DSIMBench: A Benchmark for Microarray Data Using R

Shicai Wang, Ioannis Pandis, Ibrahim Emam, David Johnson, Florian Guitton, Axel Oehmichen, and Yike Guo(✉)

Data Science Institute, William Penney Laboratory, Imperial College London, South Kensington Campus, London SW7 2AZ, UK
{s.wang11,i.pandis,i.emam,david.johnson,f.guitton, axelfrancois.oehmichen11,y.guo}@imperial.ac.uk
http://dsg.doc.ic.ac.uk

Abstract. Parallel computing in R has been widely used to analyse microarray data. We have seen various applications using various data distribution and calculation approaches. Newer data storage systems, such as MySQL Cluster and HBase, have been proposed for R data storage; while the parallel computation frameworks, including MPI and MapReduce, have been applied to R computation. Thus, it is difficult to understand the whole analysis workflows for which the tool kits are suited for a specific environment. In this paper we propose DSIMBench, a benchmark containing two classic microarray analysis functions with eight different parallel R workflows, and evaluate the benchmark in the IC Cloud testbed platform.

Keywords: Benchmark · R · MPI · MapReduce

1 Introduction

Data mining techniques applied to microarray data convert raw intensity values into useful information. R is one of the most popular data mining software tools used for medical research. With masses of data accumulating from translational research studies involving high-throughput sequencing, many high performance databases, such as MySQL Cluster [1], PostgreSQL Cluster [2], MongoDB [3] and HBase [4], and parallel computing frameworks, including Message Passing Interface (MPI) [5] and MapReduce [6], are being integrated into the traditional microarray analysis tool, R [7]. Though these new methods greatly improve the performance of R, they greatly complicate the whole analysis workflow. For example, all the databases and parallel frameworks mentioned above form eight different R workflows. Many datasets are required to fully evaluate the performance of each workflow. Thus, hundreds of, or even thousands of, tests must be performed in order to robustly evaluate and determine the most efficient and effective workflow.

Our motivation is to find an effective big data solution for our open source knowledge management software platform tranSMART [8], which was originally

J. Zhan et al. (Eds.): BPOE 2014, LNCS 8807, pp. 47–56, 2014.
DOI: 10.1007/978-3-319-13021-7_4

developed by Johnson&Johnson for in-house clinical trial and knowledge management requirements in translational studies. For the needs of various collaborative translational research projects, an instance of tranSMART is hosted at Imperial College London and has been configured to use an Oracle relational database for back-end storage. It currently holds over 70 million gene expression records. When querying the database simultaneously for hundreds of patient gene expression records, a typical exercise in translational studies, the record retrieval time can currently take up to several minutes. Furthermore, some typical analyses using R, such as marker selection and data clustering, can take up to several minutes, or even hours. These kinds of response times impede applications performed by researchers using this deployed configuration of tranSMART. Anticipating the requirement to store and analyse next generation sequencing data, where the volume of data being produced will be in the TB or PB range, the current performance exhibited by tranSMART is unacceptably poor.

In this paper, we present DSIMBench (Data Science Institute Microarray Benchmarks), which uses two common translational medical applications with six representative data mining workflows, and evaluate the benchmark on the IC Cloud [9] testbed.

2 Related Work

Benchmarks play a significant role in all domains. SPEC [10] benchmarks are gold standards used by many processor manufacturers and researchers to measure the effectiveness of their inventions. Popular benchmarking suites designed for specific application domains are also well accepted, such as TPC-H [11] for database systems, SPLASH [12] for parallel architectures, and MediaBench II [13] for media and communication processors.

Many bioinformatics benchmark suites are widely in use, such as BioBench [14], BioPerf [15] and MineBench [16]. These benchmarks contain several applications in common, including BLAST, FASTA, Clustalw, and Hmmer. The bioinformatics applications presented in DSIMBench differ from those included in these benchmark suites. BioBench contains only serial workloads. Bioperf only uses a few parallelized applications. Even in MineBench which contains full-fledged OpenMP parallelized codes of all bioinformatics work-loads, no large-scale computing framework has been integrated, such as MPI and MapReduce. In contrast to the above benchmarks, DSIMBench focuses on R scalability and performance for big data technologies with microarray data.

3 R with High Performance Plugins

3.1 Data Distribution

A standard vanilla R workflow loads the entire data before performing calculations. However, R provides many interfaces to different kinds of storage systems such as built-in functions (e.g. CSV reader), for local file system access

(e.g. Linux ext4), DBI plugins for relational database access (e.g. MySQL Cluster), and RHadoop plugin for interfacing with key-value database clusters (e.g. HBase).

