 2007.

[24] J. D. Thompson, D. G. Higgins, and T. J. Gibson, "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic Acids Research*, vol. 22, no. 22, pp. 4673–4680, 1994.

[25] K. Katoh and D. M. Standley, "MAFFT multiple sequence alignment software version 7: improvements in performance and usability," *Molecular Biology and Evolution*, vol. 30, no. 4, pp. 772–780, 2013.

[26] T. Lassmann and E. L. L. Sonnhammer, "Kalign—an accurate and fast multiple sequence alignment algorithm," *BMC Bioinformatics*, vol. 6, article 298, 2005.

[27] U. Roshan and D. R. Livesay, "Probalign: multiple sequence alignment using partition function posterior probabilities," *Bioinformatics*, vol. 22, no. 22, pp. 2715–2721, 2006.

[28] B. Morgenstern, "DIALIGN: multiple DNA and protein sequence alignment at BiBiServ," *Nucleic Acids Research*, vol. 32, supplement 2, pp. W33–W36, 2004.

[29] A. Löytynoja and N. Goldman, "Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis," *Science*, vol. 320, no. 5883, pp. 1632–1635, 2008.

[30] R. K. Bradley, A. Roberts, M. Smoot et al., "Fast statistical alignment," *PLoS Computational Biology*, vol. 5, no. 5, Article ID e1000392, 2009.

[31] P. Di Tommaso, S. Moretti, I. Xenarios et al., "T-Coffee: a web server for the multiple sequence alignment of protein and RNA sequences using structural information and homology extension," *Nucleic Acids Research*, vol. 39, supplement 2, pp. W13–W17, 2011.

[32] C. Notredame, D. G. Higgins, and J. Heringa, "T-coffee: a novel method for fast and accurate multiple sequence alignment," *Journal of Molecular Biology*, vol. 302, no. 1, pp. 205–217, 2000.

[33] C. B. Do, M. S. P. Mahabhashyam, M. Brudno, and S. Batzoglou, "ProbCons: probabilistic consistency-based multiple sequence alignment," *Genome Research*, vol. 15, no. 2, pp. 330–340, 2005.

[34] Y. Liu, B. Schmidt, and D. L. Maskell, "MSAProbs: multiple sequence alignment based on pair hidden Markov models and partition function posterior probabilities," *Bioinformatics*, vol. 26, no. 16, pp. 1958–1964, 2010.

[35] D. W. Mount, "Using iterative methods for global multiple sequence alignment," *Cold Spring Harbor Protocols*, vol. 4, no. 7, Article ID pdb.top44, 2009.

[36] O. Gotoh, "Optimal alignment between groups of sequences and its application to multiple sequence alignment," *Computer Applications in the Biosciences*, vol. 9, no. 3, pp. 361–370, 1993.

[37] C. Notredame and D. G. Higgins, "SAGA: sequence alignment by genetic algorithm," *Nucleic Acids Research*, vol. 24, no. 8, pp. 1515–1524, 1996.

[38] J. D. Thompson, F. Plewniak, and O. Poch, "A comprehensive comparison of multiple sequence alignment programs," *Nucleic Acids Research*, vol. 27, no. 13, pp. 2682–2690, 1999.

[39] A. M. Lesk and C. Chothia, "How different amino acid sequences determine similar protein structures: the structure and evolutionary dynamics of the globins," *Journal of Molecular Biology*, vol. 136, no. 3, pp. 225–270, 1980.

[40] O. O'Sullivan, K. Suhre, C. Abergel, D. G. Higgins, and C. Notredame, "3DCoffee: combining protein sequences and structures within multiple sequence alignments," *Journal of Molecular Biology*, vol. 340, no. 2, pp. 385–395, 2004.

[41] F. Armougom, S. Moretti, O. Poirot et al., "Expresso: automatic incorporation of structural information in multiple sequence alignments using 3D-Coffee," *Nucleic Acids Research*, vol. 34, supplement 2, pp. W604–W608, 2006.

[42] X. Xia, S. Zhang, Y. Su, and Z. Sun, "MICAlign: a sequence-to-structure alignment tool integrating multiple sources of information in conditional random fields," *Bioinformatics*, vol. 25, no. 11, pp. 1433–1434, 2009.

[43] Z. Zhang, A. A. Schäffer, W. Miller et al., "Protein sequence similarity searches using patterns as seeds," *Nucleic Acids Research*, vol. 26, no. 17, pp. 3986–3990, 1998.

[44] M. C. Frith, N. F. W. Saunders, B. Kobe, and T. L. Bailey, "Discovering sequence motifs with arbitrary insertions and deletions," *PLoS Computational Biology*, vol. 4, no. 4, Article ID e1000071, 2008.

[45] H. Li, J. Ruan, and R. Durbin, "Mapping short DNA sequencing reads and calling variants using mapping quality scores," *Genome Research*, vol. 18, no. 11, pp. 1851–1858, 2008.

