# Microbase2.0: A Generic Framework for Computationally Intensive Bioinformatics Workflows in the Cloud

**Keith Flanagan[1], Sirintra Nakjang[1,2], Jennifer Hallinan[1], Colin Harwood[2], Robert P. Hirt[2], Matthew R. Pocock[1], Anil Wipat[1,*]**

[1]School of Computing Science, and [2]Institute for Cell and Molecular Biosciences, Newcastle University, Newcastle upon Tyne, NE7 4RU, UK

**Summary**

As bioinformatics datasets grow ever larger, and analyses become increasingly complex, there is a need for data handling infrastructures to keep pace with developing technology. One solution is to apply Grid and Cloud technologies to address the computational requirements of analysing high throughput datasets. We present an approach for writing new, or wrapping existing applications, and a reference implementation of a framework, Microbase2.0, for executing those applications using Grid and Cloud technologies. We used Microbase2.0 to develop an automated Cloud-based bioinformatics workflow executing simultaneously on two different Amazon EC2 data centres and the Newcastle University Condor Grid. Several CPU years' worth of computational work was performed by this system in less than two months. The workflow produced a detailed dataset characterising the cellular localisation of 3,021,490 proteins from 867 taxa, including bacteria, archaea and unicellular eukaryotes. Microbase2.0 is freely available from http://www.microbase.org.uk/.

## 1 Introduction

The number of active genome sequencing projects has been increasing exponentially since 1995 [1]. Improvements in technology, coupled with large-scale deployment of sequencing hardware, have dramatically reduced the cost of producing genome sequences. Sequencing entire bacterial genomes and environment-specific metagenomes is now routine. As a result, publicly available sequence databases (such as GenBank) currently double in size approximately every two years [2]. Acquiring raw sequence data is just the beginning; a wide range of computational tools are required in order to derive new knowledge from the data.

A variety of bioinformatics tools has been developed to help derive knowledge from biological sequence data. For a given biological question, several of these tools may be run in order to generate results from which biologically relevant conclusions may be drawn. Many of these tools run in polynomial time, scaling with both the number of input sequences and their lengths. The sheer amount of data that must be manipulated leads to scalability and storage challenges as well as increasing workflow complexity. Many solutions have been proposed to address computational scalability problems in bioinformatics. These approaches range from the use of algorithm-specific dedicated hardware such as field-programmable gate arrays, to GPU implementations [3], to the use of massively parallel generic computing hardware [4]. Distributed computing approaches include Grid and Cloud computing. Grid technologies typically involve large numbers of distributed heterogeneous resources that may be spread across several geographical locations and administrative domains. Computational Grids are a means for researchers to obtain and share computational power and data storage

[*]To whom correspondence should be addressed. Email: Anil.Wipat@newcastle.ac.uk

either within their own institutions, or across institutional and geographical boundaries. Grids are extensively used in bioinformatics data processing [5, 6]. In contrast, Cloud computing typically involves the sale of computational resources by providers with large amounts of computational capacity. Cloud computing is considered by some to be an evolution of Grid computing [7], offering the ability to dynamically expand an organisation's computational power [8]. Users are often allowed complete control over their own secured virtual environments.

## 1.1 Challenges in bioinformatics

A number of individual bioinformatics applications have been successfully executed in Grid or Cloud environments [9, 10]. Many of these approaches use Grid or Cloud middleware infrastructures, such as Globus [11] or Hadoop[1]. These frameworks abstract away from the specific heterogeneous hardware configurations participating in a computation, and also provide job management functions such as the re-execution of failed jobs. However, the logistics of assembling multiple computationally-intensive analysis tools to run as a workflow are challenging. As the number of tools that are required increases, co-ordination of structured data flows between processes becomes essential. One program is often required to consume the output of another [12]. Therefore, in addition to providing scalable execution and data management, software platforms must be flexible enough to support and maintain sets of bioinformatics tools organised into workflows [13]. Automation toolkits such as Taverna [14] and Kepler [13] enable the construction of complex workflows that utilise and co-ordinate multiple remotely hosted services to achieve a particular goal. Workflow enactment permits data to flow from one service to another in an automated fashion. Upon completion of a workflow results are returned to the user. Workflow automation has been shown to save a large amount of time by removing the manual ‘copy and paste’ operations that would otherwise be required to move data between analysis tools [15].

