A Cloud Framework for High Throughput Biological Data Processing

Martin Koehler[*a], **Yuriy Kaniovskyi**[a], **Siegfried Benkner**[a], **Volker Egelhofer**[b], **Wolfram Weckwerth**[b]
[a]*University of Vienna, Research Group Scientific Computing, Austria*
[b]*University of Vienna, Department Molecular Systems Biology, Austria*
E-mail: koehler,yk,sigi@par.univie.ac.at,
volker.egelhofer,wolfram.weckwerth@univie.ac.at

The molecular systems biology community has to deal with an increasingly growing amount of data. A recent programming model that addresses the data deluge is MapReduce which facilitates processing of huge data volumes on large sets of computing resources. However, the availability of appropriate local computing resources is often limited. Cloud computing addresses this issue by providing virtually infinite resources on demand, usually following a pay per use model. In this paper we present our cloud based high throughput computing infrastructure which combines the Software as a Service approach with the MapReduce programming model for data-intensive applications, and a configurable distributed file system as provided by the Hadoop framework. Within this infrastructure we realized an application in the field of molecular systems biology which matches tryptic peptide fragmentation mass spectra data against a large scale mass spectral reference database. We evaluate this application on a local cloud resource and study the effects of different configuration parameters as provided by the application, the Hadoop framework, and the available computational and storage resources.

The International Symposium on Grids and Clouds and the Open Grid Forum - ISGC2011,
March 25-30, 2011
Academia Sinica, Taipei, Taiwan

*Speaker.

1. Introduction

Scientific applications, for example in systems biology, have to deal with an ever increasing amount of data. Hence, they require large-scale storage and computing resources which often are not available locally. As a consequence, users from diverse scientific domains seek for technologies that simplify the provisioning of computing resources for the execution of their applications. A relatively new technology in this area is Cloud computing. The concept of Cloud computing promises on demand access to virtually infinite computing and storage resources. Different Cloud providers have built huge data-centers, offering their computing and storage power to users based on a pay-per-use model. A field of increasing interest is the data deluge problem. The volume of data available in many domains increases exponentially in time. Gaining knowledge from such ever growing data sets calls for new data-intensive programming paradigms, which are scalable on distributed cloud resources, while exploiting data locality.

In the last years MapReduce [1] has gained a lot of interest in the scientific community. The MapReduce pattern is typically used for analyzing large data volumes stored over distributed IT infrastructures. MapReduce follows a massively parallel execution model, where computation takes place directly at the data sites. There are multiple implementations of the MapReduce framework available. The most common is the Hadoop framework [2] by the Apache Software Foundation.

Software as a Service (SaaS) represents an approach for facilitating access to large scale applications over the Internet. SaaS provides access to applications installed on remote systems via Web service technology. Cloud Computing [3] allows the deployment of services on computing resources that are provisioned on demand. It also enables on-demand selection of resources by taking into account the actual resource needs. As a consequence, the amount of required resources can be adapted dynamically, depending on the needs of users.

The Vienna Cloud Environment (VCE)[1] [13] follows the SaaS approach to expose virtual appliances as Web services. VCE virtual appliances are software packages preinstalled on virtual machine images that enable the provisioning of services in the Cloud. Virtual appliances hide the details of the underlying software and hardware infrastructure and provide a common set of generic interfaces to the user. Utilizing the VCE facilitates the provisioning of high performance computing applications, data resources, workflows, as well as high throughput biological data processing applications as virtual appliances, all presenting a Web service interface.

In this work we implement VCE MapReduce services for the provisioning of a high throughput biological data processing application as virtual appliance in the Cloud. The application matches tryptic peptide fragmentation mass spectra data against a large scale mass spectral reference database (ProMEX) and was recast according to the MapReduce pattern. Modifying and optimizing an application for MapReduce results in new configuration challenges at three layers: first, the application, second, the execution framework, and third, the available resources. The paper focuses on demonstrating the migration of an application into the Cloud as well as the effects of configuration parameters at all involved layers on the application's runtime.