3.2 Parallel Computation

There are many high performance R plugins that parallelize calculationd for CPU cores within one machine or for CPU cores across machines configured in a computing cluster. MPI and MapReduce are two representative technologies used for big data. For MPI, the R Snowfall [17] plugin is a usability wrapper around the Rmpi [18] plugin for more usable development of parallel R programs. Rmpi is a widely used MPI interface for the Local Area Multicomputer (LAM) [19], with MPICH2 [20], a MPI implementation. For MapReduce, the RHadoop plugin is a representative interface for the Apache Hadoop ecosystem [21], including Hadoop Distributed File System (HDFS), MapReduce and HBase.

4 DSIMBench Workflows

We designed eight R workflows based on different data distributions and computational solutions, as shown in Table 1. The first three workflows (W1–W3) are created to test the data loading performance. Each workflow loads data from one of three data sources, including local file system ext4, relational database MySQL Cluster and key-value database HBase, and performs computation in vanilla R. Workflow W4 acts as a baseline test for the parallel computations. Workflows W5 and W6 test only the performance on the parallelization of the computation in R, as the data is delegated directly from the master node through direct network sockets. Finally workflows W7 and W8 test both the data loading and parallel computation in combination, where W7 loads data to the worker nodes using the fastest data loading workflow chosen from test results of W1–W3 with MPI, while W8 loads data using RHBase and computes using MapReduce.

Table 1. The DSIMBench workflows.

Workflows	Data loading	Computation	Data source	Parallel method
W1	Single process	N/A	ext4	N/A
W2	Single process	N/A	HBase	N/A
W3	Single process	N/A	MySQL Cluster	N/A
W4	N/A	Single process	N/A	Vanilla R
W5	N/A	Multiple cores	N/A	MPI
W6	N/A	Multiple cores	N/A	MapReduce
W7	Multiple processes	Multiple cores	Best DB	MPI
W8	Multiple processes	Multiple cores	RHBase	MapReduce

4.1 Data Source Performance Assessment

In workflows W1–W3, as shown in Fig. 1, third-party R plugins are used to connect to the respective data sources. Hence, it is possible that the different implementations of these R plugins could interfere with the performance of the data source. In order to better assess the performance of the data sources in workflows W1–W3 we performed data loading tests using a user-based high-level API written in Java to directly load data from each data source to identify how much of the performance is affected by that middleware layer in R. The fastest source is then tested via a R plugin. If this R plugin on the source outperforms the other data sources via Java APIs, this data source will be used in the following W4–W6 tests.

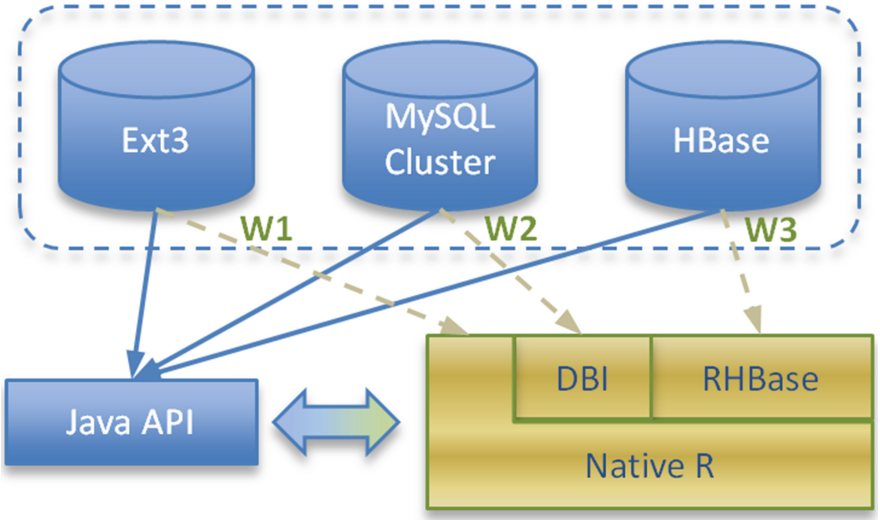


Fig. 1. Diagram illustrating how the loading test is organised.