Other challenges facing workflow developers include the extraction of information from application output files and the organisation and storage of large amounts of results. Many bioinformatics applications were written for manual operation in small-scale analyses, and produce human-readable output files that are difficult for machines to parse. Such software is often designed to execute on a single machine and is not necessarily written with parallel or distributed computing environments in mind. Since primary bioinformatics data sources are continually being updated, it is necessary to keep secondary datasets up-to-date by acquiring new data and performing new computational analyses. Re-computing entire secondary datasets each time new primary data are released is quickly becoming infeasible, even when large compute clusters are available [16]. It is therefore desirable for a workflow enactment environment to support incremental additions to existing datasets with minimal additional computational work.

## 1.2 Exploring the microbial extracytoplasmic proteome

The proteome can be conceptually divided into a core set of proteins, performing essential housekeeping tasks, common to a broad range of bacteria, and a peripheral proteome, equipping the organism for life in a particular environment [17]. The extracytoplasmic proteome is particularly likely to be important to the specific phenotype of a given organism as it mediates many primary aspects of its interaction with the environment (e.g. [18]). We developed an analysis workflow using Microbase2.0, incorporating multiple targeting-signal prediction tools to identify extracellular proteins and domains. The workflow supports the

[1] http://hadoop.apache.org/

incremental addition of new data items, and is extensible in terms of new software applications. We apply this workflow to the identification of extracytoplasmic proteins—defined as the combination of membrane, intermembrane space and secreted proteins—in all organisms for which a complete genome sequence was available in RefSeq [19] as of June, 2010.

## 2 An architecture for large-scale analysis workflows in the Cloud

We have previously developed Microbase a Grid-based framework for bioinformatics [20-22]. In this paper we describe Microbase2.0. Micobase2.0, whilst based on the principles introduced with Microbase1.0, provides a completely re-engineered and re-architectured system that has been devised to meet the challenges presented by more recent developments in Grid and Cloud computing. The Microbase2.0 system is a distributed computing bioinformatics framework, consisting of a set of separate, loosely coupled services that co-operate to provide the infrastructure required by Grid- or Cloud-based analysis workflows (Figure 1). These components are:

- A notification system: facilitates de-coupled communication between workflow components;
- The filesystem: a scalable, distributed file store specification and reference implementation;
- A distributed process manager: provides job scheduling and failure management for heterogeneous groups of worker nodes;
- Domain-specific application components (termed responders): user-written components that either perform an analysis, or delegate a task to an existing analysis program.

The Microbase2.0 architecture facilitates scalability and reliability by replicating infrastructure components over a number of servers. These components cooperate to ensure that each instance works on a different task; no computational work is unnecessarily duplicated. In a deployment of a Microbase2.0 system, the various system services may either be located on a single physical server or spread across several, potentially geographically distant machines. Therefore, a small-scale test or development system can be gradually scaled up as the requirements of a project evolve.

Responders are modular components that are written by the user. Each Microbase2.0 responder typically maps to a single bioinformatics software tool. A group of responders can be configured to form a workflow. Each responder type may be deployed to one or more servers for scalability, as required.

Command line bioinformatics applications are typically designed to run independently on a single machine, usually consuming a set of input files and producing a set of output files. These applications are typically unaware of distributed computing concepts such as obtaining data from a remote machine. However, the user of a large-scale bioinformatics analysis workflow may require result data to be stored and indexed in a database for later querying. A responder must therefore encapsulate the entire functionality of a specific bioinformatics tool by meeting the needs of the command line application (e.g., access to local data files), and the typical requirements of a large-scale analysis workflow (e.g., database storage of results). Responders typically need to provide: a means to acquire data files from the Microbase2.0

filesystem; a file parser to interpret the result files produced by the analysis tool; and a database insertion tool to permanently archive results.