The remainder of the paper is structured as follows. The next section gives an overview of our Cloud based high throughput infrastructure, the VCE. Then we explain the application matching

[1]VCE - Vienna Cloud Environment: http://www.par.univie.ac.at/project/vce/

tryptic peptide fragmentation mass spectra data against the ProMEX database in detail. Section 4 presents the recast process of the application to the MapReduce pattern and the provisioning of the application as a VCE service. A performance evaluation showing the effects of different parameters at the application, execution environment, and resource layer is presented in Section 5. The paper concludes with a review of related work, a summary and some open questions for a future study in Section 7.

2. Cloud Based High Throughput Infrastructure

Infrastructures for high throughput applications have to meet special requirements to enable fast execution for large data volumes. These applications require sizable storage resources as well as plenty computational power to process the data. To provide high throughput, fast network connections between the involved resources are needed. Computing and storage resources are often not available locally, but can be provided on demand via Cloud environments. This approach allows, on one hand, more efficient resource allocation by sharing them with other users, but, on the other hand it potentially reduces the knowledge about the involved resources. The SaaS concept enables domain scientists to access applications via the Internet.

2.1 Vienna Cloud Environment (VCE)

The development of the VCE has been based on the Vienna Grid Environment [4] (VGE), a service-oriented infrastructure for facilitating transparent access to parallel applications on remote high performance and high throughput computing systems. Compute-intensive applications, often highly optimized parallel MPI or OpenMP codes, installed on clusters or other HPC systems, can be made available over the Internet to clients as application services. Application services include support for dynamic negotiation of service level agreements based on a flexible Quality of Service infrastructure employing a business model specialized for the application. A generic Web service interface for managing the job execution on remote resources hides the details of the utilized execution environment. The uniform interface offers common operations for uploading input data, starting remote job execution, querying the state of the execution, downloading the results, and support for transferring input or output files directly between application services or Web storage resources. In addition to application services, the VCE provides data services to facilitate access to and integration of heterogeneous data sources. On top of data and application services, the VCE supports workflow services based upon the WEEP Workflow Engine [6]. All VCE services follow the same Web service interface and can utilize different Web service components, such as security or monitoring.

Recently, we Cloud-enabled the service environment by introducing the concept of virtual appliances for data and workflow services. Virtual data and mediation nodes [7] are Cloud ready virtual appliances of data services combined with a data source (e.g. relational database, structured data files, or virtual data source in the case of data mediation service). Composable workflow appliances for the provisioning of scientific workflows in the Cloud and associated adaptive load balancing mechanisms have been presented in [8].

In this work, we enhanced the VCE with Hadoop Services, providing support for data-intensive MapReduce applications based upon the Hadoop Framework [2]. Hadoop services enable the virtualization of MapReduce applications combined with a cloud execution framework. Our execution framework allows on-demand selection and configuration of resources and manages the execution of the application in the Cloud.

2.2 MapReduce Execution Framework

MapReduce is a programming model designed for processing of large volumes of data in parallel by dividing the workload into a set of independent tasks. MapReduce programs are written in a particular style that is influenced by functional programming constructs, specifically idioms for processing lists of data. Data are stored on a distributed file system and split into blocks located on each operational node. The MapReduce application is then transmitted to each node, where it is executed [1]. A MapReduce application entails two main phases: the Map-phase extracts the features of large data sets and the Reduce-phase assembles collected features for the final output.

The Hadoop framework [2] is an open-source implementation of the MapReduce model developed by the Apache Software Foundation. Hadoop offers two core software modules for each node. One is responsible for data storage (Hadoop Distributed File System - HDFS). The other module takes care of job execution tasks and consists of the JobTracker on the frontend and TaskTracker on the execution nodes. Due to the variety of features available, Hadoop turns out to be highly flexible. However, making an efficient configuration remains a form of art [5].