[46] R. Li, Y. Li, K. Kristiansen, and J. Wang, "SOAP: short oligonucleotide alignment program," *Bioinformatics*, vol. 24, no. 5, pp. 713–714, 2008.

[47] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg, "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome," *Genome Biology*, vol. 10, no. 3, article R25, 2009.

[48] "A Definition of The Cloud at Last?—Web Performance Watch," 2013, http://blogs.keynote.com/the_watch/2010/05/for-as-long-as-ive-been-involved-in-writing-about-the-cloud-and-its-related-applications-ive-seen-the-basic-question.html.

[49] "Virtualization is a key enabler of cloud computing," 2013, http://www.itnewsafrica.com/2010/03/virtualization-is-a-key-enabler-of-cloud-computing/.

[50] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics, 2007.

[51] J. Söding, "Protein homology detection by HMM-HMM comparison," *Bioinformatics*, vol. 21, no. 7, pp. 951–960, 2005.

[52] F. Sievers, D. Dineen, A. Wilm, and D. G. Higgins, "Making automated multiple alignments of very large numbers of protein sequences," *Bioinformatics*, vol. 29, no. 8, pp. 989–995, 2013.

[53] K. Katoh, K. Misawa, K.-I. Kuma, and T. Miyata, "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform," *Nucleic Acids Research*, vol. 30, no. 14, pp. 3059–3066, 2002.

[54] "What cloud computing really means," 2013, http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031.

[55] "MIT credits Irish-based entrepreneur with co-coining term "cloud computing"," 2013, http://www.siliconrepublic.com/cloud/item/24280-mit-credits-irish-based-ent.

[56] "The Benefits Of Data Center Virtualization For Businesses, CloudTweaks," 2013, http://www.cloudtweaks.com/2012/03/the-benefits-of-data-center-virtualization-for-businesses/.

[57] "VMware Virtualization Software for Desktops, Servers & Virtual Machines for Public and Private Cloud Solutions," 2013, http://www.vmware.com/.

[58] Main Page—KVM, 2013, http://www.linux-kvm.org/page/Main_Page.

[59] "Accelerating high-performance computing applications using parallel computing," 2013, http://embedded-computing.com/articles/accelerating-using-parallel-computing/.

[60] "Cloud 101: What the heck do IaaS, PaaS and SaaS companies do?" 2013, http://venturebeat.com/2011/11/14/cloud-iaas-paas-saas/.

[61] Amazon Web Services, "Cloud Computing: Compute, Storage, Database," 2013, http://aws.amazon.com/.

[62] Microsoft Home Page, Devices and Services, 2013, http://www.microsoft.com/en-us/default.aspx.

[63] Rackspace, 2013, http://www.rackspace.com/.

[64] Ensembl Genome Browser, 2013, http://www.ensembl.org/index.html.

[65] GenBank Home, 2013, http://www.ncbi.nlm.nih.gov/genbank.

[66] CRM—The Enterprise Cloud Computing Company—Salesforce.com Europe, 2013, http://www.salesforce.com/eu/.

[67] V. Choudhary, "Software as a service: implications for investment in software development," in *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS '07)*, January 2007.

[68] G. Lawton, "Developing software online with platform-as-a-service technology," *Computer*, vol. 41, no. 6, pp. 13–15, 2008.

[69] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[70] T. Nguyen, W. Shi, and D. Ruden, "CloudAligner: a fast and full-featured MapReduce based tool for sequence mapping," *BMC Research Notes*, vol. 4, article 171, 2011.

[71] M. C. Schatz, "CloudBurst: highly sensitive read mapping with MapReduce," *Bioinformatics*, vol. 25, no. 11, pp. 1363–1369, 2009.

[72] L. Pireddu, S. Leo, and G. Zanetti, "Seal: a distributed short read mapping and duplicate removal tool," *Bioinformatics*, vol. 27, no. 15, pp. 2159–2160, 2011.

[73] "Blastreduce: high performance short read mapping with mapreduce," 2013, http://www.cbcb.umd.edu/software/blastreduce.

[74] B. Langmead, M. C. Schatz, J. Lin, M. Pop, and S. L. Salzberg, "Searching for SNPs with cloud computing," *Genome Biology*, vol. 10, no. 11, article R134, 2009.

[75] M. C. Schatz, D. Sommer, D. Kelley, and M. Pop, "De novo assembly of large genomes using cloud computing," in *Proceedings of the CSHL Biology of Genomes Conference*, 2010.

[76] Y.-J. Chang, C. C. Chen, C. L. Chen, and J. M. Ho, "A de novo next generation genomic sequence assembler based on string graph and mapreduce cloud computing framework," *BMC Genomics*, vol. 13, supplement 7, p. S28, 2012.