4.2 Parallel Computation Benchmark Workflows

W4 in Fig. 2 is introduced as the baseline. W5 shows R Snowfall MPI computation via a LAM/MPICH2 cluster. The data distribution consists of two sequential steps: data loading and data copy. The input data matrix is loaded into LAM master node and then fully copied to all MPICH2 slave nodes. The calculation is carried out by the Snowfall `sfLapply()` function. `sfLapply()` mediates the distributed calculation in the slave nodes and collects the results. W6 indicates RHadoop MapReduce computation via a Apache Hadoop cluster. MapReduce in W6 utilises HDFS as Mapper task data source. Thus, the data distribution consists of two sequential steps. First, the input data matrix is split into data blocks and then uploaded into HDFS. The number of data blocks depend on the number of Mappers. After a MapReduce computation, all the results are stored in HDFS.

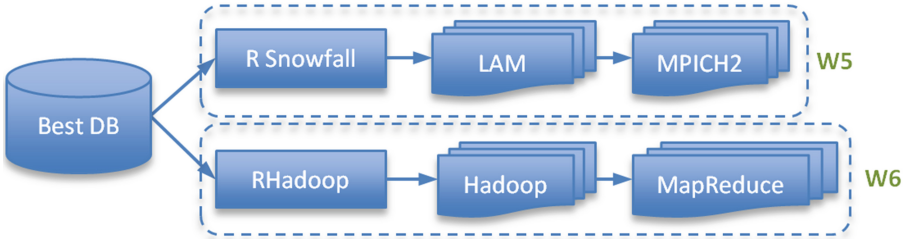


Fig. 2. Diagram illustrating how the parallel framework test is organised.

W7 in Fig. 3 manipulates MPICH2 tasks to load directly from the fastest data source based on W1–W3. If ext4 is applied to W7, the matrix data file will be split into data blocks and copied to each worker during the data preparation. The number of data blocks depends on the MPICH2 number. W8 manipulates Mapper tasks to load directly from HBase to test the built-in MapReduce HBase performance. RHadoop launches Mapper tasks without data loading. Each Mapper task loads data via built-in access to HBase Scanner and computes concurrently.

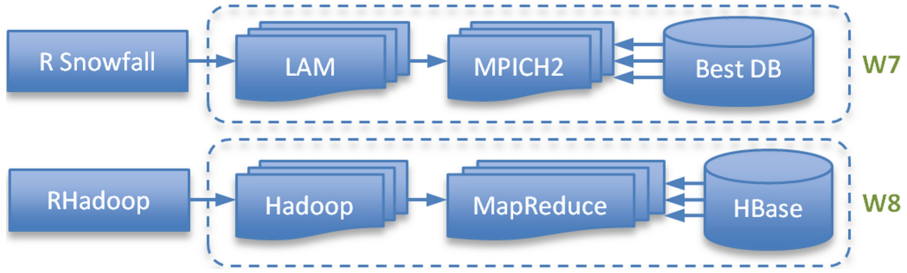


Fig. 3. Diagram illustrating R MapReduce with HBase.

5 DSIMBench Applications

5.1 Marker Selection

High-throughput gene expression analysis is a technique used to uncover disease specific gene signatures and gain further insight into disease mechanisms. In the past decade, gene expression measurements have shifted from quantitative assays capable of measuring the expression of single genes, to assays capable of assessing the levels of the majority of expressed genes in cells, tissues or organisms of interest. DNA microarray chips are the common technology platform used in recent years and are capable of simultaneous determination the entire human "transcriptome". In complex disease research, including diseases such as asthma and chronic obstructive pulmonary disease (COPD), microarray experiments are

performed on samples obtained from disease subjects and control (healthy) individuals. After the initial pre-processing steps which reduce background "noise", the expression intensity of genes present on each chip/sample are determined. Subsequently, deferentially expressed genes (DEGs) in disease compared to control samples are computed as well as the statistical significance of the difference. Finally, DEGs can be filtered by the relative levels of differential expression (fold-change) and significance (p-values; typically corrected for multiple testing: q-values).

The basic use case is to create a cohort between the patients and the control. For some more complicated ones, many clinical measurements are utilised to generate cohorts. A test case below was carried out using a large publicly available transcriptomic dataset taken from NCBI GEO [22] concerning Multiple Myeloma (GEO accession GSE24080; Popovici *et al.*, 2010 [23]). The dataset contains 559 subjects' gene expression data produced by an Affymetrix GeneChip Human Genome U133 Plus 2.0 Array. The cohorts are generated depending on patient medical therapies and survival time. This test case is utilised in workflows W1–W3 to test the data query in different number of subjects based on different cohorts.