3. Matching Tryptic Peptide Fragmentation Mass Spectra Data

The department of molecular systems biology (MoSys) at the University of Vienna is engaged in the study of the proteome (i.e. the complete set of proteins synthesized by a particular cell at a particular time). The most popular technique today for analyzing the proteome is by mass spectrometry. It relies on separating charged ions by their mass-to-charge ratio (m/z). The set of observed peptide m/z ratios is then used in two ways. First, to look for a protein database containing a set of peptide masses predicted from enzymatic digestion of each protein in the database. Second, to search a database containing already identified spectra.

There are various types of identification algorithms. They all attempt to match a spectrum (or spectrum-derived information) to sequence databases at some point during their processing. All these algorithms generate in silico peptide sequences from the available protein sequences and then construct theoretical fragmentation spectra from these sequences. These in silico spectra are then compared with the experimental fragmentation spectrum and a score is assigned to the match. However, these algorithms apply different scoring functions. Some of them report thresholds, others do not. Alternatively, one can search with an algorithm instructed to match against a mass spectral library consisting of experimentally observed and validated spectra. For example, the one implemented in ProMEX (see below). We expect this alternative provide clear advantages over genome-based prediction of mass spectra [9, 10].

3.1 ProMEX Database

MoSys is developing genome-wide quantitative molecular approaches for genome sequencing. Over the last years a large database storing public mass spectral references, the ProMEX database [11], has been created. The database consists of tryptic peptide fragmentation mass spectra derived from different plants. Currently the database contains 31,780 tryptic peptide product ion spectra entries of 27,543 different peptide sequence entries. The database is utilized to match new spectra to all entries and to compute the correlation between new spectra and already known ones. It contains several GBs of data and is growing continuously. New spectra are added into the database on a regular basis. The matching of new data to the database follows an algorithm which compares each entry of two spectra and computes the correlation of both.

BEGIN IONS	
TITLE=BSA.1003.1003.2.dta	
CHARGE=2+	
PEPMASS=409.716673	
118.121 2.5	978.85236 3
118.914 1.1	85.051 1.8
121.289 1.1	92.871 2.3
123.255 2.4	93.067 4.3
...	93.233 11.5
END IONS	...

Table 1: Samples of a MGF file (left) and a DTA file (right)

There are two basic data files in a predefined format. The .MGF file belongs to the database, which is a reference for the input .DTA files. Samples of both files are depicted in Table 1. The CHARGE and PEPMASS values denote the comparison offset parameters for the rest of the data. The first column of data represents the m/z ratio and the second column the intensity (I). DTA files on the other hand, are not structured in the same way. They do not have labeled CHARGE and PEPMASS values. These values are located in the first row of the DTA file. Each DTA file represents one sample. Each of these samples is to be compared via the matching algorithm with the MGF database.

3.2 Matching Algorithm

MoSys is currently working on a new fast matching algorithm, which will be integrated in the near future. In a first testing and implementation phase, a simplified "In-Range" algorithm is utilized. There, each database sample value has a +/- range of the percentage value. The magnitude of this variable is selected by the user. If a new observation belongs to the range, then it receives one hit point and the algorithm checks the next sample value. The more hit points a new observation receives, the more similar it is to a database sample. Each entry of the input DTA is compared to each entry in the MGF. This leads to $O(n^2)$ complexity for one input DTA file. Here n stands for the number of lines in the DTA file.

While letting this algorithm run on one input-DTA file may seem bearable, the sequential algorithm will not suffice if there are multiple input files to check, which is usually the case. The problem however, is easy to parallelize. In fact, every input file can be checked independently from the others. Furthermore, no synchronization is needed, since access will be read-only on both the input-DTA and database-MGF.