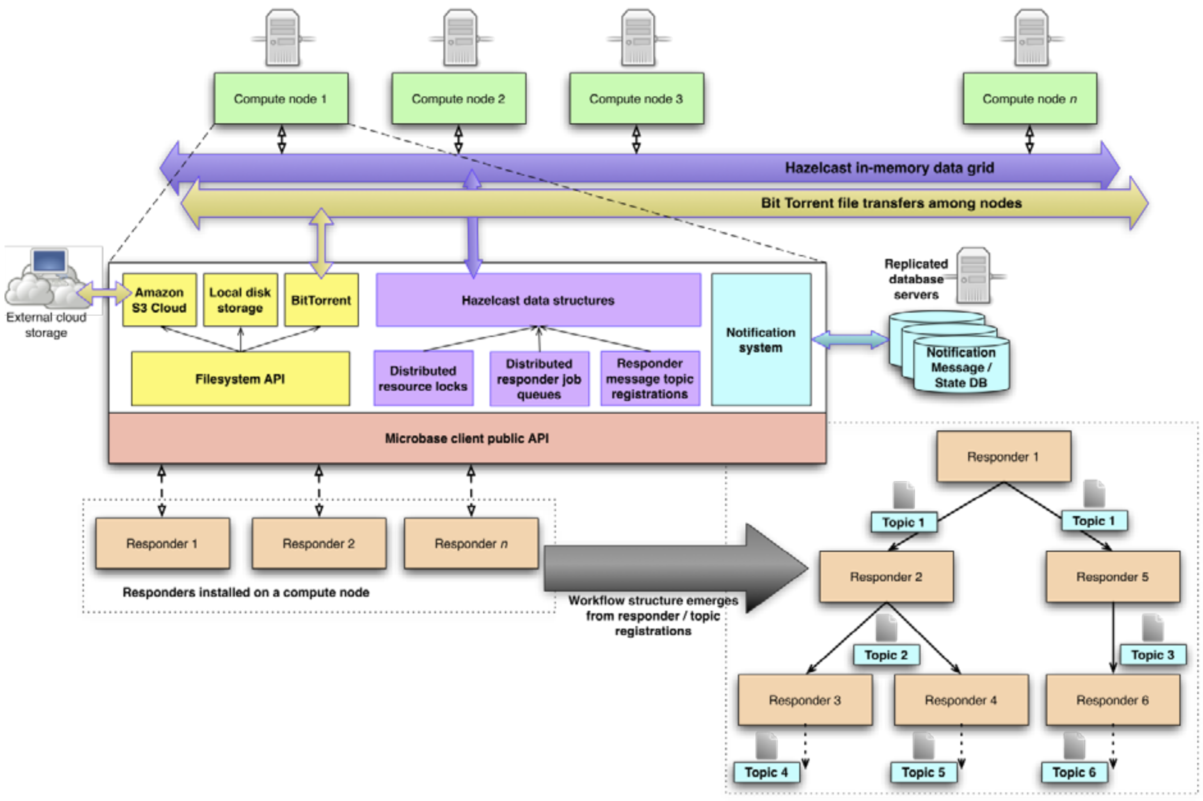
Compute node 1
Compute node 2
Compute node 3
Compute node n
Hazelcast in-memory data grid
Bit Torrent file transfers among nodes
External cloud storage
Amazon S3 Cloud
Local disk storage
BitTorrent
Hazelcast data structures
Notification system
Filesystem API
Distributed resource locks
Distributed responder job queues
Responder message topic registrations
Microbase client public API
Replicated database servers
Notification Message / State DB
Responder 1
Responder 2
Responder n
Responders installed on a compute node
Workflow structure emerges from responder / topic registrations
Topic 1
Responder 5
Topic 2
Topic 3
Responder 3
Responder 4
Responder 6
Topic 4
Topic 5
Topic 6

**Figure 1: A summary of the Microbase2.0 client architecture and its operation. Microbase2.0 provides APIs for a number of core services that facilitate workflow orchestration: the event notification system (blue), file storage and distribution system (yellow), and various distributed data structures that provide task scheduling and monitoring operations (purple). The Microbase2.0 client is installed on available computers (green). Every compute node is symmetric – there are no special nodes such as a 'master node'. In addition to the Microbase2.0 client, compute nodes may have one or more user-written components, termed 'responders', installed. A responder may register its interest in a particular message topic, and is activated by the system when a suitable message arrives in the notification system. A workflow is formed when messages generated by one responder are passed as input to another responder (bottom right).**

High-level inter-process communication between responders is carried out using a publish-subscribe notification-based approach. The Microbase2.0 notification system permits responders to register an interest in particular types of message, and thus receive past and present announcements from other workflow components. Typically, notification messages convey information about the completion of an analysis task, or the availability of a new data item. Message subscriptions therefore dictate the order in which responders execute; a 'new data available' message from one responder may result in a cascade of events as other responders react to the new data, and produce their own results, causing further messages to be published. The notification system database records every message that is sent by system or workflow responders. Communication via the notification system is performed in a loosely-coupled fashion; the publisher of a message does not have knowledge of the potential receivers, or even whether there are any receiving responders. This property is essential for supporting dynamic workflows to which new applications may be added in future. A new component added to a workflow receives the relevant message history, depending on its topic subscriptions.