5.2 Hierarchical Clustering

Genomic, proteomic and metabolic measurements have contributed to molecular profiling based patient stratification [24], such as identification of disease subgroups and the prediction of responses of individual subjects. Biomedical research is moving towards using high-throughput molecular profiling data to improve clinical decision-making. One approach for building classifiers is to classify subjects based on their molecular profiles. Unsupervised clustering algorithms can be utilised for stratification purposes.

Our benchmark applies three kinds of correlation methods used to generate correlation matrices that are used by the hierarchical clustering algorithm in tranSMART - the Pearson product-moment correlation, Spearmans rank-order correlation, and Euclidean distance correlation. The test case below was carried out using a large publicly available transcriptomic dataset taken from NCBI GEO concerning leukemia (GEO accession GSE13204; Kohlmann *et al.*, 2014 [25]). The dataset contains 2325 subjects' gene expression data produced by an Affymetrix GeneChip Human Genome U133 Plus 2.0 Array. The correlation matrix calculations could either be implemented on Hadoop, a popular and well supported distributed data storage and computation framework that supports MapReduce, or be implemented for distributed execution in R using Snowfall, a parallel computing package for R scripts. In this benchmark, all W4–W6 and the fastest one in W1–W3 are utilised to test the hierarchical clustering method.

6 Results

We performed the data loading test on marker selection and parallel tests on the hierarchical clustering on 4 virtual machines in our IC Cloud implementation.

The 4 VMs works on two physical machines with each 24 core and 64 GB memory. Each physical machine hosts 2 VMs.

- Operating system: CentOS Linux 2.6.18-308.24.1.el5xen
- CPU: 144 cores (Intel(R) Xeon(R) CPU E5-2630 0 @ 2.60GHz)
- Memory: 384 GB, DDR3, 1066 MHz
- Disk array: 24 TB (Huawei, OceanStor S5500T)
- Virtual machine: 8 virtual CPU cores, 8 GB memory

6.1 Data Loading Test

We chose 5 cohorts (10 cases) of different data size and the whole dataset of GSE24080 for a marker selection exercise, shown in Fig. 4. In the Java API test, loading from ext4 file system outperforms all the other data sources. A widely used vanilla R function scan greatly utilised for R the CSV file reading. RHBase performs better than scan only in the first data size. In the following parallel R test we choose ext4 as the data source.

6.2 R Parallel Framework Test

We utilised R function `rdist()` in package fields to calculate euclidean distance matrices, function `cor()` to calculate the Pearson and Spearman correlation matrices and function `hclust()` to cluster the correlation matrices. The result of W4 and W5 to compare parallel frameworks, shown in Fig. 5(a), indicates that when using the smaller MULTIMYEL dataset, MPI and MapReduce perform slower than vanilla R. W5 suffers from slow data transmission. The result of W7 and W8 to compare multi-thread data losing using different data sources, shown in Fig. 5(b), indicates it is faster for MPICH2 to load data directly from ext4 than HBase. W8 suffers from the long time RHBase data loading, as shown in Fig. 4. The vanilla R computation (W2) performs best in the small dataset, but does not scale up well in the large dataset. In the large dataset the better parallel methods in Fig. 5(a) (W6) and (b) (W7) are utilised to compare to W4. W6 and W7 outperformed W4. Though W6 computation time is a little longer than W7, W6 outperform W7 due to the faster data preparation.

7 Discussion

As shown in Fig. 5, the parallel methods suffer from data communication overheads such as transferring data to each worker, worker management and collecting data from the workers post-computation. But when size of the dataset increases, the advantages of parallel methods overcome these overheads. In Fig. 5(c), W6 and W7 have similar computation times, but W6 benefits from faster data preparation using HDFS. We considered using RHadoop with HBase at the beginning, however RHBase demonstrates poor data loading performance and is consequently much slower than the HBase Java API. RHBase does not perform well due to