4. Cloud Enabling Molecular Systems Biology Application

Due to the embarrassingly parallel nature of the problem as outlined before, we decided to recast the application following the MapReduce pattern. Using Hadoop also implies that data have

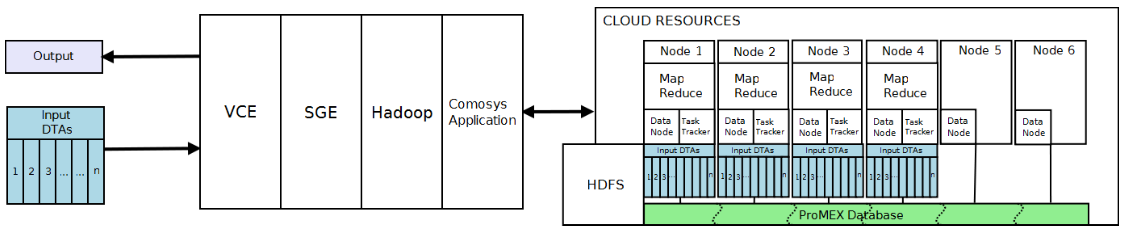

Figure 1: Service architecture: The VCE service-application which is executed on Cloud resources via SGE and the Hadoop framework.

to be available via the HDFS file system. The ProMEX database and the input DTA files have to be available in HDFS prior to job processing. To efficiently access the data, several issues concerning data locality as well as optimization of other configuration parameters have to be taken into account (see Section 4.2 and Section 5). Furthermore, in a commercial Cloud resources are not given free of charge and in a private Cloud they may be sparse as well. It is essential to evaluate the set of available resources and to consume them efficiently. For this reason, an adaptive execution mechanism has been implemented, for optimizing the allocation of resources depending on the characteristics of the application and input data. A detailed description of the adaptive mechanism can be found in [12].

4.1 Cloud Resources

The VCE includes a provisioning framework for virtual appliances on Cloud resources. In the current implementation we support cluster resources and KVM-based virtual appliances. The execution of applications is supported by utilizing the Sun Grid Engine (SGE). The architecture is shown in Figure 1. The VCE service receives the input files and provides the output to the user via a Web service interface. The application itself is executed via SGE and Hadoop on the Cloud resources. As a test environment we utilize a department cluster consisting of eight compute nodes and a front node as Cloud resources. Each node is equipped with two Nehalem QuadCore processors, 24 GB of memory, and 6 TB of disk space. The infrastructure is interconnected via InfiniBand. On top of that, the SGE framework is configured to run Hadoop v0.21.0 jobs with the demanded resources. Before executing a request, the SGE starts the job-tracker daemons on the requested number of nodes, and submits the job to the Hadoop NameNode. A job-specific setup of the Hadoop system is utilized. The front node deploys a VCE service. This allows job dependent allocation of resources and the execution of dynamic jobs via the Internet.

4.2 Migrating ProMEX to HDFS

The ProMEX database has to be made available to the MapReduce application. This is accomplished through the import of the ProMEX database into a single large file in the HDFS. The database is stored in chunks on the data nodes across the cluster. This raises two configuration problems. The first one concerns determining the size of these chunks and the second their replication factor. Both of these choices directly affect the performance of the application. The block size is inversely proportional to the number of map tasks spawned. Due to the overhead the map task creation should be considered, especially for small clusters. Replication is primarily used for