The Microbase2.0 file system is responsible for storing and distributing input and output data files for each responder. In a distributed environment a large number of worker nodes may need access to data resources simultaneously. Raw output files produced by analysis applications are permanently archived in the Microbase2.0 file system. Currently, Microbase2.0 provides a reference file system implementation that provides scalable transfers via the peer-to-peer protocol, BitTorrent. An implementation of the file system, that uses Amazon S3[2] as its data store, is under development, and other implementations for different file transfer protocols may be provided in future.

Computational work is managed in Microbase2.0 through a set of memory-resident distributed data structures. Each message published to the notification system represents a unit of work to be processed. In addition to message content, the notification system also stores message state information for each responder that subscribes to a particular topic. Responder processes, and the distributed data structures on which they rely, are entirely de-centralised and are therefore tolerant of failures. Microbase2.0 makes use of the open source data grid library, Hazelcast[3] to implement the necessary coordination of multiple distributed processes. The Microbase2.0 compute client maintains a distributed process list of the messages being processed by the system at every point in time. This data is accessible to responders via the client API, and is useful for dynamically adjusting the number of processes of a given type executing simultaneously. For example, if a particular type of responder depends on a single shared resource, such as a relational database, it may be useful to limit the number of active instances in order to improve the overall throughput of the system. The compute client may run on top of existing distributed platforms such as Condor [23], may be embedded within an Amazon EC2 virtual machine image, or may simply be started on local machines via a remote shell such as SSH. In any deployment environment, minimal configuration is required for Microbase2.0 compute client instances to find each other in order to participate in processing the same workflow.

## 3 An analysis workflow for the study of extracytoplasmic proteins

The Microbase2.0 framework has been used by a number of researchers to construct several bioinformatics analysis workflows. In this paper we describe a workflow for the identification of putative secreted and surface associated proteins using Grid and Cloud computing technologies, the Extracytoplasmic Protein Prediction Pipeline (EPPP) (Figure 2). The EPPP workflow employs several sequence analysis tools in order to identify putative extracytoplasmic proteins. A set of Microbase2.0 responders was developed to encapsulate bioinformatics tools for the identification of putative targeting signals (SignalP, LipoP, TMHMM), cell surface anchoring regions (InterProScan) and the identification of homology (BLAST-P). The workflow performs the following functions: automatic retrieval of sequence data from a public genome resource (NCBI) and the generation of appropriately formatted input data for the downstream analysis processes; processing of the input sequence through various bioinformatics tools; and extraction of a list of candidate extracytoplasmic protein based on the prediction results.

[2] http://aws.amazon.com/s3/

[3] http://www.hazelcast.com/

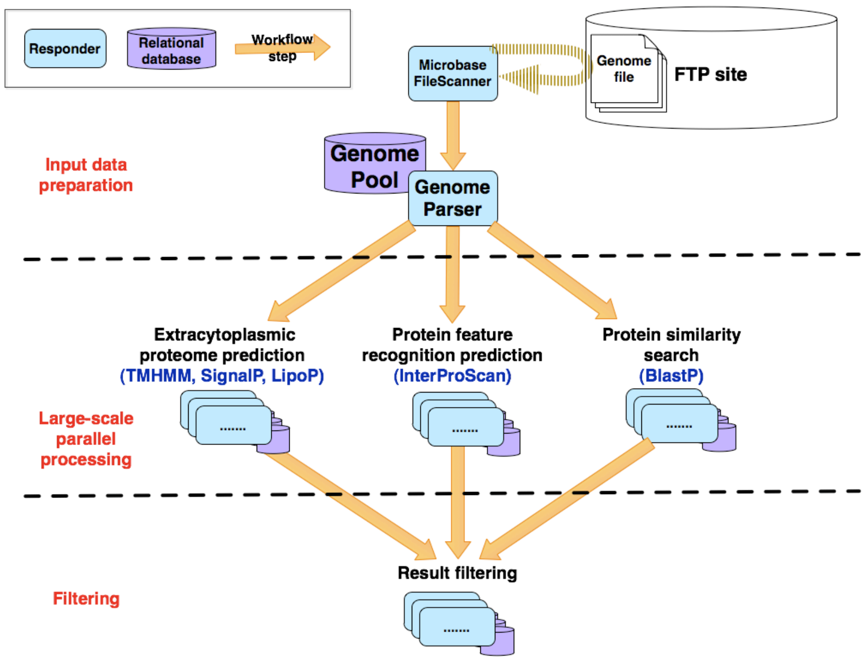
Responder
Relational database
Workflow step
Microbase FileScanner
Genome file
FTP site
Input data preparation
Genome Pool
Genome Parser
Extracytoplasmic proteome prediction (TMHMM, SignalP, LipoP)
Protein feature recognition prediction (InterProScan)
Protein similarity search (BlastP)
Large-scale parallel processing
Result filtering
Filtering