[77] B. Langmead, K. D. Hansen, and J. T. Leek, "Cloud-scale RNA-sequencing differential expression analysis with Myrna," *Genome Biology*, vol. 11, no. 8, p. R83, 2010.

[78] D. Hong, A. Rhie, S.-S. Park et al., "FX: an RNA-seq analysis tool on the cloud," *Bioinformatics*, vol. 28, no. 5, pp. 721–723, 2012.

[79] L. Jourdren, M. Bernard, M. A. Dillies, and S. Le Crom, "Eoulsan: a cloud computing-based framework facilitating high throughput sequencing analyses," *Bioinformatics*, vol. 28, no. 11, pp. 1542–1543, 2012.

[80] M. Niemenmaa, A. Kallio, A. Schumacher, P. Klemelä, E. Korpelainen, and K. Heljanko, "Hadoop-BAM: directly manipulating next generation sequencing data in the cloud," *Bioinformatics*, vol. 28, no. 6, pp. 876–877, 2012.

[81] B. D. O'Connor, B. Merriman, and S. F. Nelson, "SeqWare Query Engine: storing and searching sequence data in the cloud," *BMC Bioinformatics*, vol. 11, supplement 12, article S2, 2010.

[82] A. McKenna, M. Hanna, E. Banks et al., "The genome analysis toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data," *Genome Research*, vol. 20, no. 9, pp. 1297–1303, 2010.

[83] S. J. Matthews and T. L. Williams, "MrsRF: an efficient MapReduce algorithm for analyzing large collections of evolutionary trees," *BMC Bioinformatics*, vol. 11, supplement 1, article S15, 2010.

[84] M. E. Colosimo, M. W. Peterson, S. Mardis, and L. Hirschman, "Nephele: genotyping via complete composition vectors and MapReduce," *Source Code for Biology and Medicine*, vol. 6, article 13, 2011.

[85] P. D. Vouzis and N. V. Sahinidis, "GPU-BLAST: using graphics processors to accelerate protein sequence alignment," *Bioinformatics*, vol. 27, no. 2, pp. 182–188, 2011.

[86] C.-M. Liu, T. Wong, E. Wu et al., "SOAP3: ultra-fast GPU-based parallel alignment tool for short reads," *Bioinformatics*, vol. 28, no. 6, pp. 878–879, 2012.

[87] S. Lewis, A. Csordas, S. Killcoyne et al., "Hydra: a scalable proteomic search engine which utilizes the Hadoop distributed computing framework," *BMC Bioinformatics*, vol. 13, article 324, 2012.

[88] A. Matsunaga, M. Tsugawa, and J. Fortes, "CloudBLAST: combining MapReduce and virtualization on distributed resources

for bioinformatics applications," in *Proceedings of the 4th IEEE International Conference on eScience (eScience '08)*, pp. 222–229, December 2008.

[89] X. Feng, R. Grossman, and L. Stein, "PeakRanger: a cloud-enabled peak caller for ChIP-seq data," *BMC Bioinformatics*, vol. 12, article 139, 2011.

[90] L. Zhang, S. Gu, Y. Liu, B. Wang, and F. Azuaje, "Gene set analysis in the cloud," *Bioinformatics*, vol. 28, no. 2, pp. 294–295, 2012.

[91] S. Leo, F. Santoni, and G. Zanetti, "Biodoop: bioinformatics on hadoop," in *Proceedings of the 38th International Conference Parallel Processing Workshops (ICPPW '09)*, pp. 415–422, September 2009.

[92] H. Huang, S. Tata, and R. J. Prill, "BlueSNP: R package for highly scalable genome-wide association studies using Hadoop clusters," *Bioinformatics*, vol. 29, no. 1, pp. 135–136, 2013.

[93] D. R. Kelley, M. C. Schatz, and S. L. Salzberg, "Quake: quality-aware detection and correction of sequencing errors," *Genome Biology*, vol. 11, no. 11, article R116, 2010.

[94] Apache Hadoop, 2013, http://hadoop.apache.org/.

[95] How Hadoop Makes Short Work Of Big Data—Forbes, 2013, http://www.forbes.com/sites/netapp/2012/09/24/hadoop-big-data/.

[96] Public Data Sets on Amazon Web Services (AWS), 2013, http://aws.amazon.com/publicdatasets/.

[97] P. M. Kasson, "Computational biology in the cloud: methods and new insights from computing at scale," in *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, World Scientific, 2013.

[98] S. Vijayakumar, A. Bhargavi, U. Praseeda, and S. A. Ahamed, "Optimizing sequence alignment in cloud using hadoop and MPP database," in *Proceedings of the 5th IEEE International Conference on Cloud Computing (CLOUD '12)*, 2012.

[99] R. D. Sleator, "Proteins: form and function," *Bioeng Bugs*, vol. 3, no. 2, pp. 80–85, 2012.