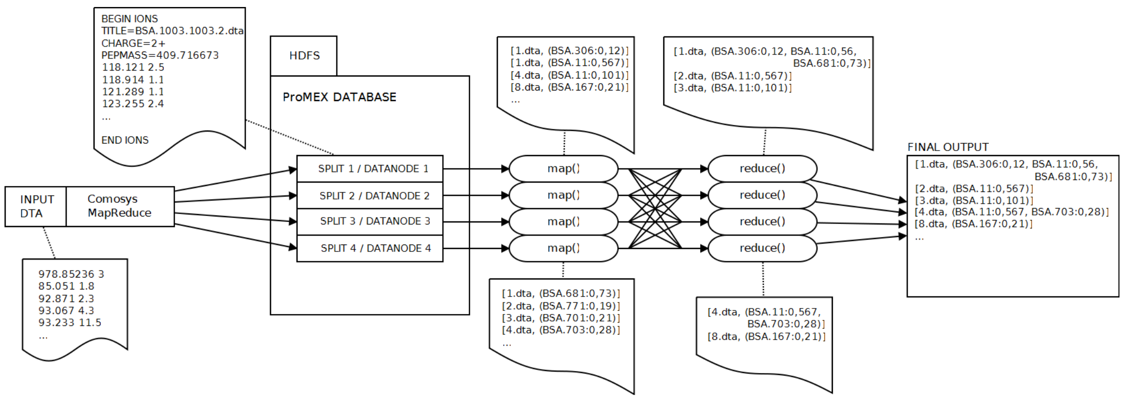

Figure 2: Data processing workflow of the MoSys-MapReduce application

fault tolerance, and secondarily for optimizing the performance of an application. By increasing the replication factor the locality of database chunks for map tasks can be increased, if not all computational resources are utilized in a job.

4.3 MapReduce Implementation

The MoSys application is utilized for protein identification in uncharacterized biological samples. It compares the samples (DTA-Format file) against the ProMEX database [11]. As stated earlier, the application and the database (MGF-format) have been recast according to the MapReduce pattern, in particular to the Apache Hadoop Framework.

The Hadoop implementation consists of Hadoop specific data structures and the business-logic needed to process the data. For reading the database in chunks and providing them to the Map tasks, we have to extend the Hadoop RecordReader-class. It interprets the information provided and converts it into an appropriate data structure processable by the application. This is realized by using the LineReader-class to read database entries split by "BEGIN / END IONS" lines in the file. The Hadoop framework initializes a map task for processing each database chunk. Each map task computes the similarity hits between the DTA-sample with the database chunk it currently processes. Upon completion, the resulting data structure, having key-value-pairs <Sample-name : hits>, is delivered to the reduce tasks to sum up all the hits of all samples and to provide the final output to the user. Figure 2 illustrates this sequence in detail.

As soon as the domain scientists have finished their experiments using the spectrometer, they are able to utilize the application for comparing the new sample with the known mass spectra stored in the ProMEX database. An extension to the basic functionality is to match more than one sample simultaneously in a comparably efficient and scalable manner. The implementation therefore provides a possibility to execute parameter studies comparing the whole set of input-files against the database. By utilizing the DistributedCache-technique of the Hadoop framework, all input files are synchronized with all allocated execution nodes which execute map tasks. The execution nodes spawn map tasks and apply the comparison algorithm on all given input files to their appropriate blocks of the database without further communication traffic.

Additionally a VCE Hadoop Service is utilized to Cloud-enable the application. The application may receive one or more DTA-files as input and provides the output as text via its generic Web service interface. The output contains summaries of all matched samples, based and sorted on the percentage of their similarity.

5. Evaluation of MapReduce Application

We evaluated the performance of the application following the MapReduce pattern (see Section 4.3) on the basis of the hardware infrastructure described in Section 4.1. The evaluation focuses on reviewing the effects on application scaling depending on the configuration parameters. The use case utilized for the evaluation is based on matching one record (stored in one input DTA-file) against the ProMEX database which has been migrated into HDFS. Due to ongoing experiments the ProMEX database is growing consecutively. Therefore, there is a need for surveying the effects on scaling the database for future usage. We inflated the current database to 100 and 500 GB of data for the experiments. Furthermore, the MapReduce Framework is configured so as to have a default block size of 128Mb and a replication factor of 1 for the data. These values have been chosen in preceding experiments. An exclusive Hadoop execution environment is configured by SGE, so that no other job can interfere the test runs. To provide correct scaling results, Hadoop is set to schedule jobs in a fair manner, using the appropriate job scheduler. It is configured to give each spawned task the same amount of resources (e.g., cores, memory) for the same period of time. We executed benchmarks related to scaling issues for hardware resources, including the number of nodes and the amount of input samples.