**Figure 2: The EPPP workflow consists of multiple Microbase2.0 responders. The workflow starts with events generated by the FileScanner responder, which is configured to periodically scan an FTP server for new files. The GenomeParser responder reacts to the presence of new genome files by parsing and storing them in a structured database, the GenomePool. Once a new genome has been stored notification messages are sent to analysis responders, resulting in a large amount of computational work being scheduled. Each responder has its own independent database. Finally, the results from each analysis tool are extracted and integrated into a relational database.**

The EPPP workflow is triggered when new GenBank-formatted genome data files become available. The Microbase2.0 FileScanner responder is responsible for detecting the arrival of a new file on an FTP site and for saving the file into the Microbase2.0 filesystem. The successful completion of this process triggers the next responder, GenomeParser. The GenomeParser responder receives messages published by the FileScanner responder. The GenomeParser is responsible for extracting genome information from the plain text GenBank files and storing it within an indexed, structured database (the GenomePool) for convenient access by other responders or users. The completion of this process results in the GenomeParser publishing messages which activate a series of downstream responders, each encapsulating a different bioinformatics tool. The output from each application is parsed and stored in independent structured databases. Finally, sequence analysis results are post-processed by the final responder, which performs a custom filtering process for the identification of putative extracytoplasmic proteins.

## 4 Application of the extracytoplasmic protein prediction workflow

### 4.1 Workflow performance

The input dataset for the workflow was created by downloading all available complete genome sequences from the RefSeq database [19]. The resulting database contained data for 3,021,490 proteins from 867 taxa (Table 1).

The EPPP workflow was utilised to identify putative extracytoplasmic proteins from the dataset. The protein sequences to be analysed were split into manageable blocks of between 100 and 1,000 sequences by the responders. This analysis involved 3,021,490 protein sequences and resulted in 101,943 computing jobs being produced by the five sequence analysis responders. The analysis jobs were assigned to the Condor Grid at Newcastle University (a minimum of 27 and a maximum of 74 worker nodes, depending upon availability, and fluctuating during the course of the workflow execution as users logged on and off desktop machines), and the Amazon Cloud computing resource.

**Table 1. Composition of the workflow input dataset**

| Group | Number of Taxa |
|---|---|
| Gram positive bacteria | 231 |
| Gram negative bacteria | 549 |
| Archaea | 55 |
| Unicellular eukaryotes | 32 |
| **Total** | **867** |

By exploiting this high-throughput computing approach, all of the jobs were successfully completed within two months. This two-month time frame includes not only compute time, but also other Microbase2.0-specific overheads, such as file management, and re-computation of failed jobs (Table 2). Overall, InterProScan processes benefit the most from the use of the distributed computing system; the four years of CPU time required to process three million sequences was reduced to 16 days of wall clock time. Likewise, for BLAST-P, the amount of active time for processing approximately 26,000 BLAST-P-pairwise and BLAST-P-refseq jobs was reduced significantly. The other, less CPU-intensive programs, LipoP, TMHMM and SignalP required four hours, 11 hours and 31 hours respectively.

## 4.2 Results of large-scale extracytoplasmic protein prediction

Of the 3,021,490 protein sequences present in the GenomePool, 981,769 protein sequences were predicted to be extracytoplasmic proteins (Table 3).

Based on the proteomes included in this study, the fractions of putative extracytoplasmic proteins across the four groups of microorganisms were estimated to be 24.6%, 25.9%, 31%, and 34.6% for microbial eukaryotes, archaea, Gram-positive bacteria and Gram-negative bacteria, respectively.

Protein sequences were considered to be putative extracytoplasmic proteins if they were predicted to have at least one of these features: 1) transmembrane region(s) predicted by TMHMM; 2) a signal peptide cleavage site predicted by SignalP; 3) a predicted signal peptidase II cleavage site assigned by LipoP; 4) or at least one well-defined functional domain indicative of a protein associated with an extracellular space. The results from the workflow were compared with a set of sequences whose subcellular location had already been determined experimentally. The experimentally verified data set was obtained from the ePSORTdb [45]. The experimental dataset was used to assess the results obtained for archaeal, Gram-positive (Gm+) and Gram-negative (Gm-) bacterial proteins - a total of 9,265 prokaryotic protein sequences. The positive predictive value and sensitivity of the workflow were computed for the bacterial and archaeal groups (Table 4).