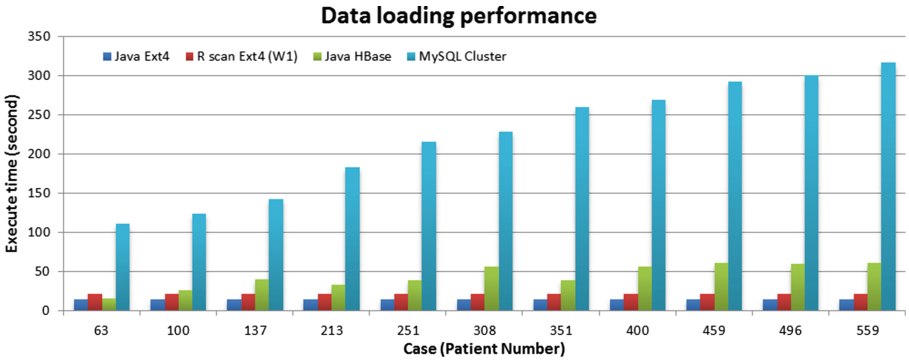
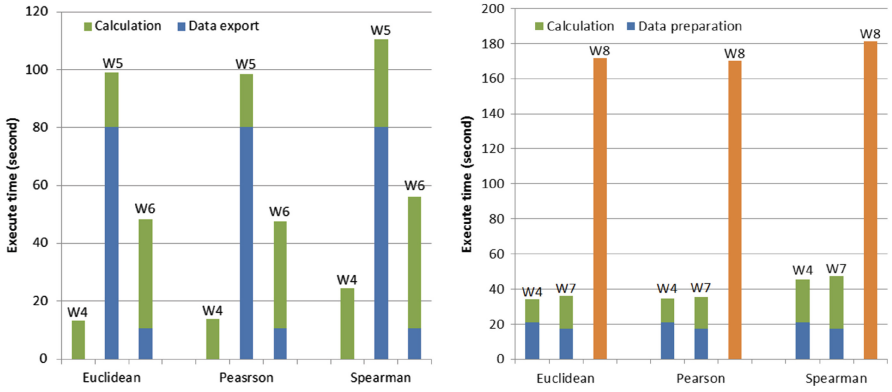
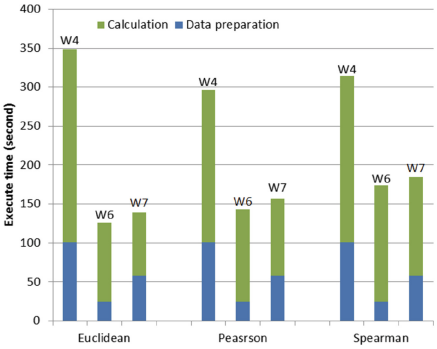


Fig. 4. Bar chart showing the performance evaluation in our data loading tests.



(a) Parallel R using single thread loading in small dataset.

(b) Parallel R using multi-thread loading in small dataset.



(c) Parallel R in large datset.

Fig. 5. Bar chart showing the performance evaluation in our computation tests.

the slow HBase Thrift [26] server. If RHBase could be implemented via rJava and HBase Java API, it may perform much better. Also, the data loading tests should introduce concurrent data loading tests before computation tests and full tests. As shown in Fig. 5, though parallel approaches can improve the data loading, optimisation of the matrix computation should not be neglected. R matrix calculations use a pure array object to gain significant performance using the CPU cache. Parallel methods divide a big matrix into small pieces and executes calculations by the low-speed R loop functions that cannot be pre-loaded in CPU cache due to potential R branch sentences. This is the reason why parallel methods can only perform 2 or 3 times faster than vanilla R when 32 CPU cores are utilised.

8 Conclusion

Big microarray data analysis using R is gaining significant focus as it's data access and computationally intensive workloads are in dire need to optimise their performance. We believe a new data mining benchmark is required to thoroughly analyse these analysis workflows and propose the most optimal workflow setup for them. In this paper, we presented DSIMBench, a benchmark containing two classic microarray analysis functions with six different parallel R workflows, and evaluated the benchmark in IC Cloud testbed platform.

Acknowledgment. This research was partially supported by the Innovative R&D Team Support Program of Guangdong Province (NO. 201001D0104726115), China, Johnson & Johnson Pharmaceutical and Research Comp, and Innovative Medicines Initiative (IMI), EU Grant Code 115446.

References

1. MySQL Cluster CGE. http://www.mysql.com/products/cluster/
2. Momjian, B.: PostgreSQL: introduction and concepts. J. Digit. Imaging Off. J. Soc. Comput. Appl. Radiol. **22**, 462 (2001). doi:10.1007/s10278-007-9097-5
3. Dirolf, K.C., Dorif, M.: MongoDB: The Definitive Guide. O'Reily Media, Sebastopol (2011)
4. George, L.: HBase The Definitive Guide. O'Reily Media, Sebastopol (2008)
5. Anon: MPI: A message passing interface. In: Proceedings of the Supercomputing Conference, pp. 878–883 (1993). doi:10.1109/SUPERC.1993.1263546
6. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008). doi:10.1145/1327452.1327492
7. R Development Core Team: R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2008). ISBN 3-900051-07-0, http://www.R-project.org
8. Athey, B.D., Braxenthaler, M., Haas, M., Guo, Y.: tranSMART: an open source and community-driven informatics and data sharing platform for clinical and translational research. In: Proceedings of the AMIA Joint Summits on Translational Science 2013, pp. 6–8, PMCID: PMC3814495 (2013)