5.1 System Scaling

The results of the experiments are presented in Figure 3. The first graph demonstrates hardware scalability depending on a cluster of 8 nodes, comprising 8 cores each. The experiment is based on four different scenarios. They entail two different databases (100 GB and 500GB) and two different amounts of input files (single input file and 1000 input files). The experiments show that the execution time of the application scales with the number of nodes as well as with the number of input files. For example in the test case 1 File/500 GB we achieve a speedup of 7.79 for the execution on 8 nodes (64 cores) against the execution of the MapReduce implementation on 1 node (8 cores) only. The second experiment (right graph) illustrates the effects of the execution of parameter studies. The graph depicts the runtime of jobs computing 1 to 5000 input samples at once on both databases. By executing 1000 DTAs within one job we can earn a significant speedup factor of 159 against the execution of 1000 sequential jobs (runtime of 1 job comparing 1 DTA/100GB: 274 sec., 1000 DTAs/100GB: 1722 sec.) each comparing only one input file.

6. Related Work

The scientific community has contributed a lot to provisioning of scientific Cloud environments. Some institutions allow access to their science Clouds on a voluntary basis (e.g., Nimbus and Stratus [14]). The Nimbus and the Stratus Clouds provide compute and storage capability to all scientists willing to run their applications in the Cloud. Additionally there are open source Cloud toolkits available. They help in deploying private Cloud environments that can be utilized for scientific applications. An example is the OpenNebula project, which is a software stack for the

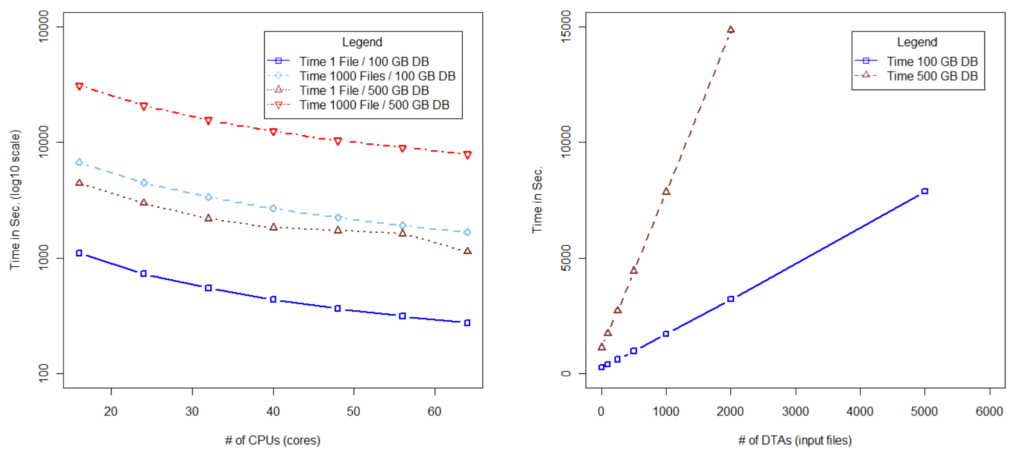

Figure 3: Summary of performance benchmarks of the MoSys-MapReduce application

provisioning of Infrastructure as a Service (IaaS). The VCE, on the other hand, can be utilized on top of the IaaS services for Cloud-enabling applications following the SaaS concept. [15] gives an overview of actual scientific Clouds and their challenges/characteristics.

The MapReduce pattern is widely used. Amazon provides an API for the execution of MapReduce application in the Amazon EC2 Cloud and [16] introduces a novel MapReduce runtime based on Microsoft Azure Cloud infrastructure services. The advantage of the VCE, is its generality and the possible usage with different IaaS providers.

7. Conclusion

The increasing amount of data available to scientists raises many new challenges. New programming paradigms, like MapReduce, aim at an easy technique to process large sets of data. They follow a massively parallel approach and thus are well suited for embarrassingly parallel problems in systems biology. However, many institutions lack computing and storage resources at the required scale. Cloud computing promises them virtually infinite resources on demand for the execution of such huge problems. Even then, those resources are not given free of charge. This work concentrates on a sustainable solution for classification of mass spectral biological protein data by leveraging relatively new computational concepts as Cloud computing and the MapReduce programming pattern. Recasting an application to MapReduce implies a lot of optimization and configuration challenges due to specifics of the underlying resources and the execution environment. We illustrated the effect of a few example parameters, such as the number of nodes, data replication, and chunk size on the performance of the application and performed a scalability study. Future tasks will include the integration of faster peptide matching algorithms as well as detailed benchmarks on deploying the application in public Clouds with respect to Quality of Service and cost models.