**Table 2: Execution times of the Microbase2.0 responders used in the EPPP workflow. 'Total CPU usage time' or total computing time shows an estimated time for a desktop with an Intel core 2 (6300) duo 1.86 GHz CPU, and 2GB memory to complete all the tasks for the specified type. 'Total active time' is the amount of time spent on both the computation time of the responders and Microbase2.0 overheads such as file. For the FileScanner, GenomeParser and TMHMM responders, one job represents one genome file. For other responders, the number of jobs varies depending on the number of protein sequences allowed per job. '-' indicates that all the protein sequences annotated in a genome file are considered as one compute job. It is notable that the total number of jobs processed by the TMHMM responder is less than the number of genome files passed into the workflow. This discrepancy is due to some genome files having no gene product annotations.**

| Responder | Total number of jobs | Limit to number of protein sequences in a job | Average time for a successful job execution (mins) | Total CPU usage time | Total active time | Average number of machines used |
|---|---|---|---|---|---|---|
| FileScanner | 3,153 | - | 0.01 | 27.99 mins | 26 hrs | 19 |
| GenomeParser | 3,153 | - | 0.32 | 16.67 hrs | 26 hrs | 40 |
| TMHMM | 2,892 | - | 1.88 | 3.78 days | 11 hrs | 74 |
| SignalP | 41,091 | 150 | 0.08 | 2.27 days | 1 day 7 hrs | 37 |
| LipoP | 2,941 | 1,500 | 0.04 | 1.93 hrs | 4 hrs | 27 |
| InterProScan | 31,924 | 100 | 68.12 | 1,510.15 days | 16 days 15 hrs | 60 |
| BLAST-P-refseq | 2,942 | 200 | 81.38 | 166.26 days | 7 days 11 hrs | 40 |

**Table 3: Summary of protein sequences assigned to different classes by the extracytoplasmic classification workflow. The workflow was applied to all protein sequences deposited in the GenomePool database. Results are shown in relation to organism groups depending on the major cell surface structures. The Gram-positive group includes members of bacterial phyla Actinobacteria, Firmicutes and Tenericutes. Other bacterial phyla are considered to belong to the Gram-negative group.**

| Organism group | Total number of proteins | Total number of predicted extracytoplasmic proteins |
|---|---:|---:|
| Gram positive | 693,402 | 214,955 |
| Gram negative | 1,922,673 | 665,194 |
| Microbial eukaryote | 272,389 | 67,121 |
| Archaea | 133,026 | 34,499 |
| Total | 3,021,490 | 981,769 |

**Table 4: Performance of the EPPP workflow. True positive (TP) and False negative (FN) denote the number of experimentally verified extracytoplasmic protein sequences that were predicted by the EPPP workflow as extracytoplasmic and intracytoplasmic, respectively. True negative (TN) and False positive (FP) represent the number of experimentally verified cytoplasmic proteins that were predicted by the EPPP workflow as intracytoplasmic and extracytoplasmic, respectively.**

| Organism group | TP | FP | TN | FN | Positive predictive value (%) | Sensitivity (%) |
|---|---|---|---|---|---|---|
| Archaea | 68 | 0 | 0 | 7 | 100.00 | 90.67 |
| Gm- | 1,779 | 89 | 4,853 | 226 | 95.24 | 88.73 |
| Gm+ | 367 | 37 | 1,621 | 57 | 90.84 | 86.56 |

## 4.3 Discussion

In the post-genomic world of modern biology, very large amounts of data are routinely generated on a day-to-day basis. As technologies for the generation and storage of biological data become faster, cheaper and more capable, the bottleneck in the generation of new knowledge is increasingly becoming the crucial annotation and analysis process. A wide range of bioinformatics tools is available to perform different types of analysis of many different types of data. Many of these tools were originally written for small scale use on a single CPU, with the output targeted at human users. In order to cope with the volume of data now available these tools can either be rewritten – a time-consuming and potentially error-prone process – or adapted to use in high-throughput, massively parallel computational environments.

Most bioinformatics analyses require the use of more than one tool, either in parallel or in sequence. The outputs of one tool frequently become or inform the inputs of another. In order to avoid the necessity for manual cutting and pasting of data between tools, some form of automated orchestration is required, usually in the form of workflows. As datasets become larger and more complex, more compute cycles are needed for their analysis. Parallel processing can bring computation time within practical limits, by distributing tasks amongst CPUs on multiple machines, either locally or on the Cloud. We have developed Microbase2.0, a computational platform which facilitates the building of bioinformatics solutions in a Cloud environment, and which can handle very large data sets.