9. Guo, L., Guo, Y., Tia, X.: IC cloud: a design space for composable cloud computing. In: Proceedings - 2010 IEEE 3rd International Conference on Cloud Computing, CLOUD 2010, pp. 394–401 (2010). doi:10.1109/CLOUD.2010.18
10. Henning, J.L.: SPEC CPU2006 benchmark descriptions. ACM SIGARCH Comput. Archit. News **34**(4), 1–17 (2006). doi:10.1145/1186736.1186737
11. TPC-H Benchmark. http://www.tpc.org/tpch/
12. Woo, S.C., Ohara, M., Torrie, E., Singh, J.P., Gupta, A.: The SPLASH-2 programs: characterization and methodological considerations. In: Proceedings 22nd Annual International Symposium on Computer Architecture (1995). doi:10.1109/ISCA.1995.524546
13. Fritts, J.E., Steiling, F.W., Tucek, J.A.: MediaBench II Video: Expediting the Next Generation of Video Systems Research. In: Proceedings of the SPIE, Embedded Processors for Multimedia and Communications II, vol. 5683, pp. 79–93 (2005)
14. Albayraktaroglu, K., Jaleel, A., Wu, X., Franklin, M., Jacob, B., Tseng, C.W., Yeung, D.: BioBench: a benchmark suite of bioinformatics applications. In: ISPASS 2005 - IEEE International Symposium on Performance Analysis of Systems and Software, vol. 2005, pp. 2–9 (2005). doi:10.1109/ISPASS.2005.1430554
15. Bader, D.A., Li, Y., Li, T., Sachdeva, V.: BioPerf: a benchmark suite to evaluate high-performance computer architecture on bioinformatics applications. In: Proceedings of the 2005 IEEE International Symposium on Workload Characterization, IISWC-2005, vol. 2005, pp. 163–173 (2005). doi:10.1109/IISWC.2005.1526013
16. Narayanan, R., Ozisikyilmaz, B., Zambreno, J., Memik, G., Choudhary, A.: MineBench: a benchmark suite for data mining workloads. In: Proceedings of the 2006 IEEE International Symposium on Workload Characterization, IISWC - 2006, pp. 182–188 (2006). doi:10.1109/IISWC.2006.302743
17. Knaus, J., Porzelius, C., Binder, H.: Easier parallel computing in R with snowfall and sfCluster. Source **1**, 54–59 (2009)
18. Yu, H.: Rmpi: parallel statistical computing in R. R News 2, 10–14 (2002). http://cran.r-project.org/doc/Rnews/Rnews_2002-2.pdf
19. Squyres, J.M.: A component architecture for LAM/MPI. ACM SIGPLAN Not. (2003). doi:10.1145/966049.781510
20. Bridges, P., Doss, N., Gropp, W., Karrels, E., Lusk, E., Skjellum, A.: User Guide to MPICH, a Portable Implementation of MPI. Argonne National Laboratory, 9700, 60439–64801 (1995)
21. White, T.: Hadoop: The Definitive Guide. O'Reilly Media, Sebastopol (2012)
22. Barrett, T., Wilhite, S.E., Ledoux, P., Evangelista, C., Kim, I.F., Tomashevsky, M., Soboleva, A.: NCBI GEO: Archive for functional genomics data sets - Update. Nucleic Acids Res.,**41** (2013). doi:10.1093/nar/gks1193
23. Popovici, V., Chen, W., Gallas, B.G., Hatzis, C., et al.: Effect of training-sample size and classification difficulty on the accuracy of genomic predictors. Breast Cancer Res. **12**(1), R5 (2010)
24. Stoughton, R.B., Friend, S.H.: How molecular profiling could revolutionize drug discovery. Nat. Rev. Drug Discov. **4**, 345–350 (2005). doi:10.1038/nrd1696
25. Kohlmann, A., Kipps, T.J., Rassenti, L.Z., Downing, J.R., et al.: An international standardization programme towards the application of gene expression profiling in routine leukaemia diagnostics: the Microarray Innovations in LEukemia study prephase. Br. J. Haematol. **142**(5), 802–807 (2008)
26. Apache Thrift. http://thrift.apache.org