References

[1] J. Dean and S. Ghemawat, *Mapreduce: simplified data processing on large clusters*, *Commun. ACM*, vol. 51, pp. 107–113, January 2008.

[2] Apache Software Foundation, *Apache Hadoop*, http://hadoop.apache.org/.

[3] I. Foster, Y. Zhao, I. Raicu, and S. Lu, *Cloud computing and grid computing 360-degree compared*, in *Grid Computing Environments Workshop, 2008. GCE '08*, 2008, pp. 1–10.

[4] S. Benkner, I. Brandic, G. Engelbrecht, and R. Schmidt, *VGE - a service-oriented grid environment for on-demand supercomputing*, in *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (Grid 2004)*, November 2004.

[5] Impetus, *Whitepaper: Deriving intelligence from large data using hadoop and applying analytics*.

[6] I. Janciak, C. Kloner, and P. Brezany, *Workflow enactment engine for wsrf-compliant services orchestration*, in *In The 9th IEEE/ACM International Conference on Grid Computing*, 2008.

[7] M. Koehler and S. Benkner, *A service oriented approach for distributed data mediation on the grid*, in *Grid and Cooperative Computing, 2009. GCC '09. Eighth International Conference on*, Lanzhou, Gansu, China, August 2009, pp. 401–408.

[8] M. Koehler, M. Ruckenbauer, I. Janciak, S. Benkner, H. Lischka, and W. N. Gansterer, *A grid services cloud for molecular modelling workflows*, *International Journal of Web and Grid Services (IJWGS)*, vol. 6, no. 2, pp. 176 – 195, 2010.

[9] H. Lam, E.W. Deutsch, J.S. Eddes, J.K. Eng, N. King, S.E. Stein SE, and R. Aebersold, *Development and validation of a spectral library searching method for peptide identification from MS/MS*, in *Proteomics*, 2007, 7(5), pp.655-667.

[10] S. Wienkoop, M. Glinski, N. Tanaka, V. Tolstikov, O. Fiehn, and W. Weckwerth, *Linking protein fractionation with multidimensional monolithic reversed-phase peptide chromatography/mass spectrometry enhances protein identification from complex mixtures even in the presence of abundant proteins*, in *Rapid Commun Mass Spectrometry* 2004, 18(6):643-650.

[11] J. Hummel, M. Niemann, S. Wienkoop, W. Schulze, D. Steinhauser, J. Selbig, D. Walther, and W. Weckwerth, *ProMEX: a mass spectral reference database for proteins and protein phosphorylation sites*, vol. 8, no. 1, 2007, p.216.

[12] M. Koehler, Y. Kaniovskyi, S. Benkner, *An adaptive framework for the execution of data-intensive MapReduce applications in the Cloud*, in *The First International Workshop on Data Intensive Computing in the Clouds - DataCloud2011*, Anchorage, USA, May 2011.

[13] M. Koehler, S. Benkner, *VCE - A Versatile Cloud Environment for Scientific Applications*, in *The Seventh International Conference on Autonomic and Autonomous Systems - ICAS 2011*, Venice, Italy, May 2011.

[14] *ScienceCloud*, "http://scienceclouds.org"

[15] Craig A. Lee, *A Perspective on Scientific Cloud Computing*, in proceedings of the *19th ACM International Symposium on High Performance Distributed Computing - HPDC'10*, ACM, Chicago, Illinois, 2010, pp. 451–459.

[16] T. Gunarathne, T. Wu, J. Qiu, G. Fox, *MapReduce in the Clouds for Science*, in proceedings of the *2010 IEEE Second International Conference on Cloud Computing Technology and Science - CLOUDCOM'10*, IEEE, Washington, DC, USA, 2010, pp. 565–572.