Microbase2.0 is a distributed systems architecture that permits multiple bioinformatics tools to be incorporated into long-running workflows. Where possible, Microbase2.0 components make extensive use of existing open source software. Two notable examples are Hazelcast for the provision of various distributed data structures and Azureus[4] for providing BitTorrent file transfers. In addition to providing a wrapper for executing existing applications in a distributed environment, the system provides data management functionality, execution provenance and job failure detection. Furthermore, processing workflows composed of responders are extensible both in terms of the handling of new data, and also the addition of new applications. When new data are added to a workflow, it is possible to incrementally update existing datasets without re-analysing existing data. Likewise, adding a new responder to an existing workflow only requires the new responder to catch up with the current system

[4] http://azureus.sourceforge.net/

state by processing existing relevant notification messages. The inherent modularity of responders enables them to be reused in different workflows with very minor changes.

The advantage of Microbase2.0 over other highly distributed, Cloud-based bioinformatics solutions, such as Hadoop, is that Microbase2.0 supports agile workflow development. There is no need for a rigid workflow definition, for example, in the form of a file. The design for a workflow in Microbase2.0 is never truly complete. New responders can be registered at any time to any message topic, and all previous messages will be delivered as if these responders had always been present. In this sense, a Microbase2.0 workflow is an emergent phenomenon of a particular set of responders, and the messages to which they are subscribed. The ability to dynamically deploy workflow components facilitates the execution of partial workflows. This ability is useful in Cloud computing environments where the price of compute resources fluctuates over time according to user demand, which in turn varies with the local time of day. A particular subset of responders can be prioritised and installed on a small number of machines. The rest of the workflow can then be executed later when the price of Cloud processing nodes is reduced. The second set of responders can then catch up to the progress made by the first set of responders, achieving the same result as if the entire workflow had run continuously. Another advantage of Microbase2.0 workflows, stemming from the loosely-coupled nature of responders, is the ability to dynamically extend a workflow, even while it is actively executing. Adding new functionality to an existing workflow avoids a complete re-execution of the workflow by re-using all existing results, permitting new tools to be added incrementally over time and as requirements or available tools change.

Microbase workflows are particularly efficient because not only is the computational work distributed, but Microbase2.0 can orchestrate the simultaneous parallel execution of multiple instances of multiple bioinformatics tools. The tools incorporated into the workflow responders have been used extensively by the bioinformatics community, and hence are well-accepted and well-understood. The only novel responders are those performing problem-specific tasks: the retrieval and parsing of genome files (a task which is the first step in many bioinformatics workflows), and the final classification and filtering of extracytoplasmic proteins. These responders, once developed, can be re-used in any workflow in which they are applicable.

The Microbase2.0 system has significant advantages over existing approaches for executing large-scale bioinformatics workflows consisting of many tools. Tools with large input files and long run times, such as InterProScan and BLAST, perform under Microbase2.0 with very high efficiency. Even when using tools with short job lengths or a requirement for many small input files, the use of Microbase2.0 is advantageous because of the systematic nature of its processing, the ability to incrementally add new data and analyses, and its ability to track workflow execution provenance information. Microbase2.0 is an open source project, and is freely available from http://www.microbase.org.uk/.

## Author contributions

KF, MP and AW conceived the idea and design for Microbase2.0. KF, SN and MP implemented the software. SN, CRH, RPH, and AW were involved in the extracellular proteome analysis. JH, KF, SN and AW wrote the manuscript. All authors proofread and edited the manuscript.

## Acknowledgements

The authors would like to thank Amazon for EC2 credit. We also wish to thank the Computing Science support team at Newcastle University for use of their desktop computer clusters and for help with debugging network issues. SN's PhD project was supported by the Faculty of Medical Sciences and the School of Computing Science at Newcastle University and an Overseas Research Students Awards Scheme. This work was also supported by funding from the EPSRC (EP/G061394/1 and EP/C536444/1). JH was supported by an RCUK Academic Fellowship.

## References

[1] K. Liolios, K. Mavromatis, and N. Tavernarakis. The Genomes On Line Database (GOLD) in 2007: status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Research*, 38:D475-D479, 2008.

[2] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and E. W. Sayers. GenBank. *Nucleic Acids Research*, 37:D26-D31, 2009.

[3] B. Harris, A. Jacob, J. Lancaster, J. Buhler, and R. Chamberlain. A banded Smith-Waterman FPGA accelerator for Mercury BLASTP. *In* International Conference on Field Programmable Logic and Applications, pages 765-769, 2007.

[4] D. Sulakhe, A. Rodriguez, M. Wilde, I. Foster, and N. Maltsev. Interoperability of GADU in using heterogeneous grid resources for bioinformatics applications. *IEEE Trans Inf Technol Biomed*, 12:241-246, 2008.

[5] M. Halling-Brown, D. Moss, and A. Shepherd. Towards a lightweight generic computational grid framework for biological research. *BMC Bioinformatics*, 9:407, 2008.

[6] T. Craddock, C. R. Harwood, J. Hallinan, and A. Wipat. e-Science: Relieving bottlenecks in large-scale genomic analyses. *Nature Reviews Microbiology*, 6:948 - 954, 2008.

[7] K. Keahey, M. Tsugawa, A. Matsunaga, and J. Fortes. Sky computing. *IEEE Internet Computing*, 13:43-51, 2009.

[8] R. Lucky. Cloud computing. *IEEE Spectrum*, 46:27, 2009.

[9] B. Langmead, M. C. Schatz, J. Lin, M. Pop, and S. L. Salzberg. Searching for SNPs with cloud computing. *Genome Biology*, 10:R134, 2009.

[10] P. Di Tommaso, M. Orobitg, F. Guirado, F. Cores, T. Espinosa, and C. Notredame. Cloud-Coffee: implementation of a parallel consistency-based multiple alignment algorithm in the T-Coffee package and its benchmarking on the Amazon Elastic-Cloud. *Bioinformatics*, 26:1903-1904, 2010.

[11] I. Foster. Globus toolkit version 4: Software for service-oriented systems. *Journal of Computer Science and Technology*, 21:513-520, 2006.

[12] A. Billion, R. Ghai, T. Chakraborty, and T. Hain. Augur--a computational pipeline for whole genome microbial surface protein prediction and classification. *Bioinformatics*, 22:2819-2820, 2006.

[13] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific workflow management and the Kepler system. *Concurrency and Computation*, 18:1039-1065, 2006.

[14] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, et al. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20:3045-3054, 2004.

[15] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. R. Pocock, P. Li, and T. Oinn. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research*, 34:W729-W732, 2006.

[16] M. C. Walter, T. Rattei, R. Arnold, U. Gueldener, M. Muensterkoetter, K. Nenova, G. Kastenmueller, P. Tischler, A. Woelling, A. Volz, et al. PEDANT covers all complete RefSeq genomes. *Nucleic Acids Research*, 37:D408-D411, 2009.

[17] S. Yudong, A. Wipat, M. Pocock, P. A. Lee, K. Flanagan, and J. T. Worthington. Exploring microbial genome sequences to identify protein families on the Grid. *IEEE Transactions on Information Technology in Biomedicine*, 11:435-442, 2007.

[18] T. T. Tseng, B. M. Tyler, and J. C. Setubal. Protein secretion systems in bacterial-host associations, and their description in the Gene Ontology. *BMC Microbiology*, 19:S2, 2009.

[19] K. D. Pruitt, T. Tatusova, W. Klimke, and D. R. Maglott. NCBI Reference Sequences: current status, policy and new initiatives. *Nucleic Acids Research*, 37:D32 - D36, 2009.

[20] Y. Sun, A. Wipat, M. Pocock, P. A. Lee, K. Flanagan, and J. T. Worthington. Exploring microbial genome sequences to identify protein families on the Grid. *IEEE Transactions on Information Technology in Biomedicine*, 11:435 - 442, 2007.

[21] Y. Sun, W. Anil, M. Pocock, P. A. Lee, P. Watson, K. Flanagan, and J. T. Worthington. A grid-based system for microbial genome comparison and analysis. *In* IEEE International Symposium on Cluster Computing and the Grid, pages 977-984, 2005.

[22] A. Wipat, Y. Sun, M. Pocock, P. A. Lee, P. Watson, and K. Flanagan. Developing Grid-based systems for microbial genome comparisons: The Microbase project. *In* Proceedings of the UK e-Science All Hands Meeting 2004, Nottingham, UK, pages 532–538, 2004.

[23] M. J. Litzkow, M. Livny, and M. W. Mutka. Condor-a hunter of idle workstations. *In* 8th International Conference on Distributed Computing Systems, pages 104-111, 1988.